บรรณานุกรม

1. ศิริจันทร์ ทองประเสริฐ, สถิติสำหรับงานวิศวกรรม,โรงพิมพ์จุฬาลงกรณ์มหาวิทยาลัย,2533.
2. Ayres, Robert U. and Miller, Steven, "Robotics, CAM and Industrial Productivity", National Productivity Review, Winter, 1981, pp.42-60.
3. Black, J.H, Cost and Optimization Engineering, 2nd ed., McGraw-Hill, New York, 1985.
4. Canada, J.R and Sullivan, W.G, Economic and Multiattribute Evaluation of Advanced Manufacturing Systems, Prentice-Hall, New Jersey, 1989.
5. Chandra, J. and Schall, S., "Ecomomic Justification of Flexible Manufacturing Systems using the Leontif Input-output Model", Engineering Economist, 34(1), 1982, pp. 29-43.
6. Dilworth, J.B, Production and Operations Management, 4th ed., McGraw-Hill, New York, 1989.
7. Doherty, W. and Leigh, W., Decision Support and Expert Systems, South Western, Amsterdam, 1986.
8. Emery, K.W., "Justifying CIM is a Snap", in: Chiantella Nathan A.(ed.) Management Guide for CIM, The Computer and Automated Systems Association of SME, Dearborn, Michigan, 1986, pp. 57-60.
9. Grady, M.T., C! :Programming Principles and Practices , McGraw-Hill, New York, 1989.

ต้นฉบับ หน้าขาดหาย

21. Ogburn, W.L., "Excutive Perspective on CIM", in: Chiantella Nathan (ed.). **Management Guide for CIM**, The Computer and Automated Systems Association of SME, Dearborn, Michigan, 1986, pp. 24-26.

22. Riggs, J.L. and West, T.M, **Engineering Economics**, 3rd ed., McGraw-Hill, New York, 1986.

23. Singhal, K. and Meredith, J.R., "Research and Models for Automated Manufacturing", **Interfaces**, 17(6), November-December, 1987.

24. Smith, L.A. and Smith, J.L., "How can an IE justify a Human factors Program to Management? ", **Industrial Engineering**, 14(2), 1982, pp. 32-43.

25. Sullivan, W.G., "Models IEs can use to include strategic, Non-monetary factors in automation decisions", **Industrial Engineering**, March, 1986, pp. 42-60.

26. Tabucanon, M.T, **Multiple Criteria Decision Making in Industry**, Elsevier, Amsterdam, 1988.

27. Tarquin, J.H and Blank, L.T, **Engineering Economy**, 3rd ed., McGraw-Hill, New York, 1989.

28. Watson, H.J and Sprague, R.H., **Decision Support Systems**, 2nd ed., Prentice-Hall, New Jersey, 1989.

ภาคผนวก

รายละเอียดของโปรแกรม

NAME	EXT	SIZE
ALT	C	6555
ALTLST	C	1246
ASST	C	501
ATTR	C	11956
ATTRSCR	C	1349
BOAARD	C	9535
CUR	C	561
FNCT	C	8162
FNCT2	C	3374
GRAPH	C	3549
HARDERR	C	3291
INFO	C	755
LOAD	C	5635
MONET	C	4738
RANK	C	1199
REPORT	C	10677
SCORE	C	2796
SCREEN	C	5870
SHOW	C	864
TEXT	C	7043
UTIL	C	9048
UTILF	C	6227
ASST	H	80
BIOSAREA	H	1619
DEFINE	H	1879
DEFINE2	H	572
EXTERN	H	226
INCLUDE	H	346

รายชื่อโปรแกรมย่อยในระบบสนับสนุนการตัดสินใจ
เพื่อการประเมินโครงการการใช้วิธีอัตโนมัติในการผลิต

