บรรณานุกรม

1. กรมวิชาการ กระทรวงศึกษาธิการ, "วิชาหลักภาษาไทย"(เล่ม 1,เล่ม 2), กรุงเทพมหานคร โรงพิมพ์คุรุสภา, 2526.
2. วิจินต์ ภาณุพงศ์, ดร., "โครงสร้างภาษาไทย ระบบไวยกรณ์", มหาวิทยาลัยรามคำแหง.
3. อนันต์ อ่วมศาสตร์, "ไทย104 ลักษณะภาษาไทย", กรุงเทพมหานคร สำนักพิมพ์ไทยวัฒนาพานิช, 2520.
4. Aho, Alfred V. and Ulman, Jeffrey D., "Principle of Compiler Design", Addison-Wesley Publishing Co., 1977.
5. Arthur B. Pyster, Ph.D., "Compiler Design and Construction", New York: Van Nostrand Reinhold Co., 1980.
6. Ashton-Tate, "dBASE II", Ashton-Tate.
7. Hunter, Robin, "The Design and Construction of Compilers", New York:
8. S.Atre, "DATA BASE", New York: John Wiley & Sons, Inc., 1980.
9. SoftCraft, Inc., "Btrieve User's Guide", SoftCraft,Inc..

ภาคผนวก ก

ไวยากรณ์ของภาษาสอบถามภาษาไทย

1. <คำสั่งเลิก> --> 'เลิก' {'เลิก'}
2. <คำสั่งสร้าง> --> 'สร้าง' <ชื่อแฟ้ม> {'สร้าง'}
3. <ชื่อแฟ้ม> --> ชื่อแฟ้ม {ชื่อแฟ้ม}
--> null {null}
4. <คำสั่งเปิดแฟ้ม> --> 'เปิดแฟ้ม' <ชื่อแฟ้ม> {'เปิดแฟ้ม'}
5. <คำสั่งแสดงโครงสร้าง> --> 'โครงสร้าง' {'โครงสร้าง'}
6. <คำสั่งเพิ่ม> --> 'เพิ่ม' {'เพิ่ม'}
7. <คำสั่ง!แสดง> --> '!แสดง' <แสดง2> {'แสดง'}
8. <แสดง2> --> <แสดง3><แสดง4> {'ถัด'}
--> <แสดง5><แสดง6> {'สำหรับ'}
--> null { null }
9. <แสดง3> --> 'ถัด' <รายการ> {'ถัด'}
10. <แสดง4> --> 'สำหรับ' <นิพจน์> {'สำหรับ'}
--> null { null }
11. <แสดง5> --> 'สำหรับ' <นิพจน์> {'สำหรับ'}
12. <แสดง6> --> 'ถัด' <รายการ> {'ถัด'}
--> null { null }
13. <รายการ> --> เลขจำนวนเต็ม {เลขจำนวนเต็ม}
14. <นิพจน์> --> <express>{ 'ไม่','(',NUMERIC,STRING,
FIELD }
15. <express> --> <land><lors>{'ไม่','(',NUMERIC,STRING
,FIELD }
16. <land> --> <lpart><lands> {'(',NUMERIC,STRING,
FIELD }
17. <lors> --> 'หรือ' <lors> {'หรือ'}
--> null { null }

18. <lands> --> 'และ' <lands> {'และ'}
--> null { null }
19. <lpart> --> 'ไม่' <rexp> {'ไม่'}
--> <rexp> {'(',NUMERIC,STRING,FIELD}
20. <rexp> --> <term><expreses>{'(',NUMERIC,STRING,
FIELD }
21. <expreses> --> <relop><term><terms>{ '>','<','=','<>'
,'>=','<='}
--> null { null }
22. <term> --> <factor><terms>{'(',NUMERIC,STRING,
FIELD }
23. <factor> --> <subfact><factors>{'(',NUMERIC,STRING,
FIELD }
24. <factors> --> <mul-op><factors> {'*','/'}
--> null { null }
25. <subfact> --> <part><subfacts>{'(',NUMERIC,STRING,
FIELD }
26. <subfacts> --> <expo-op><part><subfacts> { '**' }
--> null { null }
27. <part> --> '(' <express> ')' {'(' }
--> NUMERIC { NUMERIC }
--> STRING { STRING }
--> FIELD { FIELD }
28. <คำสั่งแก้ไข> --> 'แก้ไข' {'แก้ไข'}

29. <คำสั่งไป> --> 'ไป' <รายการ> {'ไป'}

30. <คำสั่งปิดแฟ้ม> --> 'ปิดแฟ้ม' {'ปิดแฟ้ม'}

31. <คำสั่งล้างจอ> --> 'ล้างจอ' {'ล้างจอ'}

32. <คำสั่งช่วยด้วย> --> 'ช่วยด้วย' {'ช่วยด้วย'}

ภาคผนวก ช

ข้อความแนะนำและแสดงข้อผิดพลาด

ข้อความที่	ข้อความ
1	<คำสั่ง สร้างแฟ้ม>
2	ชื่อ ชนิด ความยาว ทศนิยม
3	ใส่ชื่อแฟ้ม:
4	เกิดความผิดพลาดในการสร้างแฟ้ม
5	ไม่สามารถเปิดแฟ้มได้
6	เกิดปัญหาในการอ่านแฟ้มโครงสร้าง
7	ยังไม่ได้เปิดแฟ้ม
8	ยังไม่ได้ปิดแฟ้มปัจจุบัน
9	เลขที่รายการมีค่ามากเกินไป
10	ไม่มีรายการเลขที่นี้
11	ผิดไวยกรณ์
12	ไม่เข้าใจความหมายของคำสั่งนี้
13	ข้อความนี้หมายถึง
14	โปรดใส่หมายเลข(สองหลัก)
15	อยู่ที่จุดสิ้นสุดแฟ้ม
16	ภาษาสอบถามภาษาไทยสำหรับฐานข้อมูลแบบสัมพันธ์
17	นำข้อมูลต่างชนิดกันมาเปรียบเทียบกัน
18	เงื่อนไขมีข้อผิดพลาด

