Consiglio Nazionale delle Ricerche

ISTITUTO DI ELABORAZIONE DELLA INFORMAZIONE

PISA

Stimolatore multisensoriale per prove attitudinali.

Progetto e implementazione del software

L.Bedini, E.Bozzi, A.Ribolini

Nota Interna B4-32

Luglio 1989

Stimolatore Multisensoriale per prove attitudinali

Progetto e implementazione del software

L.Bedini, E.Bozzi, A.Ribolini

Attivita' svolta nell'ambito di un contratto di collaborazione con la Societa' Tecnobiomedica.

INDICE

1. Introduzione

Il software ha una struttura modulare e puo' essere facilmente configurato per la realizzazione dei vari tipi di prototipi.

La specializzazione del software si ottiene assegnando valori opportuni ai parametri definiti nel file MULTISEN.INI.

Nella presente documentazione vengono descritte le modalita' di specializzazione del software utilizzato nei due prototipi di stimolazione multisensoriale. Viene riportato l'elenco ed una breve descrizione dei vari file. Per comodita' dell'utente si riporta anche l'elenco in ordine alfabetico di tutte le procedure costituenti il programma con a fianco di ciascuna il nome del file che la contiene. Vengono infine allegati i listati del codice sorgente prodotto. Poiche' il codice sorgente e' stato scritto in linguaggio C, che presenta una elevata comprensibilita', come documentazione software si ritiene sufficiente fornire i listati commentati.

2. Specializzazione del software

Il file MULTISEN.INI permette la specializzazione del software per la configurazione dello strumento con il quale si deve operare. Le informazioni lette dal presente file forniscono al programma principale tutte le indicazioni necessarie al normale funzionamento. In particolare i nomi delle variabili chiave riconosciuti sono i seguenti:

MACCHINA
ARC_SEM
ARC_DISC
VISI_DEF
UDIT_DEF
SOME_DEF
FILE_FMT
FREQUENZ

Dette variabile possono essere cosi' utilizzate:

MACCHINA:

determina il tipo di strumento da inizializzare:

1=strumento completo di somestesico

2=strumento incompleto (solo visivo e uditivo).

Esempio: MACCHINA=1

ARC_SEM:

indica il nome del file contenente l'archivio relativo alle prove semplici da utilizzarsi per le successive sessioni.

Esempio: ARC_SEM=ARCHIVIO.SEM

ARC_DISC:

indica il nome del file contenente l'archivio relativo alle prove di discriminazione da utilizzarsi per le successive sessioni.

Esempio: ARC_DISC=ARCHIVIO.DIS

VISI_DEF:

indica il nome del file contenente i parametri di default utilizzati dallo stimolatore visivo.

Esempio: VISI_DEF=VISIVO.DEF

UDIT_DEF:

indica il nome del file contenente i parametri di default utilizzati dallo stimolatore uditivo.

Esempio: UDIT_DEF=UDITIVO.DEF

SOME_DEF:

indica il nome del file contenete i parametri di default utilizzati dallo stimolatore somestesico.

Esempio: SOME_DEF=SOMESTES.DEF

FILE_FMT:

indica il numero di file archivio che si possono creare sul dischetto. Dipende dal tipo di driver presente sul PC utilizzato.

da 1 a 18 consigliabile per dischetti da 360K

da 1 a 50 consigliabile per dischetti da 1200K

Esempio: FILE_FMT=18

FREQUENZ:

indica la frequenza di clock dell'oscillatore montato sulla scheda di interfaccia installata sul PC.

Esempio: FREQUENZ=1000

3. Elenco dei file contenenti il programma sorgente

Il codice sorgente che costituisce il programma MULTISEN e' suddiviso in 12 file. Di seguito viene riportata una breve descrizione del contenuto di ciascun file e l'elenco delle procedure contenute in ciascuno di essi:

FLOPPY.C:

contiene le procedure per la scrittura e la lettura degli archivi residenti sia su disco rigido che su dischetto, le procedure relative alla inizializzazione eventuale di un nuovo dischetto e le procedure di stampa dei dati intermedi, o finali, per archiviazione su supporto cartaceo.

Elenco delle procedure:

```
scelta_archiviazione(opt)
archivia_floppy()
cambia_formatta()
esegue_inizializzazione()
azzera_righe(flag)
zona_messaggi()
in_corso()
erase_corso()
write_data(ia,iv)
lista_dischetto()
mediana(array,ncelle,fmedi)
carica_array()
```

GESFILE.C:

contiene le procedure per la gestione dei vari file utilizzati dal programma, le varie procedure di caricamento, modifica e riscrittura degli archivi aggiornati e le procedure di revisione dei parametri riservati alla generazione dei numeri casuali.

Elenco delle procedure:

```
gestisce_file()
search(stringa1,stringa2)
apri_file(file_name,modo)
inizializza_ran()
chiude_file_ran()
carica_casuali(minimo,massimo,array,ncelle)
legge_dati_default(unit,array1,array2,ndata)
messaggio(indice)
carica_archivio(unit,archivio,osservazioni,ncelle)
scrive_archivio(unit,archivio,osservazioni,ncelle)
crea_indice(flag,sesso,eta,stimolatore,arto1,arto2)
aggiorna_buffer(indice,archivio,osservazioni,x,y,w,z)
```

ITBM.C:

contiene le procedure di comunicazione relative al protocollo ITBM.

Elenco delle procedure:

```
send_addr(indirizzo)
send_data(byte)
reset(time_out)
ini_per()
receive()
sense_bit(ibit,direction,time_out)
trigger(to)
send_data_some(byte)
send_addr_some()
receive_some()
ciclocoa()
```

MULTISEN.C:

contiene il programma principale e le definizioni dei vettori di programmazione degli stimolatori.

Elenco delle procedure:

main(argc,argv)
fasce()
videof(flag)
presenta(ir,ic,ia,is)

SOMESTES.C:

contiene le procedure di programmazione e di controllo dello stimolatore somestesico.

Elenco delle procedure:

somestesico(param)
edit_somestesico()
presenta_reverse(indice)
presenta_normale(indice)
carica_vettore(funzione,modo,vettore)
presenta_vettore(vettore)
tx_somestesico()
trasmette_dati(vettore,start,end)
iniz_somestesico()
attende_number(iriga,icolo,attrib,nchar_max,buffer)

TIMER.C:

contiene le procedure di gestione del timer.

Elenco delle procedure:

```
start_timer()
stop_timer()
load_timer(word)
legge_timer()
trasmette_comandi(str,cod)
```

UDITIVO.C:

contiene le procedure di programmazione e di controllo dello stimolatore uditivo.

Elenco delle procedure:

```
uditivo(param)
edit_uditivo()
tx_uditivo()
varia_parametro()
intensifica()
normal()
blinka()
rc_put(irig,icol,istri,iattr)
scrivi_attuali()
aggiorna()
aggiorna_val(para)
scrivi_param()
scrivix(par)
iniz_udi()
```

VALUTA.C:

contiene le procedure di valutazione dei risultati relativi sia alle prove semplici che a quelle di discriminazione.

Elenco delle procedure:

valutazione(opt)
valuta_semplici()
valuta_discrimi()
prinvaluta()
dati_tastiera(fmed,fdev,ioss,opt)
screen_val_sem()
screen_val_dis()

VIDEATE.C:

contiene tutte le procedure di presentazione video relative alla esecuzione delle prove di reazione semplici.

Elenco delle procedure:

test_hardware()
tipo_di_prova(flag)
anagrafe_ok()
anagrafe()
attende_char(iriga,icolo,attrib,nchar_max,buffer,tipo,array)
apre_temporaneo()
stampa_temporaneo()
prove_semplici()
clear_cursore(posizione)
write_cursore(posizione)
presito(posizione)
messaggio_assenza(posizione)
calcola_libero(posizione,direzione)
determina_stimart(posizione)
riprogramma_stimolatore()
esame_apprendimento()
apprendimento_semplice()

numero_tipo(f)
esegui_apprendi_semplice()
singola_prova(casuale,arto,lim1,lim2)
wait(tick)
wait_uno()
esame_semplice(posizione)
presenta_dato_float(iriga,icolo,attributo,valore)
presenta_dato_integer(iriga,icolo,attributo,valore)
archiviazione_fisso()
determina_arti(parola)
data_int(array)
inverte_data(array)
system_op()

VIDEATE1.C:

contiene le procedure di presentazione video relative alla esecuzione delle prove di discriminazione.

Elenco delle procedure:

prove_discriminazione()
crea_informazioni()
on(indice)
tag(indice)
off(indice)
disabilita_old(indice1,indice2,indice)
abilita(indice)
crea_possibilita(byte)
controlla_effettuati(byte,numero)
apprendimento_discriminazione()
esegui_apprendi_discriminazione()
esame_discriminazione(posizione)
tasto_start(mess0,mess1)
valuta(ireturn,nrecord,ncasual,posizione)
prepara_video(cpaziente,cdata,data_odierna)
presenta_valori(indice)
esegui_ab(tipo,riga,colo,random,stimolo,artoselezionato)
crea_stringa(byte,stringa)

VISIVO.C:

contiene le procedure di programmazione e di controllo dello stimolatore visivo.

Elenco delle procedure:

visivo(param)
edit_visivo()
tx_visivo()
scrivi(par)
scrivis(par)
brinc()
incr()
contrev()
iniz_vis()

VISUALIZ.C:

contiene la procedura per la presentazione su video dei dati anagrafici relativi al soggetto in esame.

Elenco delle procedure:

visualizza_dati(opt)

4. Elenco alfabetico delle procedure

34) dati_tastiera(fmed,fdev,ioss,opt) ..... valuta.C
35) determina_arti(parola) ..... videate.C
36) determina_stimart(posizione) ..... videate.C
37) disabilita_old(indice1,indice2,indice) videate1.C
38) edit_somestesico() ..... somestes.C
39) edit_uditivo() ..... uditivo.C
40) edit_visivo() ..... visivo.C
41) erase_corso() ..... floppy.C
42) esame_apprendimento() ..... videate.C
43) esame_discriminazione(posizione) videate1.C
44) esame_semplice(posizione) ..... videate.C
45) esegue_inizializzazione() ..... floppy.C
46) esegui_ab(tipo,riga,colo,random,stimolo,artoselezionato) videate1.C
47) esegui_apprendi_discriminazione() videate1.C
48) esegui_apprendi_semplice() ..... videate.C
49) fasce() ..... multisen.C
50) gestisce_file() ..... gesfile.C
51) in_corso() ..... floppy.C
52) incr() ..... visivo.C
53) ini_per() ..... itbm.C
54) iniz_somestesico() ..... somestes.C
55) iniz_udi() ..... uditivo.C
56) iniz_vis() ..... visivo.C
57) inizializza_ran() ..... gesfile.C
58) intensifica() ..... uditivo.C
59) inverte_data(array) ..... videate.C
60) legge_dati_default(unit,array1,array2,ndata) ..... gesfile.C
61) legge_timer() ..... timer.C
62) lista_dischetto() ..... floppy.C
63) load_timer(word) ..... timer.C
64) main(argc,argv) ..... multisen.C
65) mediana(array,ncelle,fmedi) ..... floppy.C
66) messaggio(indice) ..... gesfile.C
67) messaggio_assenza(posizione) ..... videate.C
68) normal() ..... uditivo.C
69) numero_tipo(f) ..... videate.C
70) off(indice) videate1.C
71) on(indice) videate1.C

72) prepara_video(cpaziente,cdata,data_odierna)..... videatel.C
73) presenta(ir,ic,ia,is)..... multisen.C
74) presenta_dato_float(iriga,icolo,attributo,valore)..... videate.C
75) presenta_dato_integer(iriga,icolo,attributo,valore)..... videate.C
76) presenta_normale(indice)..... somestes.C
77) presenta_reverse(indice)..... somestes.C
78) presenta_valori(indice)..... videatel.C
79) presenta_vettore(vettore)..... somestes.C
80) presito(posizione)..... videate.C
81) prinvaluta()..... valuta.C
82) prove_discriminazione()..... videatel.C
83) prove_semplici()..... videate.C
84) rc_put(irig,icol,istri,iattr)..... uditivo.C
85) receive()..... itbm.C
86) receive_some()..... itbm.C
87) reset(time_out)..... itbm.C
88) riprogramma_stimolatore()..... videate.C
89) scelta_archiviazione(opt)..... floppy.C
90) screen_val_dis()..... valuta.C
91) screen_val_sem()..... valuta.C
92) scrive_archivio(unit,archivio,osservazioni,ncelle)..... gesfile.C
93) scrivi(par)..... visivo.C
94) scrivi_attuali()..... uditivo.C
95) scrivi_param()..... uditivo.C
96) scrivis(par)..... visivo.C
97) scrivix(par)..... uditivo.C
98) search(stringa1,stringa2)..... gesfile.C
99) send_addr(indirizzo)..... itbm.C
100) send_addr_some()..... itbm.C
101) send_data(byte)..... itbm.C
102) send_data_some(byte)..... itbm.C
103) sense_bit(ibit,direction,time_out)..... itbm.C
104) singola_prova(casuale,arto,lim1,lim2)..... videate.C
105) somestesico(param)..... somestes.C
106) stampa_temporaneo()..... videate.C
107) start_timer()..... timer.C
108) stop_timer()..... timer.C
109) system_op()..... videate.C

110) tag(indice) videate1.C
111) tasto_start(mess0,mess1) videate1.C
112) test_hardware() ..... videate.C
113) tipo_di_prova(flag) ..... videate.C
114) trasmette_comandi(str,cod) ..... timer.C
115) trasmette_dati(vettore,start,end) ..... somestes.C
116) trigger(to) ..... itbm.C
117) tx_somestesico() ..... somestes.C
118) tx_uditivo() ..... uditivo.C
119) tx_visivo() ..... visivo.C
120) uditivo(param) ..... uditivo.C
121) valuta(ireturn,nrecord,ncasual,posizione) videate1.C
122) valuta_discrimi() ..... valuta.C
123) valuta_semplici() ..... valuta.C
124) valutazione(opt) ..... valuta.C
125) varia_parametro() ..... uditivo.C
126) videof(flag) ..... multisen.C
127) visivo(param) ..... visivo.C
128) visualizza_dati(opt) ..... visualiz.C
129) wait(tick) ..... videate.C
130) wait_uno() ..... videate.C
131) write_cursore(posizione) ..... videate.C
132) write_data(ia,iv) ..... floppy.C
133) zona_messaggi() ..... floppy.C