fUsIInsU ALT.C

ต้นฉบับ หน้าขาดหาย

```
#include "c:\bc\data\include.h"
#include <c:\bc\data\define2.h>
#include <c:\bc\data\extern.h>
extern int buff_key;

extern int attrindex[];
extern win boardtext;
win alttext={2,5,79,23,WHITE,BLUE,FALSE,0,RED};
int cureditprj=1;
char *period[]={"YEAR","MONTH"};
char *compound[]={"DISCRETE","CONTINUOUS"};
/*******/
altcontrol()
{
while(TRUE)
        {
        switch(altwriter())
                {
                case F10: endalt(); return;
                case F2 : break;
                }
        switch(altwriter2())
                {
                case F10: endalt(); return;
                case ESC: continue;
                }
        }
}

/****************************** altwriter */
int altwriter()
{
int rw[]={2};
int oldeditproject;
int gk;
ASST asst2110[]={{1,2111,"F2-Alernative Economic Detail; F10-Main menu; INS-insert;
DEL-delete"}};

cureditprj=1;

say("ALTERNATIVES","F10-MENU F2-ALTERNATIVE DETAIL","CUMAUM");
boxclr(&boardtext);
printcenter(&boardtext,1,"INPUT YOUR ALTERNATIVES' NAME (MAXIMUM 20
PROJECTS)");
row(&boardtext,1,rw);
assist(asst2110,1,&boardtext);
refreshproject();
hiliproject(cureditprj,1);
while(TRUE)
        {
        gk=getkey();
        oldeditproject=cureditprj;
```

```
                buff_key=gk;
                editstring(project[cureditprj-1]->name,cureditprj<=10 ? 12:52,\
                 cureditprj+(cureditprj<=10 ? 1 : -9),namelength,"");
                trim(project[cureditprj-1]->name,namelength-1);
                if(cureditprj==Nproject+1)
                        {
                        Nproject++;
                        defaultproject(Nproject);
                        }
                refreshproject();
                continue;
                }
        switch(gk)
                {
                case UP :cureditprj--; break;
                case DOWN: cureditprj++;break;
                case RIGHT: if(cureditprj<=10)cureditprj+=10;break;
                case LEFT : if(cureditprj>10)cureditprj-=10;break;
                case INS: insertproject(); refreshproject();break;
                case DEL : delproject();refreshproject();break;
/*              case F1 : help(".231") ; tx(&alttext); break;  */
                case F2 : checklastfield(); return gk;
                case F10 : if(newprogram())
                                                {
                                                checklastfield();
                                                return gk;
                                                }
                                        refreshproject();
                                        break;
                }
        adjust2(1,Nproject+1,&cureditprj);
        if(cureditprj>PROJECTMAX) cureditprj=PROJECTMAX;
        hiliproject(oldeditproject,0);
        hiliproject(cureditprj,1);
        assist(asst2110,1,&alttext);
        }
}
/***************************** refreshproject */
refreshproject()
{
int i,cl[]={41};
txclr(&alttext);
column(&alttext,1,cl);
for(i=1;i<=Nproject+1 && i<=PROJECTMAX;i++)
        {
        gotoxy(i<=10 ? 7 : 47,i<=10 ? i +1 :i -9);
        cprintf("%d",i);
        hiliproject(i,0);
        }
}

/***************************** insertproject */
insertproject()
{
```

```
/****************************** insertproject */
insertproject()
{
if(Nproject>=PROJECTMAX)   return;
memmove(project[cureditprj],project[cureditprj-1],(Nproject-cureditprj+1)*sizeof(struct
PRJ));
memset(project[cureditprj-1],0,sizeof(struct PRJ));
defaultproject(cureditprj);
Nproject++;
}

/*************************** defaultproject */
defaultproject(int curprj)
{
int i;
/**** default ****/
trim(project[curprj-1]->name,namelength-1);
project[curprj-1]->interest=10.;
project[curprj-1]->period=0;
project[curprj-1]->comp=0;

/* setup attribute value */
for(i=0;i<Nattravi;i++)
        project[curprj-1]->score[i]= attrnode[i]->positive ? attrnode[i]->lower : attrnode[i]-
>upper;
}

/********************************** delproject */
delproject()
{
if(Nproject<1 || (Nproject==PROJECTMAX-1 && cureditprj==PROJECTMAX)
                        ||cureditprj==Nproject+1) return;
if(Nproject>1)
        {
        memmove(project[cureditprj-1],project[cureditprj],(Nproject-
cureditprj+1)*sizeof(struct PRJ));
        memset(project[Nproject-1],0,sizeof(struct PRJ));
        trim(project[Nproject-1]->name,namelength-1);
        Nproject--;
        }
else
        {
        memset(project[0],0,sizeof(struct PRJ));
        trim(project[Nproject-1]->name,namelength-1);
        }
}
```

```
/****************************** hiliproject */
hiliproject(int prj,int hi)
{
if(hi) txinv(&alttext);
speak(prj<=10 ? 12 : 52,prj<=10 ? prj +1 : prj -9,project[prj-1]->name);
if(hi) txcolor(&alttext);
}

/*********************************/
checklastfield()
{
int i,j,k=YEARMAX*valuesize;
float *check;
for(j=Nproject-1;j>0;j--)
        {
        Nproject=j+1;
        for(i=0;i<namelength-1;i++)
                if(project[j]->name[i] != ' ') return;
        check = (float *)project[j]->value;
        for(i=0;i<k && check[i] == 0.0 ;i++);
        if (i < k) return;
        Nproject=j;
        }
}

/***************************/
int altwriter2()
{
int gk;
int cl[]={32,49,64};
int bcomp[PROJECTMAX];
int rowi=1,columni=1,oldrowi,oldcolumni,i;
ASST asst2100[]={{1,2111,"ESC-previous; F10-Main menu"}};
say("ALTERNATIVE ECONOMIC DETAIL","F10-MENU ESC-ALTERNATIVE
INPUT","CUMAUM");
boxclr(&boardtext);
column(&boardtext,3,cl);
speak(4,1,"ALTERNATIVE");
speak(36,1,"INTEREST");
speak(53,1,"PERIOD");
speak(66,1,"COMPOUND");
for(i=0;i<Nproject;i++)
        bcomp[i]=project[i]->comp;
for(i=0;i<Nproject;i++)
        {
        gotoxy(4,i+2); cputs(project[i]->name);
        gotoxy(36,i+2); cprintf("%10.2f",project[i]->interest);
        gotoxy(53,i+2); cprintf("%10s",period[project[i]->period]);
        gotoxy(66,i+2); cprintf("%11s",compound[project[i]->comp]);
        }
```

```
hilialt2(1,1,1);
assist(asst2100,1,&boardtext);
while(TRUE)
        {
        gk=getkey();
        oldrowi=rowi;
        oldcolumni=columni;
        if(decimal(gk) && columni==1)
                {
                float v,max=999.0;
                buff_key=gk;
                getfloat(&v,&max,36,rowi+1,10);
                project[rowi-1]->interest=v;
                project[rowi-1]->change=0;
                }
        switch(gk)
                {
                case UP: rowi--; break;
                case DOWN : rowi++; break;
                case RIGHT : columni++; break;
                case LEFT : columni--;break;
                case F10 : if(newprogram())
                                        {
                                        for(i=0;i<Nproject;i++)
                                                if(project[i]->comp!=bcomp[i])
                                                        NPVcalc(i);
                                        return gk;
                                        }
                case ESC : return gk;
                case RET : switch(columni)
                                        {
                                        case 2 : project[rowi-1]->period++;
                                                        adjust(0,1,&project[rowi-1]-
>period);
                                                        break;
                                        case 3 : project[rowi-1]->comp++;
                                                        adjust(0,1,&project[rowi-1]-
>comp);
                                                        break;
                                        }
                }
                adjust(1,3,&columni);
                adjust(1,Nproject,&rowi);
                hilialt2(rowi,columni,1);
                if(oldrowi != rowi || oldcolumni != columni)
                        hilialt2(oldrowi,oldcolumni,0);
         assist(asst2100,1,&boardtext);
         }
}
/*******************/
hilialt2(int rowi,int columni,int flag)
{
if(flag) txinv(&boardtext) ;
```

```
switch(columni)
        {
        case 1: gotoxy(36,rowi+1);
                        cprintf("%10.2f",project[rowi-1]->interest);
                        break;
        case 2: gotoxy(53,rowi+1);
                        cprintf("%10s",period[project[rowi-1]->period]);
                        break;
        case 3: gotoxy(66,rowi+1);
                        cprintf("%11s",compound[project[rowi-1]->comp]);
                        break;
        }
if(flag) txcolor(&boardtext) ;
}
/**********************/
endalt()
{
anacalc();
rankproject();
}
```

fUsIInsU ALTLST.C

```
#include <c:\bc\data\include.h>
#include  "c:\bc\data\define2.h"
#include "c:\bc\data\extern.h"
extern win boardtext;
extern int buff_key;
extern int attrindex[ATTRMAX];
extern float shiftlower,shiftupper;
extern win alttext;
/**********************/
attrlist()
{
int cl[1]={41},rw[1]={2};
int attr=1,oldattr,i;
say("ATTRIBUTES LIST","F10-menu","CUMAUM");
boxclr(&boardtext);
row(&boardtext,1,rw);
tx(&boardtext);
column(&alttext,1,cl);
for(i=0;i<Nattravi && i< 20 ;i++)
        speak (5,i+1,attrnode[i]->name);
for(;i<Nattravi;i++)
        speak (45,i-20,attrnode[i]->name);

hiliattrlst(1,1);
while(TRUE)
        {
        oldattr=attr;
        switch(getkey())
                {
                case UP : attr--; break;
                case DOWN : attr++; break;
                case RIGHT : if (attr<21) attr += 20 ; break;
                case LEFT : if( attr >20) attr -=20 ; break;
                case HOME : attr=0; break;
                case END : attr=Nattravi; break;
                case F10: if(newprogram()) return;
                                        tx(&alttext);
                                        break;
                }
        adjust2(1,Nattravi,&attr);
        hiliattrlst(attr,1);
        if(attr!=oldattr) hiliattrlst(oldattr,0);
        }
}
/***************/
hiliattrlst(int attr,int hi)
{
if(hi) txinv(&alttext);
speak(attr <21 ? 5: 45 , attr <21 ? attr : attr-21,attrnode[attr-1]->name);
if(hi) txcolor(&alttext);
}
```