ภาคผนวก ค

โปรแกรมของตัวประมวลผลภาษาสอบถาม

```
/* cfree.c  Start 2-20-86 date 4-21-86*/
/*For test context free command*/
#include <stdio.h>
#include <declare.h>
#include <thaitab.h>
#include <message.h>
main()
{
 int len,mlen,stcou=0,m,n;
 int st,end;
 char toke,*stpblk();
 char line[MAXLINE];
 char *s,*t;
 /* initialize system vars. */
 datarr[0].dst = 0;
 cjarr[0].cjst = 0;
 itgarr[0].itgst = 0;
 datptr = datarr;
 /*save function address*/
 tok_add[0]=&quit; tok_add[1]=&create;
 tok_add[2]=&use; tok_add[3]=&spec;
 tok_add[4]=&append; tok_add[5]=&list;
 tok_add[6]=&edit; tok_add[7]=&delete;
 tok_add[8]=&delf; tok_add[9]=&copy;
 tok_add[10]=&report; tok_add[11]=&go_to;
 tok_add[12] = &close_fil,tok_add[14]=&clear,.
 tok_add[16] = &help ;
/*----------*/
 clscr();
 setcur(page,1,0);
 tprint(mes16);
 tfeed();
 while(cquit==1)
  {
   t    = qbuff;
   ++t;
   *(t-1)  = ST;
   accept(".",MAXLINE,t);
   tfeed();
   s=stpblk(t);
   strcpy(t,s);
   mlen=strlen(t);
   if (mlen > 0)
    {
      if (stoke(&qbuff[1],&end,&toke) == 0)
       {
```

```
          strcpy(&qbuff[1],&qbuff[end+1]);
          s = stpblk(&qbuff[1]); /*skip white space*/
          strcpy(&qbuff[1],s);
          if(toke >= 70 && toke < 100)
           go(tok_add[toke-70]);
          else
           csent();
         }
        else
         {
          csent();
     }
 }
}
/*end of main */
go(fadd)
int (*fadd)();
{
 (*fadd)();
}
quit()
{
 clos_all();
 cquit = 0;
}
delf()
{
 printf("Delete file¥n¥n¥n");
}
copy()
{
 printf("Copy command¥n¥n¥n");
}
delete()
{
 printf("Delete command¥n¥n¥n");
}
report()
{
 printf("Report command¥n¥n¥n");
}
clear()
{
 clscr();
 setcur(page,1,0);
}
```

```
help()
{
 int i = 0,p=0;
 char *t,c1,c2;
 clear();
 t = hmes[p];
 while(*t != '¥0')
  {tprint(t);
   ++p;
   ++i;
   if(i == 8)
    {i = 0;
     c1 = kbbio(&c2);}
   tfeed();
   t = hmes[p];
  }
}
space(str,len)
char *str;
int len;
{
 while(len-->0)
  *str++ = ' ';
 *str = '¥0';
}
pagecal(reclen)
int reclen;
{ int ps=512,rem,mrem,mps=512;
  int i = 1;
  mrem = ps-(ps/reclen)*reclen;
  ps = ps+512;
  while(i++ <= 1 )
   {rem = ps-(ps/reclen)*reclen;
    if(mrem > rem)
     {mrem = rem;
      mps  = ps;}
    ps = ps+512;}
  return(mps);
}
trim(str)
char *str;
{char *p;
 p = str;
 if(*p == '¥0')
  return(0);
 while(*p++ != '¥0');
```

```
    --p;
    --p;
    while(*p == ' '&& p != str )
     --p;
    if(*p == ' ')
     *p = '¥0';
    else
     *(p+1) = '¥0';
}
```

```
/*Cffunc.c  3-7-86 today 4-24-86*/
#include <extern.h>
create()
{
 char *p,*q,c,ec,row,col,cmd,cof,fnm[3],flg;
 int i=0,x,fn,j,mi,reclen;
 char *stpblk();
 if(fsttab[0].fnum[0]!='¥0') /*Have current file?*/
  {tprint(mes8);
   tfeed();
   return(1);}
 q = &qbuff[1];
 if(*q != '¥0')
  {
   *(q+8) = '¥0';
   strcpy(fil_name,q);}
 else
  {accept(mes3,11,&fil_name);
   tfeed();
   q = stpblk(&fil_name);
   if(*q=='¥0')
   return(1);}
 strcpy(fil_name,q);
 trim(fil_name);
 strcpy(st_file,fil_name);
 strcat(fil_name,fil_ext);
 strcat(st_file,st_ext);
 clscr();
 clrget();
 say(mes1,1,0,0x70);
 say(mes2,3,0,0x0f);
 row = 2;
 mi = 0;
 c = RET;
 /*Reset value:fn = 1,i=1*/
 while(c != 23) /*^W*/
  {
   fn = i/4+1;
   if(fn <10)
    p = &fnm[1];
   else
    p = fnm;
   x=stci_d(p,fn,6);
   if (fn<10)
    fnm[0] = '0';
   fstp = &fsttab[fn-1];
```

```
    j = i-(i/4)*4;
    /*Move cursor to next item*/
    if((c == RET ;; (c == 0 && ec == 80))&&i<56)
     ++i;
    if( c == 0 && ec == 72 && i > 1) /*Arrow up*/
     --i;
    if(i > mi && i <= 53)
     { mi = i+3;
       if(i == 29)
        {row = 2;
         say(mes2,3,38,0x0f);}
       row = row+3;
       if(i <= 28)
         cof = 0; /*Collumn offset*/
       else
         cof = 38;
       say(fnm,row,cof,0x0f);
       strcpy(fstp->fnum,fnm);
       space(&fstp->fname,15);
       /*get(row,col,gpage,str,dec,typ,clr,bch,l,r,len)*/
       get(row,cof+3,0,&fstp->fname,0,'C',
         0x70,'¥0',cof+3,cof+17,15);
       space(&fstp->ftyp,1);
       get(row,cof+20,0,&fstp->ftyp,0,'C',
         0x70,'¥0',cof+20,cof+21,1);
       space(&fstp->flen,3);
       get(row,cof+23,0,&fstp->flen,0,'N',
         0x70,'¥0',cof+23,cof+25,3);
       space(&fstp->fdec,1);
       get(row,cof+28,0,&fstp->fdec,0,'N',
         0x70,'¥0',cof+28,cof+29,1);
       /*space(&fstp->fkey,1);
       get(row,cof+34,0,&fstp->fkey,0,'N',
         0x70,'¥0',cof+34,cof+35,1);*/
       vread(i-1,&c,&ec,&flg);
       ++fstp;
       fstp->fnum[0] = '¥0';
     }
     else
      vread(i-1,&c,&ec,&flg);
}
cl = pcl; /*set picture color*/
bg = pbg;
/*skip blank,and cal. record length*/
reclen = 0;
fstp = fsttab;
```

```
 while(fstp->fnum[0] != '¥0')
  { p = fstp->fname;
    q = stpblk(p);
    strcpy(p,q);
    trim(p);
    if(*p == '¥0')
     {fstp->fnum[0] = '¥0';
      ++fstp;
      fstp->fnum[0] = '¥0';}
    else
     {x = stcd_i(fstp->flen,&i);
      reclen = reclen+i;
      ++fstp;}
  }
 /*Create data file*/
 reclen = reclen+6;
 file_buf.rec_len = reclen;
 file_buf.page_siz = pagecal(reclen);
 file_buf.ndx_cnt = 1;
 file_buf.key_buf[0].key_pos = 1;
 file_buf.key_buf[0].key_len = 5;
 file_buf.key_buf[0].key_flag = MOD;DUP;
 stat = btrieve(B_CREATE,pos_blk,&file_buf,fil_name,0);
 if(stat != 0)
  {tprint(mes4);
   tfeed();
   fsttab[0].fnum[0] = '¥0';
   return(1);}
/*Create structure file*/
 file_buf.rec_len = STSIZ;
 file_buf.page_siz = pagecal(STSIZ);
 file_buf.ndx_cnt = 1;
 file_buf.key_buf[0].key_pos = 1;
 file_buf.key_buf[0].key_len = 3;
 file_buf.key_buf[0].key_flag = MOD;DUP;
 stat = btrieve(B_CREATE,pos_blk,&file_buf,st_file,0);
 if(stat != 0)
  {tprint(mes4);
   tfeed();
   fsttab[0].fnum[0]='¥0';
   return(1);}
/*Open structure file*/
 if(open_st() !=0)
  {tprint(mes4);
   tfeed();
   fsttab[0].fnum[0] = '¥0';
```

```
    return(1);}
/*Insert data to structure file*/
 fn = 0;
 while(fsttab[fn].fnum[0] != '¥0')
  {fstp = &fsttab[fn];
   strcpy(st_buff.fnumb,fstp->fnum);
   strcpy(st_buff.fnameb,fstp->fname);
   strcpy(st_buff.ftypb,fstp->ftyp);
   strcpy(st_buff.flenb,fstp->flen);
   strcpy(st_buff.fdecb,fstp->fdec);
   st_stat = btrieve(B_INS,st_pos,&st_buff,st_kbuff,0);
   ++fn;}
/*Close structure file*/
   if(close_st() != 0)
    {tprint(mes4);
     tfeed();
     fsttab[0].fnum[0]='¥0';
     return(1);}
 /*set file struct to empty*/
 fsttab[0].fnum[0] = '¥0';
}
use()
{
 char *p,*q,c,ec,row,col,cmd,cof,fnm[3];
 int i=0,x,fn,j,mi,reclen;
 char *stpblk();
 fstp = fsttab;
 if(fstp->fnum[0] != '¥0')
  clos_all();
 q = &qbuff[1];
 if(*q != '¥0')
  {;
   *(q+8) = '¥0';
   strcpy(fil_name,q);}
 else
  {accept(mes3,11,&fil_name);
   tfeed();
   q = stpblk(&fil_name);
   if(*q=='¥0')
     return(1);
   strcpy(fil_name,q);}
 trim(fil_name);
 strcpy(st_file,fil_name);
 strcat(fil_name,fil_ext);
 strcat(st_file,st_ext);
 if(open_all() != 0)
```

```
  {tprint(mes5);
   tfeed();
   clos_all();
   return(1);}
 /*Read struct to structure buff*/
 read_st();
 space2(data_buf,1024);
 stat = btrieve(B_GETLW,pos_blk,data_buf,key,0);
 if(stat == EOF_ERR)
  eof_flg = 'T';
 else
  eof_flg = 'F';
}
/*List structure Command*/
spec()
{char row,col,cof,mrow;
 int i,x,fn,reclen;
 cl = pcl;
 bg = pbg;
 fstp=fsttab;
 if(fstp->fnum[0] == '¥0')
  {tprint(mes7);
   tfeed();
   return(1);}
 clscr();
 reclen = 0;
 row = 2;
 cof = 0;
 mrow = row;    /* max. row */
 while(fstp->fnum[0] != '¥0')
  {
   say(fstp->fnum,row,cof+0,0x0f);
   say(fstp->fname,row,cof+3,0x0f);
   say(fstp->ftyp,row,cof+21,0x0f);
   say(fstp->flen,row,cof+23,0x0f);
   say(fstp->fdec,row,cof+27,0x0f);
   x = stcd_i(fstp->flen,&i);
      reclen = reclen+i;
   row=row+3;
   if(row > mrow)
    mrow = row;
   if(row > 20)
     {row = 2;
      cof = 38;}
   ++fstp;
 }
```

```
 setcur(page,mrow+3,0);
 tprint(mes19);
 printf(" %d¥n¥n¥n",reclen+6);
}
/*Close command*/
close_fil()
{
 clos_all();
 fsttab[0].fnum[0] = '¥0';
}
/*Append command*/
append()
{int len,len2,fn,mfn,row,i=0; /*fn = field number,mfn = max. fn*
 char c,ec,pg,bf,recnum[6],*p,loop=1,flg;
 unsigned power=1;
 if(fsttab[0].fnum[0] == '¥0')
  {tprint(mes7);
   tfeed();
   return(0);}
 /*Get highest record number*/
 stat = btrieve(B_GETHT,pos_blk,data_buf,key,0);
 p = &data_buf[4];
 cur_rec = 0;
 if(stat == 0)
  {while( i++ <= 5)
    {if(*p >= '0' && *p <= '9')
      cur_rec = cur_rec+(*(p--) - '0')*power;
     power = power*10;}
  }
cur_rec = cur_rec+1;
while(loop == 1)
 {
  space2(data_buf,1024);
  len = stcu_d(recnum,cur_rec,6);
  len2 = lencpy(&data_buf[5-len],recnum,len);
  len2 = lencpy(recnum,data_buf,5);
  recnum[5] = '¥0';
  fstp = fsttab;
  row = 4;
  pg = 0;
  clrget();
  clscr();
  fn =0;
  p = edit_buf + 8;
  pgsay(recnum,1,0,0x0f,0);
  while(fstp->fnum[0] != '¥0')
```

```
  {space(p,fstp->fln);
   cl = pbg;
   pgsay(fstp->fname,row,0,0x0f,pg);
   get(row,17,pg,p,0,'C',0x70,'¥0',0,78,fstp->fln);
   if(fstp->fln < 62)
    row = row+3;
   else
    row = row+3*(1+(fstp->fln-62)/79)+3;
   if(row > bm)