5. Listato del programma per file

```
/************************************ File: FLOPPY.C              */

struct disp17 {
char stringa17[50];
int posizione_riga17;
int posizione_colo17;
int attributo17;
};

struct disp17 screen17[18]={
"        STIMOLATORE MULTISENSORIALE          ",1,14,23,
"                                             ",2,14,71,
"   S C E L T A   A R C H I V I A Z I O N E   ",3,14,71,
"                                             ",4,14,71,
"                          ",7,20,23,
"    1: D I S C H E T T O       ",8,20,23,
"                          ",9,20,23,
"    2: D I S C O   F I S S O   ",10,20,23,
"                          ",11,20,23,
"    3: S T A M P A N T E       ",12,20,23,
"                          ",13,20,23,
"    4: V A L U T A Z I O N E   ",14,20,94,
"                          ",15,20,23,
"    0: E S C E                  ",16,20,71,
"                          ",17,20,23,
"                                             ",21,15,71,
" <ESC> menu` precedente     ALT-Y Reinizializza ",22,15,71,
"          Introduci: 0 - 1 - 2 - 3           ",23,15,71,
};

scelta_archiviazione(opt)
int opt;
{
   extern char nuovo_paziente;
   int  a,iprinter,stat;
   char risp,*prnt,*mess1,*mess2;
   char flag,bell=7,test;

   prnt="PRN:";
   mess1="    ARCHIVIAZIONE SU FLOPP' DEL SOGGETTO GIA' AVVENUTA      ";
   mess2="    ARCHIVIAZIONE SU FISSO I  . SOGGETTO GIA' AVVENUTA      ";
   flag=0x00;
   while(!flag)
   {
     scr_clr();
     for (a=0; a<18; a++)
       presenta(screen17[a].posizione_riga17,screen17[a].posizione_colo17,
            screen17[a].attributo17,screen17[a].stringa17);
     fasce();
     while(((risp=ci())!='1')&&(risp!='2')&&(risp!='3')&&(risp!='4')&&
          (risp!='0')&&(risp!=27)&&(risp!=21));
     switch (risp)
     {
       case '1': test=nuovo_paziente&1;
                 if(test!=0)
                 {
```

```
                presenta(19,10,71,mess1);
                printf("%c",bell);
                risp=ci();
               }
               else
               {
                archivia_floppy();
                nuovo_paziente=nuovo_paziente1;
               }
               break;
        case '2': test=nuovo_paziente&2;
               if(test!=0)
               {
                presenta(19,10,71,mess2);
                printf("%c",bell);
                risp=ci();
               }
               else
               {
                archiviazione_fisso();
                nuovo_paziente=nuovo_paziente2;
               }
               break;
        case '3': iprinter=open(prnt,1);
               write_data(iprinter,iprinter);
               stat=close(iprinter);
               break;
        case '4': valutazione(opt);
               break;
        case 21 : exit();
               break;
        case 27 : return(27);
               break;
        case '0': flag=0xFF;
               break;
      }
    }
   return(0);
}
  int  unitm,unita,unitv,status;
  char filem[15],filea[15],filev[15];
  char code,risp;
  char date[]={0,0,0,0,0,0,0,0,0,0},time[]={0,0,0,0,0,0,0,0,0,0},dummy[80];
  int  nfile1,nfile2;
  float nbytetot;

archivia_floppy()
{
  extern int ftotali;
  long  lval;
  int   a,j,k;
  char  record[132],bell=7;
  char  *mess1,*mess2,*mess3;

  mess1="                                        ";
  mess2="  A R C H I V I A Z I O N E  S U  D I S C H E T T O  ";
  scr_clr();
  presenta(1,12,71,mess1);
```

```
presenta(2,12,71,mess2);
presenta(3,12,71,mess1);
fasce();
zona_messaggi();
sprintf(filem,"A:-MASTER-.IDX");
sprintf(record,"ASSICURATI DI AVER INTRODOTTO UN DISCHETTO NELLA UNITA' A:");
presenta(10,13,23,record);
sprintf(record," E DI AVER CHIUSO LO SPORTELLO DEL DRIVER CHE LO CONTIENE ");
presenta(11,13,23,record);
sprintf(record,"      --- BATTI UN TASTO QUALSIASI QUANDO PRONTO ---     ");
presenta(12,13,23,record);
printf("%c",bell);
risp=ci();
zona_messaggi();
unitm=open(filem,2);
if(unitm==-1)
{
  zona_messaggi();
  sprintf(record,
    "DISCHETTO NON INIZIALIZZATO CORRETTAMENTE-MANCA FILE MASTER");
  presenta(10,13,23,record);
  nfile2=cambia_formatta();
}
else
  status=fscanf(unitm,"%d %d %f %s\n",&nfile1,&nfile2,&nbytetot,&dummy);
while (nfile2==0)
{
  status=close(unitm);
  erase_corso();
  zona_messaggi();
  sprintf(record,"IL DISCHETTO INSERITO NON CONTIENE PIU` SPAZIO LIBERO");
  presenta(10,13,23,record);
  nfile2=cambia_formatta();
}
nfile1++;
nfile2--;
nbytetot-=31000.0;
lval=rewind(unitm);
dates(date);
times(time);
status=fprintf(unitm,"%2d %2d %f %s %s\n",
          nfile1,nfile2,nbytetot,date,time);
status=close(unitm);
sprintf(filea,"A:ANAGRAFE.%0*d",3,nfile1);
sprintf(filev,"A:VALORICA.%0*d",3,nfile1);
unita=creat(filea);
if(unita==-1)  exit();
unitv=creat(filev);
if(unitv==-1)  exit();
status=close(unita);
status=close(unitv);
unita=open(filea,1);
if(unita==-1)  exit();
unitv=open(filev,1);
if(unitv==-1)  exit();
write_data(unita,unitv);
status=close(unita);
status=close(unitv);
```

```
}


cambia_formatta()
{
  extern int ftotali;
  int ireturn,status;
  char risp,out,record[80],flag,bell=7;
  erase_corso();
  sprintf(record,"VUOI CAMBIARE DISCHETTO O INIZIALIZZARNE UNO NUOVO (C/I)?");
  presenta(11,13,23,record);
  while(((risp=ci())!='c')&&(risp!='i')&&(risp!='C')&&(risp!='I'));
  if((risp=='i')||(risp=='I'))
  {
    esegue_inizializzazione();
    ireturn=ftotali;
  }
  else
  {
    flag=0;
    while(!flag)
    {
      zona_messaggi();
      sprintf(record,"  BATTI UN TASTO QUALSIASI QUANDO PRONTO.....");
      presenta(10,13,23,record);
      risp=ci();
      unitm=open(filem,2);
      if(unitm==-1)
      {
        printf("%c",bell);
        ireturn=0;
        status=close(unitm);
      }
      else
      {
        status=fscanf(unitm,"%d %d %f %s\n",&nfile1,&nfile2,
                &nbytetot,&dummy);
        ireturn=nfile2;
      }
    }
  }
  return(ireturn);
}


esegue_inizializzazione()
{
/* extern int printer;  */
  extern int ftotali;
  int status;
  char risposta,bell=7,record[80];
  long lval;
  zona_messaggi();
  erase_corso();
  sprintf(record,"PREGO INSERIRE UN NUOVO DISCHETTO NELL'UNITA` A:");
  presenta(10,13,23,record);
```

```
    sprintf(record,"BATTERE UN TASTO QUALSIASI E SEGUIRE LE INDICAZIONI DEL SO");
    presenta(11,13,23,record);
    printf("%c",bell);
    risposta=ci();
    zona_messaggi();
    in_corso();
    azzera_righe(1);
    scr_rowcol(11,1);
    code=exec("C:\\COMMAND.COM","/cDEL A:*.*");
    if(code==255)  exit();
 /*---disco formattato---*/
    azzera_righe(0);
    fasce();
    zona_messaggi();
    unitm=creat(filem);
    if(unitm==-1)  exit();
    unitm=open(filem,2);
    if(unitm==-1)  exit();
    dates(date);
    times(time);
    status=fprintf(unitm,"0 %2d 360000.000000 %8c %8c\n",ftotali,date,time);
    lval=rewind(unitm);
    erase_corso();
 }

 azzera_righe(flag)
 int flag;
 {
   int ir,a,b,c;
   char *riga;

 riga="================================================================================
 ==========";
   for(ir=10; ir<21; ir++)
   {
     scr_rowcol(ir,0);
     scr_clrl();
   }
   if(flag==1)
   {
     presenta(10,1,23,riga);
     presenta(20,1,23,riga);
   }
 }


 zona_messaggi()
 {
  char *spazi;
  spazi="                                                    ";
  int ir,ic=10;
  for(ir=10; ir<15; ir++)
    presenta(ir,ic,23,spazi);
 }

 in_corso()
 {
   char *mess;
```

```
  mess="      O P E R A Z I O N E   I N   C O R S O      ";
  presenta(21,11,71,mess);
}


erase_corso()
{
  char *mess;
  mess="                                                ";
  presenta(21,11,7,mess);
}

  int possibili[8]={17,20,27,30,81,84,91,94};

  char *identi[12];
  char *id0="VISIVO MANO DESTRA       ";
  char *id1="VISIVO MANO SINISTRA     ";
  char *id2="VISIVO PIEDE DESTRO      ";
  char *id3="VISIVO PIEDE SINISTRO    ";
  char *id4="UDITIVO MANO DESTRA      ";
  char *id5="UDITIVO MANO SINISTRA    ";
  char *id6="UDITIVO PIEDE DESTRO     ";
  char *id7="UDITIVO PIEDE SINISTRO   ";
  char *id8="SOMESTESICO MANO DESTRA  ";
  char *id9="SOMESTESICO MANO SINISTRA ";
  char *id10="SOMESTESICO PIEDE DESTRO ";
  char *id11="SOMESTESICO PIEDE SINISTRO ";
  char *idiscri[8];
  char *ids0="VISIVO  MANO  DS - MANO  SN ";
  char *ids1="VISIVO  MANO  SN - MANO  DS ";
  char *ids2="VISIVO  PIEDE DS - PIEDE SN ";
  char *ids3="VISIVO  PIEDE SN - PIEDE DS ";
  char *ids4="UDITIVO MANO  DS - MANO  SN ";
  char *ids5="UDITIVO MANO  SN - MANO  DS ";
  char *ids6="UDITIVO PIEDE DS - PIEDE SN ";
  char *ids7="UDITIVO PIEDE SN - PIEDE DS ";

write_data(ia,iv)
int ia,iv;
{
  extern char  work_visivo[],work_uditivo[],work_somestesico[];
  extern char  cognome[],nome[],giorno[],mese[],anno[];
  extern char  sesso,eta[],residenza[],indirizzo[];
  extern char  documento[],num_doc[],progressivo[],prev,street[];
  extern char  type_sem,type_dis,esito_sem,esito_dis;
  extern float media[],scarto[],dimedia[],discarto[];
  extern float varianz[],assolut[],divarianz[],diassolut[];
  extern int   anni,numero_prog,gia_effettuate;
  extern int   validi[],errate[],assenti[];
  extern int   divalidi[],dierrate[],diassenti[];
  extern char possibilita[];
  int a,b,c,d,j,k,status,prove,dato1,dato2,dato3;
  char risposta,ff=0x0C;

  in_corso();
  status=fprintf(ia,"%s %s Nato il: %s/%s/%s\n",
            cognome,nome,giorno,mese,anno);
  status=fprintf(ia,"Sesso: %c Eta`: %d Residente a: %s\n",
```

```
                sesso,anni,residenza);
      status=fprintf(ia,"Indirizzo: %s\n",indirizzo);
      status=fprintf(ia,"Documento: %s Numero: %s Prog.: %04d  Arto: %c\n",
                documento,num_doc,numero_prog,prev);
      status=fprintf(iv,"\n --- Parametri di calibrazione stimolatori --- \n");
      for(a=0; a<6; a++)
        status=fprintf(iv,"%0*x ",2,work_visivo[a]);
      status=fprintf(iv," - VISIVO -\n");
      for(a=0; a<19; a++)
        status=fprintf(iv,"%0*x ",2,work_uditivo[a]);
      status=fprintf(iv," - UDITIVO -\n");
      for(a=0; a<13; a++)
        status=fprintf(iv,"%0*x ",2,work_somestesico[a]);
      status=fprintf(iv," - SOMESTESICO -\n");
      status=fprintf(iv,"\n--- Valori relativi alle prove effettuate ---\n");

      for(prove=0; prove<12; prove++)
      {
        if(street[prove]==0x55)
        {
          status=fprintf(iv,"--- COMBINAZIONE: %s ---\n",identi[prove]);
          for(j=prove*20; j<(prove*20)+20; j+=10)
          {
            for(k=j; k<j+10; k++)
              status=fprintf(iv,"%4d ",validi[k]);
            status=fprintf(iv,"\n");
          }
          status=fprintf(iv,"RISPOSTE ERRATE     = %d\n",errate[prove]);
          status=fprintf(iv,"RISPOSTE ASSENTI    = %d\n",assenti[prove]);
          status=fprintf(iv,"VALORE MEDIO        = %6.1f\n",media[prove]);
          status=fprintf(iv,"DEVIAZIONE STANDARD = %6.1f\n",scarto[prove]);
          status=fprintf(iv,"MEDIANA             = %6.1f\n",varianz[prove]);
          status=fprintf(iv,"VALORE ASSOLUTO SC. = %6.1f\n",assolut[prove]);
          status=fprintf(iv,"----------------------------\n");
        }
      }
      if((esito_sem=='P')||(esito_sem=='N'))
      {
        if(esito_sem=='P')
          status=fprintf(iv,"ESITO DELLE PROVE SEMPLICI = POSITIVO   (%c)\n",type_sem);
        else
          status=fprintf(iv,"ESITO DELLE PROVE SEMPLICI = NEGATIVO   (%c)\n",type_sem);
        status=fprintf(iv,"=============================================================\n\n");
      }
      else
      {
        status=fprintf(iv,"ESITO DELLE PROVE SEMPLICI = NON ESEGUITA VALUTAZIONE\n");
        status=fprintf(iv,"=============================================================\n\n");
      }

    /* fine prove semplici */

      for(prove=0; prove<gia_effettuate; prove++)
      {
    /*    dato1=prove;
        status=fprintf(iv,"--- COMBINAZIONE: %s ---\n",idiscri[prove]);  */

        dato1=possibilita[prove];
```

```
    status=fprintf(iv,"--- COMBINAZIONE: %s ---\n",idiscri[dato1]);
    for(j=dato1*40; j<(dato1*40)+40; j+=10)
    {
      for(k=j; k<j+10; k++)
        status=fprintf(iv,"%4d ",divalidi[k]);
      status=fprintf(iv,"\n");
    }
    status=fprintf(iv,"RISPOSTE ERRATE    = %d\n",dierrate[dato1]);
    status=fprintf(iv,"RISPOSTE ASSENTI   = %d\n",diassenti[dato1]);
    status=fprintf(iv,"VALORE MEDIO       = %6.1f\n",dimedia[dato1]);
    status=fprintf(iv,"DEVIAZIONE STANDARD = %6.1f\n",discarto[dato1]);
    status=fprintf(iv,"MEDIANA            = %6.1f\n",divarianz[dato1]);
    status=fprintf(iv,"VALORE ASSOLUTO SC. = %6.1f\n",diassolut[dato1]);
    status=fprintf(iv,"----------------------------\n");
  }
  if((esito_dis=='P')||(esito_dis=='N'))
  {
    if(esito_dis=='P')
     status=fprintf(iv,"ESITO DELLE PROVE DISCRIMINAZIONE = POSITIVO
(%c)\n",type_dis);
    else
     status=fprintf(iv,"ESITO DELLE PROVE DISCRIMINAZIONE  = NEGATIVO
(%c)\n",type_dis);
    status=fprintf(iv,"================================================\n\n");
  }
  else
  {
    status=fprintf(iv,"ESITO DELLE PROVE DISCRIMINAZIONE = NON ESEGUITA
VALUTAZIONE\n");
    status=fprintf(iv,"================================================\n\n");
  }
  erase_corso();
  if(ia==iv)
    status=fprintf(ia,"%c",ff);
}


lista_dischetto()
{
/*  extern int printer;  */
  char *nome,*riga1,*riga2,*cp,risp;
  char *mess1,*mess2,*mess3,*mess4,*mess5,*mess6,*mess7,*mess8,*vuoto;
  char risposta,dummy[80],record[80],bell=7,*file,*file_open;
  int  a,b,c,status,num,unitm,indice,jrig;
  int  nbyte1,nbyte2,leng,lin=18,fr=4,fc=5,tr=22,tc=75;
  float nbytetot;

  nome="A:-MASTER-.IDX";
  riga1="                                        ";
  riga2="    L I S T A  D A T I  D E L  D I S C H E T T O     ";
  vuoto="                                        ";
  mess1="    DISCHETTO NON INIZIALIZZATO CORRETTAMENTE    ";
  mess2="    MANCA IL FILE INDICE .....      ";
  mess3="  BATTI UN TASTO QUALSIASI PER PROSEGUIRE ....    ";
  mess4="  Introduci un dischetto nell'unita` A: .....    ";
  mess5="                <esc> per uscire                 ";
  mess6="    SCEGLI OPZIONE:                             ";
  mess7="       1. : LISTA NOMINATIVI MEMORIZZATI        ";
```

```
mess8="      2. : LISTA VALORI MEMORIZZATI          ";

unitm=-1;
while(unitm==-1)
{
  scr_clr();
  presenta(0,10,71,riga1);
  presenta(1,10,71,riga2);
  presenta(2,10,71,riga1);
  presenta(23,10,71,mess5);
  fasce();
  presenta(10,12,23,vuoto);
  presenta(11,12,23,mess4);
  presenta(12,12,23,mess3);
  presenta(13,12,23,vuoto);
  risposta=ci();
  if(risposta==27)  return(27);
  unitm=open(nome,0);
  if(unitm==-1)
  {
    presenta(11,12,71,mess1);
    presenta(12,12,71,mess2);
    presenta(13,12,71,mess3);
    printf("%c",bell);
    risposta=ci();
    if(risposta==27)  return(27);
  }
}
status=fscanf(unitm,"%d %d %f %s\n",&nfile1,&nfile2,
              &nbytetot,&record);
status=close(unitm);
sprintf(record,"   IL DISCHETTO CONTIENE %2d FILE(s) ARCHIVIATI   ",
          nfile1);
presenta(7,12,23,vuoto);
presenta(8,12,23,record);
presenta(9,12,23,vuoto);
presenta(10,12,23,mess6);
presenta(11,12,23,vuoto);
presenta(12,12,23,mess7);
presenta(13,12,23,vuoto);
presenta(14,12,23,mess8);
presenta(15,12,23,vuoto);
while(((risposta=ci())!='1')&&(risposta!='2')&&(risposta!=27));
if(risposta=='1')   file="ANAGRAFE";
if(risposta=='2')   file="VALORICA";
if(risposta==27)    return(27);
scr_scrup(lin,fr,fc,tr,tc);
jrig=4;
for(indice=1; indice<=nfile1; indice++)
{
  sprintf(file_open,"A:%8s.%0*d",file,3,indice);
  unitm=open(file_open,0);
  if(unitm==-1)
  {
    sprintf(record,"   MANCA IL FILE NUMERO %d   ",indice);
    presenta(12,12,71,record);
    printf("%c",bell);
    presenta(14,12,71,mess3);
```

```
      risposta=ci();
    }
    else
    {
      if(risposta=='1')
      {
        cp=fgets(dummy,80,unitm);  /* inserisce anche CR e LF */
        leng=strlen(dummy);
        dummy[leng-1]=0;           /* elimina CR e LF       */
        sprintf(record," %0*d -- %s ",3,indice,dummy);
        presenta(indice+3,5,23,record);
      }
      else
      {
        sprintf(record,"  NUMERO DEL FILE IN CORSO : %d  ",indice);
        presenta(3,15,87,record);
        leng=255;
        while(leng!=0)
        {
          cp=fgets(dummy,80,unitm);
          if(*cp!=0)
          {
            leng=strlen(dummy);
            dummy[leng-1]=0;
            presenta(jrig,5,2,dummy);
            jrig++;
            if(jrig==23)
            {
              presenta(23,12,71,mess3);
              risp=ci();
              if(risp==27)  return(27);
              presenta(23,12,71,mess5);
              scr_scrup(lin,fr,fc,tr,tc);
              jrig=4;
            }
          }
        }
      }
    }
    status=close(unitm);
  }
  presenta(22,14,23,mess3);
  printf("%c",bell);
  risposta=ci();
}



mediana(array,ncelle,fmedi)
int array[];
int ncelle;
float *fmedi;
{
  int temp,j,k,dato1,dato2;
  for(j=0; j<ncelle-1; j++)
  {
    for(k=j+1; k<ncelle; k++)
    {
```

```
        if(array[k] < array[j])
        {
          temp=array[j];
          array[j]=array[k];
          array[k]=temp;
        }
      }
    }
    j=ncelle/2;
    *fmedi=array[j];
    if((ncelle%2)==0)        /* --- se il numero di punti e` pari --- */
    {
      dato1=array[j-1];
      dato2=array[j];
      *fmedi=(dato1+dato2)/2.0;
    }
  }

  carica_array()
  {
    identi[0]=id0;
    identi[1]=id1;
    identi[2]=id2;
    identi[3]=id3;
    identi[4]=id4;
    identi[5]=id5;
    identi[6]=id6;
    identi[7]=id7;
    identi[8]=id8;
    identi[9]=id9;
    identi[10]=id10;
    identi[11]=id11;
    idiscri[0]=ids0;
    idiscri[1]=ids1;
    idiscri[2]=ids2;
    idiscri[3]=ids3;
    idiscri[4]=ids4;
    idiscri[5]=ids5;
    idiscri[6]=ids6;
    idiscri[7]=ids7;
  }


/************************************************************************** EOF *****/

/********************************** File: GESFILE.C                  */


/*
  PROGRAMMA MULTISEN DI GESTIONE STIMOLATORE MULTISENSORIALE
  ----------------------------------------------------------

      PARTE GLOBALE DI DEFINIZIONE VARIABILI ESTERNE
*/
  int  macchina;          /* TIPO MACCHINA: 1 COMPLETA - 2 INCOMPLETA     */
  char arch_semplice[60]; /* PATH DEL FILE DI ARCHIVIO PER PROVE SEMPLICI */
  char arch_discrimi[60]; /* PATH DEL FILE DI ARCHIVIO PER PROVE DISCRIM. */
  char file_random[60];   /* PATH DEL FILE SEME RANDOM                    */
```

```
  char file_defvisivo[60]; /*FILE CONTENENTE I DEFAULT VISIVO          */
  char file_defuditiv[60]; /*FILE CONTENENTE I DEFAULT UDITIVO         */
  char file_defsomest[60]; /*FILE CONTENENTE I DEFAULT SOMESTESICO     */
  int  ftotali;          /*NUMERO FILE TOTALI DU DISCHETTO FORMATTATO  */

/* DETTE INFORMAZIONI VENGONO LETTE DAL FILE MULTISEN.INI CHE DEVE ESSERE
   RESIDENTE SUL DIRECTORY OVE RISIEDE IL PROGRAMMA STESSO.
   PER QUANTO RIGUARDA I FILES APERTI ESSI VENGONO RIFERITI RISPETTIVAMENTE
   CON GLI INTERI fp0,fp1,fp2,fp3,fp4,fp5,fp6                          */

  int fp0,fp1,fp2,fp3,fp4,fp5,fp6;

/*
   DEFINIZIONE ARRAY PARAMETRI DI DEFAULT PER I TRE STIMOLATORI
*/

  char  def_visivo[6];
  char  def_uditivo[19];
  char  def_somestesico[13];
  char  work_visivo[6];
  char  work_uditivo[19];
  char  work_somestesico[13];
  char  diver_toni[4];

/*
   definizione array di memorizzazione dati archivio
*/

  float archivio[2048];
  int   osservazioni[2048];
  int   freqpc;

gestisce_file()
{
  extern char per_status[];
  char *file_ini,*cp;
  char stringa[80],temporaneo[20];
  char *KEY1,*KEY2,*KEY3,*KEY4,*KEY5,*KEY6,*KEY7,*KEY8,*KEY9;
  char EQUA="=",SLAS="/";
  int  fp,num,result,lung,a,iseme,imacc;

  KEY1="MACCHINA";
  KEY2="ARC_SEMP";
  KEY3="ARC_DISC";
  KEY4="FILE_RAN";
  KEY5="VISI_DEF";
  KEY6="UDIT_DEF";
  KEY7="SOME_DEF";
  KEY8="FILE_FMT";
  KEY9="FREQUENZ";

  iseme=0;
  imacc=0;
  ftotali=18;
  freqpc=4700;                    /* frequenza di default      */
  file_ini="MULTISEN.INI";
  fp=open(file_ini,0);
  if (fp==-1)
```

```
    {
      messaggio(01);
      exit();
    }
    macchina=0;
  legge_stringa:
    cp=fgets(stringa,80,fp);
    if(cp==0)
    {
      num=close(fp);
      if(iseme==0)
      {
        messaggio(02);
        exit();
      }
      if((macchina!=1)&&(macchina!=2))
      {
        messaggio(03);
        exit();
      }
      if(macchina==1)
      {
        if(imacc!=3)
        {
          messaggio(04);
          exit();
        }
      }
      else
      {
        per_status[2]=-1;   /* definisce assente lo stimolatore some */
        if(imacc<2)
        {
          messaggio(05);
          exit();
        }
      }
      return(num);
    }
    if(stringa[0]==';') goto legge_stringa;
    stringa[strlen(stringa)-1]=0;

  /*
     ===== Cerca KEYWORD nel file di inizializzazione =====
  */

    if(search(stringa,KEY1)!=-1)
    {
      result=search(stringa,EQUA);
      if(result==-1)
        messaggio(06);
      else
      {
        cp=strcpy(&temporaneo,&stringa[result+1]);
        macchina=atoi(&temporaneo);
      }
      goto legge_stringa;
    }
```

```
if(search(stringa,KEY2)!=-1)
{
  result=search(stringa,EQUA);
  if(result==-1)
    messaggio(07);
  else
  {
    cp=strcpy(&arch_semplice,&stringa[result+1]);
    fp0=apri_file(arch_semplice,2);
  }
  goto legge_stringa;
}
if(search(stringa,KEY3)!=-1)
{
  result=search(stringa,EQUA);
  if(result==-1)
    messaggio(08);
  else
  {
    cp=strcpy(&arch_discrimi,&stringa[result+1]);
    fp1=apri_file(arch_discrimi,2);
  }
  goto legge_stringa;
}
if(search(stringa,KEY4)!=-1)
{
  result=search(stringa,EQUA);
  if(result==-1)
    messaggio(09);
  else
  {
    cp=strcpy(&file_random,&stringa[result+1]);
    fp2=apri_file(file_random,2);
    iseme=1;
  }
  goto legge_stringa;
}
if(search(stringa,KEY5)!=-1)
{
  result=search(stringa,EQUA);
  if(result==-1)
    messaggio(11);
  else
  {
    cp=strcpy(&file_defvisivo,&stringa[result+1]);
    fp4=apri_file(file_defvisivo,0);
    imacc+=1;
    legge_dati_default(fp4,def_visivo,work_visivo,6);
  }
  goto legge_stringa;
}
if(search(stringa,KEY6)!=-1)
{
  result=search(stringa,EQUA);
  if(result==-1)
    messaggio(12);
  else
  {
```

```
        cp=strcpy(&file_defuditiv,&stringa[result+1]);
        fp5=apri_file(file_defuditiv,0);
        imacc+=1;
        legge_dati_default(fp5,def_uditivo,work_uditivo,19);
        legge_dati_default(fp5,diver_toni,diver_toni,4);
      }
      goto legge_stringa;
    }
    if(search(stringa,KEY7)!=-1)
    {
      result=search(stringa,EQUA);
      if(result==-1)
        messaggio(13);
      else
      {
        cp=strcpy(&file_defsomest,&stringa[result+1]);
        fp6=apri_file(file_defsomest,0);
        imacc+=1;
        legge_dati_default(fp6,def_somestesico,work_somestesico,13);
      }
      goto legge_stringa;
    }
    if(search(stringa,KEY8)!=-1)
    {
      result=search(stringa,EQUA);
      if(result==-1)
        messaggio(17);
      else
      {
        cp=strcpy(&temporaneo,&stringa[result+1]);
        ftotali=atoi(&temporaneo);
      }
      goto legge_stringa;
    }
    if(search(stringa,KEY9)!=-1)
    {
      result=search(stringa,EQUA);
      if(result==-1)
        messaggio(18);
      else
      {
        cp=strcpy(&temporaneo,&stringa[result+1]);
        freqpc=atoi(&temporaneo);
      }
      goto legge_stringa;
    }
    goto legge_stringa;
}


search(stringa1,stringa2)
char stringa1[],stringa2[];
/*
  RICERCA LA STRINGA2 NELLA STRINGA1: RITORNA -1 NEL CASO NEGATIVO
  O L'INDICE RELATIVO AL PRIMO CARATTERE NEL CASO POSITIVO
*/
{
  char first,char1,char2;
```

```
  int len1,len2,a,b,iret;
  first=stringa2[0];
  len1=strlen(stringa1);
  len2=strlen(stringa2);
  iret=-1;

  for(a=0; a<len1; a++)
  {
    if(stringa1[a]==first)
    {
      iret=a;
      first=0xFF;
    }
  }

  if(iret==-1) return(iret);
  b=iret;
  iret++;
  for(a=1; a<len2; a++)
  {
    if(stringa2[a]!=stringa1[iret])
    {
      iret=-1;
      return(iret);
    }
    iret++;
  }
  return(b);
}

apri_file(file_name,modo)
char file_name[];
int  modo;
{
  int fp;
  fp=open(file_name,modo);
  if (fp==-1)
  {
    messaggio(14);
    printf(" %s",file_name);
    exit();
  }
  return(fp);
}

inizializza_ran()
{
  int seed,num;
  num=fscanf(fp2,"%d",&seed);
  if(num==-1)
  {
    messaggio(15);
    exit();
  }
  srand(seed);      /*  INIZIALIZZA SEME PER NUMERI CASUALI  */
}
```

```
chiude_file_ran()
{
  int numero,num;
  long lval;

  lval=fseek(fp2,0L,0);
  numero=rand();
  num=fprintf(fp2,"%d",numero);
  if(num==-1)
  {
    messaggio(16);
    exit();
  }
  num=close(fp2);
}


carica_casuali(minimo,massimo,array,ncelle)
int minimo,massimo,array[],ncelle;
/*
  CARICA L'ARRAY COSTITUITO DA ncelle LOCAZIONI CON ALTRETTANTI VALORI
  TEMPORALI (in millisecondi) UTILIZZANDO LA ROUTINE DI GENERAZIONE
  NUMERI CASUALI RAND()
*/
{
  float nuovomassimo,valore;
  int indice,ival;

  nuovomassimo=massimo-minimo;
  for(indice=0; indice<ncelle; indice++)
  {
    ival=rand();
    valore=ival*nuovomassimo/32768.+minimo;
    array[indice]=valore;
  }
}


legge_dati_default(unit,array1,array2,ndata)
int  unit,ndata;
char array1[],array2[];   /* array dati di default e array di lavoro */
{
  int indice,num;
  char dato;
  for (indice=0; indice<ndata; indice++)
  {
    num=fscanf(unit,"%x",&dato);
    array1[indice]=dato;  /* carica array dati di default def_   */
    array2[indice]=dato;  /* carica array di lavoro       work_  */
  }
}

messaggio(indice)
int indice;
{
  printf("\nGESFILE> ");
  switch (indice)
  {
```

```
    case (01): printf("IL FILE MULTISEN.INI NON ESISTE");
               break;
    case (02): printf("IMPOSSIBILE APRIRE FILE SEME RANDOM");
               break;
    case (03): printf("VALORE MACCHINA NON CORRETTO");
               break;
    case (04): printf("VALORI DI DEFAULT NON DEFINITI (1)");
               break;
    case (05): printf("VALORI DI DEFAULT NON DEFINITI (2)");
               break;
    case (06): printf("ERRORE DI SINTASSI CHIAVE MACCHINA");
               break;
    case (07): printf("ERRORE DI SINTASSI CHIAVE FILE_SEM");
               break;
    case (08): printf("ERRORE DI SINTASSI CHIAVE FILE_DIS");
               break;
    case (09): printf("ERRORE DI SINTASSI CHIAVE FILE_RAN");
               break;
    case (10): printf("ERRORE DI SINTASSI CHIAVE TEMPORANEO");
               break;
    case (11): printf("ERRORE DI SINTASSI CHIAVE DEFAULT VISIVO");
               break;
    case (12): printf("ERRORE DI SINTASSI CHIAVE DEFAULT UDITIVO");
               break;
    case (13): printf("ERRORE DI SINTASSI CHIAVE DEFAULT SOMESTESICO");
               break;
    case (14): printf("FILE NOT FOUND");
               break;
    case (15): printf("ERRORE SU LETTURA FILE SEME RANDOM");
               break;
    case (16): printf("ERRORE SU SCRITTURA FILE SEME RANDOM");
               break;
    case (17): printf("ERRORE DI SINTASSI CHIAVE FORMAT DISCHETTO");
               break;
    case (18): printf("ERRORE SUL VALORE DI FREQUENZA INTRODOTTO");
               break;
  }
}


carica_archivio(unit,archivio,osservazioni,ncelle)
int   unit,ncelle;
float archivio[];
int   osservazioni[];
{
  int num;
  long lval;
  lval=rewind(unit);
  num=fread(archivio,sizeof(*archivio),ncelle,unit);
  num=fread(osservazioni,sizeof(*osservazioni),ncelle,unit);
}



scrive_archivio(unit,archivio,osservazioni,ncelle)
int   unit,ncelle;
float archivio[];
int   osservazioni[];
```

```
{
  int num;
  long lval;
  lval=rewind(unit);
  num=fwrite(archivio,sizeof(*archivio),ncelle,unit);
  num=fwrite(osservazioni,sizeof(*osservazioni),ncelle,unit);
}

crea_indice(flag,sesso,eta,stimolatore,arto1,arto2)
/*
   con FLAG = 0 si intende l'archivio semplice
   con FLAG = 1 si intende l'archivio discriminazione
*/

char flag,sesso;
int  eta,stimolatore,arto1,arto2;
{
  int indice;
/* printf("\nDATI PASSATI: %c %d %d %d %d",sesso,eta,stimolatore,arto1,arto2); */
  if((eta>=18)&&(eta<=30))  eta=0;
  if((eta>30)&&(eta<=40))   eta=1;
  if((eta>40)&&(eta<=50))   eta=2;
  if((eta>50)&&(eta<=60))   eta=3;
  if(eta>60)                eta=4;
  if(flag==0)
    indice=arto1*64+stimolatore*16+eta*2;
  else
    indice=arto2*128+arto1*32+stimolatore*16+eta*2;
  if(sesso=='M'||sesso=='m') indice++;