โปรแกรม ASST.C

```
#include "c:\bc\data\include.h"
extern win usertext;

int assidnow=0;
/*********************/
int assist(ASST *asst,int assid,win *x)
{
int i;
int Nasst;
if(asst==NULL ) return;
Nasst=asst[0].key;
if(Nasst !=1)
        {
        for(i=1;i<Nasst;i++)
                if(assid==asst[i].key) break;
        if(i>=Nasst) return FALSE;
        }
else i=0;
if(asst[i].assid == assidnow) return FALSE;
tx(&usertext);

if(asst[0].assid==1)cputs(asst[0].label);

cprintf("%s",asst[i].label);
clreol();
tx(x);
return TRUE;
}
```

fUsiinsu ATTR.C

```
#include "c:\bc\data\include.h"
#include "c:\bc\data\define2.h"
ATTR far *(useattr[ATTRMAX]);
void far * attrdocalloc(ATTR far * x);

ATTR    attribute[ATTRFIX]= {
{"NET PRESENT VALUE      ","BAHT ",1,1,1, 1,0,999999.99,  1,0},
{"PAYBACK PERIOD         ","Year ",1,1,1, 1,0,20,         1,0},
{"FLEXIBILITY            ","score",1,1,1,  1,0,100,       1,0},
{"SAFETY                 ","score",1,1,1,   1,0,100,      1,0},
{"COMPATIBILITY          ","score",1,1,1,      1,0,100,          1,0},
{"MAINTAINABILITY        ","score",1,1,1,      1,0,100,          1,0},
{"PRODUCT'S QUALITY      ","score",1,1,1,    1,0,100,          1,0},
{"BETTER INFORMATION SYSTEM","score",1,1,1,    1,0,100,              1,0},
{"STAFF UPGRADING        ","score",1,1,1,     1,0,100,          1,0},
{"VENDOR VIABILITY       ","score",1,1,1,     1,0,100,          1,0},
{"COMPANY'S IMAGE IMPROVED ","score",1,1,1,     1,0,100,              1,0}};


ATTR far *(attrnode[ATTRMAX]);

extern win fulltext,boardtext,fulltext;
extern int buff_key;
extern int Nproject;
extern struct PRJ far *project[];

win attrlisttext={49,4,79,23,WHITE,BLUE,FALSE,0,RED};
win doctext={2,5,47,23,WHITE,BLUE,FALSE,0,RED};
win attrpoptext={10,10,60,21,WHITE,BLUE,FALSE,1,WHITE};

int Nattravi=0;
int Nattrselect=ATTRFIX;

int userdf=1;
int line1;
int attrindex[ATTRMAX];
/*************************/
makeattrnode()
{
int i;
for(i=0;i<ATTRFIX;i++)
        {
        attrnode[i]=&attribute[i];
        }
Nattravi=ATTRFIX;
}
extern ASST * asstgb;

/*************************/
int attrwriter()
{
int attr=1;
int oldattr;
int col[]={48};
```

```
int line1=1;
int Nline=19;
ASST asst2300[]={{3,1,"ENTER-toggle SELECT/DESELECT F2-input detail F10-
mainmenu "},
                                {1,2301,""},
                                {0,2302,"INS-insert DEL-delete"}};
say("ATTRIBUTE SELECTION","ESC-previous F2-continue","CUMAUM");
boxclr(&boardtext);
column(&boardtext,1,col);
speak(59,1,"ATTRIBUTE");
speak(20,1,"DETAIL");
speak(1,2,"ATTRIBUTE :");

refreshattr(1,Nline);
hiliattr(1,1);
assist(asst2300,1,&boardtext);
docwriter(attr);
/********** entry */
while(TRUE)
        {
        oldattr=attr;
        switch(getkey()){
                case UP : attr--;break;
                case DOWN : attr ++;   break;
                case HOME : attr=1;             break;
                case END : attr=Nattravi;       break;
                case 'x' :
                case 'X' :
                case SPACE:
                case RET : attrnode[attr-1]->use = !attrnode[attr-1]->use;
                                        speak(51,attr-line1+2,attrnode[attr-1]->use ? "X" : "
" );
                                        break;

                case INS : if ( insertattr(attr-1) )refreshattr(line1,Nline+line1-1);
                                        break;
                case DEL : if ( delattr(attr-1) )   refreshattr(line1,Nline+line1-1);
                                        break;

                case F2 :   if (attr<=ATTRFIX) attrpoptext.wbt=14;
                                        else attrpoptext.wbt=20;        /* change size
popup window */
                                        attrpop(attr);
                                        tx(&boardtext);
                                        break;
                case F10 :  makeuseattr();
                                        if(newprogram())
                                                {
                                                endattr();
                                                return;
                                                }
                                        else tx(&boardtext);
                                        break;
                }
```

```
        adjust2(1,Nattravi,&attr);
        if (scrollpage(&line1,line1+Nline-1,attr))
                refreshattr(line1,Nline+line1-1);

/*      if(attr <line1)
                {
                line1=attr;
                refreshattr (line1,Nline);
                }
        if(attr >line1+Nline-2)
                {
                line1=attr-Nline+1;
                refreshattr (line1,Nline);
                }
        if(attr < line1 || attr > line1+ Nline -1)
                {
                line1 = attr;
                refreshattr(line1,Nline);
                }
*/
        hiliattr(oldattr,oldattr-line1+1,0);
        hiliattr(attr,attr-line1+1,1);
        docwriter(attr);
        assist(asst2300,attr < ATTRFIX ? 1 :0,&boardtext);
        }
}

/************************/
int scrollpage(int *lineF,int lineL,int lineN)
{
if (lineN<*lineF)
        {
        *lineF=lineN;
        return TRUE;
        }
if (lineN>lineL)
        {
        *lineF+=lineN-lineL;
        return TRUE;
        }
return FALSE;
}

/************************/
refreshattr(int line1,int lineL)
{
int i;
txclr(&attrlisttext);
tx(&boardtext);
for(i=line1;i<=lineL && i <= Nattravi ;i++)
        {
        gotoxy(50,2+i-line1);
        cprintf("[%1s] %s",attrnode[i-1]->use ? "X" : " " ,attrnode[i-1]->name);
        }
```

```
}
/*************************/
endattr()
{
anacalc();
}
/*************************/
hiliattr(int attr,int atline,int hi)
{
if(hi) txinv(&boardtext);
speak(54,1+atline,attrnode[attr-1]->name);
if(hi)
        {
        txcolor(&boardtext);
        gotoxy(15,2); cprintf("%-25s",attrnode[attr-1]->name);
        }
}



/*************************/
attrpop(int attr)
{
int gk,i;
int x=attr-1;
int y=1,oldy;
void far **bin=NULL;
ATTR attrbuff;
ASST asst2320[]={{7,1,"F2-Attribute Selection Menu; "},
                                {1,2321,""},
                                {2,2322,"Input Unit Description"},
                                {3,2323,"Input Maximum Value"},
                                {4,2324,"Input Minimum Value"},
                                {5,2325,"ENTER-toggle MINIMIZE/MAXIMIZE"},
                                {6,2326,"Write your Attribute Description"}};

/*** back up attr value ****/
memmove(&attrbuff,attrnode[x],sizeof(ATTR));

if(textmalloc(&attrpoptext,bin)!=FALSE)
        {
        boxclr(&attrpoptext);
        speak(2,1,"ATTRIBUTE :");       hiliattrpop(&attrbuff,1,1);
        speak(2,2,"UNIT      :");
        speak(2,3,"MAXIMUM");
        speak(2,4,"MINIMUM");
        speak(2,5,"OBJECTIVE :");
        if(attr>ATTRFIX)        speak(10,6,"DESCRIPTION : ( MAX 5 LINE )");

        for(i=2;i<=10 && (attr > ATTRFIX || i<= 5);i++)
                hiliattrpop(&attrbuff,i,0);

        assist(asst2320,1,&attrpoptext);
```

```
while(TRUE)
        {
        oldy=y;
        gk=getkey();
        switch(y) {
        case 1 : if(string(gk) && !attrbuff.fix)
                                {
                                buff_key=gk;
                                editstring(attrbuff.name,15,y,26,"");
                                }
                        break;
        case 2 : if(attrbuff.unitchange && string(gk) )
                                {
                                buff_key=gk;
                                editstring(attrbuff.unit,15,y,6,"");
                                }
                        break;

        case 3 : if(decimal(gk))
                                {
                                float v,max=9999999.0;
                                buff_key=gk;
                                getfloat(&v,&max,25,y,10);
                                attrbuff.lower=v;
                                }
                        break;
        case 4 : if(decimal(gk))
                                {