    {pg = pg+1;
     pgsay(recnum,1,0,0x0f,pg);
     row = 4;}
   mfn = fn;
   p = p+fstp->fln+1;
   ++fstp;
   ++fn; }
 c = RET;
 fn = 0;
 while(c != 23 && c != 17) /*^w ,^q*/
  {
   vread(fn,&c,&ec,&flg);
   if(c==0)
    c =ec;
   if((c==RET||c == 80)&&fn < mfn)
    ++fn;
   else if(c==72 && fn > 0)
    --fn;}
  cl = pbg;
 /*^Q exit point*/
 if( c == 17)
  {space2(data_buf,1024);
   stat = btrieve(B_GETHT,pos_blk,data_buf,key,0);
   if(stat != 0)
    eof_flg = 'T';
   else
    eof_flg = 'F';
   clear();
   return(0);
  }
 /*Tailing blank,and copy from edit_buf to data_buf */
 fstp = fsttab;
 bf = 0;
 p = edit_buf+8;
 while(fstp->fnum[0] != '¥0')
  {trim(p);
   len=lencpy(fstp->fcp,p,fstp->fln);
```

```
     if(*p != '¥0')
      bf = 1;
     if(len < fstp->fln)
      *(fstp->fcp+len) ='¥0';
     p = p+fstp->fln+1;
     ++fstp;
    }
    if(bf == 1)
     {
      cur_rec = cur_rec+1;
      *(data_buf+5) = ' ';
      stat = btrieve(B_INS,pos_blk,data_buf,key,0);
     }
 }

}

/*Space and don't terminated with null*/
space2(str,len)
char *str;
int len;
{while(len-- > 0)
  *str++ = ' ';
}
/*Copy from source to dest. stop by length limit */
/*Don't terminated with null*/
lencpy(t,s,len)
char *t,*s;
int len;
{ int ol;
  ol=0;
  while(len-- > 0 )
   {*t++ = *s++;
    ++ol;}
   return(ol);
}
/*List command*/
list()
{
 char *p,*q,*t,*s,lbuff[2000],toke,expf=0,strm[6],nxtf;
 int len,upd,end,x,nxt;
 cl = pcl;
 bg = pbg;
 fstp = fsttab;
 if (fstp->fnum[0]== '¥0')
  {tprint(mes7);
```

```
  tfeed();
  return(1);}
q = &qbuff[1];
log ='T';
expf = 0;
nxtf = 0;
nxt = 1;
/*List command parser*/
if(*q != '¥0')
 {
  while(*q != '¥0')
   {if(stoke(q,&end,&toke) == 0)
     {strcpy(&qbuff[1],q+end);
      q = &qbuff[1];
      s = stpblk(q);
      strcpy(q,s);
      if(toke == 104)
        {
         if (exp_par() !=0)
          return(1);
         expf = 1;
         strcpy(q,chrp);
        }
       else if(toke == 102)
        {
         if(*q !='¥0')
          {s = q;
           len = 0;
           while(*s >= 0x30 && *s <= 0x39)
            {++s;
             ++len;}
           x =stccpy(strm,q,len+1);
           x = stcd_i(strm,&nxt);
           strcpy(q,s);
           nxtf = 1;}
         }
        else
         {tprint(mes11);
          tfeed();
          return(1);}
      }
     else
      ++q;
   }  /*Parser loop*/
 }
/*------------------*/
```

```
if(nxtf != 1)
 {space2(data_buf,1024);
  stat = btrieve(B_GETLW,pos_blk,data_buf,key,0);
  if(stat != 0)
   {if(stat == EOF_ERR)
     eof_flg = 'T';
    return(1);}
 }
else
 {if(eof_flg == 'T')
   return(0);}
/*Check error from expeval*/
if(expf == 1)
 {if( expeval() != 0)
   return(0);
 }
/*-------------------------*/
while(stat != EOF_ERR && nxt > 0)
 {if(expf == 1)
  {fval_con();
   expeval();}
  if(nxtf == 1)
   --nxt;

  if(log == 'T')
   {
    t = lbuff;
    fstp = fsttab;
    space2(t,8);
    lencpy(t,data_buf,5);  /*Record no.*/
    t = t+6;
    *t = data_buf[5];      /*del flag*/
    /*Move field content to list output*/
    while(fstp->fnum[0] != '¥0')
     {
      t = t+2;             /*space at boudary of field*/
      len = fstp->fln;
      upd = 0;
      p = fstp->fcp; /*source*/
      q = t;         /*target*/
      while(len > 0 && *p != '¥0')
       {*q++ = *p;
        --len;
        if(gtyp(*p++)!= 2)
         ++upd;
       }
```

```
        t = t+fstp->fln-len;
        /*Insert space to list ouput for collumn correction*/
        space2(t,len+upd+2);
        t = t+len+upd;
        ++fstp;
       }
      *t = '¥0';
      tprint(lbuff);
      tfeed();
     }
    space2(data_buf,1024);
    stat = btrieve(B_GETNX,pos_blk,data_buf,key,0);
   }
   if(stat == EOF_ERR )
    eof_flg = 'T';
}
go_to()
{
 char *q,*s,*t;
 int len;
 cl = pcl;
 bg = pbg;
 fstp = fsttab;
 if (fstp->fnum[0]== '¥0')
  {tprint(mes7);
   tfeed();
   return(1);}
 q = stpblk(&qbuff[1]);
 len = 0;
 s =q;
 if(*q == '¥0')
  return(1);
 while(*s >= 0x30 && *s++ <= 0x39)
  ++len;
 if(len > 5)
  {tprint(mes9);
   tfeed();
   return(1);}
 t = key;
 space2(t,5);
 t = &key[5];
 *t = '¥0';
 --t;
 --s;
 while(len-- > 0)
  *t-- = *s--;
```

```
 space2(data_buf,1024);
 stat = btrieve(B_GETEQ,pos_blk,data_buf,key,0);
 if(stat == 0)
  {eof_flg = 'F';
   return(0);}
 eof_flg = 'T';
 if(stat == 4)
  {tprint(mes10);
   tfeed();}
 return(1);
 }
/*Edit current record*/
edit()
{int len,len2,fn,mfn,row,i=0;
 char c,ec,pg,bf,*p,loop=1,eloop,flg,delchg;
 unsigned power;
 /*File is open ?*/
 if(fsttab[0].fnum[0] == '¥0')
  {tprint(mes7);
   tfeed();
   return(0);}
 /*End of file ?*/
 if (eof_flg == 'T')
  {tprint(mes15);
   tfeed();
   return(0);
  }
while(loop == 1)
 {
  len = stccpy(edit_buf,data_buf,6);
  len = stccpy(edit_buf+6,data_buf+5,2);
  /*Copy from data_buf to edit_buf */
  fstp = fsttab;
  p = edit_buf+8;
  while(fstp->fnum[0] != '¥0')
   {
    len=stccpy(p,fstp->fcp,fstp->fln+1);
    p = p+fstp->fln+1;
    ++fstp;
   }
  /*Full screen input*/
  fstp = fsttab;
  row = 4;
  pg = 0;
  clrget();
  clscr();
```

```
fn =0;
p = edit_buf + 8;
pgsay(edit_buf,1,0,0x0f,0); /*record number*/
/*Auto get loop*/
while(fstp->fnum[0] != '¥0')
 {
  cl = pbg;
  pgsay(fstp->fname,row,0,0x0f,pg);
  get(row,17,pg,p,0,'C',0x70,'¥0',0,78,fstp->fln+1);
  if(fstp->fln < 62)
   row = row+3;
  else
   row = row+3*(1+(fstp->fln-62)/79)+3;
  if(row > bm)
   {pg = pg+1;
    pgsay(edit_buf,1,0,0x0f,pg);
    row = 4;}
  mfn = fn;
  p = p+fstp->fln+1;
  ++fstp;
  ++fn;
 }
/*Convert record no. string to value*/
p = &data_buf[4];
i = 0;
power = 1;
cur_rec = 0;
while( i++ <= 5)
  {if(*p >= '0' && *p <= '9')
    cur_rec = cur_rec+(*(p--) - '0')*power;
   power = power*10;
  }
/*------*/
c = RET;
delchg = 0; /*delete changed ?. 0 = no)*/
fn = 0;
eloop = 1;
while(eloop==1) /*^w ,^q*/
 {
  /*display delete or no delete message*/
  p = data_buf+5;
  setcur(page,0,25);
  cl = pcl;
  if(*p == '*')
   tprint("4X!C8");
  else
```

```
      tprint("      ");
     vread(fn,&c,&ec,&flg);
     if(c==0)
      c =ec;
     if((c==RET||c == 80)&&fn < mfn)
      ++fn;
     else if(c==72 && fn > 0 && cur_rec != 1)
      --fn;
     else if(c == 21)
      {/*Delete record*/
       p = data_buf+5;
       if(*p == '*')
        *p = ' ';
       else
        *p = '*';
       if(delchg == 0)
        delchg = 1;
       else
        delchg = 0;
      }
     else
      eloop=0;
   }
   cl = pbg;
  if( c != 17)
   {
    /*Tailing blank,and copy from edit_buf to data_buf */
    fstp = fsttab;
    p = edit_buf+8;
    while(fstp->fnum[0] != '¥0')
     {trim(p);
      len=lencpy(fstp->fcp,p,fstp->fln);
      if(len < fstp->fln)
       *(fstp->fcp+len) ='¥0';
      p = p+fstp->fln+1;
      ++fstp;
     }
    /*Flg = 0 mean, only move cursor*/
    if(flg ==1 || delchg == 1)
     stat = btrieve(B_UPD,pos_blk,data_buf,key,0);
    if( c== 73 || c==18)
      stat = btrieve(B_GETPR,pos_blk,data_buf,key,0);
    else if(c==81||c==3)
     {stat = btrieve(B_GETNX,pos_blk,data_buf,key,0);
      if(stat != 0)
       stat = btrieve(B_GETNX,pos_blk,data_buf,key,0);
```

```
      }
    }
   else
    loop=0;
 }
 clear();
}

/*Count up/down type*/
updowc(str)
char *str;
{int i=0;
 while(*str != '¥0')
  {if(gtyp(*str++) != 2)
    ++i;}
 return(i);
}
/*Convert numeric string to numeric value*/
fval_con()
{char str[40];
 int x;
 fstp = fsttab;
 while(fstp->fnum[0] != '¥0')
  { if(fstp->ftyp[0] == 'C' )
     {x = stccpy(str,fstp->fcp,fstp->fln+1);
      str2val(&fstp->fval,str);}
    ++fstp;}
}
/*File fuction */
 /*Open structure file*/
open_st()
{
 stat = btrieve(B_OPEN,st_pos,&st_buff,st_file,1);
 return(stat);
}
 /*close structure file*/
close_st()
{
 stat = btrieve(B_CLOSE,st_pos,&st_buff,st_kbuff,0);
 return(stat);
}
/*open data file*/
0open_dat()
{
 stat = btrieve(B_OPEN,pos_blk,&data_buf,fil_name,0);
 return(stat);
```

```
}
close_dat()
{
 stat = btrieve(B_CLOSE,pos_blk,&data_buf,fil_name,0);
 return(stat);
}
open_all()
{int err;
 if((err = open_st())!= 0)
  return(err);
 else if((err = open_dat())!=0)
  return(err);
 return(0);
}
clos_all()
{close_st();
 close_dat();
}
/*Read structure to structure buff*/
read_st()
{int x;
 char *p;
 p = data_buf+6;
 stat = btrieve(B_GETLW,st_pos,&st_buff,st_kbuff,0);
 reclen = 0;
 totfld = 0;  /*Total field*/
 fstp = fsttab;
 while(stat != EOF_ERR)
 {
  if(stat != 0)
   {tprint(mes6);
    return(1);}
  strcpy(fstp->fnum,st_buff.fnumb);
  strcpy(fstp->fname,st_buff.fnameb);
  strcpy(fstp->ftyp,st_buff.ftypb);
  strcpy(fstp->flen,st_buff.flenb);
  strcpy(fstp->fdec,st_buff.fdecb);
  x = stcd_i(fstp->flen,&fstp->fln);
  x = stcd_i(fstp->fdec,&fstp->fdc);
  fstp->fcp = p;
  reclen = reclen+fstp->fln;
  totfld = totfld+1;
  p = p+fstp->fln;
  ++fstp;
  stat = btrieve(B_GETNX,st_pos,&st_buff,st_kbuff,0);
 }
```

```
 fstp->fnum[0] = '¥0';
}
/*search command keyword in string(seach only first word)
  input
  str =string to seach
  output
  end = end pos.,toke = command token */