  indice=indice*4;
/* printf("\nDATI CREATI: %c %d %d %d %d>%d",sesso,eta,stimolatore,
       arto1,arto2,indice);   ***/
  return(indice);
}

aggiorna_buffer(indice,archivio,osservazioni,x,y,w,z)
int   indice;
float archivio[];
int   osservazioni[];
float x,y,w,z;
{
  float temporaneo;
  int   itemporaneo;

  itemporaneo=osservazioni[indice];
  itemporaneo++;
  temporaneo=archivio[indice]*itemporaneo;
  temporaneo=temporaneo+x;
  temporaneo=temporaneo/itemporaneo;
  archivio[indice]=x;
  temporaneo=archivio[indice+1]*itemporaneo;
  temporaneo=temporaneo+y;
  temporaneo=temporaneo/itemporaneo;
  archivio[indice+1]=y;
  temporaneo=archivio[indice+2]*itemporaneo;
  temporaneo=temporaneo+w;
  temporaneo=temporaneo/itemporaneo;
  archivio[indice+2]=w;
```

```
  temporaneo=archivio[indice+3]*itemporaneo;
  temporaneo=temporaneo+z;
  temporaneo=temporaneo/itemporaneo;
  archivio[indice+3]=z;
  osservazioni[indice]=itemporaneo;
}

/************************************************************************ EOF *****/

/************************************ File: ITBM.C                    */


/*
  DEFINIZIONE INDIRIZZI ASSOLUTI UTILIZZATI DALLE PORTE PARALLELE
  E DAL TIMER PRESENTI SULLA SCHEDA HARDWARE
*/
  unsigned port=0x01B7;
  unsigned portim=0x01B3;
  unsigned timer=0x01BC;
/*
=============================================================================
=

   FUNCTION: send_addr(indirizzo)

   Funzione che invia l`indirizzo della periferica secondo il
   protocollo ITBM.  Necessita del parametro "indirizzo" relativo
   all'indirizzo dello strumento da interrogare.

=============================================================================
=
*/

send_addr(indirizzo)
char  indirizzo;
{
    int   val;
    char  out;
/* ------------------------------------------------------------------ */
/*   out=sense_bit(0x10,1,500);
    if (out==2)  printf ("\nERROR> Trovato COA a zero\n");          */

    _outb(0x07,port-1);       /* si pone DIR basso                */
    _outb(indirizzo,port-3);  /* invio ind. della per. sulla porta A */
    _outb(0x06,port-1);       /* si pone ADV basso                */
    out=sense_bit(0x10,0,500); /* si controlla l`arrivo del COA     */
    if(out==2)
    {
      _outb(0x07,port-1);   /* si pone ADV alto                 */
/*     printf("\nCOLLEGAMENTO INTERROTTO a");                    */
      out=2;
      return(out);
     }
    _outb(0x07,port-1);        /* si pone ADV alto                */
    _outb(0x0F,port-1);        /* si pone DIR alto                */
    out=sense_bit(0x10,1,500);  /* si controlla il ritorno alto di COA */
    if(out==2)
     {
```

```
/*      printf("\n COLLEGAMENTO INTERROTTO d");                  */
        out=2;
      }
     return(out);
}
/*
=============================================================================
=

   FUNCTION: send_data(byte)

   Funzione che invia il byte "byte" sul bus comune secondo
   protocollo ITBM.

=============================================================================
=
*/

send_data(byte)
char byte;

   char   out;
/* ------------------------------------------------------------ */
/*  out=sense_bit(0x10,1,500);
   if (out==2)  printf ("\nERROR> Trovato COA a zero\n");      */

   _outb(0x07,port-1);       /* si pone DIR basso            */
   _outb(byte,port-3);       /* invio dato sulla porta A     */
   _outb(0x03,port-1);       /* si pone DTV basso            */
   out=sense_bit(0x10,0,500); /* si controlla l'arrivo del COA */
   if(out==2)
   {
      _outb(0x07,port-1);   /* si pone DTV alto             */
/*      printf("\n COLLEGAMENTO INTERROTTO");                 */
       out=2;
       return(out);
   }
   _outb(0x07,port-1);       /* si pone DTV alto */
   _outb(0x0F,port-1);       /* si pone DIR alto */
   return(out);
}

/*
=============================================================================
=

   FUNCTION: reset(time_out)

   Esegue il reset di tutte le apparecchiature collegate al bus
   secondo protocollo ITBM. Il valore time_out viene utilizzato
   per introdurre un certo ritardo prima di rilasciare lo stato
   di reset.

=============================================================================
=
*/

reset(time_out)
```

```
int   time_out;
{
     int     a;
/* ------------------------------------------------------------------ */
     _outb(0x0A,port-1);
     for(a=0; a<=time_out; a++);      /*  CICLO DI ATTESA PER t.res */
     _outb(0x0F,port-1);
}
/*
==========================================================================================
=

   FUNCTION: ini_per()

   Inizializza la scheda contenente le porte parallele 8255 come:
   PORTA RELATIVA AL BUS ITBM
          porta A in OUT
          porta B in IN
          porta C bit 0,1,2,3 in OUT
                bit 4,5,6,7 in IN
          C0=ADV, C1=TROUT, C2=DTV, C3=DIR
          C4=COA, C5=TRIN
          parola di controllo 8A
   PORTA RELATIVA AI PULSANTI
          porta B in IN
          porta C in OUT

==========================================================================================
=
*/

ini_per()
{
     extern int freqpc;   /* frequenza di lavoro del PC in KHz */
     extern unsigned portim;  /* seconda porta                */
     extern unsigned timer;   /* timer/counter di conteggio   */
     char out,datalow,datahig;
     int  parola;

     parola=freqpc/4;
     datalow=parola&0x00FF;
     datahig=parola/256;
/* ------------------------------------------------------------ */
     _outb(0x8A,port);       /* Configurazione delle porte     */
     _outb(0x0F,port-1);     /* Tutte le linee interessate a 1 */
     _outb(0x82,portim);     /* Pulsantiera                    */
     _outb(0x00,portim-1);   /* Gate timer a zero-timer fermo  */
/* ------------------------------------------------------------ */
     _outb(0x36,timer-1);    /* contatore 0 modo 3             */
     _outb(0x74,timer-1);    /* contatore 1 modo 2             */
     _outb(datalow,timer-4); /* forma clock da 1 millisec.     */
     _outb(datahig,timer-4);
     _outb(0xFF,timer-3);    /* tempo in millisecondi          */
     _outb(0xFF,timer-3);
}


/*
```

```
=============================================================================
=

   FUNCTION: receive()

   Riceve l'indirizzo di ritorno dalla periferica selezionata

=============================================================================
=
*/

receive()
{
    extern  unsigned port;
    char    val,addr,out;
/* ------------------------------------------------------------- */

    out=sense_bit(0x10,0,500); /* Testa la linea COA per zero  */
/*   if(out==2) printf("\nNON E'ARRIVATO IL COA\n");          */
    val =_inb(port-2);       /* Prelevo valore dalla porta B */
    out=0;
    addr=val;
    return(addr);
}

/*
=============================================================================
=

   FUNCTION: sense_bit (ibit,direction,time_out)

   Controlla il bit della porta C indicato da ibit se resettato o
   settato a seconda del valore di direction. La condizione deve
   essere trovata nel periodo di tempo indicato da time_out.
   In caso di time_out spirato la funziona ritorna il valore 2 al
   programma chiamante.

=============================================================================
=
*/

sense_bit(ibit,direction,time_out)
char  ibit,direction;
int   time_out;

{
    char val,out;
    int  a,c;
/* ------------------------------------------------------------- */
    if(direction==0)
    {
      for(a=0; a<=time_out*2; a++)    /* Ciclo di attesa    */
      {
        val=_inb(port-1)&ibit;
        if(val==direction)        /* Controllo del bit  */
        {
          out=0;
          return(out);
```

```
        }
        else
          for(c=0; c<=1; c++);
      }
    }
    else
    {
    for(a=0; a<=time_out*2; a++)    /* Ciclo di attesa    */
      {
        val=_inb(port-1)&ibit;
        if(val!=0)              /* Controllo del bit  */
        {
          out=0;
          return(out);
        }
        else
          for(c=0; c<=1; c++);
      }
    }
    out=2;
    return(out);
}

/*
=============================================================================
=

   FUNCTION: trigger(to)

   Genera l'impulso di trigger per la strumentazione collegata
   con protocollo I.T.B.M. L'impulso ha durata "to" cicli.

=============================================================================
=
*/

trigger(to)           /* Genera l'impulso di trigger   */
    int to;
{
    int a,c;
    for(a=0; a<=to*3; a++)    /* ciclo di attesa*/
      for(c=0; c<=10; c++);
    _outb(0x05,port-1);     /* attivo TROUT */
    for(c=0; c<=20; c++);
    _outb(0x07,port-1);     /* disattivo TROUT */
}
/*
=============================================================================
=

   FUNCTION: send_data_some()

   Funzione di trasmissione dato allo stimolatore somestesico

=============================================================================
=
*/
send_data_some(byte)
```

```
char byte;
{
    extern  unsigned port;
    int out;
/* ------------------------------------------------------------ */

    _outb(0x07,port-1);       /* si pone DIR basso            */
    _outb(byte,port-3);       /* invio dato sulla porta A     */
    _outb(0x03,port-1);       /* si pone DTV basso            */
    out=ciclocoa();           /* aspetta la transizione del COA */
    _outb(0x0F,port-1);       /* si pone DIR alto             */
    return(out);
}
/*
=============================================================================
=

   FUNCTION: send_addr_some()

   Funzione di trasmissione indirizzo allo stimolatore somestesico

=============================================================================
=
*/
send_addr_some()
{
    char  indirizzo=0x90;
    int   val,out;
/* ------------------------------------------------------------ */
    _outb(0x07,port-1);       /* si pone DIR basso            */
    _outb(indirizzo,port-3);  /* invio ind. della per. sulla porta A */
    _outb(0x06,port-1);       /* si pone ADV basso            */
    out=ciclocoa();  ,        /* aspetta la transizione del COA */
    _outb(0x0F,port-1);       /* si pone DIR alto             */
    return(out);
}

/*
=============================================================================
=

   FUNCTION: receive_some()

   Riceve l'indirizzo di ritorno dallo stimolatore somestesico

=============================================================================
=
*/

receive_some()
{
    extern  unsigned port;
    char    val,out;
    int     a;
/* ------------------------------------------------------------ */

    for (a=0; a<10; a++);     /* Attende                      */
    val =_inb(port-2);        /* Prelevo valore dalla porta B */
```

```
    return(val);
}
/* ------------------------------------------------------------ */
/* ------------------------------------------------------------ */
ciclocoa()
{
  extern int errori,printer;
  int a,c;
  char val;

  for(a=0;a<=300;a++)
  {
    val=_inb(port-1)&0x10;
    if(val==0)                /* SEGNALE RICEVUTO*/
    {
    _outb(0x07,port-1);       /* si pone DTV o ADV alto */
      while((val==0))
        val=_inb(port-1)&0x10;
      return(0);
    }
    else
      for(c=0;c<=10;c++);
  }
/*  printf("ERRORE DI COLLEGAMENTO routine CICLOCOA-SOME");   */
  return(-1);
}
/*
============================================================================
=
============================================================================
=
*/

/************************************************************************ EOF *****/

/********************************** File: MULTISEN.C              */


/*-------------------------------------------------------------------

    PROGRAMMA DI GESTIONE DEL MODULO STIMOLATORE MULTISENSORIALE

  -------------------------------------------------------------------

   ------------------------------------------------------------
   ---        Descrizione array di utilizzo generale       -----
   ------------------------------------------------------------

   per_status[0] = byte di stato stimolatore visivo.
   per_status[1] = byte di stato stimolatore uditivo.
   per_status[2] = byte di stato stimolatore somestesico.
                Il byte di stato puo' avere i seguenti valori:
                [0] - stato di power-up:  nessuna operazione
                [1] - riconosciuto
                [2] - programmato
               [255] - errore durante il riconoscimento
               [254] - errore durante la programmazione.
```

```
                ----------------------------------------

     def_visivo = contiene i parametri di default dello stimolatore
                  visivo. (external GESFILE).
    def_uditivo = contiene i parametri di default dello stimolatore
                  uditivo. (external GESFILE).
def_somestesico = contiene i parametri di default dello stimolatore
                  somestesico. (external GESFILE).
    work_visivo = contiene i parametri da inviare allo stimolatore
                  visivo per la relativa programmazione. Utilizzato /
                  modificato dalle procedure tx_visivo()/edit_visivo().
                  (external GESFILE).
   work_uditivo = contiene i parametri da inviare allo stimolatore
                  uditivo per la relativa programmazione. Utilizzato /
                  modificato dalle procedure tx_uditivo()/edit_uditivo().
                  (external GESFILE).
work_somestesico= contiene i parametri da inviare allo stimolatore
                  somestesico per la relativa programmazione. Utilizzato /
                  modificato dalle procedure tx_someste()/edit_someste().
                  (external GESFILE).
     def_attr[] = contiene le indicazioni di stringa reverse per le
                  rappresentazioni su video dei valori di default.
        attr[] = contiene le indicazioni necessarie alla procedura
                  edit_visivo() per la visualizzazione dei valori
                  di default (modificati o no).

   ------------------------------------------------------------  */


   char    per_status[3]={0x00,0x00,0x00};

   char        attr[17]={0x0F,0x70,0x0F,0x70,0x0F,0x0F,0x0F,0x0F,0x70,
                     0x0F,0x0F,0x0F,0x0F,0x0F,0x0F,0x70,0x0F,0x0F};

   char     def_attr[17]={0x0F,0x70,0x0F,0x70,0x0F,0x0F,0x0F,0x0F,0x70,
                     0x0F,0x0F,0x0F,0x0F,0x0F,0x0F,0x70,0x0F,0x0F};

   char nuovo_paziente; /*** permette l'archiviazione dei dati
                        anagrafici e dei valori risultanti del
                        paziente di cui si sono introdotte le
                        informazioni anagrafiche            ***/
int  printer,printes,on_off;
char *printen;
int  numero_prog;

/* ================================================================== */
/* ===== PROGRAMMA PRINCIPALE ======================================= */
/* ================================================================== */

main(argc,argv)
int  argc;
char *argv[];
{
   extern char def_visivo[],work_visivo[];
   extern char def_uditivo[],work_uditivo[];
   extern char def_somestesico[],work_somestesico[];
   extern char scr_mode;
   extern char type_dis,type_sem,esito_dis,esito_sem;
```

```
   char param,cara,flag,risposta,flagg;
   int buffer[20];
float dummy1[12],dummy2[10];
int status;

   scr_setup();        /* setta le globals interne relative al tipo di */
   carica_array();     /* inizializza array di puntatori a stringhe    */
   flag=0x00;          /* adattore grafico che e' presente sul sistema */
   nuovo_paziente=0;
   type_sem=esito_sem=0;
   type_dis=esito_dis=0;
   numero_prog=0;
   flagg=0;
   on_off=0;
   if(argc>1)
     if ((*argv[1]=='p')||(*argv[1]=='P'))    on_off=1;

   scr_clr();
   gestisce_file();
   ini_per();
   reset(1000);
   inizializza_ran();
   printen="PRN:";
   printer=open(printen,1);
   while(!flag)
   {
primo_menu:
      risposta=test_hardware();
      if(risposta==21)  goto fine_multi;
      risposta=tipo_di_prova(&flagg);
      if(risposta==21)  goto fine_multi;
      if(risposta==27)  goto primo_menu;
   }
fine_multi:
  chiude_file_ran();
  exit();
}

fasce()
{
  videof(0);
  videof(1);
}
videof(flag)
int flag;
{
  int riga,colo;
  char barra[3]={0x20,0x20,0x00};

  if (flag)
    colo=1;
  else
    colo=77;
  for (riga=1; riga<24; riga++)
      presenta(riga,colo,23,barra);
}
```

```
presenta(ir,ic,ia,is)
  int ir,ic,ia;
  char is[];
{
  extern char scr_mode;
  scr_rowcol(ir,ic);
  if (scr_mode!=0x03)  /* testa se colori o monocromatico      */
  {                    /* se in monocromatico:                 */
    if (ia>0x0F)       /* controlla se e' settato il sottofondo */
      ia=0x70;         /* se si' setta il modo reverse (fondo   */
    else               /* bianco e carattere nero)             */
    {
      ia=0x07;         /* se no  setta il carattere bianco su   */
    }                  /* sfondo nero                          */
  }
  scr_aputs(is,ia);
}

/************************************************************************ EOF *****/

/********************************** File: SOMESTES.C                    */


/*-----------------------------------------------------------------

    MODULO DI GESTIONE STIMOLATORE SOMESTESICO: contiene tutte le
    routine dedicate al normale funzionamento e controllo dello
    stimolatore somestesico.

  -----------------------------------------------------------------
*/

struct disp_f {
char stringa13[62];
int posizione_riga13;
int posizione_colo13;
int attributo13;
};

struct disp_f screen_f[24]={
"            STIMOLATORE MULTISENSORIALE              ",1,8,23,
"                                                     ",2,8,71,
"   P R O G R A M M A Z I O N E  S O M E S T E S I C O  ",3,8,71,
"                                                     ",4,8,71,
" ARRAY = ",7,8,71,
"Funzionamento:",9,5,7,
"NUM.",12,44,2,
"PERIODO",12,50,2,
"DURATA",12,58,2,
"INTERV.",12,65,2,
"Stimolazione:",13,5,7,
"     ",13,51,23,
"     ",13,58,23,
"     ",14,51,23,
"     ",14,58,23,
"    ",15,44,23,
"     ",15,51,23,
"     ",15,58,23,
```

```
"     ",15,65,23,
"     ",16,51,23,
"     ",16,58,23,
"  USA LE FRECCE UP/DOWN PER SELEZIONARE IL MODO  ",20,10,71,
"        ALT-X CONVALIDA IL MODO EVIDENZIATO       ",21,10,71,
"       <ENTER> FINE CAMPO/FINE EDIT                ",22,10,71,
};


struct disp_m {
char strm[20];
int  rigm;
int  colm;
int  attm;
int  codm;
};

struct disp_m screen_m[07]={
" TENSIONE COSTANTE",9,22,7,0x00,
" CORRENTE COSTANTE",10,22,7,0x01,
" ALTERNATO        ",11,22,7,0x02,
" IMPULSI SINGOLI  ",13,22,7,0x03,
" COPPIE DI IMPULSI",14,22,7,0x04,
" TRENO  DI IMPULSI",15,22,7,0x05,
" COPPIA DI STIMOLI",16,22,7,0x06,
};


/*-----------------------------------------------------------------*/

    int     numero,durata,ripetizione,intervallo;
    int     a,i,b,d,e,f,j,c,to,tres;
    char    val,addr,out,at,br,ind,dato,check,masc,Cbit,ud,dati[20];

/*-----------------------------------------------------------------*/

somestesico(param)  /* Pacchetto software di gestione stimolatore somestesico
                       1 = riconoscimento stimolatore
                       2 = ricopia gli attributi standard in working
                       3 = non si sa' bene a cosa serva....
                       4 = permette di editare i parametri
                       5 = trasmette dati allo stimolatore somestesico.

  N.B. La cella dell'array per_status utilizzata per lo stimolatore
       somestesico e' la cella 2.                                  */

/* ---------------------------------------------------------------- */
int    param;
{
    extern char def_somestesico[];
    extern char work_somestesico[];
    extern char per_status[];
/* ---------------------------------------------------------------- */
    switch (param)
    {
      case '1':
           per_status[2]=1;
           iniz_somestesico();
```

```
            break;
        case '2':
            for(i=0;i<6;i++)  /* ricopia parametri di default */
              work_somestesico[i]=def_somestesico[i];
            break;
        case '3':
 /*          for(i=0;i<6;i++)
            {
              def_somestesico[i]=work_somestesico[i];
            }                                    */
            break;
        case '4':
            edit_somestesico();
            break;
        case '5':
            per_status[2]=2;
            tx_somestesico();
            break;
        default:
            break;
      }
 }

/* ================================================================== */
/* ======= PROCEDURA DI AGGIORNAMENTO PARAMETRI D'USO ============= */
/* ================================================================== */

 edit_somestesico()
 {
   extern char def_somestesico[];
   extern char work_somestesico[];
   extern char per_status[];
   int  a,ifunz,imodo,inpos[7]={0,0,0,11,13,15,19},index;
   char flag,ch,*messag,bell=7;
   char snum[5]={32,32,32,32,32},srip[5]={32,32,32,32,32};
   char sdur[5]={32,32,32,32,32},sint[5]={32,32,32,32,32};
   work_somestesico[0]=0x90;
   work_somestesico[3]=0x90;
   ifunz=0;
   imodo=3;
   messag=" --- Premi un tasto per continuare --- ";
   scr_clr();
   presenta_vettore(work_somestesico);
   for (a=0; a<24; a++)
     presenta(screen_f[a].posizione_riga13,screen_f[a].posizione_colo13,
           screen_f[a].attributo13,screen_f[a].stringa13);
   fasce();
   for (a=0; a<07; a++)
     presenta_normale(a);
   presenta_reverse(ifunz);
   presenta_reverse(imodo);
   numero=intervallo=0;
/*
------ seleziona tipo di funzionamento
*/
   presenta(screen_f[4].posizione_riga13,screen_f[4].posizione_colo13,
         71,screen_f[4].stringa13);
   flag=0x00;
```

```
  while(!flag)
  {
    while(((ch=ci())!=0)&&(ch!=27));  /*  27 = ESCAPE */
    switch (ch)
    {
      case 0:
        while(((ch=ci())!=72)&&(ch!=80)&&(ch!=45)&&(ch!=21));
        switch (ch)
        {
          case 72: presenta_normale(ifunz);
                   if(ifunz==0)
                     ifunz=2;
                   else
                     ifunz--;
                   presenta_reverse(ifunz);
                   break;
          case 80: presenta_normale(ifunz);
                   if(ifunz==2)
                     ifunz=0;
                   else
                     ifunz++;
                   presenta_reverse(ifunz);
                   break;
          case 45: flag=0xFF;
                   break;
          case 21: break;
        }
        break;
      case 27:  break;
    }
  }
  presenta(screen_f[4].posizione_riga13,screen_f[4].posizione_colo13,
        screen_f[4].attributo13,screen_f[4].stringa13);
/*
------ seleziona tipo di stimolazione
*/
  presenta(screen_f[10].posizione_riga13,screen_f[10].posizione_colo13,
        71,screen_f[10].stringa13);
  flag=0x00;
  while(!flag)
  {
    while(((ch=ci())!=0)&&(ch!=27));
    switch (ch)
    {
      case 0:
        while(((ch=ci())!=72)&&(ch!=80)&&(ch!=45)&&(ch!=21));
        switch (ch)
        {
          case 72: presenta_normale(imodo);
                   if(imodo==3)
                     imodo=6;
                   else
                     imodo--;
                   presenta_reverse(imodo);
                   break;
          case 80: presenta_normale(imodo);
                   if(imodo==6)
                     imodo=3;
```

```
                    else
                      imodo++;
                    presenta_reverse(imodo);
                    break;
              case 45: flag=0xFF;
                    break;
              case 21: break;
             }
            break;
        case 27:  break;
       }
     }
     presenta(screen_f[8].posizione_riga13,screen_f[8].posizione_colo13,
          screen_f[8].attributo13,screen_f[8].stringa13);
     index=inpos[imodo];
     if(imodo==5)
     {
       attende_number(screen_f[index].posizione_riga13,
                 screen_f[index].posizione_colo13,
                 screen_f[index].attributo13+128,2,&snum);
       index++;
     }
     attende_number(screen_f[index].posizione_riga13,
               screen_f[index].posizione_colo13,
               screen_f[index].attributo13+128,4,&srip);
     index++;
     attende_number(screen_f[index].posizione_riga13,
               screen_f[index].posizione_colo13,
               screen_f[index].attributo13+128,4,&sdur);
     if(imodo==5)
     {
       index++;
       attende_number(screen_f[index].posizione_riga13,
                 screen_f[index].posizione_colo13,
                 screen_f[index].attributo13+128,4,&sint);
     }
     numero=atoi(snum);
     ripetizione=(atoi(srip)/10)*10;
     ripetizione=ripetizione*2;
     durata=atoi(sdur);
     durata=durata/2.5-1;
     intervallo=atoi(sint);
     intervallo=intervallo/2.5-1;
     carica_vettore(ifunz,imodo,work_somestesico);
     presenta_vettore(work_somestesico);
     presenta(18,18,23,messag);
     printf("%c",bell);
     ch=ci();
}

/* ============================================================================ */

presenta_reverse(indice)
int  indice;
{
     presenta(screen_m[indice].rigm,screen_m[indice].colm,23,
          screen_m[indice].strm);
}
```

```
/* ========================================================================== */

presenta_normale(indice)
int   indice;
{
      presenta(screen_m[indice].rigm,screen_m[indice].colm,7,
            screen_m[indice].strm);
}

/* ========================================================================== */

carica_vettore(funzione,modo,vettore)
int  funzione,modo;
char vettore[];
{
   int check,ivett;
   char msb,lsb;
   check=0x90;
   vettore[1]=screen_m[funzione].codm;
   check=check+screen_m[funzione].codm;
   vettore[2]=check&255;
   check=0x90;
   vettore[4]=screen_m[modo].codm;
   check=check+screen_m[modo].codm;
   if(modo==5)
   {
     vettore[5]=numero;
     msb=intervallo/256;
     lsb=intervallo&255;
     vettore[10]=msb;
     vettore[11]=lsb;
     check=check+numero+msb+lsb;
     ivett=6;
   }
   else
     ivett=5;
   msb=ripetizione/256;
   lsb=ripetizione&255;
   vettore[ivett]=msb;
   vettore[ivett+1]=lsb;
   ivett+=2;
   check=check+msb+lsb;
   msb=durata/256;
   lsb=durata&255;
   vettore[ivett]=msb;
   vettore[ivett+1]=lsb;
   check=check+msb+lsb;
   if(modo==5)
     vettore[12]=check;
   else
     vettore[9]=check;
}
/* ========================================================================== */

presenta_vettore(vettore)
char vettore[];
{
  char record[10];
```

```
  int ncelle;
  int ic,ir,a;
  ir=7;
  ic=17;
  if(vettore[4]==5)
    ncelle=13;
  else
    ncelle=10;
  scr_rowcol(ir,ic);
  scr_clrl();
  for(a=0; a<ncelle; a++)
  {
    sprintf(record," %2x ",vettore[a]);
    presenta(ir,ic,71,record);
    ic+=4;
  }
  fasce();
}

/* ================================================================== */
/* ===== TRASMETTE I PARAMETRI ALLO STIMOLATORE ================== */
/* ================================================================== */

tx_somestesico()
{
  extern char def_somestesico[];
  extern char work_somestesico[];
  extern char per_status[];
  int out,fine;
/* ---------------------------------------------------------- */
/* --- Invia tipo di funzionamento ----- */
  out=send_addr_some();
  if(out!=0) per_status[2]=-2;
  trasmette_dati(work_somestesico,1,2);
  for(a=0; a<1000; a++);   /* attende prima di settare stimolo */
  out=send_addr_some();
  if(out!=0) per_status[2]=-2;
  if(work_somestesico[4]==5||work_somestesico[4]==0x0A)
    fine=12;
  else
    fine=9;
  trasmette_dati(work_somestesico,4,fine);
}
/*==========================================================================*/

trasmette_dati(vettore,start,end)
char vettore[];
int  start,end;
{
  extern char per_status[];
  int a;
  char dato,risultato;
  for(a=start; a<=end; a++)
  {
    dato=vettore[a];
    risultato=send_data_some(dato);
    if(risultato!=0) per_status[2]=-2;
  }
```

```
}


/* ================================================================== */
/* ===== ROUTINE DI RICONOSCIMENTO ================================== */
/* ================================================================== */