                                float v,max=9999999.0;
                                buff_key=gk;
                                getfloat(&v,&max,25,y,10);
                                attrbuff.upper=v;
                                }
                        break;
        case 6:
        case 7:
        case 8:
        case 9:
        case 10: if(string(gk))
                                {
                                buff_key=gk;
                                editstring(attrbuff.doc[y-6],2,y+1,45,"");
                                }
                                break;
        }
        switch(gk)
                {
                case UP : y--; break;
                case TAB:
                case DOWN :   y++; break;

                case RET  : attrbuff.positive=!attrbuff.positive; break;

/*              case LEFT : if(y==5 && attrbuff.positive) attrbuff.positive=0;
```

```
                                        break;
                        case RIGHT: if(y==5 && !attrbuff.positive) attrbuff.positive=1;
                                        break;

*/                      case F2 :       endattrpop(&attrbuff,attr);
                                        pushdown(&attrpoptext,*bin);
                                        return;
                        }
                adjust(1,x<ATTRFIX ? 5:10,&y);
                hiliattrpop(&attrbuff,oldy,0);
                hiliattrpop(&attrbuff,y,1);
                assist(asst2320,min(y,6),&attrpoptext);
                }
        }
}
/*******************/
endattrpop(ATTR far * attrbuff,int attr)
{
int x=attr-1,j;
float *buff; /* shold be revis later */
if(attrbuff->lower != attrnode[x]->lower ||
        attrbuff->upper != attrnode[x]->upper ||
        attrbuff->positive != attrnode[x]->positive ||
        strncmp(attrbuff->unit,attrnode[x]->unit,6) != 0 ||
        strncmp(attrbuff->name,attrnode[x]->name,25) != 0 )
        {
        memmove(attrnode[x],attrbuff,sizeof(ATTR));

        /* reset value */
        for(j=0;j<Nproject;j++)
                {
                buff=&project[j]->score[attrindex[x]];
                maxfloat(buff,buff,&attrnode[x]->lower);
                minfloat(buff,buff,&attrnode[x]->upper);
                }
        attrnode[x]->type=0;
        attrnode[x]->utilchange=0;
        if(attrnode[x]->use)
                anacalc();
        }
}
char *positive[]={"MINIMIZE","MAXIMIZE"};

/*********************/
hiliattrpop(ATTR far * attrbuff,int y,int hi)
{
if(hi) txinv(&attrpoptext);
switch(y)
        {
        case 1: speak(15,1,attrbuff->name); break;
        case 2: speak(15,2,attrbuff->unit);
                        txcolor(&attrpoptext);

                        gotoxy(15,3); cprintf("%5s =",attrbuff->unit);
```

```
                        gotoxy(15,4); cprintf("%5s =",attrbuff->unit);
                        break;

        case 3: gotoxy(25,3); Tprintf(&attrbuff->lower); break;
        case 4: gotoxy(25,4); Tprintf(&attrbuff->upper); break;

        case 5: speak(15,5,positive[attrbuff->positive]);
                        txcolor(&attrpoptext);
                        speak(25,5,attrbuff->positive ? "MORE IS BETTER":
                                "LESS IS BETTER");
                        break;

/*      case 5: speak(attrbuff->positive ? 21 :11 ,5,"X");
                        txcolor(&attrpoptext);
                        speak(attrbuff->positive ? 11 : 21,5," " );
                        break;
*/
        case 6:
        case 7:
        case 8:
        case 9:
        case 10: speak(2,y+1,attrbuff->doc[y-6]); break;

        }
if(hi) txcolor(&attrpoptext);
}


/*************************/
docwriter(int attr)
{
int i;
txclr(&doctext);
        for(i=0;i<5;i++)
                if(attr>ATTRFIX)
                        cprintf("%s\n\r",attrnode[attr-1]->doc[i]);
                else
                        cprintf("%s",attrnode[attr-1]->doc[i]==NULL ? "": attrnode[attr-1]-
>doc[i]);
tx(&boardtext);
}

/*************************/
/*attrfixdoc()
{
int i;
for(i=0;i<ATTRFIX;i++)
        attrdocalloc(attrnode[i]);
} */

/*********************/
void far * attrdocalloc(ATTR far * attrp)
{
int i;
```

```
void far * (docbuff[5]);
for(i=0;i<5;i++)
        {
        attrp->doc[i]=(char far *)farcalloc(1,46);
        trim(attrp->doc[i],45);
        }

return docbuff;
}
/*********************/
attrclr(int x)
{
attrdocclr(x);
farfree(attrnode[x]);
attrnode[x]=NULL;
}

/*********************/
attrdocclr(int x)
{
int i;
for (i=0;i<5;i++)
        {
        farfree(attrnode[x]->doc[i]);
        attrnode[x]->doc[i]=NULL;
        }
}
/*********************/
makeuseattr()
{
int i,j=0;
for(i=0;i<Nattravi;i++)
        {
        if(!attrnode[i]->use) continue;
        useattr[j]=attrnode[i];
        attrindex[j]=i;
        j++;
        }
Nattrselect=j;
}
/*********************/
int initattr(int x)
{
attrnode [x] = (ATTR far *) farcalloc( 1,sizeof(ATTR) ) ;
attrdocalloc(attrnode[x]);
}

/*********************/
int insertattr(int x)
{
int i;
ATTR far * buff;
if( x < ATTRFIX -1 || Nattravi >= ATTRMAX ) return FALSE ;
if( ( buff=(ATTR far *) farcalloc( 1,sizeof(ATTR) ) )==NULL ) return FALSE;
```

```
memmove(&attrnode[x+2],&attrnode[x+1],(Nattravi-x-1)*sizeof(ATTR far *));

/*** set ***/
attrnode[x+1]=(ATTR far *) buff;

/*** move value in project ***/
for(i=0;i<Nproject;i++)
        {
        memmove(&project[i]->score[x+2],&project[i]->score[x+1],(Nattravi-x-
1)*sizeof(float));
        project[i]->score[x+1]=0;
        memmove(&project[i]->util[x+2],&project[i]->util[x+1],(Nattravi-x-1)*sizeof(float));
        project[i]->util[x+1]=0;
        }

/*** change attrindex ***/
Nattravi++;
makeuseattr();

        /*** default **/
if(userdf==99) userdf=0;
sprintf(attrnode[x+1]->name,"%-22s %2d","USER DEFINE",userdf++);
strcpy(attrnode[x+1]->unit,"score");
attrdocalloc(attrnode[x+1]);
attrnode[x+1]->fix=0;
attrnode[x+1]->unitchange=TRUE;
attrnode[x+1]->lower=0;
attrnode[x+1]->upper=100;
attrnode[x+1]->positive=1;
return TRUE;
}
/********************/
int delattr(int x)
{
int i;
if( x < ATTRFIX ) return FALSE;

attrdocclr(x);

farfree(attrnode[x]);
memmove(&attrnode[x],&attrnode[x+1],(Nattravi-x-1) *sizeof(ATTR far *));

attrnode[Nattravi-1]=NULL;

        /*** move value in project ***/
        for(i=0;i<Nproject;i++)
                {
                memmove(&project[i]->score[x],&project[i]->score[x+1],(Nattravi-x-
1)*sizeof(float));
                memmove(&project[i]->util[x],&project[i]->util[x+1],(Nattravi-x-
1)*sizeof(float));
                }
```

```
        /*** change attrindex ***/
        Nattravi--;
        makeuseattr();
return TRUE;
}



/*************************/
attrdoc()
{
FILE *fp;
char far * docbuff;
char * mark;
int docsize,stack;
int i,times,j;
if((docbuff = farcalloc (1,46*15))!=NULL)
        {
        if((fp=fopen("cumaum.doc","rb")) != NULL)
                {
                for(i=0;i<ATTRFIX &&  !feof (fp);i++)
                        {
                        stack=fread(docbuff,1,46*15,fp);
                        if(stack>=46*15) stack=46*15;
                        docbuff[stack-1]=0;     /* make null-terminate */
                        mark=strstr(docbuff,"@"); /*mark document*/
                        strncpy(mark,"\0",1);
                        docsize=strlen(docbuff);
                        times=0.5+docsize/(46*3);
                        for(j=0;j<5 && j <= times;j++)
                                {
                                attribute[i].doc[j]=farcalloc(1,46*3+1);
                                trim(attribute[i].doc[j],46*3);

                                strncpy(attribute[i].doc[j],&docbuff[j*46*3],min(46*3,strlen(
&docbuff[j*46*3])));
                                }
                          /*    attribute[i].doc[0]=farmalloc(docsize);
                                strcpy(attribute[i].doc[0],docbuff);      */
                                fseek(fp,(46*15-docsize-1+stack-46*15)* -1,SEEK_CUR);
                                }
                fclose(fp);
                }
        farfree(docbuff);
        }
}
```