stoke(str,end,toke)
char *str,*toke;
int  *end;
{
 int  row,col;
 char *keywp,*ostr,spf;
  commp = comktab;
  /*search first maching character by fix. string char.*/
  while(commp->commch != *str && commp->commch != '¥0')
   {
   ++commp;
   }
  if (commp->commch == '¥0')
   return(1);
  if (commp->commch == *str)
   {
    /*search word that start with same
        character (& fix. string)*/
    row = commp->commadd;
    keywp = tok_tab[row].keyword;
    ostr  = str;
    while(*keywp != WBRK)
    { /*search next character of word */
      str = ostr;
      col = 0;
      spf = 1;
      while (spf == 1)
      {
       while (*keywp != '¥0' && *keywp == *str)
       {
          ++keywp;
          ++str;
          ++col;
       }
       /*skip space in word*/
       if (*str == ' ')
        {
         while(*str == ' ')
          {++col;
```

```
            ++str;}
          }
         else
          spf = 0;
        }
        if (*keywp == '¥0')
        {
          /*this word matched*/
          *end = col;
          *toke = tok_tab[row].token;
          return(0);
         }
        else
        {
         /*skip to next word in table*/
         ++row;
         keywp = tok_tab[row].keyword;
        }
       }/*end of search word that start with same charater*/
     }/*end if of find  first macthing charater */
  return(1);
}
```

```
/*CSENT.C Date 5-01-86*/
#include <extern.h>
/*Semi thai netural language parser*/
csent()
{
 char *p,*t,*s,tbuff[500],loop=1,stat;
 static char m[] = "GOHA]8";
 s = m;
 if (sdatval() != 0)
  {tprint(mes12);
   tfeed();
   return(0);
  }
 fstp = fsttab;
 if(fstp->fnum[0] == '¥0')
  {tprint(mes7);
   tfeed();
   return(0);
  }
 /*-------- Init parameter ----------*/
 defflg = 0;
 odtyp = ' ';
 datptr = datarr;
 sdat = datptr->dst;
 qp = &qbuff[1];
 /*-----------------------------------*/
 if(conj () != 0);
   {qp = stpblk(qp);
    pronoun();
    qp = stpblk(qp);
    tv1();
    itg();}              /*Suppanarm chaitarm*/
 tprint(qp);
 tfeed();
 t = tbuff;
 while(*s != '¥0')
  *t++ = *s++;
 *t++ = ' ';
 /*Search field and save pos. to table*/
 fldscan();
 fldptr = fldtab;
 stat = fsearch(t,&t);
 if(stat == 0)
  {
   while(loop == 1)
    {
```

```
      /*Comparative operator word search*/
      comps(t,&t);
      /*Move data value to tbuff*/
      qp = datptr->dst;
      while(qp <= datptr->dend)
       *t++ = *qp++;
      *t++ =' ';
      /*---------------*/
      odtyp = datptr->dtyp;
      ++datptr;
      sdat = datptr->dst;
      if(datptr->dtyp != '¥0')
       logs(t,&t);       /*Logical word search*/
      if (*qp == '¥0')
       loop = 0;
      else
       {
        qp = stpblk(qp);
        if((*qp >= 0x30 && *qp <= 0x39) || *qp == '.' || *qp == '
          || *qp == '¥''|| *qp == '¥"')
         defflg = 1;       /* 1 = can default*/
        else
         defflg = 0;
        stat = fsearch(t,&t);
       }
     }
    *t = '¥0';
    strcpy(&qbuff[1],tbuff);
    tprint(&qbuff[1]);
    tfeed();
    list();
   }
}
fldscan()
{
 char *p,*s,*t,loop;
 fldptr = fldtab;
 fldptr->ft = '¥0';
 p = qp;
 while(*p != '¥0')
  {
   loop = 1;
   fstp = fsttab;
   while(loop==1)
    {t = fstp->fname;
     s = p;
```

```
      while(*t == *s && *t != '¥0')
       {++s;
        ++t;}
      if(*t != '¥0')
       {++fstp;
        if(fstp->fnum[0] == '¥0')
         loop = 0;
       }
      else
       {loop = 0;
        fldptr->fst = p;
        fldptr->fend = s-1;
        fldptr->ft = fstp->ftyp[0];
        fldptr->fp = fstp->fname;
        ++fldptr;
        fldptr->ft = '¥0';
        p = s-1;}
     }
    ++p;
  }
 }
/*Field search*/
fsearch(tt,tt2)
char *tt;
unsigned *tt2;
{
 char *s,*t,*s2,lop,loop,eflg;
 int cou;
 s = qp;
 cou = 0;
 while(fldptr->fend < sdat && fldptr->ft != '¥0')
  {++fldptr;
   ++cou;}
 if(cou > 0)
  {--fldptr;
   t = fldptr->fp;
   ++fldptr;
  }
 else
  {
   if(datptr->dtyp != odtyp || defflg == 0)
     {if( askf() == 0)
       t = fstp->fname;
      else
       {*tt2 = tt;
        return(1);}
```

```
      }
     else
      t = ofname;
   }
  strcpy(ofname,t);
  negs();              /*Find negation*/
  if(negf ==1 && datptr->dtyp == 'J')
   {
    *tt++ = 'N';
    *tt++ = 'O';
    *tt++ = 'T';
    *tt++ = ' ';
    negf = 0;
   }
  /*move field name to command buff*/
  while(*t != '¥0')
   *tt++ = *t++;
  *tt++ = ' ';
  *tt2 = tt;
  return(0);
}
/*Ask user for field name*/
askf()
{
 char dbuff[200],fnb[4],*t,*s,loop,flg;
 loop=1;
 s = sdat;
 t = dbuff;
 while(s <= datptr->dend)
  *t++ = *s++;
 *t++ = ' ';
 *t   = '¥0';
 while(loop ==1)
 {
  fstp = fsttab;
  tprint(dbuff);
  tprint(mes13);
  tfeed();
  while(fstp->fnum[0] != '¥0')
   {tprint(fstp->fnum);
    tprint(" ");
    tprint(fstp->fname);
    tprint(" ");
    ++fstp;}
  tfeed();
  accept(mes14,4,fnb);
```

```
    tfeed();
    fstp = fsttab;
    flg = 1;
    while(fstp->fnum[0] != '¥0' && flg==1)
     {if(fstp->fnum[0] == fnb[0] && fstp->fnum[1] == fnb[1])
       {flg = 0;
        loop = 0;}
      else
       ++fstp;
     }
   }
   return(0);
  }
  /*Neagtion word search*/
  negs()
  {
   char *s;
   negf = 0;
   s = qp;
   while(s < sdat )
    {if (*s == 'T')
      {if(*(s+1) == '?' && *(s+2) == ' ')
        {negf = 1;
         qp = s+3;
         return(0);}
      }
     ++s;
    }
  }
  /*Logical conjunction search*/
  logs(tt,tt2)
  char *tt;
  unsigned *tt2;
  {
   int end;
   char *p,*t,toke,loop;
   static char andm[] = "AND",orm[]="OR";
   loop = 1;
   p = qp;
   while(p < sdat && loop==1)
    {
     if(stoke(p,&end,&toke) == 0)
      {if(toke == 34 || toke == 35)
        {qp = p+end;
         loop = 0;}
       else
```

```
        ++p;
      }
     else
      ++p;
    }
   if(loop==1)
    toke = '"';
   if(toke == '"')
    t = andm;
   else
    t = orm;
   while(*t != '¥0')
    *tt++ = *t++;
   *tt++ = ' ';
   *tt2 = tt;
   return(0);
}
/*Relational conjunction search*/
comps(tt,tt2)
char *tt;
unsigned *tt2;
{
 int end;
 char *p,*t,toke,loop;
 loop=1;
 p = qp;
 while(p < sdat && loop==1)
  {
   if(stoke(p,&end,&toke) == 0)
    {if(toke >= 60 && toke <= 62)
      {qp = p+end;
       loop = 0;}
     else
      ++p;
    }
   else
    ++p;
  }
 /*Default relational operator*/
 if(loop==1)
  {if (datptr->dtyp == 'J')
    toke = '$';
   else
    toke = '=';
  }
 if(negf == 1)
```

```
   {switch(toke) {
    case '=':
     {*tt++ = '<';
      *tt++ = '>';
      break;}
    case '>':
     {*tt++ = '<';
      *tt++ = '=';
       break;}
    case '<':
     {*tt++ = '>';
      *tt++ = '=';
      break;}
    }
   }
  else
   *tt++ = toke;
  *tt++ = ' ';
  *tt2 = tt;
  return(0);
}
/*Search Data value */
sdatval()
{char *pt,*st,*sst,*sen,*nst,*nen;
 char flg,sf,nf;
 flg = 1;
 st = &qbuff[1];
 pt = st;
 sen = st;
 nen = st;
 datptr = datarr;
 while(flg == 1)
 {
  while((*pt < 0x30 || *pt > 0x39) && *pt != '.' && *pt != ','
        && *pt != '¥''&& *pt != '¥"'&&*pt != '¥0')
    ++pt;
  sf = 0;
  nf = 0;
  if (*pt == '¥'' || *pt == '¥"')
   {
    /*0 mean found*/
    if (sdatvals(pt,&sst,&sen) == 0)
     {sf = 1;
      pt = sen+1;
      datptr->dtyp = 'J';
      datptr->dst  = sst;
```

```
        datptr->dend = sen;
        ++datptr;
        datptr->dtyp = '¥0';
       }
     }
    else if ((*pt >= 0x30 && *pt <= 0x39)||*pt=='.'
              ||*pt == ',' )
     {
      if (sdatvaln(pt,&nst,&nen) == 0)
       {nf = 1;
        pt = nen+1;
        datptr->dtyp = 'C';
        datptr->dst  = nst;
        datptr->dend = nen;
        ++datptr;
        datptr->dtyp = '¥0';
       }
     }
    if (nf == 0 && sf == 0)
     flg = 0;
  }
  if(sen == st  && nen == st)
   return(1);          /*No found data value*/
  else
   {--datptr;
    pt = datptr->dend+1;
    /*Cut right side of right-most data.val*/
    *pt = '¥0';
    return(0);}
}
/*end of sdatval*/
sdatvals(pt,st,en) /*st < en*/
char *pt;
unsigned *st,*en;
{
 char c;
 while(*pt != '¥0' && *pt != '¥''&& *pt != '¥"')
  ++pt;
 if (*pt != '¥0')
 {
  c = *pt;
  *st = pt++;
  while(*pt != c && *pt != '¥0')
   ++pt;
  if (*pt != '¥0')
   *en = pt;
```

```
   else
     return(1);
  }
  else
    return(1);
  return(0);
}
/*end of sdatvals */
sdatvaln(pt,st,en) /*st < en*/
char *pt;
unsigned *st,*en;
{
 while((*pt < 0x30 || *pt > 0x39) &&
        *pt != '.' && *pt != ','&& *pt != '¥0')
   ++pt;
 if (*pt != '¥0')
  *st = pt++;
 else
  return(1);
 while((*pt >= 0x30 && *pt <= 0x39 ||
        *pt == '.'|| *pt==',')&& *pt != '¥0' )
   ++pt;
 *en = pt-1;
 ++pt;
 return(0);
}
/*end of sdatvaln*/
/*Search conjuntion-search it all*/
conj()
{
 int end;
 char *p,*s,*t,*cend,toke,flg,loop;
 flg = 0;
 p = qp;
 while( p < sdat)
  {if( stoke(p,&end,&toke) == 0)
    {
     if(toke == 122)
     {
      loop = 1;
      fstp = fsttab;
      while(fstp->fnum[0] != '¥0' && loop==1)
       {s = p;
        t = fstp->fname;
        while(*s ==  *t && *t != '¥0')
         {++s;
```

```
            ++t;}
          if(*t == '¥0')