iniz_somestesico() /* Routine di riconoscimento dello stimolatore
                      somestesico tramite inizializzazione dello stesso */
{
   extern char per_status[];
   char out;

   per_status[2]=1;
   out=send_addr_some();
   if(out!=0)
     per_status[2]=-1;
   out=send_data_some(0x09);
   if(out!=0)
     per_status[2]=-1;
   out=receive_some();
   if(out!=0x90)
     per_status[2]=-1;
}
/* ================================================================== */

/*
  GESTISCE I CAMPI NUMERICI DELLA VIDEATA RELATIVA ALLA
  RIPROGRAMMAZIONE DEL SOMESTESICO
*/

attende_number(iriga,icolo,attrib,nchar_max,buffer)
int  iriga,icolo,nchar_max;
char attrib,*buffer;
{
  char cifra,flag,bell=7;
  char string2[2]={0X00,0X00};
  int indice,posizione,finest;
  posizione=icolo;
  indice=0;
  flag=0;
  attrib=attrib&0xF2;     /* sfondo come richiesto,caratteri verdi */
  scr_rowcol(iriga,icolo);
  while(!flag)
  {
    cifra=ci();
    switch (cifra)
    {
      case 8:  if(indice!=0)
               {
                 posizione--;
                 indice--;
                 scr_rowcol(iriga,posizione);
                 string2[0]=0X20;
                 presenta(iriga,posizione,attrib,string2);
                 scr_rowcol(iriga,posizione);
               }
```

```
            break;

        case 13: if(indice!=0)
                 {
                   buffer[indice]=0;
                   flag=0xFF;
                 }
                 else
                   printf("%c",bell);
                 break;
        default: if(isdigit(cifra))
                 {
                   string2[0]=cifra;
                   presenta(iriga,posizione,attrib,string2);
                   buffer[indice]=cifra;
                   indice++;
                   posizione++;
                   if(indice>nchar_max-1)
                   {
                     flag=0xFF;
                     buffer[indice]=0;
                   }
                   break;
                 }
     }
   }
   attrib=attrib&0x7F;  /* elimina il blink */
   attrib=attrib|0x07;  /* setta caratteri bianchi (sfondo come desiderato */
   presenta(iriga,icolo,attrib,buffer);
}

/* ===== END MODULO STIMOLATORE SOMESTESICO ========================== */

/************************************************************************ EOF *****/

/********************************** File: TIMER.C                */



start_timer()
{
  extern unsigned portim;
  unsigned tempo_letto;
  _outb(0x01,portim-1);
  legge_timer();
}

stop_timer()
{
  extern unsigned portim;
  _outb(0x00,portim-1);
}


load_timer(word)
unsigned  word;
{
  extern unsigned timer;
```

```
  char lsb,msb;
  msb=word/256;
  lsb=word&0xFF;
  _outb(lsb,timer-3);
  _outb(msb,timer-3);
  start_timer();
}


legge_timer()
{
  extern unsigned timer;
  char lsb,msb;
  unsigned word;

  _outb(0x40,timer-1);       /* richiede il latch del counter 0 */
  lsb=_inb(timer-3);
  msb=_inb(timer-3);
  word=msb*256+lsb;
  return(word);
}

/*************************************************************************/


trasmette_comandi(str,cod)
char str,cod;
{
  extern char per_status[];
extern printer;
  int prove,a,indice,status;
  char bell=7,flag;
  indice=0;
  flag=0;
  prove=5;
/***** status=fprintf(printer,"str = %c cod = %c\n",str,cod);   *****/
  switch (str)
  {
    case 'V': while((indice<prove)&&(flag==0))
            {
              visivo(cod);
/*** status=fprintf(printer,"(%d)\n",per_status[0]);   *****/
              if((per_status[0]==1)||(per_status[0]==2))
                flag=255;
/*** status=fprintf(printer,"I%dI\n",indice);         *****/
              indice++;
            }
           break;
    case 'U': while((indice<prove)&&(flag==0))
            {
              uditivo(cod);
/*** status=fprintf(printer,"(%d)\n",per_status[1]);   ***/
              if((per_status[1]==1)||(per_status[1]==2))
                flag=255;
/*** status=fprintf(printer,"I%dI\n",indice);   ***/
              indice++;
            }
           break;
```

```
      case 'S': while((indice<prove)&&(flag==0))
                {
                  somestesico(cod);
/*** status=fprintf(printer,"(%d)\n",per_status[2]);  ***/
                  if((per_status[2]==1)||(per_status[2]==2))
                    flag=255;
/*** status=fprintf(printer,"|%d\n",indice);  ***/
                  indice++;
                }
               break;
    }
    if(flag==0)
    {
/*    scr_clr();
      printf("\n%cERRORE SULLO STIMOLATORE %c - CODICE %c",bell,str,cod);
      exit();                                                    */
      printf("%c",bell);
    }
/***** status=fprintf(printer,"STATI> %d %d %d\n",
            per_status[0],per_status[1],per_status[2]);  *****/
}

/*
=============================================================================
*/


/************************************************************************ EOF *****/

/************************************ File: UDITIVO.C                  */


/*
  ---------------------------------------------------------------
      MODULO UDITIVO: contiene tutte le routines di gestione dello
      stimolatore uditivo.
  ---------------------------------------------------------------

  --- Definizione strutture parametri e stringhe per rappresentazione
  --- video.
*/

struct disp {
char stringa[74];
int posizione_riga;
int posizione_colo;
int attributo;
};

/*
 ----- Contiene tutti messaggi alfabetici da presentare, le relative
 ----- posizioni riga/colonna su video e l'attributo colore di ogni
 ----- stringa.
*/

struct disp screen[39]={
"                                          ",1,6,71,
"          PROGRAMMAZIONE STIMOLATORE UDITIVO              ",2,6,71,
```

```
"                                                    ",3,6,71,
"ATTUALE",5,21,32,
"ATTUALE",5,64,32,
" 1 ",6,1,32,
"Indice stimolo.",6,5,7,
" 8 ",6,39,32,
" Frequenza.....",6,42,7,
" 2 ",8,1,32,
"Stato masking..",8,5,7,
" 9 ",8,39,32,
"Sviluppo rise/fall..",8,43,7,
" 3 ",10,1,32,
"Indice output..",10,5,7,
" 10",10,39,32,
"Durata  rise/fall..",10,43,7,
" 4 ",12,1,32,
"Att. stimolo...",12,5,7,
" 11",12,39,32,
"Durata plateau Btono",12,43,7,
" 5 ",14,1,32,
"Att. masking...",14,5,7,
" 12",14,39,32,
"Durata plateau noise",14,43,7,
" 6 ",16,1,32,
" Polarita`.....",16,4,7,
" 7 ",18,1,32,
"Durata click...",18,5,7,
" 13",16,39,32,
"Numero stimoli.....",16,43,7,
" 14",18,39,32,
"Ritardo stimolo.....",18,43,7,
"-------------------------------------------------------------------------",20,3,23,
"  UP  ^ Seleziona parametro precedente  <ESC> Predispone variazione  ",21,3,23,
"  DOWN V  Seleziona parametro successivo                              ",22,3,23,
"  LEFT <  Seleziona nuovo default        <ENTER> FINE CAMPO/FINE EDIT ",23,3,23
};


/*
-- All'accensione vengono presentati i campi workd e pard: i primi relativi
   ai parametri attuali e i secondi relkativi ai parametri possibili tra i
   quali scegliere quelli di lavoro. Il comando 2 (settaggio parametri di
   default) ricopia defd in workd e pard defpar in pard.
*/


struct dispd {
char defd[10];
char pard[10];
char defpar[10];
char workd[10];
};

struct dispd videod[14]={
" CLICK "," CLICK "," CLICK "," CLICK ",
"  ON   "," OFF  "," OFF  "," ON   ",
"Rph/Lph","Rph/Lph","Rph/Lph","Rph/Lph",
"  64   "," 60   "," 60   "," 64   ",
```

```
" 24  "," 20  "," 20  "," 24  ",
" POSIT "," POSIT "," POSIT "," POSIT ",
" 100uS "," 100uS "," 100uS "," 100uS ",
"  250 ","  250 ","  250 ","  250 ",
"LINEARE","LINEARE","LINEARE","LINEARE",
"  1mS ","  1mS ","  1mS ","  1mS ",
"  1mS ","  1mS ","  1mS ","  1mS ",
"  1mS ","  1mS ","  1mS ","  1mS ",
"   0 ","   0 ","   0 ","   0 ",
"  1mS ","  1mS ","  1mS ","  1mS "
};



struct dispv {
char string[10];
int posr;
int posc;
char cod;
int appart;
};

/*
-- Videox contiene le nomenclature di tutti i parametri possibili
*/


struct dispv videox[83]={
" CLICK ",6,21,0x00,1,
" LOGON ",6,21,0x01,1,
" BTONO ",6,21,0x02,1,
" BNOISE",6,21,0x03,1,
/* ------------------------------------ */
"  OFF  ",8,21,0x00,2,
"  ON   ",8,21,0x01,2,
/* ------------------------------------ */
"Rph/Lph",10,21,0x00,3,
"Lph/Rph",10,21,0x01,3,
"Rph/Rph",10,21,0x02,3,
"Lph/Lph",10,21,0x03,3,
"RLp/off",10,21,0x04,3,
"off/RLp",10,21,0x05,3,
"Rff/Lff",10,21,0x06,3,
"Lff/Rff",10,21,0x07,3,
/* ------------------------------------ */
"  60   ",12,21,60,4,
"  64   ",12,21,64,4,
"  68   ",12,21,68,4,
"  72   ",12,21,72,4,
"  76   ",12,21,76,4,
"  80   ",12,21,80,4,
"  84   ",12,21,84,4,
"  88   ",12,21,88,4,
/* ------------------------------------ */
"  20   ",14,21,20,5,
"  24   ",14,21,24,5,
"  28   ",14,21,28,5,
"  32   ",14,21,32,5,
```

```
" 36  ",14,21,36,5,
" 40  ",14,21,40,5,
" 44  ",14,21,44,5,
" 48  ",14,21,48,5,
/* ------------------------------------ */
" POSIT ",16,21,0x00,6,
" NEGAT ",16,21,0x01,6,
"ALTERN ",16,21,0x02,6,
/* ------------------------------------ */
" 100uS ",18,21,0x00,7,
" 200uS ",18,21,0x01,7,
" 500uS ",18,21,0x02,7,
"1000uS ",18,21,0x03,7,
/* ------------------------------------ */
"  250 ",6,64,0x00,8,
"  500 ",6,64,0x01,8,
" 1000 ",6,64,0x02,8,
" 2000 ",6,64,0x03,8,
" 3000 ",6,64,0x04,8,
" 4000 ",6,64,0x05,8,
" 6000 ",6,64,0x06,8,
" 8000 ",6,64,0x07,8,
/* ------------------------------------ */
"LINEARE",8,64,0x00,9,
"COSENIC",8,64,0x01,9,
/* ------------------------------------ */
"  1mS ",10,64,0x00,10,
"  2mS ",10,64,0x01,10,
"  3mS ",10,64,0x02,10,
"  4mS ",10,64,0x03,10,
"  5mS ",10,64,0x04,10,
"  6mS ",10,64,0x05,10,
"  7mS ",10,64,0x06,10,
"  8mS ",10,64,0x07,10,
"  9mS ",10,64,0x08,10,
" 10mS ",10,64,0x09,10,
" 20mS ",10,64,0x0A,10,
" 30mS ",10,64,0x0B,10,
" 40mS ",10,64,0x0C,10,
" 50mS ",10,64,0x0D,10,
" 60mS ",10,64,0x0E,10,
" 70mS ",10,64,0x0F,10,
" 80mS ",10,64,0x10,10,
" 90mS ",10,64,0x11,10,
" 100mS ",10,64,0x12,10,
" 200mS ",10,64,0x13,10,
" 300mS ",10,64,0x14,10,
" 400mS ",10,64,0x15,10,
" 500mS ",10,64,0x16,10,
" 600mS ",10,64,0x17,10,
" 700mS ",10,64,0x18,10,
" 800mS ",10,64,0x19,10,
" 900mS ",10,64,0x1A,10,
"1000mS ",10,64,0x1B,10,
/* ------------------------------------ */
"   0 ",16,72,0,11,
"   2 ",16,72,2,11,
"   4 ",16,72,4,11,
```

```
  "   6 ",16,72,6,11,
  "   8 ",16,72,8,11,
  "  10 ",16,72,10,11,
  "  12 ",16,72,12,11,
  "  14 ",16,72,14,11
  };


  /*
  ----- DESCRIZIONE ARRAY INTERNI ----------------------------------

    limiti  = contiene per ognuno dei quattordici campi da presentare
              il limite inferiore ed il limite superiore tra i quali
              puo` essere variato il parametro relativo al campo
              in corso. Detta coppia di dati e' riferita all'indice
              numerico della struttura videox.

    parametro = contiene per ogni campo l'indice riferito alla struttura
                videox di rappresentazione stringa. Detti valori sono
                aggiornati dalla funzione aggiorna.

    defparame = contiene i valori da ricare in parametro ogni qualvolta
                si riselezionano i parametri di default.

    prig - pcol = contengono le coordinate riga/colonna relative alla
                  rappresentazione su video dei parametri.

    prig - pcol = contengono le coordinate riga/colonna relative alla
                  rappresentazione su video dei valori attuali di lavoro.

    corrisponde = contiene per ogni campo la corrispondenza tra l'array
                  parametro e l'array work_uditivo/def_uditivo (vedi:
                  alcune informazioni da trasmettere allo stimolatore
                  sono costituite da due byte).

    ------------------------------------------------------------------
  */

    int  voce;
    int  corrisponde[14]={2,3,4,5,7,9,10,11,12,13,14,15,16,18};
    int  prig[14]={6,8,10,12,14,16,18,6,8,10,12,14,16,18};
    int  dcol[14]={21,21,21,21,21,21,21,64,64,64,64,64,64,64};
    int  pcol[14]={29,29,29,29,29,29,29,72,72,72,72,72,72,72};
    int  limiti[28]={0,3,4,5,6,13,14,21,22,29,30,32,33,36,37,44,
                     45,46,47,74,47,74,47,74,75,83,47,74};
    int  defparame[14]={0,4,6,14,22,30,33,37,45,47,47,47,75,47};
    int  parametro[14]={0,4,6,14,22,30,33,37,45,47,47,47,75,47};

  /*
   ---------------------------------------------------------------
   --- Entry point globale modulo uditivo ------------------------
   ---------------------------------------------------------------
  */


  uditivo(param)
    int param;
  {
```

```
  int k,j;
  extern char per_status[];
  extern char work_uditivo[];
  extern char def_uditivo[];
  switch (param)
  {
    case '1':
          per_status[1]=1;
          iniz_udi();
          break;
    case '2':
          for (k=0; k<14; k++)
          {
            for (j=0; j<10; j++)
              videod[k].workd[j] = videod[k].defd[j];
            for (j=0; j<10; j++)
              videod[k].pard[j] = videod[k].defpar[j];
            work_uditivo[k] = def_uditivo[k];
            parametro[k]=defparame[k];
          }
          break;
    case '3':
          break;
    case '4':
          edit_uditivo();
          break;
    case '5':
          per_status[1]=2;
          tx_uditivo();
          break;
    default:
          break;
  }
}



edit_uditivo()
{
   extern def_uditivo[];
   extern work_uditivo[];
   int  a,b,index;
   char ch,flag;
/* ---------------------------------------------------------------- */
   scr_clr();
   for (a=0; a<39; a++)
     scrivix(&screen[a]);
   scrivi_attuali();
   scrivi_param();
   index=0;
   voce=0;
   intensifica();
   flag=0x00;
   while(!flag)
   {
     while(((ch=ci())!=0)&&(ch!=13)&&(ch!=27));
     switch (ch)
```

```
        {
          case 0:
                while(((ch=ci())!=80)&&(ch!=72)&&(ch!=75)&&(ch!=77));
                  switch (ch)
                  {
                    case 72:
                          if(voce!=0)
                          {
                            normal();
                            voce--;
                            intensifica();
                          }
                          break;
                     case 75:
                          if(voce>=7)
                          {
                             normal();
                             voce -= 7;
                             intensifica();
                          }
                          break;
                     case 77:
                          if(voce<=6)
                          {
                             normal();
                             voce += 7;
                             intensifica();
                          }
                          break;
                    case 80:
                          if(voce!=13)
                          {
                            normal();
                            voce++;
                            intensifica();
                          }
                          break;
                  }
                break;
        case 27:
                blinka();
                varia_parametro();
                break;
        case 13:
                flag=0xFF;
/*************   scr_clr(); ****************/
                break;
      }
    }
  }
```

```
/* ================================================================== */
/* ===== TRASMETTE I PARAMETRI ALLO STIMOLATORE ================== */
/* ================================================================== */

tx_uditivo()
```

```
{
    char ind,out,check,dato;
    int j;

    extern char def_uditivo[];
    extern char work_uditivo[];
    extern char per_status[];
/* ------------------------------------------------------------------ */
/*----- Inizio trasmissione vera e propria -----*/

    ind=work_uditivo[0];
    out=send_addr(ind);
    if(out==2)              /* non si e` ricevuto il COA*/
       per_status[1]=-2;
    check=work_uditivo[0];
    for(j=1;j<19;j++)
    {
      out=send_data(work_uditivo[j]);
/*     printf("\n%d --- %x",j,work_uditivo[j]);              */
      check=check+work_uditivo[j];
      if(out==2)              /*non si e' ricevuto il COA */
         per_status[1]=-2;
    }

    dato=~check+1;
    out=send_data(dato);
    if(out==2)              /*non si e` ricevuto il COA del cs*/
       per_status[1]=-2;

/**** PER UN ERRORE NEL FIRMWARE DELLO STIMOLATORE IL PRIMO TRIGGER
    NON VIENE RICONSCIUTO. IL TRIGGER POSTO SUBITO DI SEGUITO
    RIALLINEA LA SEQUENZA DEGLI STIMOLI ****************************/
    trigger(100);

}




varia_parametro()
{
  char ch,flag,temporanea[10];
  int liminf,limsup,indice,j;

  flag=0;
  while (!flag)
  {
    while(((ch=ci())!=0)&&(ch!=27));
    switch (ch)
    {
      case 0:
         while(((ch=ci())!=80)&&(ch!=72));
         liminf=limiti[voce*2];
         limsup=limiti[voce*2+1];
         switch (ch)
         {
```

```
            case 72:
                if (parametro[voce]!=limsup)
                {
                  parametro[voce]=parametro[voce]+1;
                  indice=parametro[voce];
                  rc_put(prig[voce],pcol[voce],
                       videox[indice].string,0xCF);
                }
                break;
            case 80:
                if (parametro[voce]!=liminf)
                {
                  parametro[voce]=parametro[voce]-1;
                  indice=parametro[voce];
                  rc_put(prig[voce],pcol[voce],
                       videox[indice].string,0xCF);
                }
                break;
          }
          break;
        case 27:
          flag=0xFF;
          indice=parametro[voce];
          for (j=0; j<10; j++)
          {
            videod[voce].workd[j] = videox[indice].string[j];
            videod[voce].pard[j] = videox[indice].string[j];
          }
          rc_put(prig[voce],dcol[voce],
               videod[voce].workd,0x70);
          intensifica();
          aggiorna();
          break;
      }
    }
}



intensifica()
{
  rc_put(prig[voce],pcol[voce],videod[voce].pard,71);
}


normal()
{
  rc_put(prig[voce],pcol[voce],videod[voce].pard,0x07);
}


blinka()
{
  rc_put(prig[voce],pcol[voce],videod[voce].pard,0xCF);
}

rc_put(irig,icol,istri,iattr)
```

```
   int  irig,icol;
   char istri[],iattr;
{
   scr_rowcol(irig,icol);
   scr_aputs(istri,iattr);
}


scrivi_attuali()
{
   int  a;
   for (a=0; a<14; a++)
     rc_put(prig[a],dcol[a],videod[a].workd,0x70);
}


aggiorna()
{
  switch (voce)
  {
    case  3:
    case  4:
          aggiorna_val(0);
          break;
    case 12:
          aggiorna_val(1);
          break;
    default:
          aggiorna_val(2);
          break;
  }
}



aggiorna_val(para)
  int para;
{
  extern char work_uditivo[];
  int valore,indice,index;

  indice=parametro[voce];
  valore=videox[indice].cod;
  index=corrisponde[voce];
    work_uditivo[index]=valore;
    switch (para)
    {
      case 0:
           work_uditivo[index+1]=~valore;
           break;
      case 1:
           work_uditivo[index+1]=0;
           break;
    }
}
```

```
scrivi_param()
{
   int  a,indice,j;

   for (a=0; a<14; a++)
   {
     indice=parametro[a];
     for (j=0; j<10; j++)
       videod[a].pard[j]=videox[indice].string[j];
     rc_put(prig[a],pcol[a],videod[a].pard,0x07);
   }
}


scrivix(par)
struct disp *par;
{
  scr_rowcol(par->posizione_riga,par->posizione_colo);
  scr_aputs(par->stringa,par->attributo);
}



/* ================================================================== */
/* ===== ROUTINE DI RICONOSCIMENTO ================================== */
/* ================================================================== */


iniz_udi()
{
   extern char per_status[];
   char ind,out;

   ind=0x70;
   out=send_addr(ind);
   if(out==2)              /* non si e` ricevuto il COA */
      per_status[1]=-1;
   out=send_data(0x01);
   if(out==2)              /* non si e` ricevuto il COA */
      per_status[1]=-1;
   out=receive();
   if(out==2)              /* non si e` ricevuto il COA */
      per_status[1]=-1;
/*  if(ind==out)
     printf("---------- CORRETTO");
   else                                          */
   if(ind!=out)
   {
/*    printf("INDIRIZZO DI RITORNO NON CORRETTO");          */
     per_status[1]=-1;
   }
}


/************************************************************************ EOF *****/

/********************************** File: VALUTA.C                */
```

```
    char *val_sem [12] = {
      "                                                    ",
      "                                                    ",
      "                                                    ",
      "                                                    ",
      "                                                    ",
      "                                                    ",
      "                                                    ",
      "                                                    ",
      "                                                    ",
      "                                                    ",
      "                                                    ",
      "                                                    ",
      "                                                    ",
      "                                                    "
    };


    char *val_dis [8] = {
      "                                                    ",
      "                                                    ",
      "                                                    ",
      "                                                    ",
      "                                                    ",
      "                                                    ",
      "                                                    ",
      "                                                    "
    };

    char esito_sem;
    char esito_dis;
    char type_sem;
    char type_dis;

valutazione(opt)
int opt;
{
    char *mess0,*mess1,record[80],risposta;
    mess0="                                                    ";
    mess1="                V A L U T A Z I O N E                ";
    if(opt==0)
      valuta_semplici();
    else
      valuta_discrimi();
    scr_clr();
    presenta(1,10,71,mess0);
    presenta(2,10,71,mess1);
    presenta(3,10,71,mess0);
    fasce();
    sprintf(record," Desideri la stampa della scheda di valutazione (S/N)  ");
    presenta(12,15,23,record);
    risposta=ci();
    if((risposta=='s')||(risposta=='S'))
      prinvaluta();
}
```

```
valuta_semplici()
{
  char *vdis1,*vdis2,*vdis3,*vdis4,*vdis5,*vdis6;
  char *vdis7,*vdis8,*vdis9,*vdis10,*vdis11,*vdis12;
  char *vdisa[12];
  extern float archivio[];
  extern int osservazioni[];
  extern float media[],scarto[];
  extern char cognome[],nome[],sesso,street[];
  extern int anni;
  extern int fp0;

  char *mess0,*mess3,*mess4,*mess5,*mess6;
  char *file_inp,record[80];
  char risposta,key,data_odierna[10];
  int len,npold,stat,index;
  int stim[12]={0,0,0,0,1,1,1,1,2,2,2,2};
  int arti[12]={0,1,2,3,0,1,2,3,0,1,2,3};
  float fmed[12],fdev[12],val,dev;
  int ioss[12],ipossi;
  int a,iriga,icolo,nerrate;
  vdis1=" VIS MD";
  vdis2=" VIS MS";
  vdis3=" VIS PD";
  vdis4=" VIS PS";
  vdis5=" UDI MD";
  vdis6=" UDI MS";
  vdis7=" UDI PD";
  vdis8=" UDI PS";
  vdis9=" SOM PS";
  vdis10=" SOM PS";
  vdis11=" SOM PS";
  vdis12=" SOM PS";
  vdisa[0]=vdis1;
  vdisa[1]=vdis2;
  vdisa[2]=vdis3;
  vdisa[3]=vdis4;
  vdisa[4]=vdis5;
  vdisa[5]=vdis6;
  vdisa[6]=vdis7;
  vdisa[7]=vdis8;
  vdisa[8]=vdis9;
  vdisa[9]=vdis10;
  vdisa[10]=vdis11;
  vdisa[11]=vdis12;
  mess0="                                                    ";
  mess3="  Dati da tastiera o da disco ? (T/D): ";
  mess5="  Introduci valore medio e deviazione standard:  ";
  mess6="  Introduci nome del file : ";
  data_int(data_odierna);
  screen_val_sem();
  len=strlen(cognome)+1;
  presenta(5,8,23,mess0);
  presenta(5,8,23,cognome);
  presenta(5,8+len,23,nome);
  len=len+strlen(nome)+1;
  presenta(5,8+len,23,data_odierna);
  presenta(12,12,23,mess3);
```

```
while(((key=ci())!='T')&&(key!='t')&&(key!='D')&&(key!='d'));
if((key=='T')||(key=='t'))
{
  type_sem='T';
  dati_tastiera(fmed,fdev,ioss,'S');
  screen_val_sem();
}
else
{
  type_sem='D';
  carica_archivio(fp0,archivio,osservazioni,1024);
  for(a=0; a<12; a++)
  {
    index=crea_indice(0,sesso,anni,stim[a],arti[a],arti[a]);
    fmed[a]=archivio[index];
    fdev[a]=archivio[index+1];
    ioss[a]=osservazioni[index];
  }
}
iriga=7;
icolo=9;
nerrate=0;
ipossi=0;
for(a=0; a<12; a++)
  if(street[a]!=0x55)
    ipossi++;
if(ipossi==12) nerrate=1;
for(a=0; a<12; a++)
{
  if(street[a]==0x55)
  {
    val=media[a];
    dev=fmed[a]+fdev[a]*2.0;       /* valore massimo */
    if(val < dev)
    {
      sprintf(record,
        "%2d) %6.1f %6.1f %6.1f %6.1f - OK - %4d %s",
        iriga-6,media[a],scarto[a],fmed[a],fdev[a],ioss[a],vdisa[a]);
      sprintf(val_sem [iriga-7],
        "%2d) %6.1f %6.1f %6.1f %6.1f - OK - %4d %s\n",
        iriga-6,media[a],scarto[a],fmed[a],fdev[a],ioss[a],vdisa[a]);
      presenta(iriga,icolo,23,record);
    }
    else
    {
      sprintf(record,
        "%2d) %6.1f %6.1f %6.1f %6.1f - NO - %4d %s",
        iriga-6,media[a],scarto[a],fmed[a],fdev[a],ioss[a],vdisa[a]);
      sprintf(val_sem [iriga-7],
        "%2d) %6.1f %6.1f %6.1f %6.1f - NO - %4d %s\n",
        iriga-6,media[a],scarto[a],fmed[a],fdev[a],ioss[a],vdisa[a]);
      presenta(iriga,icolo,71,record);
      nerrate++;
    }
  }
  else
  {
   sprintf(record,
```

```
      "%2d) ------  ------  ------  ------  ------ ---- %s",
      iriga-6,vdisa[a]);
    sprintf(val_sem [iriga-7],
      "%2d) ------  ------  ------  ------  ------ ---- %s\n",
      iriga-6,vdisa[a]);
    presenta(iriga,icolo,23,record);
    }
   iriga++;
  }
  if(nerrate==0)
  {
    sprintf(record,"     Esito delle prove: POSITIVO     ");
    presenta(20,20,32,record);
    esito_sem='P';
  }
  else
  {
    sprintf(record,"     Esito delle prove: NEGATIVO     ");
    presenta(20,20,71,record);
    esito_sem='N';
  }
  sprintf(record,"    BATTI UN TASTO PER PROSEGUIRE    ");
  presenta(22,20,71,record);
  risposta=ci();
}


valuta_discrimi()
{
  extern float archivio[];
  extern int osservazioni[];
  extern float dimedia[],discarto[];
  extern char cognome[],nome[],sesso,street[];
  extern int anni,dierrate[];
  extern int fp1;
  extern char possibilita[];
  extern int possibili[],gia_effettuate;

  char *mess0,*mess3,*mess4,*mess5,*mess6;
  char *vdis1,*vdis2,*vdis3,*vdis4,*vdis5,*vdis6,*vdis7,*vdis8,*vdisa[8];
  char record[80],risposta,key,data_odierna[10];
  int len,npold,stat,index;
  int stim[8]={0,0,0,0,1,1,1,1};
  int art1[8]={0,1,2,3,0,1,2,3};
  int art2[8]={1,0,3,2,1,0,3,2};
  float fmed[8],fdev[8],val,dev;
  int ioss[8];
  int a,iriga,icolo,nerrate,prove,dato1,dato2,idd;
  vdis1=" VIS MD MS";
  vdis2=" VIS MS MD";
  vdis3=" VIS PD PS";
  vdis4=" VIS PS PD";
  vdis5=" UDI MD MS";
  vdis6=" UDI MS MD";
  vdis7=" UDI PD PS";
  vdis8=" UDI PS PD";
  vdisa[0]=vdis1;
  vdisa[1]=vdis2;
```

```
vdisa[2]=vdis3;
vdisa[3]=vdis4;
vdisa[4]=vdis5;
vdisa[5]=vdis6;
vdisa[6]=vdis7;
vdisa[7]=vdis8;
mess0="                                                    ";
mess3="  Dati da tastiera o da disco ? (T/D): ";
mess5="  Introduci valore medio e deviazione standard:  ";
data_int(data_odierna);
screen_val_dis();
len=strlen(cognome)+1;
presenta(5,8,23,mess0);
presenta(5,8,23,cognome);
presenta(5,8+len,23,nome);
len=len+strlen(nome)+1;
presenta(5,8+len,23,data_odierna);
presenta(12,12,23,mess3);
while(((key=ci())!='T')&&(key!='t')&&(key!='D')&&(key!='d'));
if((key=='T')||(key=='t'))
{
  type_dis='T';
  dati_tastiera(fmed,fdev,ioss,'D');
  screen_val_dis();
}
else
{
  type_dis='D';
  carica_archivio(fp1,archivio,osservazioni,2048);
  for(a=0; a<8; a++)
  {
    index=crea_indice(1,sesso,anni,stim[a],art1[a],art2[a]);
    fmed[a]=archivio[index];
    fdev[a]=archivio[index+1];
    ioss[a]=osservazioni[index];
  }
}
iriga=9;
icolo=10;
if(gia_effettuate>0)
  nerrate=0;
else
  nerrate=1;
for(dato2=0; dato2<8; dato2++)
{
  for(prove=0; prove<gia_effettuate; prove++)
  {
    sprintf(record,
      "%2d) ------  ------  ------  ------ -- ------ ---- %s",
      dato2+1,vdisa[dato2]);
    sprintf(val_dis [iriga-9],
      "%2d) ------  ------  ------  ------ -- ------ ---- %s\n",