fujsiinsu ATTRSCR.C

```
#include "c:\bc\data\include.h"
#include "c:\bc\data\define2.h"
#include <c:\bc\data\extern.h>


extern win fulltext,fulltext2;
struct BAR attrbar;
char legend[]={"score"};

/**** 444444444444444444444444444 **************/
attrscale()
{
int i;
ASST asst2400[]={{1,2411,"Use \x1B\x1A to weight Attribute; F10-Main menu"}};

for(i=0;i<Nattrselect;i++)
        {
        attrbar.name[i]=useattr[i]->name;
        attrbar.value[i]=&useattr[i]->scaling;
        }
attrbar.Allrec=Nattrselect;
attrbar.Nrecord=min(16,Nattrselect);
attrbar.min=0;
attrbar.max=100;
attrbar.namesize=25;
attrbar.decimal=FALSE;
attrbar.legend=legend;

/****/

txclr(&fulltext2);
say("ATTRIBUTE WEIGHTING","ESC-previous F2-continue","CUMAUM");
assist(asst2400,1,&fulltext);
switch(barwriter(&attrbar))
        {
        case F10: if(newprogram())
                        {
                        endattr4();
                        return;
                        }
assist(asst2400,1,&fulltext);
        }
}
/**********/
attrscaling(struct BAR *strip)
{
int i;
for(i=0;i<Nattrselect;i++)
         useattr[i]->scaling=*strip->value[i];
}
/**********/
attrratio()
{
int i;
```

```
float sum=0;
for(i=0;i<Nattrselect;i++)
        sum+=useattr[i]->scaling;
for(i=0;i<Nattrselect;i++)
        if(sum==0)useattr[i]->ratio=1/Nattrselect;
        else    useattr[i]->ratio=useattr[i]->scaling/sum;
}
/**********************/
endattr4()
{
attrscaling(&attrbar);
attrratio();
anacalc();
}
```