           loop = 0;
          else
           ++fstp;
         }
        if(loop == 0)    /*Conj. is equal with field*/
         p = s;
        else
         {flg = 1;
          cend = p+end-1;
          p = p+end;}
       }
      else
       ++p;
      }
    else
     ++p;
   }
   if(flg == 1)
    {qp = cend+1;
     return(0);}
   else
    return(1);
}
/*Search pronoun,olny one (when found then exit)*/
pronoun()
{
 int end;
 char *p,toke,loop;
 loop = 1;
 p = qp;
 while(loop==1)
  {if( stoke(p,&end,&toke) == 0)
     {
      if(toke == 122)
       {
        qp = p+end;
        p = qp;
       }
      else
       loop = 0;
     }
    else
     loop = 0;
  }
```

```
}
/*Search Transitive verb type 1*/
tv1()
{
 int end;
 char *p,toke,loop;
 loop = 1;
 p = qp;
 while(loop==1 && p<sdat)
  {if( stoke(p,&end,&toke) == 0)
    {
     if(toke == 124)
      {
       qp = p+end;
       p = qp;
      }
     else
      loop = 0;
    }
   else
    loop = 0;
  }
}
itg()
{int end;
 char *p,toke;
 p = qp;
 while( p < sdat)
  {if( stoke(p,&end,&toke) == 0)
    {
     if(toke == 125)
      {
       qp = p+end;
       p = qp;
      }
     else
      ++p;
    }
   else
    ++p;
  }
}
/*-------------
/*search word in string
  input
  keytab = key table,wordtab = seached word table,
```

```
   str =string to seach
   st = col. of char. in array. that sync. with str
   output
   st = start pos. ,end = end pos. */
sword(keytab,keyp,wordtab,str,st,end)
char *str,wordtab[][15];
int  *st,*end;
struct ktab {
        char keych;
        int  wadd; };
struct ktab keytab[], *keyp;
{
 int col,ocol,flag,row;
 char *tabp,*ostr,*tabp2;
 col = *st;
 while (*str != '¥0') /* end of string ? */
 {
  keyp = keytab;
  ocol = col;
  ostr = str;
  /*search first maching character by fix. string char.*/
  while(keyp->keych != *str && keyp->keych != '¥0')
   ++keyp;
  if (keyp->keych == *str)
   {
    /*search word that start with same character */
    row  = keyp->wadd;
    tabp = &wordtab[row][0];
    while(*tabp != WBRK)
    { /*search next character of word */
      while (*tabp != '¥0' && *tabp != WBRK
        && *str !='¥0'&& *tabp == *str)
      { ++tabp;
         ++str;
         col++;}
      if (*tabp == '¥0'|| *tabp == WBRK )
      {
        /*this word matched*/
        *st = ocol;
        *end = col-1;
        return(0);
       }
      else
      {
       /*skip to next word in table*/
       while(*tabp != '¥0'&& *tabp != WBRK)
```

```
          ++tabp;
         if (*tabp != WBRK)
         {str   = ostr;
          col = ocol;
          ++row;
          tabp = &wordtab[row][0];
          /*while (*tabp++ == '¥0');*/
         }
        }
     }/*end of search word that start with same charater*/
    }/*end if of find  first macthing charater */
   /*skip to next string character*/
   ++ostr;
   str   = ostr;
   col = ocol+1;
   while(*str != '¥0' && *str == ' ') /*skip space*/
    {++col;
     ++str;}
  }
  return(1);
}
----------*/
```

```
/*Cfword.c start 3-1-86 Date 4-11-86 ,
  Editor Functions.*/
#include <stdio.h>
#include <ctype.h>
#include <extern.h>
say(str,row,col,att)
 char *str,att;
 int row,col;
{
 char *t;
 t = temp;
 while(*str != '¥0')
 {
  *t++ = *str;
  *t++ = att;
  ++str;
 }
 *t++ = STDP;
 setcur(0,row,col);
 dsplngb(typtab,combtab,lmar,temp);
}
pgsay(str,row,col,att,page)
char *str,page;
int row,col,att;
{
 ++saycou;
 sayp = &saytab[saycou-1];
 sayp->mesp = str;
 sayp->syrow = row;
 sayp->sycol = col;
 sayp->syatt = att;
 sayp->sypage = page;
 if(page == 0)
  say(str,row,col,att);
}
/*Resay message in sayrec*/
autosay(pg)
char pg;
{int i;
 if(saycou == 0)
  return(0);
 sayp = saytab;
 i = 1;
 while(sayp->sypage != pg && i++ <= saycou )
  ++sayp;
 if(sayp->sypage != pg || i > saycou)
  return(1);
 while(sayp->sypage == pg)
  {
```

```
    say(sayp->mesp,sayp->syrow,sayp->sycol,sayp->syatt);
    ++sayp;
   }
}
/*Format input*/
get(row,col,pg,str,dec,typ,clr,bch,l,r,ln)
char row,col,pg,l,r,*str,clr,bch,typ;
int dec,ln;
{
 char att=7,*s,*t;
 grp = &grectab[gecou];
 grp->grow = row;
 grp->gcol = col;
 grp->gpage = pg;
 grp->gdat = str;
 grp->gbc  = bch;
 grp->gcl  = clr;
 grp->gdec = dec;
 grp->gt   = typ;
 grp->glm  = l;     /*left margin*/
 grp->grm  = r;     /*right margin*/
 grp->glen = ln;
 ++gecou;
 grectab[gecou].gt = '¥0';
 ccol = col;
 crow = row;
 bc = bch;
 lm = l;
 rm = r;
 cl = clr;
 cpos=1;
 stb = str;
 ptr = str;
 if(dec != 0)
  *(ptr+(rlen-dec)) = '.';
 rlen = grp->glen;
 mlen = rlen;
 slen = t2len(ptr);
 if(pg == 0)
  {
   if(bch != '¥0')
    {setcur(page,crow,ccol-1);
     dspmid(bch,page,clr);}
   setcur(page,crow,ccol);
   copget();
   dsplngb(typtab,combtab,lm,temp);
  }
}
/*Clear get*/
```

```
clrget()
{
 gecou = 0;
 saycou = 0;
 epage = 0; /*used by ger*/
 grectab[0].gt = '¥0'; /*gt normaly have to type C,N*/
}
/*Redisplay page*/
reget(pg)
char pg;
{int i = 0;
 clscr(page,bg);
 grp = grectab;
 while(grp->gpage != pg && i++ <= 23&&grp->gt != '¥0')
  ++grp;
 if(grp->gpage != pg || grp->gt == '¥0')
  return(1);
 while(grp->gpage == pg && grp->gt != '¥0')
  {
   ccol = grp->gcol;
   crow = grp->grow;
   bc = grp->gbc;
   lm = grp->glm;
   rm = grp->grm;
   cl = grp->gcl;
   stb = grp->gdat;
   ptr = grp->gdat;
   vdec = grp->gdec;
   cpos = 1;
   rlen = grp->glen;
   mlen = rlen;
   slen = t2len(ptr);
   if(bc != '¥0')
    {setcur(page,crow,ccol-1);
     dspmid(bc,page,cl);}
   setcur(page,crow,ccol);
   copget();
   dsplngb(typtab,combtab,lm,temp);
   ++grp;
  }
}
/*Operate  get function*/
vread(itc,c,exc)
int itc;
char *c,*exc;
{
 char rrow,rcol,rbc,rlm,rrm,rcl,*rpt,com;
 char eflg=1,pg,*s,*t;
 /*cpos offset frim rcol+1*/
```

```
    grp = &grectab[itc];
    pg = grp->gpage;
    if(epage != pg)
     {reget(pg);
      autosay(pg);
      epage = pg;}
    grp = &grectab[itc];
    srow = grp->grow; /*start row*/
    scol = grp->gcol;
    vtyp  = grp->gt;
    vdec = grp->gdec;
    bc   = grp->gbc;    /*Boudary charater ,'¥0' = none*/
    cl   = grp->gcl;    /*Foreground color/att.*/
    lm   = grp->glm;
    rm   = grp->grm;
    mlen = grp->glen;/*maximum len*/
    s    = grp->gdat;
    t = wbuff;
    rlen = 0;          /*real len*/
    /*copy from gdat to wbuff*/
    while(*s!='¥0')
     {*(t++) = *(s++);
      ++rlen;}
    *t = '¥0';
    ptr = wbuff;
    stb  = wbuff;
    slen = t2len(ptr);
    crow = srow;
    ccol = scol;
    cpos = 1;
  while (eflg==1)
   {
    seco();
    setcur(page,crow,ccol);
    lgram = lgf(ptr);
    rgram = rgf(ptr,'¥0');
    chr = kbinp(&echr);
    curtyp = gtyp(chr);
    dgram = dgf(lgram,curtyp);
    if(isdigit(chr) != 0)
     ct = 'N';
    else
     ct = 'C';
    if (chr<32)
     {
      if(chr==0)
       com = echr;
      else
       com = chr;
```

```
    if (com==22||com==82) /*^v*/
     {
      if (ins == 1)
       ins = 0;
      else
       ins = 1;
     }
    else if(com==1) /*^A*/
     wordl();
    else if(com==6) /*^F*/
     wordr();
    else if((com==7 || com == 83)&& rgram != 1)
     rdel();
    else if(com==4 || com == 77)  /*^D*/
     {cra();
      if(vtyp == 'N'&& cpos == mlen-vdec+1 )
        cra();}
    else if(com==19 || com == 75 || com == 8)  /*^S*/
     {cla();
      if(vtyp == 'N' && cpos == mlen-vdec+1)
        cla();}
    else if(com==20)  /*^T*/
     rcut();
    else if(com==25)  /*^Y*/
     delal();
    else if(com==23||com==RET||com==72||
            com==80||com==17||com==27) /*Exit */
     eflg=0;
    else if(com==71||com==73||com==79||
            com==81||com==3||com==18||com==21)
     eflg=0;
    }
   else
   {if(ct == vtyp || vtyp == 'C' ) /*Typ checking*/
    {if(ins == 0 && curtyp == 2 && rgram != 1)
      replac();
     else if(rgram == 1 && dgram != 0 && rlen < mlen)
      dea();
     else if(rgram != 1 && dgram != 0)
      insert();
     else if(dgram == 0)
      putch(7);
    }
    else
     putch(7);
   }
  }
*c = chr;
*exc = echr;
```

```
t = grp->gdat;
s = wbuff;
while(*s != '¥0')
  *t++ = *s++;
*t = '¥0';
}
/*Type 2 length*/
t2len(pt)
char *pt;
{
 int i = 0;
 while(*pt != '¥0')
  { if (gtyp(*(pt++)) == 2)
      ++i;}
 return(i);
}
/*delete right side*/
rdel()
{
 int i;
 if(rgram == 8) /*cursor point at rm+1*/
  {cra();
   rgram = rgf(ptr,'¥0'); }
 if(rgram==2||rgram==4||rgram==5)
   pulll(2);
 else if(rgram==3)
  pulll(1);
 else if(rgram==6||rgram==7)
   pulll(3);
 i = mlen-cpos+1+(rlen-slen);
 --slen;
 /*pulsl(1,crow,ccol,slen);*/ /*pull screen left*/
 copget();
 dsplngb(typtab,combtab,lm,temp);
 }
/*Delete all*/
delal()
{
 ccol = scol;
 crow = srow;
 rlen = 0;
 *stb = '¥0';
 ptr  = stb;
 slen = 0;
 cpos = 1;
 copget();
 setcur(page,crow,ccol);
 dsplngb(typtab,combtab,lm,temp);
}
```

```
/*Delete all char in right(right cut)*/
rcut()
{
 *ptr = '¥0';
 rlen = cpos-1;
 slen = t2len(stb);
 copget();
 dsplngb(typtab,combtab,lm,temp);
}
/*Data entry*/
dea() /*ptr point to end text*/
{
 /*enter type 2 case*/
 if(dgram == 1||(dgram==3 &&ccol <= rm))