      dato2+1,vdisa[dato2]);
    presenta(iriga,icolo,23,record);
    dato1=possibilita[prove];
    idd=possibili[dato1];
    if(idd==possibili[dato2])
    {
```

```
            val=dimedia[dato1];
            dev=fmed[dato1]+fdev[dato1]*1.5;       /* valore massimo */
            if((val < dev)&&(dierrate[dato1] <= 8))
            {
              sprintf(record,
              "%2d) %6.1f  %6.1f  %6.1f  %6.1f %2d - OK - %4d %s",
              dato2+1,dimedia[dato1],discarto[dato1],fmed[dato1],fdev[dato1],
              dierrate[dato1],ioss[dato1],vdisa[dato2]);
              sprintf(val_dis [iriga-9],
              "%2d) %6.1f  %6.1f  %6.1f  %6.1f %2d - OK - %4d %s\n",
              dato2+1,dimedia[dato1],discarto[dato1],fmed[dato1],fdev[dato1],
              dierrate[dato1],ioss[dato1],vdisa[dato2]);
              presenta(iriga,icolo,23,record);
            }
            else
            {
              sprintf(record,
              "%2d) %6.1f  %6.1f  %6.1f  %6.1f %2d - NO - %4d %s",
              dato2+1,dimedia[dato1],discarto[dato1],fmed[dato1],fdev[dato1],
              dierrate[dato1],ioss[dato1],vdisa[dato2]);
              sprintf(val_dis [iriga-9],
              "%2d) %6.1f  %6.1f  %6.1f  %6.1f %2d - NO - %4d %s\n",
              dato2+1,dimedia[dato1],discarto[dato1],fmed[dato1],fdev[dato1],
              dierrate[dato1],ioss[dato1],vdisa[dato2]);
              presenta(iriga,icolo,71,record);
              nerrate++;
            }
            prove=gia_effettuate+1;      /* ----- by-pass for loop ----- */
          }
        }
        iriga++;
      }
      if(nerrate==0)
      {
        sprintf(record,"     Esito delle prove: POSITIVO     ");
        presenta(20,20,32,record);
        esito_dis='P';
      }
      else
      {
        sprintf(record,"     Esito delle prove: NEGATIVO     ");
        presenta(20,20,71,record);
        esito_dis='N';
      }
      sprintf(record,"    BATTI UN TASTO PER PROSEGUIRE    ");
      presenta(22,20,71,record);
      risposta=ci();
}


prinvaluta()
{
  extern int printer;
  extern int  numero_prog;
  extern char nuovo_paziente;
  extern int  printer,on_off;
  extern char cognome[],nome[],giorno[],mese[],anno[];
  extern char sesso,eta[],residenza[],indirizzo[];
```

```
extern char documento[],num_doc[],progressivo[];
extern char type_dis,type_sem,esito_dis,esito_sem;
char data[10],tempo[10];
char ff=0x0C;
char setta[3],rese[3],settal[4],resel[4];
int status,a;

setta[0]=0x1B;
setta[1]='E';
setta[2]=0;
rese[0]=0x1B;
rese[1]='F';
rese[2]=0;
settal[0]=0x1B;
settal[1]='l';
settal[2]=0x0A;
settal[3]=0;
resel[0]=0x1B;
resel[1]='l';
resel[2]=1;
resel[3]=0;

data_int(data);
times(tempo);
status=fprintf(printer,"%c%s\n",ff,settal);
status=fprintf(printer,"                %sSTIMOLATORE MULTISENSORIALE%s",setta,rese);
status=fprintf(printer,"\n\n\n");
status=fprintf(printer,"          %sS C H E D A   D I   V A L U T A Z I O N E%s",
          setta,rese);
status=fprintf(printer,"\n          %s----------------------------------------%s",
          setta,rese);
status=fprintf(printer,"\n\n");
status=fprintf(printer,"\n  %sDATI DEL: %s ore %s%s",setta,data,tempo,rese);
status=fprintf(printer,"\n\n  %sDati relativi al soggetto:%s\n",setta,rese);
status=fprintf(printer,"\nNUMERO PROGRESSIVO: %s%d%s",setta,
          numero_prog,rese);
status=fprintf(printer,"\nCOGNOME: %s%s%s    NOME: %s%s%s",
          setta,cognome,rese,setta,nome,rese);
status=fprintf(printer,
      "\nSESSO: %s%c%s      DATA DI NASCITA: %s%s-%s-%s%s",
          setta,sesso,rese,setta,giorno,mese,anno,rese);
status=fprintf(printer,"\nINDIRIZZO: %s%s%s",
          setta,indirizzo,rese);
status=fprintf(printer,"\nRESIDENTE A: %s%s%s",
          setta,residenza,rese);
status=fprintf(printer,"\nESTREMI DI RICONOSCIMENTO: %s%s %s%s",
          setta,documento,num_doc,rese);
status=fprintf(printer,"\n\n\n");
status=fprintf(printer,"  %sValutazione prove semplici:%s\n\n",setta,rese);
for(a=0; a<12; a++)
  status=fprintf(printer,"%s",val_sem[a]);
status=fprintf(printer,"\n\n\n");
status=fprintf(printer,"  %sValutazione prove di discriminazione:%s\n\n",setta,rese);
for(a=0; a<8; a++)
  status=fprintf(printer,"%s",val_dis[a]);
status=fprintf(printer,"\n\n\n");
status=fprintf(printer,"%sESITO DELLE PROVE = .....%s%c%c",setta,rese);
status=fprintf(printer,"\n\n\n");
```

```
  status=fprintf(printer,"      %sL'ANALISTA = .....%s%c%c",setta,rese,ff,ff);
  status=fprintf(printer,"%s",resel);
}


dati_tastiera(fmed,fdev,ioss,opt)
float fmed[],fdev[];
int  ioss[];
char opt;
{
  extern char *identi[];
  extern char *idiscri[];
  char *mess0,*mess1,*mess2,*mess3,*mess4;
  char risp,record[80];
  int  a;
  mess0="                                                ";
  mess1="         INTRODUZIONE  DATI  DA  TASTIERA              ";
  mess3="  Introduzione completa o solo per stimolatori (C/S)    ";
  mess4="    RICORDATI DI SEPARARE I DUE VALORI CON LO SPAZIO   ";
  scr_clr();
  presenta(1,10,71,mess0);
  presenta(2,10,71,mess1);
  presenta(3,10,71,mess0);
  fasce();
  presenta(12,10,23,mess3);
  while(((risp=ci())!='S')&&(risp!='s')&&(risp!='C')&&(risp!='c'));
  presenta(12,10,7,mess0);
  if((risp=='C')||(risp=='c'))
  {
    presenta(7,10,71,mess4);
    if(opt=='S')
    {
      for(a=0; a<12; a++)
      {
        sprintf(record,"Combinazione ");
        presenta(a+9,10,23,record);
        presenta(a+9,23,23,identi[a]);
      }
      for(a=0; a<12; a++)
      {
        scr_rowcol(a+9,50);
        scanf("%f %f",&fmed[a],&fdev[a]);
      }
    }
    else
    {
      for(a=0; a<8; a++)
      {
        sprintf(record,"Combinazione ");
        presenta(a+9,10,23,record);
        presenta(a+9,23,23,idiscri[a]);
      }
      for(a=0; a<8; a++)
      {
        scr_rowcol(a+9,51);
        scanf("%f %f",&fmed[a],&fdev[a]);
      }
    }
```

```
    }
    else
    {
      presenta(7,10,71,mess4);
      if(opt=='S')
      {
        sprintf(record,"Stimolatore VISIVO      ");
        presenta(10,10,23,record);
        sprintf(record,"Stimolatore UDITIVO     ");
        presenta(11,10,23,record);
        sprintf(record,"Stimolatore SOMESTESICO ");
        presenta(12,10,23,record);
        scr_rowcol(10,34);
        scanf("%f %f",&fmed[0],&fdev[0]);
        fmed[3]=fmed[2]=fmed[1]=fmed[0];
        fdev[3]=fdev[2]=fdev[1]=fdev[0];
        scr_rowcol(11,34);
        scanf("%f %f",&fmed[4],&fdev[4]);
        fmed[7]=fmed[6]=fmed[5]=fmed[4];
        fdev[7]=fdev[6]=fdev[5]=fdev[4];
        scr_rowcol(12,34);
        scanf("%f %f",&fmed[8],&fdev[8]);
        fmed[11]=fmed[10]=fmed[9]=fmed[8];
        fdev[11]=fdev[10]=fdev[9]=fdev[8];
      }
      else
      {
        sprintf(record,"Stimolatore VISIVO      ");
        presenta(10,10,23,record);
        sprintf(record,"Stimolatore UDITIVO     ");
        presenta(11,10,23,record);
        scr_rowcol(10,34);
        scanf("%f %f",&fmed[0],&fdev[0]);
        fmed[3]=fmed[2]=fmed[1]=fmed[0];
        fdev[3]=fdev[2]=fdev[1]=fdev[0];
        scr_rowcol(11,34);
        scanf("%f %f",&fmed[4],&fdev[4]);
        fmed[7]=fmed[6]=fmed[5]=fmed[4];
        fdev[7]=fdev[6]=fdev[5]=fdev[4];
      }
    }
    if(opt=='S')
    {
      for(a=0; a<12; a++)
        ioss[a]=31;
    }
    else
    {
      for(a=0; a<8; a++)
        ioss[a]=31;
    }
}



screen_val_sem()
{
  char *mess0,*mess1,*mess2;
```

```
  mess0="                                          ";
  mess1="          V A L U T A Z I O N E           ";
  mess2="            PROVE  SEMPLICI               ";
  scr_clr();
  presenta(1,8,71,mess0);
  presenta(2,8,71,mess1);
  presenta(3,8,71,mess2);
  fasce();
}



screen_val_dis()
{
  char *mess0,*mess1,*mess2;
  mess0="                                          ";
  mess1="          V A L U T A Z I O N E           ";
  mess2="          PROVE DISCRIMINAZIONE                 ";
  scr_clr();
  presenta(1,8,71,mess0);
  presenta(2,8,71,mess1);
  presenta(3,8,71,mess2);
  fasce();
}

/************************************************************** EOF *****/

/********************************** File: VIDEATE.C              */


/*                  --->start                   */
struct disp01 {
char stringa01[58];
int posizione_riga01;
int posizione_colo01;
int attributo01;
};

struct disp01 screen01[11]={
"        STIMOLATORE MULTISENSORIALE          ",1,10,23,
"                                    ",2,10,71,
"        T E S T   H A R D W A R E          ",3,10,71,
"                                    ",4,10,71,
"STIMOLATORE VISIVO .......",9,15,7,
"STIMOLATORE UDITIVO ......",11,15,7,
"STIMOLATORE SOMESTESICO ....",13,15,7,
"PULSANTIERA ..........",15,15,7,
"                                    ",20,10,71,
"          <ENTER> PER CONTINUARE          ",21,10,71,
"           ALT-Y  REINIZIALIZZA PROGRAMMA       ",22,10,71,
};
  char rispostaok[9]="PRESENTE";
  char rispostano[9]="ASSENTE ";

test_hardware()
{
  extern char per_status[];
  extern int macchina;
```

```
extern unsigned portim;
        int irrisp[4]={9,11,13,15};
        int icrisp=51,attribok=32,attribno=71;
        char indirizzi[3]={0x50,0x70,0x90};
        char key,pulsanti,out;
int  a,b,c,d,istimolo;
unsigned word,word1,word2,word3,word4;

scr_clr();
if(macchina==2)
{
  screen01[7].posizione_riga01=screen01[6].posizione_riga01;
  irrisp[3]=irrisp[2];
}

for (a=0; a<11; a++)
presenta(screen01[a].posizione_riga01,screen01[a].posizione_colo01,
     screen01[a].attributo01,screen01[a].stringa01);
fasce();

/* inizio test su periferiche collegate ------
                    esegue cicli di riconoscimento ----------*/
istimolo=0;
trasmette_comandi('V','1');    /* ciclo di riconoscimento visivo */
if (per_status[istimolo]==1)
  presenta(irrisp[istimolo],icrisp,attribok,rispostaok);
else
{
  presenta(irrisp[istimolo],icrisp,attribno,rispostano);
}
istimolo+=1;
trasmette_comandi('U','1');    /* ciclo di riconoscimento uditivo */
if (per_status[istimolo]==1)
  presenta(irrisp[istimolo],icrisp,attribok,rispostaok);
else
{
  presenta(irrisp[istimolo],icrisp,attribno,rispostano);
}
istimolo+=1;
if(macchina==1)
{
  trasmette_comandi('S','1');  /* ciclo di riconoscimento somestesico */
  if (per_status[istimolo]==1)
    presenta(irrisp[istimolo],icrisp,attribok,rispostaok);
  else
    presenta(irrisp[istimolo],icrisp,attribno,rispostano);
}
else
  per_status[istimolo]=-1;

/*--- Testa presenza quattro pulsanti ---------------------------- */
  istimolo+=1;
  pulsanti=_inb(portim-2)&0x1F;
  if (pulsanti==0x0F)
    presenta(irrisp[istimolo],icrisp,attribok,rispostaok);
  else
    presenta(irrisp[istimolo],icrisp,attribno,rispostano);
```

```
/* --- Test sul funzionamento dei timer ----------------------------*/

   word=0x1234;
   load_timer(word);
   start_timer();
   for(b=0; b<500; b++);  /*    WAIT    */
   word4=legge_timer();
   for(b=0; b<500; b++);  /*    WAIT    */
   word1=legge_timer();
   for(b=0; b<500; b++);  /*    WAIT    */
   word2=legge_timer();
   for(b=0; b<500; b++);  /*    WAIT    */
   word3=legge_timer();
   stop_timer();
/*  printf("IMPOSTATO %u -- 1-2-3 %u %u %u",word,word1,word2,word3); */
   if(word3==word2) printf("  --- FAIL");
   if(word2==word1) printf("  --- FAIL");
   if(word1==word4) printf("  --- FAIL");

/* --- ATTESA COMANDO ------------------------------------------- */

   while (((key=ci())!=21)&&(key!=13));
   return(key);
}
/*
=============================================================================
*/

  int flop;

/*
=============================================================================
*/


struct disp02 {
char stringa02[55];
int posizione_riga02;
int posizione_colo02;
int attributo02;
};

struct disp02 screen02[10]={
"          STIMOLATORE MULTISENSORIALE          ",1,10,23,
"                                               ",2,10,71,
"           T I P O   D I   P R O V A           ",3,10,71,
"                                               ",4,10,71,
"1. : PROVE DI REAZIONE SEMPLICI    ",10,20,2,
"2. : PROVE DI DISCRIMINAZIONE      ",12,20,2,
"3. : LISTA CONTENUTO DEL DISCHETTO ",14,20,2,
"             <ESC> menu' precedente            ",18,10,71,
"             ALT-Y Reinizializza programma     ",19,10,71,
"             Introduci: 1 - 2 - 3              ",20,10,71,
};

tipo_di_prova(flag)
char *flag;
{
```

```
    extern int macchina,printer,on_off;
    char risposta,key;
    char op1=0x31,op2=0x32,op3=0x33;
    int  a,status;
    if((macchina==2)&&(*flag==0))
    {
      op3=op2;
      screen02[6].posizione_riga02=screen02[5].posizione_riga02;
      screen02[5].posizione_riga02=screen02[4].posizione_riga02;
      screen02[6].stringa02[0]=screen02[5].stringa02[0];
      screen02[5].stringa02[0]=screen02[4].stringa02[0];
      screen02[9].stringa02[34]=0x20;
      screen02[9].stringa02[36]=0x20;
      *flag=0xFF;
    }
    risposta=0x00;
    while (!risposta)   /* ---- loop generale -------*/
    {
      scr_clr();
      for (a=0; a<10; a++)
      presenta(screen02[a].posizione_riga02,screen02[a].posizione_colo02,
           screen02[a].attributo02,screen02[a].stringa02);
      fasce();
      while (((key=ci())!=21)&&(key!=27)&&(key!=32)&& /* ALT-Y o ESCape */
           (key!=op1)&&(key!=op2)&&(key!=op3));   /* Numerico      */
      apre_temporaneo();
      if((macchina==2)&&(key>'0'))  key++;
      switch (key)
      {
        case 32:   system_op();
                   break;
        case 21:   exit();
                   break;
        case 27:   risposta=0x0FF;
                   return(27);
                   break;
        case '1':  visivo('2');
                   uditivo('2');
                   if(anagrafe_ok()==27)
                     risposta=0xFF;
                   else
                     prove_semplici();
                   break;
        case '2':  visivo('2');
                   uditivo('2');
                   if(anagrafe_ok()==27)
                     risposta=0xFF;
                   else
                     prove_discriminazione();
                   break;
        case '3':  lista_dischetto();
                   break;
      }
    }
    return(key);
}

/*****************************************************************************/
```

```
  float media[12],scarto[12],dimedia[128],discarto[128];
  float varianz[12],assolut[12],divarianz[128],diassolut[128];
  int validi[240],errate[12],assenti[12];
  int divalidi[320],dierrate[8],diassenti[8];          /* MODIFICATI */
  int voto[12],divoto[8],gia_effettuate,incurs;
  char effettuate[8],possibilita[8];
  char scelta[14],attr[14],inside[14],street[12];

/*****************************************************************************/

anagrafe_ok()
{
   extern int  numero_prog;
   extern char nuovo_paziente;
   extern int  printer,on_off;
   extern char cognome[],nome[],giorno[],mese[],anno[];
   extern char sesso,eta[],residenza[],indirizzo[];
   extern char documento[],num_doc[],progressivo[];
   extern char type_dis,type_sem,esito_dis,esito_sem;
   int a,status;
   char *mess0,*mess1,*vuoi_anagrafe,risposta,bell=7,test;
   char data[10],tempo[10],setta[3],rese[3],risparc,ff=12;
   vuoi_anagrafe="  VUOI INTRODURRE I DATI ANAGRAFICI (S/N)?  ";
   mess0="         DATI NON COMPLETAMENTE ARCHIVIATI          ";
   mess1="     VUOI ARCHIVIARE PRIMA DI PROSEGUIRE (S/N)?     ";
   setta[0]=0x1B;
   setta[1]='E';
   setta[2]=0;
   rese[0]=0x1B;
   rese[1]='F';
   rese[2]=0;
   for(a=0; a<14; a++)
   {
     scelta[a]=0;
     attr[a]=2;
     inside[a]=0;
   }
   if(numero_prog>0)
   {
     presenta(16,15,23,vuoi_anagrafe);
     printf("%c",bell);
     risposta=ci();
     if((risposta=='s')||(risposta=='S'))
     {
       test = nuovo_paziente&3;
       if(test != 3)
       {
         printf("%c",bell);
         presenta(15,10,87,mess0);
         presenta(16,10,87,mess1);
         while(((risparc=ci())!='s')&&(risparc!='S')&&
              (risparc!='n')&&(risparc!='N'));
         if((risparc=='s')||(risparc=='S'))
           scelta_archiviazione(0);
       }
     }
   }
   else
```

```
  risposta='s';

 if(risposta=='s'||risposta=='S')
 {
  if(anagrafe()==27) return(27);
  for(a=0; a<12; a++)
  {
     street[a]=0;
     errate[a]=0;
     assenti[a]=0;
     media[a]=0.0;
     scarto[a]=0.0;
     varianz[a]=0.0;
     assolut[a]=0.0;
  }
  type_sem=esito_sem=0;
  for(a=0; a<320; a++)
    divalidi[a]=0.0;
  for(a=0; a<240; a++)
    validi[a]=0.0;
  for(a=0; a<8; a++)
  {
    effettuate[a]=0;
    possibilita[a]=0;
    dierrate[a]=0;
    diassenti[a]=0;
    dimedia[a]=0.0;
    discarto[a]=0.0;
    divarianz[a]=0.0;
    diassolut[a]=0.0;
  }
  type_dis=esito_dis=0;
  nuovo_paziente=0;
  gia_effettuate=0;
  data_int(data);
  times(tempo);
  if(on_off!=0)
  {
   status=fprintf(flop,"%c\n ---------- %s --- %s----------",ff,
              data,tempo);
   status=fprintf(flop,"\nNUMERO PROGRESSIVO: %s%d%s",setta,
              numero_prog,rese);
   status=fprintf(flop,"\nCOGNOME: %s%s%s    NOME: %s%s%s",
              setta,cognome,rese,setta,nome,rese);
   status=fprintf(flop,
         "\nSESSO: %s%c%s      DATA DI NASCITA: %s%s-%s-%s%s",
              setta,sesso,rese,setta,giorno,mese,anno,rese);
   status=fprintf(flop,"\nINDIRIZZO: %s%s%s",
              setta,indirizzo,rese);
   status=fprintf(flop,"\nRESIDENTE A: %s%s%s",
              setta,residenza,rese);
   status=fprintf(flop,"\nESTREMI DI RICONOSCIMENTO: %s%s %s%s",
              setta,documento,num_doc,rese);
   status=fprintf(flop,
        "\n ----------------------------------------------------\n");
   stampa_temporaneo();
  }
}
```

```
   apre_temporaneo();
}

/*****************************************************************************/

struct disp04 {
char stringa04[59];
int posizione_riga04;
int posizione_colo04;
int attributo04;
};

struct disp04 screen04[29]={
"          STIMOLATORE MULTISENSORIALE          ",1,10,23,
"                                               ",2,10,71,
"        D A T I   A N A G R A F I C I          ",3,10,71,
"                                               ",4,10,71,
"COGNOME: ",7,4,7,
"                    ",7,13,23,
" NOME: ",7,40,7,
"                      ",7,47,23,
"DATA DI NASCITA: ",9,4,7,
" / / ",9,21,23,
" SESSO: ",9,31,7,
"      ",9,39,23,
"ETA` ANNI: ",9,48,7,
"             ",9,59,23,
"RESIDENTE A: ",11,4,7,
"                                    ",11,17,23,
"INDIRIZZO :  ",13,4,7,
"                                       ",13,17,23,
"TIPO DEL DOCUMENTO: ",15,4,7,
"                ",15,24,23,
" N.: ",15,45,7,
"                    ",15,50,23,
"ARTO PREVALENTE ",17,4,7,
"           ",17,20,23,
"PROGRESSIVO   ",17,38,7,
"           ",17,52,23,
"                                               ",20,10,71,
"           <ENTER> Abilitazione/Continua       ",21,10,71,
"                                               ",22,10,71,
};

  char cognome[26],nome[28],giorno[3],mese[3],anno[3];
  char sesso,eta[16],residenza[48],indirizzo[58];
  char documento[20],num_doc[25],progressivo[14],prev[2],trixer;
  int  lunghezze[13]={26,28,2,2,2,1,16,48,58,20,25,1,14};
  int  anni;
  unsigned indirizzi[13]={&cognome,&nome,&giorno,&mese,&anno,
            &sesso,&eta,&residenza,&indirizzo,&documento,
            &num_doc,&prev,&progressivo};
  int  stipo[13]={0,0,1,1,1,3,0,2,2,2,2,3,0};
  char idents[2]={'M','F'};
  char identa[2]={'D','S'};

anagrafe()
{
```

```
extern int numero_prog;
extern char per_status[];
int iriga[13]={7,7,9,9,9,9,9,11,13,15,15,17,17};
int icolo[13]={13,47,21,24,27,39,59,17,17,24,50,20,52};
char buffer[20],today[9],bell=0x07,fanni;
char day[3],mou[3],yea[3];
char *messaggio,*abblenka,risposta,record[80];
int a,b,igiorno,imese,ianno,idata,ioggi,nonvalidi;

scr_clr();
messaggio=" PREGO CONFERMARE LA CORRETTA INTRODUZIONE DATI (S/N) ";
abblenka="                                                    ";
today[8]=0;
giorno[2]=0;
mese[2]=0;
anno[2]=0;
day[2]=0;
mou[2]=0;
yea[2]=0;
prev[2]=0;
numero_prog++;
giorno[0]=giorno[1]='0';
mese[0]=mese[1]='0';
anno[0]=anno[1]='0';
for (a=0; a<29; a++)
presenta(screen04[a].posizione_riga04,screen04[a].posizione_colo04,
     screen04[a].attributo04,screen04[a].stringa04);
fasce();
sprintf(record," %05d ",numero_prog);
presenta(iriga[12],icolo[12],23,record);
b=0;
nonvalidi=0;
while ((nonvalidi==0)||(nonvalidi==1))
{
 for (a=b; a<6; a++)
 {
 attesa:
   attende_char(iriga[a],icolo[a],151,lunghezze[a],indirizzi[a],
            stipo[a],idents);
   presenta(iriga[a],icolo[a],23,indirizzi[a]);
   if (((igiorno=atoi(&giorno)) < 0) || (igiorno > 31))
   {
     presenta(iriga[a],icolo[a],148,indirizzi[a]);
     printf("%c",bell);
     goto attesa;
   }
   if (((imese=atoi(&mese)) < 0) || (imese > 12))
   {
     presenta(iriga[a],icolo[a],148,indirizzi[a]);
     printf("%c",bell);
     goto attesa;
   }
 }
 ianno=atoi(&anno);
 idata=ianno*365+imese*30+igiorno;
 data_int(today);
 day[0]=today[0];
 day[1]=today[1];
```

```
    mou[0]=today[3];
    mou[1]=today[4];
    yea[0]=today[6];
    yea[1]=today[7];
    igiorno=atoi(&day);
    imese=atoi(&mou);
    ianno=atoi(&yea);
    ioggi=ianno*365+imese*30+igiorno;
    anni=(ioggi-idata)/365.0;
    fanni=0;
    if((anni<18)||(anni>100))
    {
      sprintf(record,"     %d     ",anni);
      presenta(iriga[6],icolo[6],23+128,record);
      printf("%c",bell);
      fanni=255;
    }
    else
    {
      sprintf(record,"     %d     ",anni);
      presenta(iriga[6],icolo[6],23,record);
      nonvalidi=0;
      fanni=0;
      b=0;
    }
    if(fanni!=0)
    {
      nonvalidi=1;
      b=2;
    }
    if(nonvalidi==0)
    {
      for (a=7; a<12; a++)
      {
        attende_char(iriga[a],icolo[a],151,lunghezze[a],indirizzi[a],
                stipo[a],identa);
        presenta(iriga[a],icolo[a],23,indirizzi[a]);
      }
      presenta(19,10,199,messaggio);
      printf("%c",bell);
      while(((risposta=ci())!=83)&&(risposta!=115)&&
          (risposta!=78)&&(risposta!=110));
      presenta(19,10,7,abblenka);
      if((risposta==78)||(risposta==110))
        b=0;
      else
        nonvalidi=2;
    }
  }
}

/*
  GESTISCE I CAMPI NELLA VIDEATA ANAGRAFE
*/

attende_char(iriga,icolo,attrib,nchar_max,buffer,tipo,array)
int  iriga,icolo,nchar_max;
char attrib,*buffer;
```

```
int  tipo;
char array[];
{
  char carattere,flag,oktoload;
  char string2[2]={0X00,0X00};
  char bell=7,*spazi;
  int indice,posizione,finest;
  posizione=icolo;
  indice=0;
  flag=0;
  spazi="                                                    ";
  attrib=attrib&0xF4;     /* sfondo come richiesto,caratteri rossi */
  attrib=attrib|0x07;
  scr_rowcol(iriga,icolo);
  while(!flag)
  {
    carattere=ci();
    if((carattere>96)&&(carattere<123))
       carattere-=32;
    switch (carattere)
    {
      case 8:  if(indice!=0)
               {
                 posizione--;
                 indice--;
                 scr_rowcol(iriga,posizione);
                 string2[0]=0X20;
                 presenta(iriga,posizione,attrib,string2);
                 scr_rowcol(iriga,posizione);
               }
               break;

      case 13: if(indice!=0)
               {
                 buffer[indice]=0;
                 finest=nchar_max-indice;
                 spazi[finest]=0x00;
                 scr_aputs(spazi,attrib);
                 spazi[finest]=0x20;
                 flag=0xFF;
               }
               else
                 if((tipo==3)||(tipo==1))
                   flag=0;
                 else
                   flag=0xFF;
               break;
      case 21: exit();
               break;
      case 27: return(27);
               break;
      default: oktoload=0;
               switch (tipo)
               {
                 case 0: if ((isalpha(carattere)) || (carattere==0x20))
                           oktoload=1;
                         break;
                 case 1: if (isdigit(carattere))
```

```
                    oktoload=1;
                  break;
            case 2: if ((isalnum(carattere)) || (carattere==0x20))
                    oktoload=1;
                  break;
            case 3: if (isalpha(carattere))
                    if ((carattere==array[0])||
                       (carattere==array[1]))  oktoload=1;
                  break;
          }
          if (oktoload!=0)
          {
            string2[0]=carattere;
            presenta(iriga,posizione,attrib,string2);
            buffer[indice]=carattere;
            indice++;
            posizione++;
            if(indice>nchar_max-1)   flag=0xFF;
          }
          else
            printf("%c",bell);
          break;
      }
    }
}

/*
==========================================================================
*/

apre_temporaneo()
{
  extern int on_off,printer;
  char *floptemp="FILETEMP.DAT";
  int status;
  if(on_off!=0)
  {
    flop=creat(floptemp);
    status=close(flop);
    flop=open(floptemp,1);
  }
}

/*
==========================================================================
*/

stampa_temporaneo()
{
  extern int flop,printer,on_off;
  char *floptemp="FILETEMP.DAT";
  char record[132],sta;
  int status;
  if(on_off!=0)
  {
    status=close(flop);
    flop=open(floptemp,0);
    while((sta=fgets(record,132,flop))!=0)
```

```
        fprintf(printer,"%s",record);
      status=close(flop);
    }
}

/*
================================================================================
*/

struct disp05 {
char stringa05[62];
int posizione_riga05;
int posizione_colo05;
int attributo05;
};

struct disp05 screen05[19]={
"            STIMOLATORE MULTISENSORIALE            ",1,10,23,
"                                                   ",2,10,71,
"         S E S S I O N E   I N   C O R S O         ",3,10,71,
"              PROVE  SEMPLICI                      ",4,10,71,
"    USA LE FRECCE <>^v  PER SELEZIONARE LA MISURA   ",5,10,23,
"-----------------------------------------",6,24,7,
"! MANO ds ! MANO sn ! PIEDEds ! PIEDEsn !",7,24,7,
"-----------------------------------------------------",8,12,7,
"! VISIVO   !       !       !       !       !",9,12,7,
"-----------------------------------------------------",10,12,7,
"! UDITIVO  !       !       !       !       !",11,12,7,
"-----------------------------------------------------",12,12,7,
"! SOMESTESI !       !       !       !       !",13,12,7,
"-----------------------------------------------------",14,12,7,
"1.: ESECUZIONE MISURA     2.: VISUALIZZAZIONE DATI ACQUISITI",16,8,2,
"3.: ARCHIVIAZIONE DATI    4.: RIPROGRAMMAZIONE STIMOLATORE",18,8,2,
"      ALT-Y  reinizializza programma               ",20,10,71,
"      <ESC>  menu` precedente                      ",21,10,71,
"           Introduci: 1 - 2 - 3 - 4                ",22,10,71,
};

  char curriga[12]={9,9,9,9,11,11,11,11,13,13,13,13};
  char curcolo[12]={25,35,45,55,25,35,45,55,25,35,45,55};
  char cursore[10]={32,32,32,32,32,32,32,32,32,0};
  char attributo_on=32,attributo_off=7;
  char stim_selezionato; /* 0:VISIVO  1:UDITIVO  2:SOMESTESICO      */
  char arto_selezionato; /* contiene gia' la maschera corrispondente */
  char apprendarray[3];  /* [0]:VISIVO  [1]:UDITIVO  [2]:SOMESTESICO */
                         /* 00:DA EFFETUARSI APPRENDIMENTO          */
                         /* FF:APPRENDIMENTO GIA` EFFETTUATO        */
  char arto_numerico;    /* 0:mds  1:msn  2:pds  3:psn              */
  char arto_num1,arto_num2;

prove_semplici()
{
  extern int macchina;
  extern char per_status[];
  extern printer;
  int  a,b,iflag,incurs,istep,status,nliberi;
  char ch,out,risposta,*finesec;
  char flag=0x00;
```

```
  char *messag,bell=0x07;
  char *str1,*str2,nflag,somok;   /* soglia stim. somestesico */
  somok=0;
  apprendarray[0]=0x00;
  apprendarray[1]=0x00;
  apprendarray[2]=0x00;
  messag="   ERRORE SULLO STIMOLATORE SELEZIONATO   ";
  finesec="1.: ESECUZIONE MISURA    ";

/***** status=fprintf(flop,"STATI> %d %d %d\n",
          per_status[0],per_status[1],per_status[2]); ****************/

/* NEL CASO DI ASSENZA DI TUTTI E TRE GLI STIMOLATORI ESCE DALLA FUNZIONE */
  if((per_status[0]!=1)&&(per_status[0]!=2)
     &&(per_status[1]!=1)&&(per_status[1]!=2)
       &&(per_status[2]!=1)&&(per_status[2]!=2))
  {
     scr_clr();
     printf("\n*** IMPOSSIBILE ATTIVARE LA FUNZIONE RICHIESTA ***");