fUsiinsu BOARD.C

```
#include "c:\bc\data\include.h"
char *cashlabel[]={"INVESTMENT . CAPITALIZED",
"Equipment cost",
"Accessories,tooling cost",
"Other",
"TOTAL capital investment",
"Investment tax credit (%)",
"Investment tax credit",
"INVESTMENT . EXPENDED",
"Engineering",
"Installation",
"Startup",
"Other",
"Total expended Investment",
"Total invest. after credit",
"OPERATING SAVINGS (COSTS)",
"Direct Labor savings",
"Indirect Labor savings",
"Inventory savings",
"Maintenance savings",
"Other savings",
"Other costs",
"Total operating saving",
"ANALYSIS",
"Depreciation",
"Net before tax saving",
"Net after tax saving rate%",
"Net after tax cash saving",
"Net after tax cash flow"};

win boardtext={2,3,79,23,WHITE,BLUE,FALSE,0,RED};
win cashtext={2,5,79,23,WHITE,BLUE,FALSE,0,RED};
extern int buff_key;
extern win fulltext;
extern int Nproject;
extern char *period[2];
int startboardfield[2]={0,8};
int startcfield[2]={0,4};
int projectNO;
int boardNO=1;
int startyear;
int curyear;
int curvline;
int Nvalidline[]={8,8};
int Nvalidcline[]={4,4};
int validcline[][5]={{5,7,15,16},{8,12,14,16}};
int validline[][10]={
                                        {2,3,4,6,11,12,13,14,0,0},
                                        {2,3,4,5,6,7,11,13,0,0}
                                        };

win destext={2,3,34,23,WHITE,BLUE,FALSE,0,RED};

struct PRJ far *project[PROJECTMAX];
```

```
struct PRJ   projectdemo[1]={{"ALTERNATIVE    NO. 1 "}};


/************************/
projectalloc()
{
int i;
for(i=0;i<PROJECTMAX;i++)
        {
        project[i]=farcalloc(1,sizeof(struct PRJ));
        trim(project[i]->name,namelength-1);
        }
}
/************************/
demo()
{
int i;
for(i=0;i<1;i++)
memcpy(project[i],&projectdemo[i],sizeof(struct PRJ));
recalcall(); /*****/
checklastyear();
monetcalc();
makeuseattr();
attrratio();
anacalc();
}
/************************/

boardwriter()
{
int col[]={36,47,58,69},rw[]={2};
int ncol=4;
int oldboardNO=-1;
win desctext={2,5,35,23,WHITE,BLUE,FALSE,0,RED};
projectNO=1;
startyear=1;
curyear=curvline=1;
if(projectNO>Nproject) projectNO=Nproject;
say("CASH FLOW","F10-menu ESC-alternatives","CUMAUM");
while (TRUE)
        {
        if(boardNO!=oldboardNO)   /* for newboard */
                {
                if(oldboardNO !=-2)    /* for first time */
                        {
                        boxclr(&boardtext);
                        refreshprj(); /* rewrite attribute's name */
                        row(&boardtext,1,rw);
                        column(&cashtext,ncol,col);
                        }
                txclr(&desctext);
                if(boardNO==1) board1();
                else board2();
                oldboardNO=boardNO;
```

```
                tx(&cashtext);
                }
        refreshdata();
        calcdisplay();
        if(fieldentry()==F10)
                if(newprogram()==TRUE)
                        {
                        endcashflow();
                        return;
                        }
                else tx(&cashtext);
        }
}
/*******************/


board1()  /* can concise */
{
gotoxy(1,1);
cprintf("%-27s\n\r",cashlabel[0]);
cprintf("...1) %-27s\n\r",cashlabel[1]);
cprintf("...2) %-27s\n\r",cashlabel[2]);
cprintf("...3) %-27s\n\r",cashlabel[3]);
cprintf("** %-27s\n\r",cashlabel[4]);
cprintf("...4) %-27s\n\r",cashlabel[5]);
cprintf("** %-27s\n\r\n\r\n\r",cashlabel[6]);
cprintf(" %-27s\n\r",cashlabel[7]);
cprintf("...5) %-27s\n\r",cashlabel[8]);
cprintf("...6) %-27s\n\r",cashlabel[9]);
cprintf("...7) %-27s\n\r",cashlabel[10]);
cprintf("...8) %-27s\n\r",cashlabel[11]);
cprintf("** %-27s\n\r",cashlabel[12]);
cprintf("...%-27s\n\r",cashlabel[13]);
}
/*************************/
board2()
{
gotoxy(1,1);
cprintf("%-27s\n\r",cashlabel[14]);
cprintf("...9) %-27s\n\r",cashlabel[15]);
cprintf("..10) %-27s\n\r",cashlabel[16]);
cprintf("..11) %-27s\n\r",cashlabel[17]);
cprintf("..12) %-27s\n\r",cashlabel[18]);
cprintf("..13) %-27s\n\r",cashlabel[19]);
cprintf("..14) %-27s\n\r",cashlabel[20]);
cprintf("%-27s\n\r\n\r",cashlabel[21]);
cprintf("%-27s\n\r",cashlabel[22]);
cprintf("..%-27s\n\r",cashlabel[23]);
cprintf("..%-27s\n\r",cashlabel[24]);
cprintf("..%-27s\n\r",cashlabel[25]);
cprintf("..%-27s\n\r\n\r",cashlabel[26]);
cprintf("..%-27s\n\r",cashlabel[27]);
}
/*******************/
```

```
refreshdata()
{
int i,j;
for(i=0;i<4;i++)
        {
        gotoxy(cellNO(i+1)+1,1);
        cprintf("%-5s %2d",period[project[projectNO-1]->period],startyear+i-1);
        for(j=0;j<Nvalidline[boardNO-1];j++)
                {
                gotoxy(cellNO(i+1),validline[boardNO-1][j]);
                Tprintf(&project[projectNO-1]->value[startyear-
1+i][j+startboardfield[boardNO-1]]);
/*              cprintf("%10.2f",project[projectNO-1]->value[startyear-
1+i][j+startboardfield[boardNO-1]]);
*/              }
        }
calcdisplay();
}

/*************************/
int cellNO(int year)
{
return(36+(year-1)*11);
}
/**********************/
int fieldentry()
{
int gk;
int oldcurvline;
int oldcuryear;
ASST asst2210[]={{3,0,""},
                                {1,2211,"ESC-Alternative change F10-Main menu PgDn-
continue"},
                                {2,2212,"ESC-Alternative change F10-Main menu PgUp-
previous"}};
hili(curvline,curyear,1);
assist(asst2210,boardNO,&cashtext);
while((gk=getkey()) != F10)
        {
        oldcurvline=curvline;
        oldcuryear=curyear;
        if(decimal(gk))
                {
                float v,max=999999999.0;
                buff_key=gk;
                getfloat(&v,&max,cellNO(curyear-startyear+1),
                                validline[boardNO-1][curvline-1],10);
                project[projectNO-1]->value[curyear-1][startboardfield[boardNO-
1]+curvline-1]=v;