  {
   *ptr++ = chr;
   ++ccol;
   ++slen;
   dspmid(chr,cl);
  }
 else if(dgram == 3 && ccol > rm)
  {*ptr++ = chr;
   ccol = lm;
   crow=crow+3;
   setcur(page,crow,ccol);
   dspmid(chr,cl);
   ++slen;
   ++ccol;}
 else if(dgram == 2)
  {
    *ptr++ =chr;
    dspup(chr,cl);}
 else if(dgram == 4)
  {
   *ptr++ = chr;
   dspdow(chr,bg);}
 else if(dgram == 5)
   {
     *ptr++ = chr;
     dspup(wcomb(chr,*(ptr-2)),cl);}
*ptr = '¥0';
++rlen;
++cpos;
/*-------
if(dgram==2 || dgram == 4 ||dgram == 5)
 {bleft(1);
  copget();
  dsplngb(typtab,combtab,lm,temp);
 }
```

```
-------*/
setcur(page,crow,ccol);
}
/*if curtyp != 2 ,no care ins = ?,-->insert by auto*/
insert()
{
 pullr(1);
 if(curtyp == 2) /*insert type 2 case*/
  {
   if(rgram == 8) /*Cursor point at rm+1*/
    cra();
   *ptr = chr;
   ++slen;
   /*rgram is changed &used for cra()*/
   rgram = rgf(ptr,'¥0');
   copget();
   dsplngb(typtab,combtab,lm,temp);
   cra();
  }
 else
  {
   if(dgram == 2)
    {
      *(ptr++) =chr;
      ++cpos;
      dspup(chr,cl);}
   else if(dgram == 4)
    {
     *(ptr++) = chr;
     ++cpos;
     dspdow(chr,bg);}
   else if(dgram == 5)
     {
       *(ptr++) = chr;
       ++cpos;
       dspup(wcomb(chr,*(ptr-2)),cl);}
  }
if(dgram==2 || dgram == 4 || dgram == 5)
 {if(rgram == 8)
   setcur(page,crow+3,lm);
  copget();
  dsplngb(typtab,combtab,lm,temp);}
setcur(page,crow,ccol);
}
/*replace */
replac()
{
 *ptr = chr;
 if (rgram == 8) /*Cursor point at rm+1 */
```

```
   cra();
  ++slen;
  dspmid(chr,cl);
  rgram = rgf(ptr,'¥0');  /*Reset old rgram */
  /*------Auto delete up/down*/
  ++ptr; /*set pointer point to up/down*/
  if (rgram == 2 || rgram == 4 || rgram == 5)
    pulll(1);
  else if(rgram == 6 || rgram == 7)
    pulll(2);
  --ptr;
  /*--------*/
  setcur(page,crow,ccol);
  copget();
  dsplngb(typtab,combtab,lm,temp);
  /*---*/
  rgram = rgf(ptr,'¥0');
  cra();
  setcur(page,crow,ccol);
  clrup(cl);
  clrdow(bg);
}
/*Move cursor to next word*/
wordr()
{
 rgram = rgf(ptr,'¥0');
 while(*ptr < 'A'&& rgram != 1)
  {cra();
   rgram = rgf(ptr,'¥0');}
 rgram = rgf(ptr,'¥0');
 while(rgram != 1 && *ptr >= 'A')
  {cra();
   rgram = rgf(ptr,'¥0');}
}
/*Move cursor to front word*/
wordl()
{
 lgram = lgf(ptr);
 while(*ptr < 'A'&& lgram != 1)
  {cla();
   lgram = lgf(ptr);}
 lgram = lgf(ptr);
 while(*ptr >=  'A' && lgram != 1)
  {cla();
   lgram = lgf(ptr);}
}
```

```
/*Cfwfunc.c*/
/*start 3-1-86 today 4-2-86*/
#include <stdio.h>
#include <extern.h>
/*Search coordinate of end-text*/
seco()
{
 int i,j,k;
 i = slen+mlen-rlen; /*total collumn on screen*/
 j = rm-scol+1; /*collumns of first line*/
 if (i<=j)
  {ecol = scol+i-1;
   erow = srow;}
 else
  {i = i-j;
   j = rm-lm+1;
   k = i/j;
   if(k*j == i)
    ecol = rm;
   else
    {ecol = lm+i-k*j-1;
     ++k;}
   erow = k*3+srow;
  }
}
/*Boudary left*/
bleft(n)
int n;
{
 int col,mcol,lcol;
 if(bc != '¥0')
  {col = ecol+1;
   mcol = rm+1;
   lcol = lm+1;}
 else
  {col = ecol;
   mcol = rm;
   lcol = lm;}
 while (n-- > 0)
  {setcur(page,erow-1,col);
   dspmid(' ',page,bg);
   setcur(page,erow+1,col);
   dspmid(' ',page,bg);
   setcur(page,erow,col);
   dspmid(' ',page,bg);
   if(col == lcol)
    {col = mcol;
     erow = erow-3;}
   else
```

```
    --col;}
 if(bc!= '¥0')
  {ecol = col-1;
   bound();}
 else
  ecol = col;
 setcur(page,crow,ccol);
}
/*Boundary right*/
bright(n)
int n;
{
 int col,mcol,lcol;
 char a;
 if(bc != '¥0')
  {col = ecol+1;
   mcol = rm+1;
   lcol = lm+1;
   a = bg;}
 else
  {col = ecol;
   mcol = rm;
   lcol = lm;
   a = cl;}
 while (n-- > 0)
  {setcur(page,erow-1,col);
   dspmid(' ',page,a);
   setcur(page,erow+1,col);
   dspmid(' ',page,a);
   setcur(page,erow,col);
   dspmid(' ',page,a);
   if(col == mcol)
    {col = lcol;
     erow = erow+3;}
   else
    ++col;}
 if(bc!= '¥0')
  {ecol = col-1;
   bound();}
 else
  ecol = col;
 setcur(page,crow,ccol);
}
/*Display boundary */
bound()
{
 setcur(page,erow,ecol+1);
 dspmid(bc,page,cl);
}
```

```
/*copy form getvar to disp buff*/
/*Must start with type 2*/
copget()
{
 int i;
 char *s,*t,col,t2,t3,att;
 col = ccol;
 t = temp;
 s = ptr;
 while(*s !='¥0')
  {*t++ = *s++;
   *t++ = cl;
   t2 = gtyp(*s);
   t3 = gtyp(*(s+1));
   if(t2 == 3)
    att = bg;
   else
    att = cl;
   if(t2 == 1 || t2 == 3 || t2 == 4)
    {*t++ = *s++;
     *t++ = att;
     if(t3 == 1)
      {*t++ = *s++;
       *t++ = cl;}
     }
    ++col;
    if(col == rm+1)
     {*t++ = RET;
      col = lm;
      ++t;}
  }
 i = mlen-slen; /*Don't change boundary*/
 while(i-- > 0)
  {*t++ = ' ';
   *t++ = cl;
   ++col;
   if(col == rm+1)
    {col = lm;
     *t++ = RET;
     ++t;}
  }

 *t = STDP;
}
/*pull left*/
pulll(n)
int n;
{
 char *p;
```

```
 unsigned i;
 p = ptr;
 if (rlen > 1)
 {
  i= rlen-cpos-n+1;
  while(i-- > 0)
  {*p = *(p+n);
   ++p;}
 }
 rlen = rlen-n;
 *(stb+rlen) = '¥0';
}
/*pull right*/
pullr(n)
int n;
{
 char *p;
 unsigned i,ps;
 p = stb+rlen;
 i = rlen-cpos+2;
 while(i-- > 0)
  *(p+n)=*(p--);
 rlen = rlen+n;
 if (rlen > mlen)
  {rlen = mlen;
   *(stb+rlen) = '¥0';}
}
cla() /* Cursor left action */
{
 if (lgram == 1)
  putch(7);
 else if(lgram == 2|| lgram == 4 || lgram == 5)
  {
   ptr = ptr - 2;
   cpos = cpos-2;
   --ccol;}
  else if(lgram == 3)
   {
    --ptr;
    --cpos;
    --ccol;}
  else if(lgram == 6|| lgram == 7)
   {
    ptr = ptr - 3;
    cpos = cpos-3;
    --ccol;}
  else if (lgram==8)
  {ccol = rm+1;
   crow=crow-3;}
```

```
  setcur(page,crow,ccol);
 }
cra() /* cursor right action */
{
 if (rgram == 1)
  putch(7);
 else if (rgram == 2;; rgram==4 ;; rgram == 5)
  {ptr = ptr +2;
   cpos = cpos+2;
  ++ccol;}
 else if (rgram == 3)
  {
   ++ptr;
   ++cpos;
   ++ccol;}
else if(rgram == 6 ;; rgram == 7)
  {ptr = ptr+3;
   cpos = cpos+3;
   ++ccol;}
 else if(rgram == 8) /*Cursor point at rm+1 */
 {ccol = lm;
  crow=crow+3;}
 setcur(page,crow,ccol);
}
/*gramma function*/
lgf(str) /*left gramma function */
char *str;
{
 char lchr, st = 0;
 char typ, *lptr;
 if(cpos==1)
  return(1);
 if(ccol==lm)
  return(8);
 lptr = str;
 lptr = lptr-1;
 lchr = *lptr;
 lptr = lptr-1;
 typ = gtyp(lchr);
 if (typ == 8)
  {
  return(1);
  }
 else if (typ == 1)
  {
    lchr = *lptr;
    lptr = lptr-1;
    typ = gtyp(lchr);
    switch (typ) {
```

```
      case 2 :
       return(2);
      case 3 :
       return(6);
      case 4 :
       return(7);
      default:       }
     }
    else if (typ == 2)
     return(3);
    else if (typ == 3)
     return(4);
    else if (typ == 4)
     return(5);
    else if (typ == 7)
     return(9);
    else
     return(8);
  }
rgf(rptr,eot) /*right grammar funtion*/
char *rptr,eot;
 {
  int i = 0;
  char t1,t2,t3;
  if (*rptr==eot)
   t1 = 9;
  else
   t1 = gtyp(*rptr++);
  if (*rptr==eot)
   t2 = 9;
  else
   t2 = gtyp(*rptr++);
  if (*rptr==eot)
   t3 = 9;
  else
   t3 = gtyp(*rptr);
  while ( (rgtab[i][0] != t1 && rgtab[i][0] !=0) ||
          (rgtab[i][1] != t2 && rgtab[i][1] !=0) ||
          (rgtab[i][2] != t3 && rgtab[i][2] !=0))
    i++;
   /*When ptr point at EOT and cursor point
     at rm+1,rgram != 8,because this
     editor don't save RET*/
   if(rgram != 1 && ccol == rm+1)
    return(8);
   return(rgtab[i][3]);
}
/*Data entry gramma function  */
 dgf()
```

```
{
 int i;
 i = (lgram-1)*6;
 while (curtyp!=dgtab[i][1]&&lgram==dgtab[i][0])
  i++;
 if (lgram == dgtab[i][0])
  return(dgtab[i][2]);
 else
  return(0);
}
```

```
/*CFIO.C do i&o function 4-12-86 */
#include <stdio.h>
#include <extern.h>
/*High level I&O*/
getline(s,lim)
char *s;
int lim;
{
 /*ct = current type,ot = old type*/
 char c,d,ext,*b,ct,ot;
 int i;
 unsigned cmd,cmd2,row,col,srow,scol,row2,col2;
 rdcur(page,&srow,&scol,&cmd);
 i = 0;
 *s = '¥0';
 while ( i<lim-1 && (c=kbinp(&ext))!= RET)
  {
   ct = gtyp(c);
   ot = gtyp(*(s-1));
   if (c == 8 && i != 0)
    {
     rdcur(page,&row,&col,&cmd);
     curlft();
     rdcur(page,&row2,&col2,&cmd2);
     if(ot == 2)
      {
       dspmid(' ',cl);
       if(col == 0)
        {col = col2;
         row = row2;}
       else
        --col;
      }
     else if(ot == 1 || ot == 4)
      {
       setcur(page,row2-1,col2);
       dspmid(' ',cl);
       if( i >= 3)
        {if(gtyp(*(s-2)) == 4)
          {setcur(page,row2-1,col2);
           dspmid(*(s-2),cl);}
        }
      }
     else if(ot == 3)
       {setcur(page,row2+1,col2);
        dspmid(' ',bg);}
     setcur(page,row,col);
     --i;
     --s;
```

```
      *s = '¥0';
     }
    else if( c != 0 && c >= 32)
     {
      if(pflg == 1)
       prt_out(c);
      if(ct == 2)
       {dspmid(c,cl);
        ++i;
        *s++ = c;}
      else if(i != 0)
       { d = c;
         rdcur(page,&row,&col,&cmd);
         if(ot == 4 && ct == 1)
          d = wcomb(*(s-1),c);
         if((ct == 1 || ct == 3 || ct == 4) && ot != 1 )
         {
          if(col == 0 && row != srow)
           setcur(page,row-3,79);}
         if(ct == 3 && ot != 3 )
          {dspdow(d,bg);