     printf("\n***   Batti un tasto qualsiasi per proseguire  ***");
     ch=ci();
     return;
  }

  if(macchina==2)
  {
    for(a=0; a<53; a++)
    {
      screen05[12].stringa05[a]=0x20;
      screen05[13].stringa05[a]=0x20;
    }
    street[8]=0xFF;          /* il cursore non puo` entrare  */
    street[9]=0xFF;
    street[10]=0xFF;
    street[11]=0xFF;
  }
  flag=0x00;

nuovo_display:

  scr_clr();
  for (a=0; a<19; a++)
  presenta(screen05[a].posizione_riga05,screen05[a].posizione_colo05,
       screen05[a].attributo05,screen05[a].stringa05);
  fasce();

/* PER OGNI SINGOLO STIMOLATORE ASSENTE NE IMPEDISCE L'USO */
/* ----- VISIVO ----- */
  if((per_status[0]!=1)&&(per_status[0]!=2))
  {
    for(a=0; a<4; a++)
      messaggio_assenza(a);
  }
/* ----- UDITIVO ----- */
  if((per_status[1]!=1)&&(per_status[1]!=2))
  {
    for(a=4; a<8; a++)
      messaggio_assenza(a);
```

```
  }
/* ----- SOMESTESICO ----- */
  if((per_status[2]!=1)&&(per_status[2]!=2))
  {
    for(a=8; a<12; a++)
      messaggio_assenza(a);
  }
/* ================================================================================ */
/* reset(1000);                          /*** RESET BUS ITBM ***/
/* ================================================================================ */
  nliberi=12;
  incurs=-1;
  for(a=0; a<12; a++)
  {
    if(street[a]==0x55)
    {
      presito(a);        /* scrive ESEGUITA in corrispondenza */
      nliberi--;
    }
    if(street[a]==0xFF)
      nliberi--;
    if(street[a]==0);
      incurs=a;          /* c`e` almeno un'altra possibilita' */
  }
  if(nliberi==0)
    presenta(16,8,71,finesec);
  if(incurs!=-1)
  {
    flag=0x00;
    incurs=0;
    if(street[incurs]!=0)
    {
      istep=calcola_libero(incurs,1);
      if(istep!=-1)
      {
        iflag=street[incurs+istep];
        if(iflag==0)
        {
          incurs+=istep;
          write_cursore(incurs);
        }
      }
    }
    else
    {
      write_cursore(incurs);
    }
  }
  else
  {
    flag=0xFF;
  }
  while(!flag)
  {
    while(((ch=ci())!='1')&&(ch!='2')&&(ch!='3')&&(ch!='4')&&(ch!=0)
                  &&(ch!=21)&&(ch!=27));
    switch (ch)
    {
```

```
case 0:  while(((ch=ci())!=72)&&(ch!=75)&&
         (ch!=77)&&(ch!=80)&&(ch!=21));
        switch (ch)
        {
          case 72: if(incurs>3)
                   {
                    iflag=street[incurs-4];
                    if(iflag==0)
                    {
                     clear_cursore(incurs);
                     incurs-=4;
                     write_cursore(incurs);
                    }
                   }
                   break;
          case 75: if(incurs>0)
                   {
                    istep=calcola_libero(incurs,0);
                    if(istep!=-1)
                    {
                     iflag=street[incurs-istep];
                     if(iflag==0)
                     {
                      clear_cursore(incurs);
                      incurs-=istep;
                      write_cursore(incurs);
                     }
                    }
                   }
                   break;
          case 77: if(incurs<11)
                   {
                    istep=calcola_libero(incurs,1);
                    if(istep!=-1)
                    {
                     iflag=street[incurs+istep];
                     if(iflag==0)
                     {
                      clear_cursore(incurs);
                      incurs+=istep;
                      write_cursore(incurs);
                     }
                    }
                   }
                   break;
          case 80: if(incurs<8)
                   {
                    iflag=street[incurs+4];
                    if(iflag==0)
                    {
                     clear_cursore(incurs);
                     incurs+=4;
                     write_cursore(incurs);
                    }
                   }
                   break;
          case 21: flag=0xFF;
                   break;
```

```
                }
                break;
        case '1': determina_stimart(incurs);
                switch (stim_selezionato)
                {
                  case 0: reset(500);
                        trasmette_comandi('V','5');
                        break;
                  case 1: reset(500);
                        trasmette_comandi('U','5');
                        break;
                  case 2: reset(500);
                        trasmette_comandi('S','5');
                        trigger(10);
                        send_data_some(0x0A);
/*-------------------------------------------------------------------------*/
   str1="   DETERMINAZIONE SOGLIA STIMOLATORE SOMESTESICO   ";
   str2=" <esc> = TERMINA              <barra> = STIMOLA  ";
   scr_clr();
   presenta(10,10,23,str1);
   presenta(11,10,23,str2);
   printf("%c",bell);
   nflag=0;
   somok=1;
   while(!nflag)
   {
     while(((risposta=ci())!=0x1B)&&(risposta!=0x20));
     switch (risposta)
     {
       case 0x1B: nflag=0xFF;
                break;
       case 0x20: trigger(10);
                send_data_some(0x0A);
                break;
     }
   }
/*-------------------------------------------------------------------------*/
                        break;
                }
                if(per_status[stim_selezionato]!=2)
                {
                     presenta(19,10,23,messag);
                     printf("%c",bell);
                     risposta=ci();
                     goto nuovo_display;
                }
/*              if(apprendarray[stim_selezionato]==00)
                {   */
                  out=esame_apprendimento();
                  if(out==21)   exit();
                  if(out!=27)
                  {
                    if(out==1)                  /* esame        */
                    {
                      if(esame_semplice(incurs)!=27)
                          street[incurs]=0x55;
                      flag=0x55;
                    }
```

```
                    else
                    {                           /* apprendimento */
                      if(apprendimento_semplice()!=27)
                      {
                        apprendarray[stim_selezionato]=0xFF;
                        if(esame_semplice(incurs)!=27)
                           street[incurs]=0x55;
                      }
                      flag=0x55;
                    }
                  }
/*                  else
                  {
                    if(esame_semplice(incurs)!=27)
                        street[incurs]=0x55;
                    flag=0x55;
                  }
                }          */
                break;
        case '2': visualizza_dati('S');
                flag=0x55;
                break;
        case '3': if(scelta_archiviazione(0)==27)
                    flag=0x55;
                else
                    flag=0xFF;
                break;
        case '4': determina_stimart(incurs);
                riprogramma_stimolatore();
                flag=0x55;
                break;
        case 21:  exit();
                break;
        case 27:  return(27);
                break;
      }
    }
    if(flag==0x55) goto nuovo_display;
}


clear_cursore(posizione)
int posizione;
{
  presenta(curriga[posizione],curcolo[posizione],
        attributo_off,cursore);
}


write_cursore(posizione)
int posizione;
{
  presenta(curriga[posizione],curcolo[posizione],
        attributo_on,cursore);
}

presito(posizione)
int posizione;
```

```
{
  char *esito;
  esito=" ESEGUITA";
  presenta(curriga[posizione],curcolo[posizione],71,esito);
}


messaggio_assenza(posizione)
int posizione;
{
   char *assente;
   assente=" ASSENTE ";
   assente[9]=0;
   presenta(curriga[posizione],curcolo[posizione],71,assente);
   street[posizione]=0xFF;
}


calcola_libero(posizione,direzione)
int posizione,direzione;          /* 0=PRECEDENTE   1=SUCCESSIVO */
{
  int iret,istep,indice;
  istep=256;
  if(direzione==0)
  {
    for(indice=posizione-1; indice>=0; indice--)
    {
      if(street[indice]==0)
      {
        istep=indice;
        iret=posizione-istep;
        indice=-1;
      }
    }
  }
  else
  {
    for(indice=posizione+1; indice<12; indice++)
    {
      if(street[indice]==0)
      {
        istep=indice;
        iret=istep-posizione;
        indice=12;
      }
    }
  }
  if(istep==256)
    iret=-1;
  return(iret);
}

determina_stimart(posizione)
int posizione;
{
  char fonte_stimolo[12]={0,0,0,0,1,1,1,1,2,2,2,2};
  char arto_risposta[12]={1,2,4,8,1,2,4,8,1,2,4,8};
  char out;
```

```
  stim_selezionato=fonte_stimolo[posizione];
  arto_selezionato=arto_risposta[posizione];
  arto_numerico=posizione&0x03;
}



riprogramma_stimolatore()
{
  extern char per_status[];
  char *mess1,*mess2,bell=0x07,risposta;
  mess1="  STIMOLATORE NON INIZIALIZZATO  ";
  mess2="   STIMOLATORE NON PROGRAMMATO   ";
  switch (stim_selezionato)
  {
    case 0: if((per_status[0]==1)||(per_status[0]==2))
            {
              visivo('4');
              trasmette_comandi('V','5');
              if(per_status[0]!=2)
              {
                presenta(19,10,23,mess2);
                printf("%c",bell);
                risposta=ci();
              }
            }
            else
            {
              presenta(19,10,23,mess1);
              printf("%c",bell);
              risposta=ci();
            }
            break;
    case 1: if((per_status[1]==1)||(per_status[1]==2))
            {
              uditivo('4');
              trasmette_comandi('U','5');
              if(per_status[1]!=2)
              {
                presenta(19,10,23,mess2);
                printf("%c",bell);
                risposta=ci();
              }
            }
            else
            {
              presenta(19,10,23,mess1);
              printf("%c",bell);
              risposta=ci();
            }
            break;
    case 2: if((per_status[2]==1)||(per_status[2]==2))
            {
              somestesico('4');
              trasmette_comandi('S','5');
              if(per_status[2]!=2)
                presenta(19,10,23,mess2);
            }
```

```
            else
            {
              presenta(19,10,23,mess1);
            }
            printf("%c",bell);
            risposta=ci();
            break;
    }
}

/* ========================================================================== */

struct disp07 {
char stringa07[57];
int posizione_riga07;
int posizione_colo07;
int attributo07;
};

struct disp07 screen07[9]={
"            STIMOLATORE MULTISENSORIALE          ",1,10,23,
"                                                 ",2,10,71,
"            T I P O   D I   P R O V A            ",3,10,71,
"                                                 ",4,10,71,
"1. : PROVE DI APPRENDIMENTO",10,23,2,
"2. : PROVE DI ESAME",13,23,2,
"                                                 ",20,10,71,
"  <ESC> menu` precedente     ALT-Y reinizializza    ",21,10,71,
"           Introduci: 1 - 2                       ",22,10,71,
};


esame_apprendimento()          /* Ritorna 0:ESAME o 1:APPRENDIMENTO    */
{
   int  a;
   char risposta,key;
   scr_clr();
   for (a=0; a<9; a++)
   presenta(screen07[a].posizione_riga07,screen07[a].posizione_colo07,
        screen07[a].attributo07,screen07[a].stringa07);
   fasce();
   risposta=0x00;
   while (!risposta)                   /* ---- loop generale -------*/
   {
     while (((key=ci())!=21)&&(key!=27)&&          /* ALT-Y o ESCape */
          (key!='1')&&(key!='2'));              /* Numerico      */
     switch (key)
     {
       case 21:  exit();
                 break;
       case 27:  risposta=0x0FF;
                 break;
       case '1': key=0;
                 risposta=0x0FF;
                 break;
       case '2': key=1;
                 risposta=0x0FF;
                 break;
```

```
      }
    }
    return(key);
}
/*
==========================================================================
*/



struct disp08 {
char stringa08[59];
int posizione_riga08;
int posizione_colo08;
int attributo08;
};

struct disp08 screen08[18]={
"            STIMOLATORE MULTISENSORIALE            ",1,10,23,
"                                                   ",2,10,71,
"      P R O V E   D I   A P P R E N D I M E N T O  ",3,10,71,
"                                                   ",4,10,71,
" NOME SOGGETTO =                    DATA=       ",5,10,23,
"STIMOLATORE = ",6,11,2,
"          ",6,25,23,
"ARTO = ",6,40,2,
"               ",6,47,23,
"NUMERO DELLE PROVE = ",10,18,2,
"       ",10,39,23,
"TIPO = ",12,17,2,
" AUTOMATICO ",12,24,23,
"  /  ",12,36,7,
" MANUALE    ",12,41,7,
"                                                   ",19,10,71,
"          <ESC>  menu` precedente                  ",20,10,71,
"         <ENTER> abilitazione/continua              ",21,10,71,
};

    int nprove,auto_man;

apprendimento_semplice()
{
    char out;
    out=numero_tipo(0);
    if(out==27)  return(27);
    if(esegui_apprendi_semplice()==27)   return(27);
}


numero_tipo(f)
char f;
{
    extern char possibilita[];
    extern int gia_effettuate;
    extern int  possibili[];
    char *nmaxprove;
    int  a,idd;
    int cpaziente=27,cdata=59,len;
```

```
char *cursore,*spazi,*fonte,*arisp,data_odierna[9];
char buffer[9],key,risposta,out,flag,dum[2],dato;
char record[80],c_modo[7]={0,0,0,0,0,0,0};
buffer[2]=0;
nmaxprove="          NUMERO MASSIMO DELLE PROVE = 8          ";
spazi="                                        ";
cursore=" <----- ";
spazi="       ";
cursore[8]=0;
spazi[8]=0;
flag=0x00;
scr_clr();
for (a=0; a<7; a++)
presenta(screen08[a].posizione_riga08,screen08[a].posizione_colo08,
     screen08[a].attributo08,screen08[a].stringa08);
for (a=9; a<18; a++)
presenta(screen08[a].posizione_riga08,screen08[a].posizione_colo08,
     screen08[a].attributo08,screen08[a].stringa08);
if(f==0)
{
  for (a=7; a<9; a++)
  presenta(screen08[a].posizione_riga08,screen08[a].posizione_colo08,
       screen08[a].attributo08,screen08[a].stringa08);
  switch (stim_selezionato)
  {
    case 0: fonte="  VISIVO  ";
            break;
    case 1: fonte="  UDITIVO  ";
            break;
    case 2: fonte=" SOMESTESICO";
            break;
  }
  switch (arto_selezionato)
  {
    case 1: arisp="  MANO DESTRA    ";
            break;
    case 2: arisp="  MANO SINISTRA  ";
            break;
    case 4: arisp="  PIEDE DESTRO   ";
            break;
    case 8: arisp="  PIEDE SINISTRO  ";
            break;
  }
  presenta(6,25,23,fonte);
  presenta(6,47,23,arisp);
}
else
{
  sprintf(record," COMBINAZIONE=                    ");
  presenta(6,11,23,record);
  dato=possibilita[gia_effettuate-1];
  idd=possibilita[gia_effettuate-1];
  dato=possibili[idd];
  crea_stringa(dato,c_modo);
  presenta(6,26,23,c_modo);
}
fasce();
presenta(8,10,71+128,nmaxprove);
```

```
  data_int(data_odierna);
  len=strlen(cognome)+1;
  presenta(screen08[4].posizione_riga08,cpaziente,23,cognome);
  cpaziente+=len;
  presenta(screen08[4].posizione_riga08,cpaziente,23,nome);
  presenta(screen08[4].posizione_riga08,cdata,23,data_odierna);
  presenta(10,58,23,cursore);
  attende_char(10,39,151,2,buffer,1,dum);
  presenta(10,58,7,spazi);
  presenta(12,58,23,cursore);
  presenta(10,39,23,buffer);
  nprove=atoi(buffer);
  auto_man=0;
  while(!flag)
  {
    while(((key=ci())!=27)&&(key!=21)&&(key!=0)&&(key!=13));
    switch (key)
    {
      case 21: exit();
               break;
      case 13: key=0;
               flag=0xFF;
               break;
      case 27: flag=0xFF;
               break;
      case 00: while(((key=ci())!=75)&&(key!=77));
               switch (key)
               {
                 case 77: auto_man=1;
                          presenta(12,24,7,screen08[12].stringa08);
                          presenta(12,41,23,screen08[14].stringa08);
                          break;
                 case 75: auto_man=0;
                          presenta(12,41,7,screen08[14].stringa08);
                          presenta(12,24,23,screen08[12].stringa08);
                          break;
               }
               break;
    }
  }
  return(key);
}

/*
================================================================================
*/

struct disp09 {
char stringa09[58];
int posizione_riga09;
int posizione_colo09;
int attributo09;
};

struct disp09 screen09[14]={
"          STIMOLATORE MULTISENSORIALE          ",1,10,23,
"                                               ",2,10,71,
"    P R O V E  D I  A P P R E N D I M E N T O   ",3,10,71,
```

```
"                                          ",4,10,71,
" NOME SOGGETTO =                              DATA        ",5,10,23,
"STIMOLATORE = ",6,11,2,
"          ",6,25,23,
"ARTO = ",6,40,2,
"               ",6,47,23,
"MODALITA` =      ",7,25,2,
"NUMERO PROVE =      ",8,22,2,
"                                          ",20,10,71,
"                                          ",21,10,71,
"                                          ",22,10,71,
};


esegui_apprendi_semplice()
{
  unsigned b,c,d;
  int  casuali[8],riga,colo1,colo2,ireturn,len,a;
  int cpaziente=27,cdata=59,next_prov;
  char *automatico,*manuale,*premi_barra,record[80];
  char *errata,*assente,*in_corso,*start,*spazi,*enter,*out_corso;
  char bell=0x07,risposta,uscita;
  char *fonte,*arisp,data_odierna[9];
  char *nuova_prova;
  nuova_prova=" VUOI RIPETERE LA PROVA DI APPRENDIMENTO (S/N) ";
  automatico=" AUTOMATICO ";
  manuale="   MANUALE  ";
  premi_barra="  PREMI LA BARRA PER INIZIALIZZARE LO STIMOLO  ";
  spazi="                                    ";
  errata="  ERRATA         ";
  assente="  ASSENTE        ";
  in_corso=" IN CORSO ";
  out_corso="          ";
  start="  PREMI UN TASTO PER PARTIRE ";
  enter="PREMI UN TASTO PER CONTINUARE";
  next_prov=0;
 while(next_prov==0)
 {
  scr_clr();
  for (a=0; a<14; a++)
  presenta(screen09[a].posizione_riga09,screen09[a].posizione_colo09,
       screen09[a].attributo09,screen09[a].stringa09);
  data_int(data_odierna);
  len=strlen(cognome)+1;
  presenta(screen09[4].posizione_riga09,cpaziente,23,cognome);
  cpaziente+=len;
  presenta(screen09[4].posizione_riga09,cpaziente,23,nome);
  presenta(screen09[4].posizione_riga09,cdata,23,data_odierna);

  switch (stim_selezionato)
  {
    case 0: fonte="   VISIVO   ";
          break;
    case 1: fonte="  UDITIVO   ";
          break;
    case 2: fonte=" SOMESTESICO";
          break;
  }
```

```
switch (arto_selezionato)
{
  case 1: arisp="  MANO DESTRA    ";
        break;
  case 2: arisp="  MANO SINISTRA   ";
        break;
  case 4: arisp="  PIEDE DESTRO    ";
        break;
  case 8: arisp="  PIEDE SINISTRO  ";
        break;
}
presenta(6,25,23,fonte);
presenta(6,47,23,arisp);
if(auto_man==0)
  presenta(7,43,23,automatico);
else
  presenta(7,43,23,manuale);
if(nprove>8)  nprove=8;
sprintf(record,"     %d     ",nprove);
presenta(8,43,23,record);
scr_rowcol(9,1);
for(a=0; a<nprove; a++)
  printf("\n              %d:             RISPOSTA:",a+1);
fasce();
if(auto_man==0)            /* --- E' STATO SCELTO IL MODO AUTOMATICO */
{
  carica_casuali(2000,5000,casuali,nprove);  /* TRA 2 E 5 SECONDI    */
  riga=10;
  colo1=20;
  colo2=44;
  presenta(21,10,71,start);
  printf("%c",bell);
  risposta=ci();
  presenta(21,10,71,spazi);
  for(a=0; a<nprove; a++)
  {
    presenta(riga,colo1,23,in_corso);
    load_timer(casuali[a]);
    trigger(10);
    if(stim_selezionato==2)   send_data_some(0x0A);
    start_timer();
    ireturn=singola_prova(casuali[a],arto_selezionato,100,500);
    if(ireturn==0)  presenta(riga,colo2,71,assente);
    if(ireturn==-1)
    {
      presenta(riga,colo2,71,errata);
      wait_uno();
    }
    if((ireturn!=0)&&(ireturn!=-1))
    {
      sprintf(record,"     %4d mS     ",ireturn);
      presenta(riga,colo2,23,record);
      wait_uno();
    }
    presenta(riga,colo1,7,out_corso);
    riga++;
    uscita=csts();
    if(uscita!=0)
```

```
      {
        uscita=ci();
        if(uscita==21) exit();
        if(uscita==27) return(27);
      }
    }
  }
  else                    /* --- E' STATO SCELTO IL MODO MANUALE   */
  {
    carica_casuali(2000,5000,casuali,nprove);  /* TRA 2 E 5 SECONDI    */
    riga=10;
    colo1=20;
    colo2=44;
    for(a=0; a<nprove; a++)
    {
      presenta(21,10,71,premi_barra);
      while((risposta=ci())!=0x20);
      presenta(21,10,71,spazi);
      presenta(riga,colo1,23,in_corso);

/* ---------------------- Ritardo casuale tra tasto e trigger ------------ */
      for(d=0; d<casuali[a]*12; d++)   /* per ritardare in modo casuale */
      {
        b = d*b;
        c = b/(d+1);
        b = b+c;
      }
/* ------------------------------------------------------------------------ */
      load_timer(casuali[a]);
      trigger(10);
      if(stim_selezionato==2)    send_data_some(0x0A);
      start_timer();
      ireturn=singola_prova(casuali[a],arto_selezionato,100,500);
      if(ireturn==0)  presenta(riga,colo2,71,assente);
      if(ireturn==-1)
      {
        presenta(riga,colo2,71,errata);
        wait_uno();
      }
      if((ireturn!=0)&&(ireturn!=-1))
      {
        sprintf(record,"      %4d mS     ",ireturn);
        presenta(riga,colo2,23,record);
        wait_uno();
      }
      presenta(riga,colo1,7,out_corso);
      riga++;
      uscita=csts();
      if(uscita!=0)
      {
        uscita=ci();
        if(uscita==21) exit();
        if(uscita==27) return(27);
      }
    }
  }
  presenta(21,10,23,nuova_prova);
  printf("%c",bell);
```

```
   risposta=ci();
   presenta(21,10,71,spazi);
   if((risposta=='n')||(risposta=='N'))   next_prov=1;
  }
 }



singola_prova(casuale,arto,lim1,lim2)
unsigned casuale;
char arto;
int  lim1,lim2;
/* ---------------------------------------------------------------------------

 Esegue il calcolo del tempo di risposta relativamente alla singola prova
 ---------------------------------------------------------------------------

 Parametri di ingresso:
 ----------------------
  casuale : e' l'intervallo temporale settato dalla routine chiamante
            nel quale l'operatore deve effettuare la risposta
    arto : e' il numero (un unico bit settato a 1) che identifica
           (mascheratura) l'arto con cui deve avvenire la risposta
 lim1,lim2 : determinano il campo di convalida della risposta;

 Parametro di uscita:   iret
 ---------------------
    cosi' configurato:

    iret =  n  --> n e` il tempo di risposta in millisecondi
    iret =  0  --> risposta assente
    iret = -1  --> risposta errata

--------------------------------------------------------------------------- */

{
  extern portim;
  unsigned iret;
  unsigned legge_timer();
  unsigned tempo;
  int  a;
  char byte;

  arto=~arto&0x0F;
  iret=0;
  while(((tempo=legge_timer())!=1)&&(iret==0))
  {
    byte=_inb(portim-2)&0x0F;
    if(byte!=0x0F)
    {
      if(byte==arto)
      {
        iret=legge_timer();
        for(a=0; a<300; a++);
        iret=casuale-iret;
        if((iret<lim1)||(iret>lim2))
          iret=-1;
      }
```

```
        else
          iret=-1;
      }
    }
    while (((tempo=legge_timer())>50)&&(tempo < casuale));
    stop_timer();
    return(iret);
}



wait(tick)
unsigned tick;
{
  unsigned letto;
  stop_timer();
  load_timer(tick);
  start_timer();
  letto=0xFFFF;
  while(letto!=1)
  {
    letto=legge_timer();
  }
  stop_timer();
}


wait_uno()              /* aspetta che venga rilasciato il tasto  */
{
  extern portim;
  char byte;
  while((byte=_inb(portim-2)&0x0F)!=0x0F);
  wait(100);
}


/*
=============================================================================
=
*/

struct disp10 {
char stringa10[58];
int posizione_riga10;
int posizione_colo10;
int attributo10;
};

struct disp10 screen10[20]={
"          STIMOLATORE MULTISENSORIALE          ",1,10,23,
"                                ",2,10,71,
"          P R O V E  D I  E S A M E          ",3,10,71,
"                                ",4,10,71,
" NOME SOGGETTO =                    DATA       ",5,10,23,
" STIMOLATORE = ",6,10,23,
"          ",6,25,23,
"   ARTO = ",6,37,23,
```

```
"              ",6,47,23,
"RISPOSTE ERRATE = ",19,10,2,
"       ",19,28,23,
"RISPOSTE ASSENTI = ",19,40,2,
"        ",19,59,23,
"VALORE MEDIO ATTUALE = ",20,5,2,
"       ",20,28,23,
"DEVIAZ. STANDARD  = ",20,39,2,
"        ",20,59,23,
"                                        ",21,10,71,
"   <ESC>  per terminare                      ",22,10,71,
"  <ENTER> per proseguire      <PrtSc> stampa schermo  ",23,10,71,
};


esame_semplice(posizione)
int  posizione;
{
  #include "math.h"
  extern char cognome[],nome[],stim_selezionato,arto_selezionato;
  extern int printer,on_off;
  int riga[20]={8,9,10,11,12,13,14,15,16,17,
          8,9,10,11,12,13,14,15,16,17};
  int colo1[20]={16,16,16,16,16,16,16,16,16,16,
          43,43,43,43,43,43,43,43,43,43};
  int colo2[20]={24,24,24,24,24,24,24,24,24,24,
           50,50,50,50,50,50,50,50,50,50};
  int dati[2],riga1=19,riga2=20,cerrate=33,cassenti=65;
  int cpaziente=27,cdata=59,ir,a,b,len,prime_due;
  int cstimolo=25,carto=47,prove_massime,nrecord;
  int casuali[200],nvalide,ireturn,ncasual,jdex,status;
  int singola_prova(),temporaneo[20],c;
  float somma_dati,somma_quadrati,quadrato;
  double somma_varia,tempor;
  double x;
  char buffer[8],record[80],data_odierna[9],ff=0x0C;
  char risposta,bell=0x07,uscita,dum[2],nuova_prova;
  char *fonte,*arisp,*mess0,*mess1,*mess2,*mess3,*mess4,*curson,*cursoff;

  mess0="  BATTI UN TASTO PER INIZIARE LE PROVE  ";
  mess1="                                       ";
  mess2="    BATTI UN TASTO PER PROSEGUIRE      ";
  mess3="      Introduci numero di prove:       ";
  mess4=" VUOI RIPETERE LA PROVA COMPLETA  (S/N) ";
  apre_temporaneo();
  curson=" ---> ";
  cursoff="      ";
  nuova_prova='S';
  nrecord=1;
  for(a=0; a<8; a++)
    buffer[a]=0;

 while (nuova_prova=='S')
 {
  if(on_off!=0)
    status=fprintf(flop,"\nPROVE SEMPLICI --- INIZIO NUOVA PROVA");
  scr_clr();
  for (a=0; a<20; a++)
```

```
presenta(screen10[a].posizione_riga10,screen10[a].posizione_colo10,
     screen10[a].attributo10,screen10[a].stringa10);

ir=8;
for(a=0; a<10; a++)
{
  sprintf(record,"        %2d.              mSec",a+1);
  presenta(ir,1,2,record);
  sprintf(record,"%2d.              mSec",a+11);
  presenta(ir,39,2,record);
  ir++;
}
fasce();
data_int(data_odierna);
len=strlen(cognome)+1;
presenta(screen10[4].posizione_riga10,cpaziente,23,cognome);
cpaziente+=len;
presenta(screen10[4].posizione_riga10,cpaziente,23,nome);
presenta(screen10[4].posizione_riga10,cdata,23,data_odierna);

switch (stim_selezionato)
{
  case 0: fonte="  VISIVO  ";
        break;
  case 1: fonte="  UDITIVO  ";
        break;
  case 2: fonte=" SOMESTESICO";
        break;
}
switch (arto_selezionato)
{
  case 1: arisp="  MANO DESTRA     ";
        break;
  case 2: arisp="  MANO SINISTRA    ";
        break;
  case 4: arisp="  PIEDE DESTRO     ";
        break;
  case 8: arisp="  PIEDE SINISTRO   ";
        break;
}
if(on_off!=0)
  status=fprintf(flop,"\nCOMBINAZIONE = %s %s",fonte,arisp);

presenta(screen10[5].posizione_riga10,cstimolo,23,fonte);
presenta(screen10[5].posizione_riga10,carto,23,arisp);
presenta_dato_integer(screen10[10].posizione_riga10,
        screen10[10].posizione_colo10,23,errate[posizione]);
presenta_dato_integer(screen10[12].posizione_riga10,
       screen10[12].posizione_colo10,23,assenti[posizione]);
presenta_dato_float(screen10[14].posizione_riga10,
         screen10[14].posizione_colo10,23,media[posizione]);
presenta_dato_float(screen10[16].posizione_riga10,
        screen10[16].posizione_colo10,23,scarto[posizione]);
presenta(21,14,23,mess3);
printf("%c",bell);
attende_char(21,50,151,2,buffer,1,dum);
prove_massime=atoi(buffer);
if(prove_massime>20) prove_massime=20;
```

```
   sprintf(record," ---------- NUMERO PROVE = %3d ---------- ",
         prove_massime);
   presenta(7,10,87,record);
   presenta(21,14,23,mess2);
   risposta=ci();
   presenta(21,14,71,mess1);

/*** INIZIALIZZA VARIABILI DI UTILIZZO **********************************/
   somma_dati=0.0;
   somma_quadrati=0.0;