                project[projectNO-1]->change=0;
                recalc(projectNO-1,curyear-1);
                calcdisplay();
                }
```

```
        switch(gk)
                {
/*              case F1: help(boardNO==1 ? ".121" : ".122");
                                tx(&cashtext);break; */
                case UP : curvline--;break;
                case DOWN :curvline++;break;
                case LEFT : curyear--;break;
                case RIGHT : curyear++;break;
                case TAB  : curyear+=4;break;
                case 3840  : curyear-=4;break;
                case HOME  : curyear=1; break;
                case END   : curyear=YEARMAX;break;
                case PgUp : if(boardNO==2)
                                                {
                                                boardNO=1;
                                                return gk;
                                                }
                                        break;
                case PgDn : if(boardNO==1)
                                                {
                                                boardNO=2;
                                                return gk;
                                                }
                                        break;
                case ESC : projectpop();
                                        refreshdata();
                }
        if(curvline<1) curvline=1;
        else if(curvline>Nvalidline[boardNO-1]) curvline=Nvalidline[boardNO-1];
        if(curyear<1) curyear=1;
        else if(curyear>YEARMAX) curyear=YEARMAX;
        if(curyear<startyear || curyear >startyear+3)
                {
                if(curyear<startyear) startyear=curyear;
                else startyear=curyear-3;
                refreshdata();
                }
        else hili(oldcurvline,oldcuryear,0);
        hili(curvline,curyear,1);
        assist(asst2210,boardNO,&cashtext);
        }
return gk;
}
/************************/
hili(int vline,int year,int hi)
{
if(hi) txinv(&cashtext);
gotoxy(cellNO(year-startyear+1),validline[boardNO-1][vline-1]);
Tprintf(&project[projectNO-1]->value[year-1][startboardfield[boardNO-1]+vline-1]);
/*cprintf("%10.2f",project[projectNO-1]->value[year-1][startboardfield[boardNO-1]+vline-
1]);
*/
if(hi) txcolor(&cashtext);
}
```

```
Tprintf(float *x)
{
cprintf(fabs((double)*x) > 999999.99
                        ? (fabs((double)*x) > 9999999.9 ? "%10.0f" : "%10.1f")
                        : "%10.2f",*x);
}




win projecttext={40,10,67,15,WHITE,BLUE,FALSE,0,RED};
menubar projectmenu={20,2,1,BLACK,WHITE,NULL};
/*******************/
projectpop()
{
char *(charbuff[PROJECTMAX]);
void far **bin=NULL;
int ch,i;
int legal[]={0};
ASST asst2330[]={{1,2231,"Use ↑↓-select; ESC-previous"}};
projectmenu.Nchoice=Nproject;
projecttext.wt=13-Nproject/2;
projecttext.wbt=Nproject+projecttext.wt-1;
for(i=0;i<PROJECTMAX;i++)
        {
        charbuff[i]=project[i]->name;
        }
projectmenu.choice=charbuff;
textmalloc(&projecttext,bin);
if((ch=popup(&projecttext,&projectmenu,legal,asst2330))==ESC)
        {
        pushdown(&projecttext,*bin);
        tx(&cashtext);
        return;
        }
else {
        pushdown(&projecttext,*bin);
        tx(&cashtext);
        if(projectNO != projectmenu.bar)
                {
                projectNO=projectmenu.bar;
                refreshprj();
                refreshdata();
                }
        }
}
/***************************/
recalc(int projectx,int yearx)
{
float *calcbuff=project[projectx]->calc[yearx];       /* should be change to pointer */
float *valuebuff=project[projectx]->value[yearx];
```

```
calcbuff[0]=valuebuff[0]+valuebuff[1]+valuebuff[2]; /*Total cap invest*/
calcbuff[1]=calcbuff[0]/100*valuebuff[3]; /* investment tax credit */
calcbuff[2]=valuebuff[4]+valuebuff[5]+valuebuff[6]+valuebuff[7];/* Total exp invest*/
calcbuff[3]=calcbuff[0]+calcbuff[2]-calcbuff[1];/*Total invest after credit*/
calcbuff[4]=valuebuff[8]+valuebuff[9]+valuebuff[10]+valuebuff[11]+\
                        valuebuff[12]-valuebuff[13];/*Total ops saving*/

calcbuff[5]=calcbuff[4]-valuebuff[14];/*Net before tax saving*/
calcbuff[6]=valuebuff[14]+calcbuff[5]/100*(100-valuebuff[15]);/*net after tax saving*/
calcbuff[7]=calcbuff[6]-calcbuff[3];/*Net after tax cash flow*/
}
/***************/
recalcall()
{
int i,j;
for(i=0;i<Nproject;i++)
        {
        for(j=0;j<YEARMAX;j++)
                {
                recalc(i,j);
                }
        }
}
/*************************/
calcdisplay()
{
int i,j;
txinv(&cashtext);
for(i=0;i<4;i++)
        {
        for(j=0;j<Nvalidcline[boardNO-1];j++)
                {
                gotoxy(cellNO(i+1),validcline[boardNO-1][j]);
                Tprintf(&project[projectNO-1]->calc[startyear-1+i][j+startcfield[boardNO-
1]]);
                }
        }
tx(&cashtext);
}
/*****************/
refreshprj()
{
tx(&boardtext);
cprintf("ALTERNATIVE NUMBER %2d   : %25s",projectNO,project[projectNO-1]->name);
tx(&cashtext);
}
/******************/
checklastyear()
{
int i,j,p,use;
for(p=0;p<Nproject;p++)
        {
        if(project[p]->change!=0) continue;
        project[p]->yearuse=1;
```

```
        for(i=YEARMAX-1;i>=0;i--)
                {
                use=FALSE;
                for(j=0;j<valuesize;j++)
                        if(project[p]->value[i][j]!=0)
                                {
                                use=TRUE;
                                break;
                                }
                if(use==TRUE)
                        {
                        project[p]->yearuse=i+1;
                        break;
                        }
                }
        }
}
/**********************/
endcashflow()
{
checklastyear();
monetcalc();
}
```