           *s++ =c;
           ++i;}
         else if((ct == 4 && ot != 4 )|| ct == 1 )
          {dspup(d,cl);
           *s++  = c;
           ++i;}
         setcur(page,row,col);
       }
     }
   }
  /*if (c == RET)
      *s++ = c;*/
  *s = '¥0';
  if(pflg == 1)
   {prt_out(13);
    prt_out(10);}
  return(i);
}
accept(str,lim,ip)
char *str,*ip;
int lim;
{int len;
 tprint(str);
 len=getline(ip,lim);
 return(len);
}
/*input char from keyboard*/
kbinp(ext)
```

```
char *ext;
{
 char chr,chr2,x;
 chr2 = 60;
 while (chr2 == 60)
 {
  chr = kbbio(&chr2);
  if (chr == 0 && chr2 ==60)
   {
    cmflg = cmflg ^ 1;
    putch(7);
    if(cmflg == 0)
     cursize(7,7);
    else
     cursize(5,7);
   }
  /* check ^p */
  if(chr == 16)
   {pflg = pflg ^ 1;
    chr2 = 60;
   }
 }
 if (chr < 32 || cmflg == 0 )
  {*ext = chr2;
   return(chr);}
 else
  {x = convtab[0]; /*get offset val.*/
   *ext = chr2;
   return(convtab[chr-x+1]);}
 }
/*Get type function*/
gtyp(gchr) /* get general type of char */
 char gchr;
 {
  char offs;
  offs = typtab[0];
  return(typtab[gchr-offs+1]);
  }
ktyp(kchr) /*keyboard charracter typ*/
char kchr;
{
 char ktyp;
 ktyp = gtyp(kchr);
 if (kchr == 0x0d)
  return(11); /*soft return */
 else
  return(ktyp);
 }
```

```
/*combine to type <5>,sara_up+van */
wcomb(up,van)
 char up,van;
 {
  char typ,tchr,upoff,vaoff;
  int  gsize;
  upoff = combtab[0]; /*get sa-up offset val.*/
  vaoff = combtab[2]; /*get van   offset val.*/
  gsize = combtab[1]; /*group size of sa-up  */
  typ = gtyp(van);
  if (typ == 1)
   {
    up = up-upoff;
    van = van-vaoff;
   }
  else
   {
    tchr = van;
    van   = up-vaoff;
    up    = tchr-upoff;
   }
    return(combtab[up*gsize+van+3]);
 }
/*screen function*/
/*High Level I&O*/
tprint(s)
char *s;
{
 char *t,t2,t3,att,c1,c2,ret;
 int r;
 unsigned  cmd,row,col;
 t = temp;
 rdcur(page,&row,&col,&cmd);
 while(*s != '¥0')
  {
   /*Check keyboard status,check ^s*/
    kbstat(&c1);
    if(c1 == 19)
    {
     c1 = kbbio(&c2);
    }
    if(c1 == 16)       /* ^p */
    {
     c1 = kbbio(&c2);
     pflg = pflg ^ 1;
    }
   /*-----------------------*/
   if(pflg == 1)
    prt_out(*s);
```

```
    ret = *s;           /*save for check RET*/
    *t++ = *s++;
    *t++ = cl;
    t2 = gtyp(*s);
    t3 = gtyp(*(s+1));
    if(t2 == 3)
     att = cl;
    else
     att = bg;
    if(t2 == 1 || t2 == 3 || t2 == 4)
     {if(pflg == 1)
       prt_out(*s);
      *t++ = *s++;
      *t++ = att;
      if(t3 == 1)
       {if(pflg == 1)
          prt_out(*s);
        *t++ = *s++;
        *t++ = cl;}
     }
    ++col;
    if(col > 79|| ret == 13)
     {*t = STDP;
      dsplngb(typtab,combtab,0,temp);
      t =temp;
      col = 0;
      tfeed();
      if(row + 3 > 23)
       row = 23;
      else
       row =row+3;
      setcur(page,row,col);
     }
   }
   *t++ = STDP;
   dsplngb(typtab,combtab,0,temp);
   setcur(page,row,col);
}
/*Thai feed*/
tfeed()
{unsigned row,col,cmod;
 int i;
 rdcur(page,&row,&col,&cmod);
 if(row+3 <= 23)
  setcur(page,row+3,0);
 else
  {setcur(page,23,0);
   i = (row+3)-23;
   while(i-- > 0)
```

```
    scrllup();
   }
  /*Hard copy*/
  if(pflg == 1)
   {prt_out(13);
    prt_out(10);
   }
}

/*Low level I&O ,link with BIOS*/
/*cursor left*/
curlft()
{
 unsigned curmd;
 int row,col;
 rdcur(page,&row,&col,&curmd);
 if (col != 0)
  setcur(page,row,col-1);
 else if(row-3 >= 1)
  setcur(page,row-3,79);
}
currgh()
{
 unsigned pos,curmd,row,col;
 rdcur(page,&row,&col,&curmd);
 if (col = 79)
  {
   if (row < 24)
    setcur(page,row+1,0);
   else
    {scrllup();
     setcur(page,row,0);}
  }
 else
  setcur(page,row,col+1);
}
dspmid(dchr,a)
char dchr,a;
{
 unsigned  curmd;
 int row,col;
 wrtatt(page,1,dchr,a);
 rdcur(page,&row,&col,&curmd);
 ++col;
 if (col > 79)
  tfeed();
 else
  setcur(page,row,col);
}
```

```
dspup(dchr,a)
char dchr,a;
{
 unsigned pos, curmd, row, col;
 rdcur(page, &row,&col, &curmd);
 /*set cursor pos. to up line */
 setcur(page,row-1,col-1);
 wrtatt(page,1,dchr,a);
 /*set to old position */
 setcur(page,row,col);
}
/*display down*/
dspdow(dchr,a)
char dchr,a;
{
 unsigned curmd, row,col;
 rdcur(page, &row,&col, &curmd);
 /*set cursor pos. to up line */
 setcur(page,row+1,col-1);
 wrtatt(page,1,dchr,a);
 /*set to old position */
 setcur(page,row,col);
}
/*Clear screen routine*/
clrlft()
{
 unsigned curmd, row,col;
 rdcur(page, &row,&col,&curmd);
 /*set cursor pos. to left */
 setcur(page,row,col-1);
 wrtatt(page,1,32,cl);
}
clrup(a)
{
 unsigned curmd,row,col;
 rdcur(page, &row,&col, &curmd);
 /*set cursor pos. to up line */
 setcur(page,row-1,col-1);
 wrtatt(page,1,32,a);
 /*set to old position */
 setcur(page,row,col);
}
clrdow(a)
{
 unsigned curmd, row,col;
 rdcur(page,&row,&col,&curmd);
 /*set cursor pos. to up line */
 setcur(page,row+1,col-1);
 wrtatt(page,1,32,a);
```

```
 /*set to old position */
 setcur(page,row,col);
}
/* clear screen */
clscr()
{
 unsigned lin, lfcorn,rgcorn,pos;
 lin = 0;
 lfcorn = 0;
 scldow(lin,0,0,24,79,bg);
 setcur(page,0,0);
}
/*Clear screen no. 2,Don't clear line 0*/
clscr2()
{
 scldow(0,1,0,24,79,bg);
}
/*scroll function */
scrllup()
{
 unsigned lin, lfcorn,rgcorn,pos;
 lin = 1;
 sclup(lin,0,0,24,79,cl);
}
/*Printer output*/
prt_out(c)
char c;
{
 prtbio(c);
}
```

```
/*File name: declare.h*/
#define MAXLINE 500
#define ST '¥1'    /*Start text*/
#define WBRK '~'   /*word break symbol*/
#define ETAB ';'   /*table end symbol*/
#define EOT 255
#define RET 13
#define FEED 10
#define EOF_ERR 9
#define DUP 1
#define MOD 2
#define BIN 4
#define B_OPEN  0
#define B_CLOSE 1
#define B_INS 2
#define B_UPD 3
#define B_DEL 4
#define B_GETEQ 5
#define B_GETNX 6
#define B_GETPR 7
#define B_GETGT 8
#define B_GETGE 9
#define B_GETLT 10
#define B_GETLE 11
#define B_GETLW 12
#define B_GETHT 13
#define B_CREATE 14
#define B_STAT 15
#define B_SET_DIR 17
#define B_GET_DIR 18
#define STSIZ 27

/*symbol table,use for save field*/
struct fldt
 {
  char ft;      /*Field type*/
  char *fst;    /*start position*/
  char *fend;   /*end pos.*/
  char *fp;    /*Point to field name in fstru*/
 };
/*data value table*/
struct datt
 {
  char dtyp; /*data value type 'c' = char 'n' = num*/
  char *dst; /*start data value pos.*/
  char *dend;
```

```
 };
/*Conjunction saved table*/
struct cjt
 {
  int cjst; /*start pos.*/
  int cjend; /*end pos.*/
  char cjflg; /* flag='Y' = OK*/
 };
/*Interrogators table(suppanarm chaitarm)*/
struct itgt
 {
  int itgst;
  int itgend;
  char itflg;
 };
/*Transitive verb 1 saved table*/
struct tv1t
 {
  int tv1st;
  int tv1end;
  char tv1flg;
 };
 /*vaiables of i&o,first byte is offset val. */
 char convtab[] = { 32,32,33,46,50,51,52,235,165,54,55,53,
                    57,191,162,211,187,166,49,47,95,190,
                    180,215,219,163,179,169,196,176,168,
                    201,45,49,194,46,167,173,172,210,170,
                    223,177,227,198,197,37,228,214,171,48,
                    175,164,182,226,203,34,41,222,40,184,92,
                    195,216,56,96,189,217,209,161,207,178,
                    208,225,193,224,206,199,181,220,183,192,
                    213,188,200,204,218,202,212,185,221,186,
                    174, 124, 39, 126
                   };
/*combination code table saup.offset val,
group size,van offset val */
char combtab[] ={
  217,5,224,238,239,240,241,242,243,244,245,246,
  0,247,248,249,250,0,251,252,253,254,
  0,234,235,236,237,0 };
/*type table
start from code a1h(161),it up on data in fstcode,
first byte is offset val.
*/
char typtab[] = {0,
 00,13,13,13,13,13,13,13,13,13,07,13,13,
```

```
 06,13,13,13,13,13,13, 13,13,13,13,13,13,
 13,13,13,13,13,13,02,02,02,02,02,02,02,02,
 2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,
 2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,
 2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,
 2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,
 2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,
 2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,
 2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,
 2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,
 2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,3,3,4,4,4,4,
 4,4,1,1,1,1,1,1,5,5,5,5,5,5,5,5,5,5,5,5,
 5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,9 }; /*end at 255*/
 /*end of io var.*/
 char qbuff[500]; /*query buffer*/
 char *qp;        /*query buff. pointer*/
 char tokbuff[500];
 char epbuff[500];
 int stack[500];
 int stp=0; /*stack pointer*/
 int spt;
 struct fldt fldtab[24],*fldptr;
 struct datt datarr[24], *datptr;
 struct cjt cjarr[20],*cjptr;
 struct itgt itgarr[24],*itgptr;
 struct tvlt tvlarr[10],*tvlptr;
 int datrow = 0; /* Empty row */
 int cjrow = 0;
 int itgrow = 0;
 int tvlrow = 0;
 int sexp,eexp; /*start & end of expression */
 int querp; /* query pointer*/

 /* dgram table.(Data entry gramma table).
 {lgram,curtyp,dgram}*/
char dgtab[54][3] = {