   carica_casuali(2000,5000,casuali,100);
   nvalide=posizione*20;  /* posizione indica la combinazione [0..11] */
   somma_dati=0.0;
   somma_quadrati=0.0;
   jdex=0;
   ncasual=0;
   prime_due=0;
   while(jdex!=prove_massime)
   {
     presenta(riga[jdex],colo1[jdex],23,curson);
     load_timer(casuali[ncasual]);
     trigger(10);
     if(stim_selezionato==2)    send_data_some(0x0A);
     start_timer();
     ireturn=singola_prova(casuali[ncasual],arto_selezionato,100,500);
     if(ireturn==0)
     {
       if(on_off!=0)
         status=fprintf(flop,"\n%2d --- PROVA ASSENTE ---",nrecord);
       assenti[posizione]++;
       presenta_dato_integer(screen10[12].posizione_riga10,
            screen10[12].posizione_colo10,23,assenti[posizione]);
       ncasual++;
     }
     if(ireturn==-1)
     {
       if(on_off!=0)
         status=fprintf(flop,"\n%2d --- PROVA ERRATA ----",nrecord);
       errate[posizione]++;
       presenta_dato_integer(screen10[10].posizione_riga10,
        screen10[10].posizione_colo10,23,errate[posizione]);
       ncasual++;
     }
     if((ireturn!=0)&&(ireturn!=-1))
     {
       sprintf(record,"   %d ",ireturn);
       presenta(riga[jdex],colo2[jdex],2,record);
       if(on_off!=0)
           if(prime_due==0)
            status=fprintf(flop,"\n%2d  TEMPO DI RISPOSTA =  %5d mS -NO-",
                       nrecord,ireturn);
           else
            status=fprintf(flop,"\n%2d  TEMPO DI RISPOSTA =  %5d mS ***",
                       nrecord,ireturn);
       validi[nvalide]=ireturn;
       presenta(riga[jdex],colo1[jdex],7,cursoff);
       jdex++;
```

```
      nvalide++;
      somma_dati+=ireturn;
      media[posizione]=somma_dati/jdex;
      quadrato=ireturn;
      quadrato=quadrato*quadrato;
      somma_quadrati+=quadrato;
      presenta_dato_float(screen10[14].posizione_riga10,
            screen10[14].posizione_colo10,23,media[posizione]);
      x=somma_quadrati/jdex-media[posizione]*media[posizione];
      x=sqrt(x);
      scarto[posizione]=x;
      presenta_dato_float(screen10[16].posizione_riga10,
           screen10[16].posizione_colo10,23,scarto[posizione]);
      ncasual++;
    }
    if((jdex==2)&&(prime_due==0))
    {
      somma_dati=0.0;
      somma_quadrati=0.0;
      media[posizione]=0.0;
      scarto[posizione]=0.0;
      assolut[posizione]=0.0;
      varianz[posizione]=0.0;
      errate[posizione]=0;
      assenti[posizione]=0;
      presenta_dato_float(screen10[14].posizione_riga10,
             screen10[14].posizione_colo10,23,media[posizione]);
      presenta_dato_float(screen10[16].posizione_riga10,
             screen10[16].posizione_colo10,23,scarto[posizione]);
      presenta_dato_integer(screen10[10].posizione_riga10,
             screen10[10].posizione_colo10,23,errate[posizione]);
      presenta_dato_integer(screen10[12].posizione_riga10,
            screen10[12].posizione_colo10,23,assenti[posizione]);
      jdex=0;
      ncasual=0;
      nvalide=posizione*20;
      prime_due=1;
    }
    if(ncasual==100)
    {
      carica_casuali(2000,5000,casuali,100);
      ncasual=0;
    }
    uscita=csts();
    if(uscita!=0)
    {
      uscita=ci();
      if(uscita==21) exit();
      if(uscita==27) return(27);
    }
    nrecord++;
  }
  b=posizione*20;
  somma_varia=0.0;
  for (a=b,c=0; a<b+20; a++,c++)
  {
    tempor=validi[a]-media[posizione];
    somma_varia+=fabs(tempor);
```

```
      temporaneo[c]=validi[a];
    }
    somma_varia=somma_varia/20;
    assolut[posizione]=somma_varia;
    mediana(temporaneo,prove_massime,&varianz[posizione]);
    presenta(21,14,23,mess4);
    printf("%c",bell);
    risposta=ci();
    presenta(21,14,71,mess1);
    if ((risposta=='s')||(risposta=='S'))
    {
      media[posizione]=0;
      scarto[posizione]=0;
      errate[posizione]=0;
      assenti[posizione]=0;
      nuova_prova='S';
    }
    else
      nuova_prova='N';
  }
  if(on_off!=0)
  {
    status=fprintf(flop,"\n ------------------------------------");
    sprintf(record,"\nVALORI: [%.1f] [%.1f] [%.1f] [%.1f]",media[posizione],
          scarto[posizione],varianz[posizione],assolut[posizione]);
    status=fprintf(flop,"%s",record);
    status=fprintf(flop,"\n ------------------------------------");
  }
  stampa_temporaneo();
}



presenta_dato_float(iriga,icolo,attributo,valore)
int   iriga,icolo;
float valore;
char  attributo;
{
  char record[20];
  sprintf(record,"  %3.1f  ",valore);
  presenta(iriga,icolo,attributo,record);
}

presenta_dato_integer(iriga,icolo,attributo,valore)
int   iriga,icolo,valore;
char  attributo;
{
  char record[20];
  sprintf(record,"  %5d  ",valore);
  presenta(iriga,icolo,attributo,record);
}


archiviazione_fisso()
{
  extern char def_visivo[],work_visivo[];
  extern char def_uditivo[],work_uditivo[];
  extern char def_somestesico[],work_somestesico[];
```

```
extern float archivio[];
extern int  osservazioni[];
extern int  fp0,fp1;
extern int  possibili[];
int ieta,indice,index,num,a,idd;
char risposta,flag,record[80];
char *riga1,*riga2;
char *mess1,*mess2,*mess3,*mess4,*mess5,*erro1,bell=7;
riga1="                                                    ";
riga2="          A R C H I V I A Z I O N E   D A T I        ";
mess1="   STO ARCHIVIANDO I DATI RELATIVI ALLE PROVE SEMPLICI      ";
mess2=" STO ARCHIVIANDO I DATI RELATIVI ALLE PROVE DI DISCRIMINAZIONE ";
mess3="   ARCHIVIO PROVE SEMPLICI AGGIORNATO.....  ";
mess4="   ARCHIVIO PROVE DI DISCRIMINAZIONE AGGIORNATO.....  ";
mess5=" Batti un tasto qualsiasi per continuare.....  ";
erro1="   IMPOSSIBILE AGGIORNARE ARCHIVIO: DATI NON STANDARD.....  ";
scr_clr();
presenta(0,10,71,riga1);
presenta(1,10,71,riga2);
presenta(2,10,71,riga1);
presenta(22,10,71,riga1);
fasce();
flag=0;
presenta(10,10,23,mess1);
work_visivo[4]=def_visivo[4];
work_uditivo[4]=def_uditivo[4];
work_uditivo[2]=def_uditivo[2];
work_uditivo[5]=def_uditivo[5];
work_uditivo[6]=def_uditivo[6];
work_uditivo[11]=def_uditivo[11];
work_uditivo[14]=def_uditivo[14];
for(a=0; a<6; a++)
{
  if(def_visivo[a]!=work_visivo[a])
  {
    flag=255;
    printf("\nDATO DIVERSO --- VISIVO #%d",a);
  }
}
for(a=0; a<19; a++)
{
  if(def_uditivo[a]!=work_uditivo[a])
  {
    flag=255;
    printf("\nDATO DIVERSO --- UDITIVO #%d",a);
  }
}
for(a=0; a<13; a++)
{
  if(def_somestesico[a]!=work_somestesico[a])
  {
    flag=255;
    printf("\nDATO DIVERSO --- SOMESTESICO #%d",a);
  }
}
if(flag!=0)
{
  presenta(22,10,71,erro1);
```

```
    printf("%c",bell);
    risposta=ci();
    return;
  }
  carica_archivio(fp0,archivio,osservazioni,1024);
  ieta=anni;
  for(a=0; a<12; a++)
  {
    if(street[a]==0x55)
    {
      determina_stimart(a);
      indice=crea_indice(0,sesso,ieta,stim_selezionato,
                  arto_numerico,arto_numerico);
/**   printf("\nSTO AGGIORNANDO L'INDICE: %d",indice);  **/
      sprintf(record,"  PROVE SEMPLICI INDICE #%d  ",indice);
      presenta(11,10,23,record);
      aggiorna_buffer(indice,archivio,osservazioni,
            media[a],scarto[a],varianz[a],assolut[a]);  /* x,y,w,z */
    }
  }
  scrive_archivio(fp0,archivio,osservazioni,1024);
  presenta(12,10,23,mess3);
  presenta(22,10,71,mess5);
  printf("%c",bell);
  risposta=ci();
  scr_clr();
  presenta(0,10,71,riga1);
  presenta(1,10,71,riga2);
  presenta(2,10,71,riga1);
  presenta(22,10,71,riga1);
  fasce();
  presenta(10,10,23,mess2);
  if(flag!=0)
  {
    presenta(22,10,71,erro1);
    printf("%c",bell);
    risposta=ci();
    return;
  }
  carica_archivio(fp1,archivio,osservazioni,2048);
  ieta=anni;
  for(a=0; a<gia_effettuate; a++)
  {
    idd=possibilita[a];
    index=possibili[idd];
    determina_arti(index);
    indice=crea_indice(1,sesso,ieta,stim_selezionato,arto_num1,arto_num2);
/**   printf("\nSTO AGGIORNANDO L'INDICE: %d",indice);   **/
    sprintf(record,"  PROVE DI DISCRIMINAZIONE INDICE #%d  ",indice);
    presenta(11,10,23,record);
    aggiorna_buffer(indice,archivio,osservazioni,dimedia[idd],
        discarto[idd],divarianz[idd],diassolut[idd]);  /* x,y,w,z */
  }
  scrive_archivio(fp1,archivio,osservazioni,2048);
  presenta(12,10,23,mess4);
  presenta(22,10,71,mess5);
  printf("%c",bell);
  risposta=ci();
```

```
}

/*****************************************************************************/

determina_arti(parola)
int parola;
{
  stim_selezionato=(parola&64)/64;
  arto_num1=(parola&12)/4;
  arto_num2=parola&3;
}



data_int(array)
char array[];
{
  dates(array);
  inverte_data(array);
}



inverte_data(array)
char array[];
{
  char temp;
  int  a;
  for (a=0; a<2; a++)
  {
    temp = array[a];
    if(temp==0x20) temp=0x30;
    array[a] = array[a+3];
    array[a+3] = temp;
  }
}

system_op()
{
  char *mess0,*mess1,*mess2;
  char code;

 mess0="Stimolatore multisensoriale -- IEI/CNR -- Rev. 1.0 -- Febbraio 1989";
 mess1="------------------------------------------------------------";
 mess2="          Digita 'exit' per tornare al programma              ";
 scr_clr();
 presenta(1,1,23,mess0);
 presenta(2,1,23,mess1);
 presenta(3,1,23,mess2);
 scr_rowcol(4,1);
 code=exec("C:\\COMMAND.COM","");
}

/************************************************************************ EOF *****/

/********************************** File: VIDEATE1.C              */
```

```
/*    --->start:        */

struct disp14 {
char stringa14[65];
int posizione_riga14;
int posizione_colo14;
int attributo14;
};


struct disp14 fisso[28]={
"          STIMOLATORE  MULTISENSORIALE          ",1,13,23,
"                                    ",2,13,71,
"       S E S S I O N E   I N   C O R S O       ",3,13,71,
"           PROVE  DI DISCRIMINAZIONE            ",4,13,71,
"STIMOLATORE:  ",6,15,2,
" / ",6,45,7,
" SORGENTE STIMOLO A: ",7,9,2,
" / ",7,45,7,
" SORGENTE STIMOLO B: ",8,9,2,
" / ",8,45,7,
"RISPOSTA A: ",9,6,2,
" / ",9,28,7,
" / ",9,41,7,
" / ",9,55,7,
"RISPOSTA B: ",10,6,2,
" / ",10,28,7,
" / ",10,41,7,
" / ",10,55,7,
"               COMBINAZIONI GIA' EFFETTUATE:              ",11,5,71,
"                                          ",15,5,71,
"-------------------------------------------------------------",16,5,23,
"1: ESECUZIONE MISURA",17,8,2,
"2: VISUALIZZAZIONE DATI",17,36,2,
"3: ARCHIVIAZIONE DATI",18,8,2,
"4: RIPROGRAMMAZIONE STIMOLATORE",18,36,2,
" <> RIGHT/LEFT PER SELEZIONARE / <RET> Convalida    ",21,13,71,
" <ESC> menu` precedente       / ALT-Y Reinizializza ",22,13,71,
"             Introduci: 1 - 2 - 3 - 4             ",23,13,71,
};

struct display {
char dstr[65];
int  dprig;
int  dpcol;
int  dcode;
};

struct display mobile[14]={
"  VISIVO  ",6,33,0,
" UDITIVO  ",6,48,1,
"   ALTO   ",7,33,2,
"  BASSO   ",7,48,3,
"   ALTO   ",8,33,4,
"  BASSO   ",8,48,5,
" MANO DS ",9,18,6,
" MANO SN ",9,31,7,
" PIEDE DS",9,45,8,
```

```
" PIEDE SN",9,58,9,
" MANO DS ",10,18,10,
" MANO SN ",10,31,11,
" PIEDE DS",10,45,12,
" PIEDE SN",10,58,13
};

  char attr_cur=71,attr_sel=23,attr_mix=87;
  char stima,stimb,arto_stima,arto_stimb;
  char *cursore,*cspazi;
  char c_sorgenti[5]={'A','B','A','B',0};
  char c_stimoli[9]={'M','D','M','S','P','D','P','S',0};
  char c_modo[7]={0,0,0,0,0,0,0};

prove_discriminazione()
{
  extern char per_status[];
  extern char stim_selezionato,apprendarray[];
  extern char possibilita[];
  extern int  gia_effettuate;
  extern char scelta[],attr[],inside[];
  extern int  possibili[];
extern int printer;
int status;
  int  a,b,c,d,defindex[5]={0,2,6,10,14},posizione,vecchiapos;
  int  uditivo_visivo,stimoloff,puntatore1,puntatore2,esito,ccc;
  int  doffset,offseta,offsetb,offsetab,offsetbb,riga,colo,incurs;
  char risp,ch,flag,bell=0X07,opzione,selezionato,conferma,fret,flagp;
  char dato,mask,out,risposta,idd;
  char *messag,*mess1,*mess2,*spazi;
  stim_selezionato=stima=stimb=arto_stima=arto_stimb=0;
  flagp=0;
  mess1="*** IMPOSSIBILE ATTIVARE LA FUNZIONE RICHIESTA ***";
  mess2="***       PROVA  GIA'  EFFETTUATA          ***";
  spazi="                                          ";
  messag="  ERRORE SULLO STIMOLATORE SELEZIONATO  ";
  for(a=0; a<3; a++)
    apprendarray[a]=0;

/* === disegna il display da utilizzare ========================== */

display:
  for(a=0; a<14; a++)
  {
    scelta[a]=0;
    attr[a]=2;
  }
  scr_clr();
  for (a=0; a<28; a++)
    presenta(fisso[a].posizione_riga14,fisso[a].posizione_colo14,
          fisso[a].attributo14,fisso[a].stringa14);
  fasce();
  if(((per_status[0]==255)||(per_status[0]==254))&&
    ((per_status[1]==255)||(per_status[1]==254)))
  {
      presenta(20,12,23,mess1);
      printf("%c",bell);
      ch=ci();
```

```
      return;
    }
    scelta[2]=255;
    scelta[5]=255;
    scelta[6]=255;
    scelta[11]=255;
    attr[2]=71;
    attr[5]=71;
    attr[6]=23;
    attr[11]=23;
    flagp=0;
    for(a=0; a<2; a++)
    {
      if((per_status[a]!=255)&&(per_status[a]!=254))
      {
        if(flagp==0)
        {
          posizione=a;
          scelta[a]=255;
          attr[a]=23;
          flagp=255;
        }
        presenta(mobile[a].dprig,mobile[a].dpcol,attr[a],mobile[a].dstr);
      }
      else
        inside[a]=255;
    }
    for (a=2; a<14; a++)
      presenta(mobile[a].dprig,mobile[a].dpcol,attr[a],mobile[a].dstr);
    on(posizione);
    if(gia_effettuate>0)
    {
      riga=12;
      colo=5;
      for (a=0; a<gia_effettuate; a++)
      {
        scr_rowcol(riga,colo);
        idd=possibilita[a];
        dato=possibili[idd];
        crea_stringa(dato,c_modo);
        printf("%s/",c_modo);
        colo+=7;
        if(colo>65)
        {
          riga++;
          colo=5;
        }
      }
    }
    flag=0;
  /* ================================================================================*/
    while(!flag)
    {
      while(((ch=ci())!=0)&&(ch!=27)&&(ch!='1')&&(ch!='2')
                  &&(ch!=21)&&(ch!='3')&&(ch!='4')&&(ch!=13));
      switch (ch)
      {
        case 0:  while(((ch=ci())!=72)&&(ch!=75)&&(ch!=77)&&(ch!=80));
```

```
            off(posizione);
            vecchiapos=posizione;
            switch (ch)
            {
              case 72: if(posizione>1)
                       {
                        if((posizione>7)&&(posizione<=9))
                          posizione=1;
                        else
                          posizione=0;
                       }
                       break;
              case 75: if(posizione>0)
                        if(posizione==6)
                          posizione=1;
                        else
                          posizione--;
                       break;
              case 77: if(posizione<9)
                        if(posizione==1)
                          posizione=6;
                        else
                          posizione++;
                       break;
              case 80: if(posizione<2)
                       {
                        if(posizione==0)
                          posizione=6;
                        else
                          posizione=8;
                       }
                       break;
            }
            for (a=0; a<2; a++)
            {
              if((posizione==a)&&((per_status[a]==255)
                          ||(per_status[a]==254)))
                posizione=vecchiapos;
            }
            on(posizione);
            break;
case 27 : return(27);
            break;
case 21 : exit();
            break;
case '1': crea_informazioni();
            crea_possibilita(&possibilita[gia_effettuate]);
            esito=controlla_effettuati(&possibilita,gia_effettuate);
            incurs=possibilita[gia_effettuate];
            if(esito!=0)
            {
              presenta(20,12,23,mess2);
              printf("%c",bell);
              ch=ci();
              presenta(20,12,7,spazi);
            }
            else
            {
```

```
                gia_effettuate++;
                reset(100);
                if(stim_selezionato==0)
                  trasmette_comandi('V','5');
                else
                  trasmette_comandi('U','5');
                if(per_status[stim_selezionato]!=2)
                {
                    presenta(19,10,23,messag);
                    printf("%c",bell);
                    risp=ci();
                    goto display;
                }
/*              if(apprendarray[stim_selezionato]==0)
                {                                   */
                  out=esame_apprendimento();
                  if(out==21)  exit();
                  if(out!=27)
                  {
                    if(out==1)                  /* esame */
                    {
                      if(esame_discriminazione(incurs)!=27)
                        apprendarray[stim_selezionato]=0xFF;
                      else
                        gia_effettuate--;
                    }
                    else
                    {                           /* apprendimento */
                      if(apprendimento_discriminazione()!=27)
                      {
                        if(esame_discriminazione(incurs)!=27)
                          apprendarray[stim_selezionato]=0xFF;
                        else
                          gia_effettuate--;
                      }
                      else
                        gia_effettuate--;
                    }
                    flag=0x55;
                  }
                  else
                  {
                    flag=0xFF;
                  }
/*              }
                else
                {
                  if(esame_discriminazione(incurs)!=27)
                    apprendarray[stim_selezionato]=0xFF;
                  flag=0x55;
                }                   */
              }
              break;
        case '2': visualizza_dati('D');
              flag=0x55;
              break;
        case '3': if(scelta_archiviazione(1)==27)
                  flag=0x55;
```

```
                else
                  flag=0xFF;
                break;
        case '4': crea_informazioni();
                riprogramma_stimolatore();
                flag=0x55;
                break;
        case 21:  flag=0xFF;
                break;
        case 13:  switch (posizione)
                {
                  case 0:
                  case 1: disabilita_old(0,1,posizione);
                        break;
                  case 2: disabilita_old(2,3,2);
                        disabilita_old(4,5,4);
                        abilita(5);
                        break;
                  case 3: disabilita_old(2,3,3);
                        disabilita_old(4,5,5);
                        abilita(4);
                        break;
                  case 6: disabilita_old(6,9,6);
                        disabilita_old(10,13,11);
                        abilita(11);
                        break;
                  case 7: disabilita_old(6,9,7);
                        disabilita_old(10,13,10);
                        abilita(10);
                        break;
                  case 8: disabilita_old(6,9,8);
                        disabilita_old(10,13,13);
                        abilita(13);
                        break;
                  case 9: disabilita_old(6,9,9);
                        disabilita_old(10,13,12);
                        abilita(12);
                        break;
                 default: disabilita_old(10,13,posizione);
                        break;
                }
               on(posizione);
               break;
     }
   }
   if(flag==0x55) goto display;
   return(0);
}

/* ------------------------------------------------------------------------------ */

crea_informazioni()
{
  extern char stim_selezionato,stima,stimb,arto_stima,arto_stimb;
  extern char scelta[];
  int a;
  for (a=0; a<14; a++)
  {
```

```
    if(scelta[a]!=0)
    {
      switch (a)
      {
        case 0:
        case 1: stim_selezionato=a;
                break;
        case 2: stima=0;
                stimb=1;
                break;
        case 3: stima=1;
                stimb=0;
                break;
        case 4:
        case 5: stimb=a-4;
                break;
        case 6: arto_stima=0;
                arto_stimb=1;
                break;
        case 7: arto_stima=1;
                arto_stimb=0;
                break;
        case 8: arto_stima=2;
                arto_stimb=3;
                break;
        case 9: arto_stima=3;
                arto_stimb=2;
                break;
       default: arto_stimb=a-10;
                break;
      }
    }
  }
}

/* ========================================================================== */

on(indice)
int indice;
{
  extern char attr[];
  char attributo;
  attributo=attr[indice]|71|128;
  presenta(mobile[indice].dprig,mobile[indice].dpcol,attributo,
        mobile[indice].dstr);
}
/* ========================================================================== */

tag(indice)
int indice;
{
  extern char attr[];
  char attributo;
  attributo=attr[indice]|23;
  presenta(mobile[indice].dprig,mobile[indice].dpcol,attributo,
        mobile[indice].dstr);
}
/* ========================================================================== */
```

```
off(indice)
int indice;
{
  extern char attr[];
  char attributo;
  attributo=attr[indice];
  presenta(mobile[indice].dprig,mobile[indice].dpcol,attributo,
        mobile[indice].dstr);
}

/* ========================================================================== */

disabilita_old(indice1,indice2,indice)
int indice1,indice2,indice;
{
  extern char scelta[];
  extern char attr[];
  int a;
  for(a=indice1; a<=indice2; a++)
  {
    scelta[a]=0;
    attr[a]=2;
    off(a);
  }
  scelta[indice]=255;
  attr[indice]=23;
}

/* ========================================================================== */

abilita(indice)
int indice;
{
  extern char scelta[];
  extern char attr[];
  scelta[indice]=255;
  attr[indice]=23;
  tag(indice);
}

/* ========================================================================== */

crea_possibilita(byte)
char *byte;
{
  extern char stim_selezionato;
  extern int  possibili[];
  char a,b;
  char forma;

  forma=0;
  forma+=stim_selezionato*64;
  forma+=stima*32;
  forma+=stimb*16;
  forma+=arto_stima*4;
  forma+=arto_stimb;
  for(a=0; a<8; a++)
```

```
    if(forma==possibili[a])
      forma=a;
  *byte=forma;
}
/* ============================================================================ */

controlla_effettuati(byte,numero)
char *byte;
int  numero;
{
extern int printer;
  int a;
  char dato;
  if(numero==0)  return(0);
  dato=*(byte+numero);
  for (a=0; a<numero; a++)
  {
    if(*(byte+a)==dato)
      return(255);
  }
  return(0);
}

/* ============================================================================ */

apprendimento_discriminazione()
{
   char out,record[80];
   out=numero_tipo(1);
   if(out==27)  return(27);
   if(esegui_apprendi_discriminazione()==27)   return(27);
   return(0);
}

/*
============================================================================
*/

struct disp19 {
char stringa19[58];
int posizione_riga19;
int posizione_colo19;
int attributo19;
};

struct disp19 screen19[9]={
"          STIMOLATORE MULTISENSORIALE          ",1,10,23,
"            E S E C U Z I O N E              ",2,10,71,
"    P R O V E  D I  A P P R E N D I M E N T O    ",3,10,71,
"             DISCRIMINAZIONE                 ",4,10,71,
"MODALITA' =     ",6,25,2,
"NUMERO PROVE =     ",8,22,2,
"                                ",20,10,71,
" CASO PRESCELTO =                          ",21,10,71,
"                                ",22,10,71,
};
```

```
esegui_apprendi_discriminazione()
{
  extern int auto_man,nprove;
  extern int gia_effettuate;
  extern char possibilita[];
  extern int  possibili[];
  int  a,casuali[100],riga,colo1,colo2,ireturn,valore,next_prov;
  int  n_consa,n_consb,soglia=3500;  /*  ((5000-2000)/2)+2000  */
  char *automatico,*manuale,*premi_barra,*comman,record[80];
  char *errata,*assente,*start,*spazi,*spaz1;
  char *enter,*out_corso;
  char bell=0x07,risposta,dato,uscita,idd;
  char *nuova_prova;
  nuova_prova=" VUOI RIPETERE LA PROVA DI APPRENDIMENTO (S/N) ";
  automatico=" AUTOMATICO ";
  manuale="   MANUALE  ";
  comman="      1 : stimolo A    /    2 : stimolo B      ";
  spazi="                                          ";
  errata="  ERRATA          ";
  assente="  ASSENTE         ";
  out_corso="          ";
  start="  PREMI UN TASTO PER PARTIRE ";
  spaz1="                          ";
  enter="PREMI UN TASTO PER CONTINUARE";
  next_prov=0;
 while(next_prov==0)
 {
  scr_clr();
  for (a=0; a<9; a++)
  presenta(screen19[a].posizione_riga19,screen19[a].posizione_colo19,
       screen19[a].attributo19,screen19[a].stringa19);
  if(auto_man==0)
    presenta(6,43,23,automatico);
  else
    presenta(6,43,23,manuale);
  idd=possibilita[gia_effettuate-1];
  dato=possibili[idd];
  crea_stringa(dato,c_modo);
  presenta(21,28,71,c_modo);
  if(nprove>8)   nprove=8;
  sprintf(record,"      %d      ",nprove);
  presenta(8,43,23,record);
  scr_rowcol(9,1);
  for(a=0; a<nprove; a++)
    printf("\n              %d:             RISPOSTA:",a+1);
  fasce();
  if(auto_man==0)             /* --- E' STATO SCELTO IL MODO AUTOMATICO */
  {
    carica_casuali(2000,5000,casuali,100);   /* TRA 2 E 5 SECONDI     */
    riga=10;
    colo1=20;
    colo2=44;
    presenta(21,35,23,start);
    printf("%c",bell);
    risposta=ci();
    presenta(21,35,71,spaz1);
    n_consa=n_consb=0;
    for(a=0; a<nprove; a++)
```

```
  {
    valore=casuali[a];
    if(valore>soglia)
    {
      if(n_consb>=3)
      {
        ireturn=esegui_ab('A',riga,colo1,casuali[a],stima,arto_stima);
        n_consb=0;
        n_consa++;
      }
      else
      {
        ireturn=esegui_ab('B',riga,colo1,casuali[a],stimb,arto_stimb);
        n_consb++;
      }
    }
    else
    {
      if(n_consa>=3)
      {
        ireturn=esegui_ab('B',riga,colo1,casuali[a],stimb,arto_stimb);
        n_consa=0;
        n_consb++;
      }
      else
      {
        ireturn=esegui_ab('A',riga,colo1,casuali[a],stima,arto_stima);
        n_consa++;
      }
    }
    if(ireturn==0)  presenta(riga,colo2,71,assente);
    if(ireturn==-1)
    {
      presenta(riga,colo2,71,errata);
      wait_uno();
    }
    if((ireturn!=0)&&(ireturn!=-1))
    {
      sprintf(record," %d msecondi ",ireturn);
      presenta(riga,colo2,23,record);
      wait_uno();
    }
    presenta(riga,colo1,7,out_corso);
    riga++;
    uscita=csts();
    if(uscita!=0)
    {
      uscita=ci();
      if(uscita==21) exit();
      if(uscita==27) return(27);
    }
  }
  presenta(21,10,71,enter);
  printf("%c",bell);
  risposta=ci();
  presenta(21,10,71,spaz1);
}
else                    /* --- E' STATO SCELTO IL MODO MANUALE    */
```

```
    {
      carica_casuali(2000,5000,casuali,100);  /* TRA 2 E 5 SECONDI    */
      riga=10;
      colo1=20;
      colo2=44;
      for(a=0; a<nprove; a++)
      {
        presenta(21,11,71,comman);
        while(((risposta=ci())!='1')&&(risposta!='2'));
        presenta(21,11,71,spazi);
        if(risposta=='1')
          ireturn=esegui_ab('A',riga,colo1,casuali[a],stima,arto_stima);
        else
          ireturn=esegui_ab('B',riga,colo1,casuali[a],stimb,arto_stimb);
        if(ireturn==0)  presenta(riga,colo2,71,assente);
        if(ireturn==-1)
        {
          presenta(riga,colo2,71,errata);
          wait_uno();
        }
        if((ireturn!=0)&&(ireturn!=-1))
        {
          sprintf(record," %d msecondi ",ireturn);
          presenta(riga,colo2,23,record);
          wait_uno();
        }
        presenta(riga,colo1,7,out_corso);
        riga++;
        uscita=csts();
        if(uscita!=0)
        {
          uscita=ci();
          if(uscita==21) exit();
          if(uscita==27) return(27);
        }
      }
    }
    presenta(21,10,23,nuova_prova);
    printf("%c",bell);
    risposta=ci();
    presenta(21,10,71,spazi);
    if((risposta=='n')||(risposta=='N'))   next_pro  1;
   }
}

/*
==========================================================================
=
*/

struct disp20 {
char stringa20[58];
int posizione_riga20;
int posizione_colo20;
int attributo20;
};

struct disp20 screen20[18]={
```

```
"          STIMOLATORE MULTISENSORIALE          ",1,10,23,