  {1,2,1},{1,11,7},{1,12,6},{0,0,0},{0,0,0},{0,0,0},
  {2,2,3},{2,11,7},{2,12,6},{0,0,0},{0,0,0},{0,0,0},
  {3,1,2},{3,2,3},{3,3,4},{3,4,2},{3,11,7},{3,12,6},
  {4,1,2},{4,2,3},{4,11,7},{4,12,6},{0,0,0},{0,0,0},
  {5,1,5},{5,2,3},{5,11,7},{5,12,6},{0,0,0},{0,0,0},
  {6,2,3},{6,11,7},{6,12,6},{0,0,0},{0,0,0},{0,0,0},
  {7,2,3},{7,11,7},{7,12,6},{0,0,0},{0,0,0},{0,0,0},
  {8,2,1},{8,11,7},{8,12,6},{0,0,0},{0,0,0},{0,0,0},
  {9,2,1},{9,11,7},{9,12,6},{0,0,0},{0,0,0},{0,0,0}};
/* right gramma table first 3 are seq, 4th is rgram */
```

```
char rgtab[18][4] = {
  {2,1,0,2},{2,1,6,2},{2,2,0,3},{2,3,2,4},{2,3,6,4},
  {2,3,9,4},{2,3,1,6},{2,4,1,7},{2,4,2,5},{2,4,6,5},
  {2,4,9,5},{2,6,7,3},{2,6,10,3},{2,9,0,3},{6,7,0,9},
  {6,10,0,8},{9,0,0,1},{0,0,0,0}};
char temp[6000]; /*temporay area for display long*/
char *malloc();
char cmflg = 0; /*character mode flag*/
char pflg = 0;  /*printer output flag*/
char lgram, rgram, dgram;
char lmar = 0;
char bell,rmflg=0,sclflg = 0, scdflg=0;
int tlin=0,maxlin=0;
 struct getrec {
        char grow;
        char gcol;
        char gpage;
        char *gdat;
        char gt;      /*Item type*/
        int  glen;
        int  gdec;    /*decimal*/
        char gbc;     /*boundary char, '¥0' = no*/
        char gcl;     /*color*/
        char glm;
        char grm;
 } grectab[100];
 struct getrec *grp;
struct sayrec {
        char *mesp;   /*Point to say message*/
        char syrow;
        char sycol;
        char syatt;
        char sypage;
 } saytab[100];
 struct sayrec *sayp;
 int saycou;          /*Say item count,anf flag(0=no)*/
 /*File structure*/
 struct fstru
   {
    char fnum[4];
    char fname[16];
    char ftyp[2];
    char flen[4];
    char fdec[2];
    int  fln;
    int  fdc;
```

```
   char *fcp;        /*Field content pointer.
                       (point to data buff)*/
   double fval;      /*Current field value.*/
  } fsttab[24];
struct fstru *fstp;
/*File manager vars*/
/*Btrieve vars*/
struct key_spec
 {int key_pos;
  int key_len;
  int key_flag;
  char not_use1[4];
  char reserve1[6];
};
struct fil_spec
 {int rec_len;
  int page_siz;
  int ndx_cnt;
  char not_use2[4];
  char reserve2[6];
  struct key_spec key_buf[2];
 };
 struct fil_spec file_buf;
 char fil_name[15];
 char fil_ext[5] = ".DBF";
 char pos_blk[128];
 int stat;
 struct st_rec    /*record of file structure */
  {
   char fnumb[4];
   char fnameb[16];
   char ftypb[2];
   char flenb[4];
   char fdecb[2];};
 struct st_rec st_buff;
 char st_pos[128];
 int st_stat;
 char st_kbuff[20];
 char st_file[15];
 char st_ext[5] = ".STC";
 char key[20];
 char data_buf[1024];   /*File buffer,rec.len. not > 1024*/
 char edit_buf[2000];   /*Data Base edit/list buffer*/
 unsigned cur_rec;
 int reclen;            /*Record length*/
 int totfld;            /*Total field*/
```

```
 char eof_flg = 'T';     /*End-of-file flag*/
/*Editor vars.*/
int gecou = 0;
char bc,cl=7,lm,rm,tm,bm=23,
char *stb,ccol,crow,scol,srow;
char chr,echr,curtyp,ins=0,*ptr,wbuff[600];
unsigned cpos,mlen,rlen,slen;
char ecol,erow,bg=7,page=0,vtyp,ct,epage,vpage;
int vdec;
/*Parser vars*/
int quit(),create(),use(),spec();
int append(),list(),edit(),delete();
int delf(),copy(),report(),go_to();
int close_fil(),clear(),help();
int tok_add[15];
char cquit=1;
/*Csent vars*/
char *sdat;    /*Start point of current data value*/
char *oedat;   /*End point of previous data*/
char negf;     /*Negation flag*/
char ltoke;    /*Token of logical opt.*/
char defflg;   /*Default field flag. 1 = can default*/
char odtyp;          /*Old data value type*/
char ofname[16];    /*Old field name*/
/*Expression parser vars*/
char *chrp;
char tbuff[20];
char *token;
char *opr;
char *opt;
char toke_type;
char toke_no;
char posfixb[256];
char *posf;
char serr=0;
char perr=0;
char t1[20];
char *a;
char t2[20];
char *b;
/*Identifier table*/
struct id_rec {
      char id_no[3];
      char id_typ;      /*F =field, C = constant*/
      char id_dtyp;     /*C = character , N = numeric */
      char id_len;
```

```
      char *id_add;
      double id_val;
} id_tab[50];
struct id_rec *idp;
int idcou;              /*Point to empty element*/
char id_buff[1024];     /*String constant buffer*/
char *idbp;
/*------*/
struct toke_rec {
      char *tword;
      char ttoke;
} toke_tab[] = { {"AND",'"'},{"QCL",'"'},
{"OR",'#'},{"HA¥J",'#'},
{"NOT",'!'},{"T?`",'!'},{"¥O",'¥O'}};
struct toke_rec *trp;
/*Operator stack*/
struct opt_stack {
       char *opt_sp;
} opt_tab[50];
struct opt_stack *optsp;
int opt_cou;
char opt_stbuf[1024];
char *opt_mem;
/*Operand stack*/
struct opr_stack {
       char *opr_sp;
} opr_tab[50];
struct opr_stack *oprsp;
char opr_stbuf[1024];
char *opr_mem;
int opr_cou;
/*--------End of exp. pars. vars.----------*/
/*Expression evaluation vars.*/
struct val_rec {
 char otyp;
 char ologic;
 char *oadd;
 double oval;
} val_tab[50];
int val_stp;
struct val_rec *op;
struct val_rec *op2;
char typ,log,*add;
double val;
/*----------*/
int pop_val();
```

```
char *stpblk();
/*Parameter*/
char pbg=0x0f,pcl=0x0f,ptm,pbm;
```

```
/* Message.h */
char *hmes[] = {"   โปรแกรมนี้มีหน้าที่ ในการ
 จัดการข้อมูลซึ่ง งถูกเก็บอยู่ ในแฟ้มข้อมูล ",
"ถ้าท่านยังไม่มีแฟ้มข้อมูลอยู่ เลย
 ท่านจะต้องสร้างแฟ้มข้อมูลขึ้นมาเสียก่อน",
"เครื่องหมายที่ใช้",
"ตัวอักษร  หมายถึงให้พิมพ์ตาม เช่น
 สร้าง  หมายถึงพิมพ์คำว่า สร้าง",
"<...>          ,,    ให้ใช้ตามข้อกำหนดของมัน",
"[....]         ,,    สิ่ง งที่อยู่ ในวงเล็บนี้
  จะมีหรือไม่มีก็ได้",
"{.....}        ,,    ให้เลือกใช้สิ่ง งที่
 อยู่ ในวงเล็บนี้อย่าง ใดอย่าง หนึ่ง",
"โปรแกรมนี้ประกอบด้วยคำสั่ง งซึ่ง งทำหน้าที่ ต่าง ๆดังนี้คือ",
"1.{สร้าง, สร้างแฟ้ม} [ <ชื่อแฟ้ม> ]
    เช่น สร้าง      หรือ  สร้าง    B:MLIST",
"2.{เปิดแฟ้ม, ใช้แฟ้ม} [ <ชื่อแฟ้ม> ]
    เช่น เปิดแฟ้ม   ,,    เปิดแฟ้ม B:MLIST",
"3.ปิดแฟ้ม",
"4.{เพิ่ม, ใส่ ข้อมูล}",
"5.ไป <ตัวเลข>",
"6.{แก้ไข, แก้ที่ ผิด}",
"7.{โครงสร้าง, แสดงโครงสร้าง}",
"8.{ล้างจอ, ลบจอ}",
"9.{เลิก, เลิกทำงาน}",
"¥0"};
char mes1[] = "<คำสั่ง งสร้างแฟ้ม>";
char mes2[] = "   ชื่อฟิลด์              ชนิด ขนาด ทศนิยม";
char mes3[] = "ใส่ชื่อแฟ้ม:";
char mes4[] = "เกิดความผิดพลาดในการสร้างแฟ้ม";
char mes5[] = "ไม่สามารถเปิดแฟ้มได้";
char mes6[] = "เกิดปัญหาในการอ่านแฟ้มโครงสร้าง";
char mes7[] = "ยังไม่ได้เปิดแฟ้ม";
char mes8[] = "ยังไม่ได้ปิดแฟ้มปัจจุบัน";
char mes9[] = "เลขที่ รายการ มีค่ามากเกินไป";
char mes10[] = "ไม่มี รายการเลขที่นี้";
char mes11[] = "ผิดไวยากรณ์";
char mes12[] = "ไม่เข้าใจความหมายของคำสั่ง งนี้";
char mes13[] = " ข้อความนี้หมายถึง";
char mes14[] = "โปรดใส่หมายเลข(สองหลัก):";
char mes15[] = "อยู่ ที่ จุดสิ้นสุดแฟ้ม";
char mes16[] = "ภาษาสอบถามภาษาไทย สำหรับฐานข้อมูลแบบสัมพันธ์";
char mes17[] = "นำข้อมูลคนละชนิด มาเปรียบเทียบกัน";
char mes18[] = "เงื่อนไขมีข้อผิดพลาด";
char mes19[] = "ความยาวของรายการ = ";
```

```
/*Thaitab.h*/
/*Date 4-29-86*/

struct tok_rec {
        char *keyword;
        char  token;
} tok_tab[] = {
"<",60,"~",0,"=",61,"~",0,">",62,"~",0,
"!แสดง",75,"~",0,"จง",124,"~",0,
"คำนหาว่า",124,"คำนหา",124,"คำนว่า",
 124,"คำน",124,"~",0,
"ช่วยดำเนิน",86,"ช่วยคำน",123,"~",0,
"ชี้ว่า",124,"ชี้",124,"ชี้ว่าง",122,"~",0,
"ตอบว่า",124,"ตอบ",124,"ต้องการว่า",
 124,"ต้องการ",124,"~",0,
"บอกว่า",124,"บอก",124,"บรรยายว่า",
 124,"บรรยาย",124,"~",0,
"เลือกทำงาน",70,"เลือก",70,"เพื่อ",74,
"เปลี่ยนแปลง",72,"เผยว่า",124,"เผย",124,
"เสนอว่า",124,"เสนอ",124,"เท่าไร",125,
"เมื่อใด",125,"เท่ากับ",61,
"เท่ากับ",61,"~",0,
"แก้ไข",76,"แก้ทื่อผิด",76,"แสดงโครงสร้าง",
 73,"แสดงว่า",124,"แสดง",124,
"แจ้งว่า",124,"แจ้ง",124,"และ",34,"~",0,
"ไปที่",81,"ไป",81,"ไฉน",125,"ไหน",125,"~",0,
"ให้ดำเนินว่า",124,"ให้ดำเนิน",124,"ใด",125,
"ใคร",125,"ใส่ข้อมูล",74,"ใช้แปลง",72,
"~",0,
"สร้างแปลง",71,"สร้าง",71,"สำเนาไป",
 79,"สำเนา",79,"สำหรับ",104,"~",0,
"โครงสร้าง",73,"โดยใช้",101,"~",0,
"ลบแปลง",78,"ลบจอ",84,"ล้างจอ",84,"ลบ",77,"~",0,
"ทื่อไหน",125,"ทื่อใด",125,"ทื่อ",122,"ทำรายงาน",
 80,"ทำไม",125,"ทั้งหมด",103,
"ทราบว่า",124,"ทราบ",124,"~",0,
"รายงาน",80,"รายการทื่อ",100,"รายการ",
 100,"รับว่า",124,"รับ",124,"~",0,
"ถัดไป",102,"ถัด",102,"~",0,
"ปรากฏ",83,"ปลดแปลง",82,"~",0,
"มาชี้ว่า",124,"มาหน่อยว่า",124,
"มาว่า",124,"มากกว่า",62,"~",0,
"น้อยกว่า",60,"~",0,
"หรือ",35,"~",0,
"อะไร",125,"~",0,"ผม",123,"~",0,
"ดิฉัน",123,"~",0,"ฉัน",123,"~",0};
```

```
struct tok_rec *tokp;
struct commkey {
     char commch;
     int  commadd;
 } comktab[] = { '<',0,'=',2,'>',4,'!',6,
 'จ',8,'ค',10,'ช',15,'ซ',18,'ด',22,
 'บ',27,'เ',32,'แ',45, 'ไ',54,'ใ',59,'ส',
 66,'ิ',72,'ล',75,'ท',80,'ร',89,
 'ก',95,'ป',98,'ม',101,'น',106,'ห',108,'อ',
 110,'ผ',112,'ด',114,'จ',116,'¥0',0};

struct commkey *commp;
```

ประวัติผู้เขียน

นายดิเรก เอกบูรณะวัฒน์ เกิดที่จังหวัดนครปฐม เมื่อวันที่ 2 กรกฎาคม พ.ศ. 2501 ได้รับปริญญาวิทยาศาสตร์บัณฑิต สาขาวิชาวิทยาศาสตร์ทั่วไปจากคณะวิทยาศาสตร์ มหาวิทยาลัยสงขลานครินทร์ เมื่อปีการศึกษา 2523 ได้เข้าศึกษาต่อในระดับปริญญาโทในระดับ ปริญญามหาบัณฑิต ภาควิชาวิศวกรรมคอมพิวเตอร์ บัณฑิตวิทยาลัยจุฬาลงกรณ์มหาวิทยาลัย ปี พ.ศ. 2524