"                                    ",2,10,71,
"          P R O V E  D I  E S A M E          ",3,10,71,
"             DISCRIMINAZIONE                  ",4,10,71,
" NOME SOGGETTO =                    DATA      ",5,10,23,
" COMBINAZIONE= ",6,11,2,
"          ",6,26,23,
"RISPOSTE ERRATE = ",19,10,2,
"       ",19,28,23,
"RISPOSTE ASSENTI = ",19,40,2,
"       ",19,59,23,
"VALORE MEDIO ATTUALE = ",20,5,2,
"       ",20,28,23,
" DEVIAZ. STANDARD = ",20,39,2,
"       ",20,59,23,
"                                    ",21,10,71,
"   <ESC>  per terminare                  ",22,10,71,
"   <ENTER> per proseguire      <PrtSc> stampa schermo  ",23,10,71,
};

   char valstim;   /* indica il tipo di stimolo trasmesso */

esame_discriminazione(posizione)
int  posizione;
{
   #include "math.h"
   extern int divalidi[],dierrate[],diassenti[];
   extern float dimedia[],discarto[],divarianz[],diassolut[];
   extern char cognome[],nome[],stim_selezionato;
   extern int  gia_effettuate,flop,on_off;
   extern char possibilita[];
   extern int  possibili[];
   int riga[40]={8,9,10,11,12,13,14,15,16,17,8,9,10,11,12,13,14,15,16,17,
            8,9,10,11,12,13,14,15,16,17,8,9,10,11,12,13,14,15,16,17};
   int colo1[40]={8,8,8,8,8,8,8,8,8,8,25,25,25,25,25,25,25,25,25,25,
            42,42,42,42,42,42,42,42,42,42,59,59,59,59,59,59,59,59,59,59};
   int colo2[40]={14,14,14,14,14,14,14,14,14,14,31,31,31,31,31,31,31,31,31,31,
            48,48,48,48,48,48,48,48,48,48,65,65,65,65,65,65,65,65,65,65};
   int dati[2],riga1=19,riga2=20,cerrate=33,cassenti=65;
   int cpaziente=27,cdata=59,ir,a,b,len;
   int cstimolo=25,carto=47,soglia=3500,n_consa,n_consb,valore;
   int c[uali[200],nvalide,ireturn,ncasual,jdex,jdexa,jdexb;
   int si[gola_prova(),prime_due,nrecord,status,c,tempor[neo[40];
   float somma_dati,somma_quadrati,quadrato,tempor;
   double x,somma_varia;
   char record[80],data_odierna[9],uscita,nuova_prova,ff=12;
   char risposta,bell=0x07,dato,info[10]={0,0,0,0,0,0,0,0,0,0},idd;
   char *fonte,*arisp,*mess0,*mess1,*mess2,*mess4,*cursoff;

   mess0="BATTI UN TASTO PER INIZIARE LE PROVE";
   mess1="                                    ";
   mess2="   BATTI UN TASTO PER PROSEGUIRE    ";
   mess4="  VUOI RIPETERE LA PROVA COMPLETA  (S/N)  ";
   apre_temporaneo();
   cursoff="      ";
   nuova_prova='S';
   nrecord=1;
/***** status=fprintf(printer,"COMBINAZIONE/INDICE CORRENTE> %d",posizione); */
```

```
  while(nuova_prova=='S')
  {
    if(on_off!=0)
      status=fprintf(flop,"\nDISCRIMINAZIONE --- INIZIO NUOVA PROVA");
    prepara_video(cpaziente,cdata,data_odierna);
    idd=possibilita[gia_effettuate-1];
    dato=possibili[idd];
    crea_stringa(dato,c_modo);
    if(stim_selezionato==0)
    {
      sprintf(record,"VISIVO - %s",c_modo);
      presenta(6,26,23,record);
      if(on_off!=0)
        status=fprintf(flop,"\nVISIVO - COMBINAZIONE = %s",c_modo);
    }
    else
    {
      sprintf(record,"UDITIVO - %s",c_modo);
      presenta(6,26,23,record);
      if(on_off!=0)
        status=fprintf(flop,"\nUDITIVO - COMBINAZIONE = %s",c_modo);
    }
    presenta_valori(posizione);
    tasto_start(mess0,mess1);

/*** INIZIALIZZA VARIABILI DI UTILIZZO **********************************/

    somma_dati=0.0;
    somma_quadrati=0.0;

    carica_casuali(2000,5000,casuali,200);
    nvalide=posizione*40;  /* posizione indica la combinazione [0..127] */
    n_consa=n_consb=0;
    jdex=jdexa=jdexb=0;
    ncasual=0;
    prime_due=0;
    while((jdexa!=20)||(jdexb!=20))
/*  while((jdexa!=3)||(jdexb!=3))          <<<<<<<<<<< */
    {
      valore=casuali[ncasual];
      if(valore>soglia)
      {
        if(n_consb>=3)
        {
          ireturn=esegui_ab('A',riga[jdex],colo1[jdex],casuali[ncasual],
                    stima,arto_stima);
          n_consb=0;
          n_consa++;
        }
        else
        {
          ireturn=esegui_ab('B',riga[jdex],colo1[jdex],casuali[ncasual],
                    stimb,arto_stimb);
          n_consb++;
          n_consa=0;
        }
      }
```

```
    else
    {
      if(n_consa>=3)
      {
        ireturn=esegui_ab('B',riga[jdex],colo1[jdex],casuali[ncasual],
                    stimb,arto_stimb);
        n_consa=0;
        n_consb++;
      }
      else
      {
        ireturn=esegui_ab('A',riga[jdex],colo1[jdex],casuali[ncasual],
                    stima,arto_stima);
        n_consa++;
        n_consb=0;
      }
    }
    valuta(ireturn,nrecord,&ncasual,posizione);
    if((ireturn!=0)&&(ireturn!=-1))
    {
      if(((valstim=='A')&&(jdexa<20))||((valstim=='B')&&(jdexb<20)))
/* <<<<<< if(((valstim=='A')&&(jdexa<3))||((valstim=='B')&&(jdexb<3)))  */
      {
        sprintf(record,"  %d ",ireturn);
        presenta(riga[jdex],colo2[jdex],36,record);
        if(on_off!=0)
          if(prime_due==0)
status=fprintf(flop,"\n%2d STIMOLO %c - TEMPO DI RISPOSTA =  %5d mS -NO-",
            nrecord,valstim,ireturn);
          else
status=fprintf(flop,"\n%2d STIMOLO %c - TEMPO DI RISPOSTA =  %5d mS ***",
            nrecord,valstim,ireturn);
        presenta(riga[jdex],colo1[jdex],7,cursoff);
        if(valstim=='A')
        {
          jdexa++;
          sprintf(info,"  A%2d ",jdexa);
          presenta(riga[jdex],colo1[jdex],23,info);
        }
        else
        {
          jdexb++;
          sprintf(info,"  B%2d ",jdexb);
          presenta(riga[jdex],colo1[jdex],71,info);
        }
        divalidi[nvalide]=ireturn;
        nvalide++;
        jdex++;
        somma_dati+=ireturn;
        dimedia[posizione]=somma_dati/jdex;
        quadrato=ireturn;
        quadrato=quadrato*quadrato;
        somma_quadrati+=quadrato;
        presenta_dato_float(screen20[12].posizione_riga20,
            screen20[12].posizione_colo20,23,dimedia[posizione]);
        x=somma_quadrati/jdex-dimedia[posizione]*dimedia[posizione];
        x=sqrt(x);
        discarto[posizione]=x;
```

```
        presenta_dato_float(screen20[14].posizione_riga20,
              screen20[14].posizione_colo20,23,discarto[posizione]);
        ncasual++;
       }
       if((jdex==2)&&(prime_due==0))
       {
        somma_dati=0.0;
        somma_quadrati=0.0;
        dimedia[posizione]=0.0;
        discarto[posizione]=0.0;
        diassolut[posizione]=0.0;
        divarianz[posizione]=0.0;
        dierrate[posizione]=0;
        diassenti[posizione]=0;
        presenta_valori(posizione);
        jdex=jdexa=jdexb=0;
        ncasual=0;
        nvalide=posizione*40;
        prime_due=1;
       }
     }
     if(ncasual==200)
     {
      carica_casuali(2000,5000,casuali,200);
      ncasual=0;
     }
     uscita=csts();
     if(uscita!=0)
     {
      uscita=ci();
      if(uscita==21) exit();
      if(uscita==27) return(27);
     }
     nrecord++;
   }
   b=posizione*40;
   somma_varia=0.0;
   for (a=b,c=0; a<b+40; a++,c++)
   {
    tempor=divalidi[a]-dimedia[posizione];
    somma_varia+=fabs(tempor);
    temporaneo[c]=divalidi[a];
   }
   somma_varia=somma_varia/40;
   diassolut[posizione]=somma_varia;
   mediana(temporaneo,40,&divarianz[posizione]);
   presenta(21,14,23,mess4);
   printf("%c",bell);
   risposta=ci();
   presenta(21,14,71,mess1);
   if ((risposta=='s')||(risposta=='S'))
   {
    dimedia[posizione]=0.0;
    discarto[posizione]=0.0;
    dierrate[posizione]=0;
    diassenti[posizione]=0;
    nuova_prova='S';
   }
```

```
    else
      nuova_prova='N';
  if(on_off!=0)
  {
    status=fprintf(flop,"\n ------------------------------------");
    sprintf(record,"\nVALORI: [%.1f] [%.1f] [%.1f] [%.1f]",dimedia[posizione],
          discarto[posizione],divarianz[posizione],diassolut[posizione]);
    status=fprintf(flop,"%s",record);
    status=fprintf(flop,"\n ------------------------------------");
  }
  stampa_temporaneo();
 }
}

/* ================================================================== */

tasto_start(mess0,mess1)
char *mess0,*mess1;
{
  char risposta,bell=7;
  presenta(21,14,23,mess0);
  printf("%c",bell);
  risposta=ci();
  presenta(21,14,71,mess1);
}

/* ================================================================== */

valuta(ireturn,nrecord,ncasual,posizione)
int nrecord,ireturn,posizione;
int *ncasual;
{
   extern int on_off,flop;
   extern int diassenti[],dierrate[];
   int status;
   if(ireturn==0)
   {
     if(on_off!=0)
       status=fprintf(flop,"\n%2d STIMOLO %c --- PROVA ASSENTE ---",
                 nrecord,valstim);
     diassenti[posizione]++;
     presenta_dato_integer(screen20[10].posizione_riga20,
           screen20[10].posizione_colo20,23,diassenti[posizione]);
     *ncasual++;
   }
   if(ireturn==-1)
   {
     if(on_off!=0)
       status=fprintf(flop,"\n%2d STIMOLO %c --- PROVA ERRATA ----",
                 nrecord,valstim);
     dierrate[posizione]++;
     presenta_dato_integer(screen20[08].posizione_riga20,
       screen20[08].posizione_colo20,23,dierrate[posizione]);
     *ncasual++;
   }
}

/* ================================================================== */
```

```
prepara_video(cpaziente,cdata,data_odierna)
int cpaziente,cdata;
char data_odierna[];
{
  extern char cognome[],nome[];
  int a,ir,len;
  char record[80];
  scr_clr();
  for (a=0; a<18; a++)
  presenta(screen20[a].posizione_riga20,screen20[a].posizione_colo20,
       screen20[a].attributo20,screen20[a].stringa20);
  ir=8;
  for(a=0; a<10; a++)
  {
    sprintf(record,
" %2d          %2d          %2d          %2d          ",
a+1,a+11,a+21,a+31);
    presenta(ir,4,36,record);
    ir++;
  }
  fasce();
  data_int(data_odierna);
  len=strlen(cognome)+1;
  presenta(screen20[4].posizione_riga20,cpaziente,23,cognome);
  cpaziente+=len;
  presenta(screen20[4].posizione_riga20,cpaziente,23,nome);
  presenta(screen20[4].posizione_riga20,cdata,23,data_odierna);
}

/* ================================================================== */

presenta_valori(indice)
int indice;
{
  extern int dimedia[],discarto[],dierrate[],diassenti[];
  presenta_dato_integer(screen20[08].posizione_riga20,
        screen20[08].posizione_colo20,23,dierrate[indice]);
  presenta_dato_integer(screen20[10].posizione_riga20,
        screen20[10].posizione_colo20,23,diassenti[indice]);
  presenta_dato_float(screen20[12].posizione_riga20,
        screen20[12].posizione_colo20,23,dimedia[indice]);
  presenta_dato_float(screen20[14].posizione_riga20,
        screen20[14].posizione_colo20,23,discarto[indice]);
}

/* ================================================================== */

esegui_ab(tipo,riga,colo,random,stimolo,artoselezionato)
char stimolo,tipo,artoselezionato;
int riga,colo,random;
{
  extern char stim_selezionato,work_visivo[];
  extern char work_uditivo[],diver_toni[];
  char *in_corso,artosel;
  int  ireturn,lind1,lind2;
  valstim=tipo;
  if(tipo=='A')
```

```
    in_corso="STIM A";
  else
    in_corso="STIM B";
  presenta(riga,colo,23,in_corso);
  if(stim_selezionato==0)
  {
    work_visivo[4]=stimolo+1;
    trasmette_comandi('V','5');
  }
  else
  {
    lind1=(stimolo*2);
    lind2=(stimolo*2)+1;
    work_uditivo[4]=4;
    work_uditivo[2]=2;
    work_uditivo[14]=19;
    work_uditivo[5]=diver_toni[lind1];
    work_uditivo[6]=diver_toni[lind2];
    if(stimolo==0)
    {
      work_uditivo[11]=06;
    }
    else
    {
      work_uditivo[11]=02;
    }
    trasmette_comandi('U','5');
  }
  switch (artoselezionato)
  {
    case 0: artosel=1;
          break;
    case 1: artosel=2;
          break;
    case 2: artosel=4;
          break;
    case 3: artosel=8;
          break;
  }
  load_timer(random);
  trigger(10);
  start_timer();
  ireturn=singola_prova(random,artosel,100,700);
  return(ireturn);
}

/* ========================================================================== */

crea_stringa(byte,stringa)
char byte;
char stringa[];
{
  char mask;
  int  doffset,offseta,offsetb,offsetab,offsetbb;
  mask=byte&0x40;
  doffset=0;
  if(mask!=0) doffset+=2;
  mask=byte&0x20;
```

```
  if(mask!=0)
    offseta=doffset+1;
  else
    offseta=doffset;
  mask=byte&0x10;
  if(mask!=0)
    offsetb=doffset+1;
  else
    offsetb=doffset;
  mask=byte&0x0C;
  offsetab=(mask/4)*2;
  offsetbb=(byte&0x03)*2;
  sprintf(stringa,"%c%c%c%c%c%c",c_sorgenti[offseta],
  c_sorgenti[offsetb],c_stimoli[offsetab],c_stimoli[offsetab+1],
  c_stimoli[offsetbb],c_stimoli[offsetbb+1]);
}

/************************************************************** EOF *****/

/********************************** File: VISIVO.C              */


/*-----------------------------------------------------------------

      MODULO DI GESTIONE STIMOLATORE VISIVO: contiene tutte le
      routine dedicate al normale funzionamento e controllo
      dello stimolatore visivo.

 -----------------------------------------------------------------
*/

struct disp {
char stringa[40];
int posr;
int posc;
int attr;
};

struct disp video[8]={
"PROGRAMMAZIONE STIMOLATORE VISIVO",1,20,30,
"Selezione tipo stimolatore",4,4,15,
"Selezione tipo di stimolo",7,4,15,
"Selezione lato stimolazione",11,4,15,
"Selezione durata stimolo",16,4,15,
" Premere le freccie dir. per individ.  ",21,4,30,
" Premere la freccia <- per selezionare ",22,4,64,
" <RETURN> PER MEMORIZZARE I PARAMETRI  ",23,4,64
};

struct dispv {
char stringa[10];
int posr;
int posc;
int appar;
char cod;
};

struct dispv videos[17]={
```

```
"LED",4,40,1,0x00,
"GOGGLE",5,40,1,0x01,
"APPEAR",7,40,2,0x00,
"FLASH",8,40,2,0x01,
"REV",9,40,2,0x02,
"INIBITA",11,40,3,0x00,
"DESTRO",12,40,3,0x01,
"SINISTRO",13,40,3,0x02,
"BITONALE"14, 40,3,0x03,
"5",16,40,4,0x00,
"10",17,40,4,0x01,
"15",18,40,4,0x02,
"50",19,40,4,0x03,
"100",16,55,4,0x04,
"200",17,55,4,0x04,
"300",18,55,4,0x05,
"500",19,55,4,0x06
};

/*-----------------------------------------------------------------*/

    int     a,i,b,d,e,f,j,c,to,tres;
    char    val,addr,out,at,br,ind,dato,check,masc,Cbit,ud,dati[20];

/*-----------------------------------------------------------------*/

visivo(param)  /* Pacchetto software di gestione stimolatore visivo
                  1 = riconoscimento stimolatore visivo
                  2 = ricopia gli attributi standard in working
                  3 = non si sa' bene a cosa serva....
                  4 = permette di editare i parametri
                  5 = trasmette dati allo stimolatore.

  N.B. La cella dell'array per_status utilizzata per lo stimolatore
       visivo e' la cella 0.                                    */

/* ---------------------------------------------------------------- */
int   param;
{
    extern char def_visivo[];
    extern char work_visivo[];
    extern char per_status[];
    extern char def_attr[];
    extern char attr[];
/* ---------------------------------------------------------------- */
    switch (param)
    {
      case '1':
            per_status[0]=1;
            iniz_vis();
            break;
      case '2':
            for(i=0;i<6;i++)  /* ricopia parametri di default */
            {
              work_visivo[i]=def_visivo[i];
/*              printf("%x\n",work_visivo[i]);              */
            }
            for(i=0;i<17;i++)  /* ricopia attributi di default */
```

```
            {
              attr[i]=def_attr[i];
            }
            break;
      case '3':
/*          for(i=0;i<6;i++)
            {
              def_visivo[i]=work_visivo[i];
              printf("%x\n",work_visivo[i]);
            }                                  */
            break;
      case '4':
            edit_visivo();
            break;
      case '5':
            per_status[0]=2;
            tx_visivo();
            break;
      default:
            break;
    }
}

/* ================================================================== */
/* ====== PROCEDURA DI AGGIORNAMENTO PARAMETRI D'USO ============== */
/* ================================================================== */

edit_visivo()
{
    extern char def_visivo[];
    extern char work_visivo[];
    extern char per_status[];
    extern char def_attr[];
    extern char attr[];
/* ------------------------------------------------------------------ */
  br=0x8F;    /* Inizializza lo schermo con le informazioni relative*/
  scr_clr();
  for(a=0;a<8;a++)
    scrivi(&video[a]);
  for(a=0;a<17;a++)
    scrivis(&videos[a]);
  d=1;
  a=0;
  b=0;
  scr_rowcol(4,40);
  brinc();
  while(b==0)
    {
    while(((c=ci())!=0)&&(c!=13));
      switch (c)
      {
        case 13 :
              e=0;
              for(a=0;a<16;a++)
              {
                if((attr[a])==0x70)
                {
                  dati[e]=videos[a].cod;
```

```
                e++;
                }
              }
              b=1;
              break;
        case 00 :
              while(((c=ci())!=80)&&(c!=72)&&(c!=75));
                switch (c)
                {
                  case 80 :  if (a!=16)
                               incr();
/*                             else            {
                               scrivis(&videos[a]);
                               a=0;
                               brinc();
                             }                 */
                             break;
                  case 72 :   if (a!=0)          {
                               scrivis(&videos[a]);
                               a--;
                               brinc();
                             }
/*                           else              {
                               scrivis(&videos[a]);
                               a=16;
                               brinc();
                             }                 */
                             break;
                  case 75:    contrev();
                             brinc();
                             break;
                }
          break;
        }
     }
/**  scr_clr();         **********/
     for (j=0; j<4; j++)  /* ricopia i dati nell'array di lavoro */
     {
       work_visivo[j+2]=dati[j];
     }
}
/* ============================================================================== */
/* ======= TRASMETTE I PARAMETRI ALLO STIMOLATORE ============================= */
/* ============================================================================== */

tx_visivo()
{
     extern char def_visivo[];
     extern char work_visivo[];
     extern char per_status[];
     extern char def_attr[];
     extern char attr[];
/* ------------------------------------------------------------------- */
/*----- Inizio trasmissione vera e propria -----*/

     ind=work_visivo[0];
     out=send_addr(ind);
/*    printf("%2x\n",ind);                    */
```

```
      if(out==2)              /* non si e` ricevuto il COA */
        per_status[0]=-2;
      check=work_visivo[0];
      for(j=1;j<6;j++)
      {
        out=send_data(work_visivo[j]);
/*      printf("%2x\n",work_visivo[j]);                    */
        check=check+work_visivo[j];
        if(out==2)              /*non si e' ricevuto il COA */
          per_status[0]=-2;
      }

      dato=~check+1;
      out=send_data(dato);
/*    printf("%2x\n",dato);                                */
      if(out==2)              /* non si e` ricevuto il COA del cs */
        per_status[0]=-2;
}
/* ================================================================== */
/* ===== STRUTTURE DI SCRITTURA STRINGHE SU SCHERMO ================= */
/* ================================================================== */

scrivi(par)
struct disp *par;
{
    scr_rowcol(par->posr,par->posc);
    scr_aputs(par->stringa,par->attr);
}

/* ================================================================== */


scrivis(par)
struct dispv *par;
{
    extern char attr[];

    scr_rowcol(par->posr,par->posc);
    scr_aputs(par->stringa,attr[a]);
}

/* ================================================================== */
/* ===== PONE IN MODO BLINK LA STRINGA RIFERITA ALLA RIGA "a" ==== */
/* ================================================================== */



brinc()     /* Salva l'attributo della stringa scritta nella posizione
               a, la riscrive in modo blink quindi ripristina
               l'attributo salvato                          */
{
    extern char attr[];

    at=attr[a];
    attr[a]=br;
    scrivis(&videos[a]);
    attr[a]=at;
    d=1;
```

```
}

/* ================================================================== */
/* ===== ATTIVA IL MODO BLINK SULLA RIGA SUCCESSIVA AD "a" ======== */
/* ================================================================== */



incr()      /* Scrive la stringa, incrementa puntatore e chiama
               la funzione BRINC di settaggio modo blink          */
{
   scrivis(&videos[a]);
   a++;
   brinc();
}
/* ================================================================== */
/* ===== RICERCA E VISUALIZZA IL NUOVO VALORE DI DEFAULT ========== */
/* ================================================================== */


contrev()    /* Esegue la ricerca del valore attuale di default in
                base al gruppo di appartenenza in cui si trova il
                cursore e setta il default alla posizione attuale  */
{
   extern char attr[];

   e=videos[a].appar;
   for(f=0;f<17;f++)
   {
     if(((videos[f].appar)==e)&&((attr[f])==0x70))
     {
       attr[f]=0x0F;
       scrivis(&videos[f]);
     }
   }
   attr[a]=0x70;
   scrivis(&videos[a]);
}

/* ================================================================== */
/* ===== ROUTINE DI RICONOSCIMENTO ================================= */
/* ================================================================== */


iniz_vis()    /* Routine di riconoscimento dello stimolatore visivo
                 tramite inizializzazione della stessa          */
{
   extern char per_status[];
   char ind,out;

   ind=0x50;
   out=send_addr(ind);
   if(out==2)            /* non si e` ricevuto il COA*/
     per_status[0]=-1;
   out=send_data(0x01);
   if(out==2)            /* non si e` ricevuto il COA*/
     per_status[0]=-1;
   out=receive();
```

```
    if(out==2)          /* non si e` ricevuto il COA*/
      per_status[0]=-1;
/*  if(ind==out)
      printf("INDIRIZZO DI RITORNO CORRETTO");
    else                                        */
    if(ind!=out)
    {
/*    printf("INDIRIZZO DI RITORNO NON CORRETTO");          */
      per_status[0]=-1;
    }
}

/* ===== END MODULO STIMOLATORE VISIVO ============================== */

/************************************************************** EOF *****/

/********************************** File: VISUALIZ.C            */



visualizza_dati(opt)
char opt;
{
  extern float archivio[];
  extern int  osservazioni[];
  extern int  fp0,fp1;
  extern int  possibili[];
  extern int  numero_prog;
  extern char nuovo_paziente;
  extern int  printer,on_off;
  extern char cognome[],nome[],giorno[],mese[],anno[];
  extern char sesso,eta[],residenza[],indirizzo[];
  extern char documento[],num_doc[],progressivo[];
  extern char type_dis,type_sem,esito_dis,esito_sem;
  extern int  divalidi[],validi[],errate[],dierrate[];
  extern int  assenti[],diassenti[];
  extern char effettuate[],possibilita[];
  extern float media[],dimedia[],scarto[],discarto[];
  extern float varianz[],assolut[],divarianz[],diassolut[];
  extern char  arto_numerico,arto_num1,arto_num2;
  extern char  arto_selezionato,stim_selezionato,street[];
  extern int   anni,gia_effettuate;
  int ieta,indice,index,num,a;
  int index,jdex,adex,a,b,c,ir,ic,aaa;
  char risposta,flag,record[80],*fonte,*resti,*resti1,data_odierna[9];
  char *mess1,*mess2,*mess3,*mess4,bell=7,chmodo[7]={0,0,0,0,0,0,0};
  mess1="            STIMOLATORE MULTISENSORIALE              ";
  mess2="                                                     ";
  mess3="       V I S U A L I Z Z A Z I O N E   D A T I       ";
  mess4=" <return> successiva                  <esc> termina  ";
  scr_clr();
  data_int(data_odierna);
  presenta(0,10,23,mess1);
  presenta(1,10,71,mess2);
  presenta(2,10,71,mess3);
  presenta(3,10,71,mess2);
  sprintf(record," %s %s %s %d",cognome,nome,data_odierna,anni);
  presenta(21,12,23,record);
```

```
presenta(23,10,71,mess4);
fasce();
if(opt=='S')
{
  carica_archivio(fp0,archivio,osservazioni,1024);
  ieta=anni;
  ir=4;
  ic=5;
  for(a=0; a<12; a++)
  {
    if(street[a]==0x55)
    {
      determina_stimart(a);
      switch (stim_selezionato)
      {
        case 0: fonte="  VISIVO  ";
                break;
        case 1: fonte="  UDITIVO  ";
                break;
        case 2: fonte=" SOMESTESICO";
                break;
      }
      switch (arto_selezionato)
      {
        case 1: resti="  MANO DESTRA    ";
                break;
        case 2: resti="  MANO SINISTRA  ";
                break;
        case 4: resti="  PIEDE DESTRO   ";
                break;
        case 8: resti="  PIEDE SINISTRO  ";
                break;
      }
      indice=crea_indice(0,sesso,ieta,stim_selezionato,
                         arto_numerico,arto_numerico);
      sprintf(record,"  [ %6.2f %6.2f %6.2f %6.2f #%d] %s %s ",
                archivio[indice],archivio[indice+1],
                archivio[indice+2],archivio[indice+3],
                osservazioni[indice],fonte,resti);
      presenta(ir,4,87,record);
      printf("\n");
      jdex=a*20;
      for(b=jdex; b<jdex+10; b++)
      {
        sprintf(record," %4d ",validi[b]);
        presenta(ir+1,ic,23,record);
        ic+=7;
      }
      ic=5;
      for(b=jdex+10; b<jdex+20; b++)
      {
        sprintf(record," %4d ",validi[b]);
        presenta(ir+2,ic,23,record);
        ic+=7;
      }
      sprintf(record,
"DATI RELATIVI AL SOGGETTO ---> [ %6.2f %6.2f %6.2f %6.2f ]      ",
                media[a],scarto[a],varianz[a],assolut[a]);
```

```
        presenta(ir+3,4,23,record);
        ir+=4;
        if(ir>16)
        {
          presenta(23,10,71,mess4);
          printf("%c",bell);
          while(((risposta=ci())!=0x0D)&&(risposta!=0x1B));
          switch (risposta)
          {
            case 0x00D: for(c=4; c<21; c++)
                        {
                          scr_rowcol(c,1);
                          scr_clrl();
                        }
                        fasce();
                        ir=4;
                        break;
            case 0X01B: return(27);
                        break;
          }
        }
      }
      ic=5;
    }
    presenta(23,10,71,mess4);
    printf("%c",bell);
    while(((risposta=ci())!=0x1B));
  }
  else
  {
    carica_archivio(fp1,archivio,osservazioni,2048);
    ieta=anni;
    ir=4;
    ic=5;
    for(a=0; a<gia_effettuate; a++)
    {
      index=possibilita[a];
      adex=possibili[index];
      determina_arti(adex);
      crea_stringa(adex,chmodo);
      indice=crea_indice(1,sesso,ieta,stim_selezionato,
                         arto_num1,arto_num2);
      sprintf(record,"[ %6.2f %6.2f %6.2f %6.2f #%d] %s",
                archivio[indice],archivio[indice+1],
                archivio[indice+2],archivio[indice+3],
                osservazioni[indice],chmodo);
      presenta(ir,4,87,record);
      printf("\n");
      jdex=index*40;
      for(aaa=jdex; aaa<jdex+40; aaa+=10)
      {
        ir++;
        for(b=aaa; b<aaa+10; b++)
        {
          sprintf(record," %4d ",divalidi[b]);
          presenta(ir,ic,23,record);
          ic+=7;
        }
```

```
        ic=5;
       }
       sprintf(record,
    "DATI RELATIVI AL SOGGETTO ---> [ %6.2f %6.2f %6.2f %6.2f ]       ",
       dimedia[index],discarto[index],divarianz[index],diassolut[index]);
       ir++;
       presenta(ir,4,23,record);
       ir++;
       if(ir>16)
       {
         presenta(23,10,71,mess4);
         printf("%c",bell);
         while(((risposta=ci())!=0x0D)&&(risposta!=0x1B));
         switch (risposta)
         {
           case 0x00D: for(c=4; c<25; c++)
                       {
                         scr_rowcol(c,1);
                         scr_clrl();
                       }
                       fasce();
                       ir=4;
                       break;
           case 0X01B: return(27);
                       break;
         }
       }
     }
     ic=5;
     presenta(23,10,71,mess4);
     printf("%c",bell);
     while(((risposta=ci())!=0x1B));
   }
}


/************************************************************** EOF *****/
```