โปรแกรม CUR.C

```
/*      CURSOR.C:  Cursor appearance functions  */

#include "c:\bc\data\include.h"
extern BIOSDATA far * bios;
void curson (void);
void cursoff (void);

void cursoff (void)
{
union REGS reg;

        reg.h.ah = 1;
        reg.h.ch = bios->cursTop | 0x20;
        reg.h.cl = bios->cursBottom;
        int86 (0x10, &reg, &reg);
} /*-------------------------------------*/

void curson (void)
{
union REGS reg;

        reg.h.ah = 1;
        reg.h.ch = bios->cursTop & 0xDF;
        reg.h.cl = bios->cursBottom;
        int86 (0x10, &reg, &reg);
} /*-------------------------------------*/
```

fUsIInsU FNCT.C

```
#include "c:\bc\data\include.h"
void box(win *x);
void boxclr(win *x);
void tx(win *x);
void txd(win *x);
void txclr(win *x);
void txdclr(win *x);
void invdi (win *x,char *ax);
void speak(int x,int y,char *s);
char *fullline(char *,int);
void putcenter(char *line,int st,int en);

/*void *bin;*/
extern char linebin[80];
extern int gk,buff_key;
extern win headwin;
int textlenght(win *pwin)
{
return (pwin->wr-pwin->wl+1);
}
int texthieght(win *pwin)
{
return ( pwin->wbt-pwin->wt+1);
}
/*****************/
int legalcheck(int legal[],int arg)
{
int i;
for(i=0;legal[i]!=0;i++)
        if (legal[i]==arg) return TRUE;
return FALSE;
}
/******************************** popup */
int popup(win *pwin,menubar *pmenu,int legal[],ASST * asst)
{
int h,l;
int gk;
int lineF=1;
int oldbar=pmenu->bar;
boxclr(pwin);
l=textlenght(pwin)-2;
h=texthieght(pwin);
refreshpopup(pwin,pmenu,lineF,l);
hilipopup(pwin,pmenu,1,pmenu->bar,l,1);
assist(asst,pmenu->bar,pwin);
while((gk=getkey()) != ESC)
        {
         if(legalcheck(legal,gk))return(gk);

         oldbar=pmenu->bar;
         switch(gk){
                case UP:    pmenu->bar--; break;
                case DOWN:  pmenu->bar++; break;
                case RET :  return(gk);
```

```
                        }
        adjust2(1,pmenu->Nchoice,&(pmenu->bar));
        if(scrollpage(&lineF,lineF+pmenu->lineM-1,pmenu->bar))
                refreshpopup(pwin,pmenu,lineF,l);

        hilipopup(pwin,pmenu,lineF,oldbar,l,0);
        hilipopup(pwin,pmenu,lineF,pmenu->bar,l,1);

/*      gotoxy(2,pmenu->bar);
        cputs(fullline(pmenu->choice[pmenu->bar-1],l));
                textcolor(pmenu->fore);
                textbackground(pmenu->back);
                gotoxy(2,newbar);
                cputs(fullline(pmenu->choice[newbar-1],l));
                txcolor(pwin);*/

        assist(asst,pmenu->bar,pwin);
        }
return(ESC);
}
/*******************/
refreshpopup(win *pwin,menubar *pmenu,int lineF,int l)
{
int i;
int lineL=pmenu->lineM+lineF-1;
for(i=lineF;i<=lineL && i <= pmenu->Nchoice ;i++)

        {
        gotoxy(2,i-lineF+1);
        if(i==pmenu->bar)
                {
                textcolor(pmenu->fore);
                textbackground(pmenu->back);
                cputs(fullline(pmenu->choice[i-1],l));
                txcolor(pwin);
                }
        else    cputs(fullline(pmenu->choice[i-1],l));
        }
}

/*******************/
hilipopup(win *pwin,menubar *pmenu,int lineF,int barN,int l,int hi)
{
gotoxy(2,barN-lineF+1);
cputs(fullline(pmenu->choice[barN-1],l));
if(hi)
        {
        textcolor(pmenu->fore);
        textbackground(pmenu->back);
        }
gotoxy(2,barN-lineF+1);
cputs(fullline(pmenu->choice[barN-1],l));
if (hi) txcolor(pwin);
}
```

```
/**********************/
void pushdown(win * pwin,void far *bin)
{
loadwin(pwin,bin);
farfree(bin);
bin=NULL;
}
/*****************/
char *fullline(char* line,int lenght)
{
int len;
memset(linebin,' ',lenght);
linebin[lenght]='\x0';
len=strlen(line);
strncpy(linebin,line,strlen(line)<lenght ? len : lenght);
return(linebin);
}

menubar headmenu={5,5,1 BLACK,WHITE,NULL};

/***********************************/
int menurow(int instant)
{
int start[10]={1,17,33,49,64,0,0,0,0,0};
int end[10]={10,26,42,58,74,0,0,0,0,0};
int i;
char * headchoice[]={"FILE","INPUT","ANALYSIS","REPORT","INFO"};
ASST asst00[]={{6,0,""},
                                {1,1,"Load, Save Data"},
                                {2,2,"Input to CUMAUM"},
                                {3,3,"Analysis of CUMAUM"},
                                {4,4,"Print report of CUMAUM"},
                                {5,5,"Information of Automation Project"}};


headmenu.choice=(char **)&headchoice;



txclr(&headwin);
for(i=1;i<=headmenu.Nchoice;i++)
        {
        gotoxy(start[i-1],1);
        if(i==headmenu.bar)
                {
                textcolor(headmenu.fore);
                textbackground(headmenu.back);
                putcenter(headmenu.choice[i-1],start[i-1],end[i-1]);
                textcolor(headwin.tc);
                textbackground(headwin.tb);
                }
        else putcenter(headmenu.choice[i-1],start[i-1],end[i-1]);
```

```
        }
if(instant==TRUE)
        {
        return(headmenu.bar);
        }
assist(asst00,headmenu.bar,&headwin);
while((gk=getkey())!=ESC)
        {
        int newbar=headmenu.bar;
        switch(gk){
                case LEFT: newbar--; break;
                case RIGHT: newbar++;break;
                case DOWN:
                case RET: return(headmenu.bar);
                }
        if(newbar<1) newbar=headmenu.Nchoice;
        else if(newbar>headmenu.Nchoice) newbar=1;
        putcenter(headmenu.choice[headmenu.bar-1],start[headmenu.bar-
1],end[headmenu.bar-1]);
        textcolor(headmenu.fore);
        textbackground(headmenu.back);
        putcenter(headmenu.choice[newbar-1],start[newbar-1],end[newbar-1]);
        textcolor(headwin.tc);
        textbackground(headwin.tb);
        headmenu.bar=newbar;
        assist(asst00,newbar,&headwin);
        }
return(gk);
}
/*********************************/
void putcenter(char *line,int st,int en)
{
gotoxy(st,1);
cputs(fullline("                ",en-st+1));
gotoxy((en+st-strlen(line)+1)/2,1);
cputs(line);
}
/********************************* editstring */

void editstring(char *s,int x,int y,int maxlength, char *legal)
{
int c, len=strlen(s)-1, pos = 0, insert = FALSE;
curson();
maxlength--;
while(len>=1)           /**************TRIMMING *******/
        {
        if(s[len]!=' ') break;
        len--;
        }


di(s,x,y,maxlength);
do
        {
```

```
        gotoxy(pos+x,y);
        switch(c = getkey())
                {
                case HOME :   pos = 0;
                                break;
                case END   :    pos = len;
                                break;
                case INS   :    insert = !insert;
                                break;
                case LEFT :   if (pos > 0)      pos--;
                                break;
                case RIGHT:   if (pos < len)
                                pos++;
                                break;
                case BACK :   if (pos > 0)
                                        {
                                        movmem(&s[pos], &s[pos - 1], len - pos +
1);
                                        pos--;
                                        s[len]=' ';
                                        len--;
                                        }
                                break;
                case DEL :    if (pos <= len)
                                        {
                                        movmem(&s[pos + 1], &s[pos], len - pos);
                                        s[len]=' ';
                                        len--;

                                        }
                                break;
                case RET   :    break;
                case UP    :    break;
                case DOWN :   break;
                case ESC :    len = 0;
                                break;
                default :
                                if (((legal[0] == 0) || (strchr(legal, c) != NULL)) &&
                                string(c)  )
                                        {
                                        /** insert mode ***/
                                        if (insert)
                                                {
                                                if (len > maxlength-1) break;
                                                if      (len >= maxlength && s[maxlength]
== ' ')

                                                        {
                                                        s[maxlength]= '\x0';
                                                /*      len--;*/
                                                        }
                                                memmove(&s[pos + 1], &s[pos], len - pos
+ 1);

                                                len++;
                                                }
```

```
                                        /*** ovr mode ***/
                                        else
                                                {
                                                if (pos >= maxlength) break;
                                                if (pos >= len)
                                                        len++;
                                                }
                                        s[pos++] = c;
                                        }
                                break;
                }
/*      s[len] = 0;*/
        di(s,x,y,maxlength);
        }               while ((c != RET) && (c != ESC) && (c != DOWN) && (c != UP));
buff_key=c;
cursoff();
}

/********************************* getint */
void getint(int *number,int max,int x,int y,int len)
{
 int i;
 char s[10];

 strcpy(s,"      ");
 editstring(s,x,y,len+1, "1234567890");
 i = atoi(s);
 if(i>max) i=max;
 *number = i;
}

/********************************* getfloat */
void getfloat (float *number,float *max,int x,int y,int len)
{
float i;
char s[]={"      "};
editstring(s,x,y,len+1,"1234567890.");
i= atof(s);
if(i>*max) *number=*max;
else *number =i;
}


di(char *s,int x,int y,int length)
{
gotoxy(x,y);
cputs(fullline(s,length));
}
int integer(int x)
{
return( (x>=48 && x<=57) || x==46 ? TRUE : FALSE);
}
int decimal(int x)
{
```

```
return( (x>=48 && x<=57)  ? TRUE : FALSE );
}
int string(int x)
{
return(( x<=255 && x>=32) || x<0 ? TRUE : FALSE);
}


/* help(char * legend)
{
FILE * fp;
char *helpdoc,*mark=NULL;
char *markend=NULL;
char badge[5];
void far **bin=NULL;
long indict;
win helptext={5,5,75,20,WHITE,BLUE,FALSE,1,WHITE);
if(textmalloc(&helptext,bin)!=FALSE
        && (fp=fopen("help.doc","rb"))!=NULL
        && (helpdoc=farmalloc(160*sizeof(char)))!=NULL)
        {
        boxclr(&helptext);
        fread(helpdoc,1,159,fp);
        helpdoc[160]=0;

        mark=strstr(helpdoc,legend);
        strncpy(badge,&mark[5],5);
        badge[5]=0;
        indict= atol(badge);
        fseek(fp,indict,SEEK_SET);

        while( markend==NULL && !feof(fp))
                {
                fread(helpdoc,1,159,fp);
                helpdoc[160]=0;
                markend=strstr(helpdoc,"@");
                if(markend!=NULL) *markend=0;
                cprintf(helpdoc);
                }
        fclose(fp);
        farfree(helpdoc);
        }
while(getkey()!=ESC);
pushdown(&helptext,*bin);
} */
/*******************/
adjust(int mini,int maxi,int *x)
{
if(*x< mini) *x=maxi;
else if(*x> maxi ) *x=mini;
}
/*******************/
adjust2(int mini,int maxi,int *x)
{
```

```
if(*x< mini) *x=mini;
else if(*x> maxi ) *x=maxi;
}
```

fusiinsu FNCT2.C

```
#include "c:\bc\data\include.h"
#include "c:\bc\data\define2.h"
#include "c:\bc\data\extern.h"

extern struct BAR attrbar;
extern win fulltext,boardtext,fulltext;
extern int buff_key;
extern win doctext;
extern int line1;
extern int attrindex[ATTRMAX];
extern char legend[];

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

win bartext={1,3,80,24,WHITE,BLUE,FALSE,0,WHITE};
win fulltext2={1,1,80,25,WHITE,BLUE,FALSE,0,WHITE};

/**********************/
int barwriter(struct BAR *strip)
{
int i,gk;
int oldatt,attvalue=1;
int oldvalue;
int line1=1;
int len;
float block;
len=strlen(strip->legend);
block=(strip->max-strip->min)/100;

bartext.wt=(int)((23-strip->Nrecord)/2);
txclr(&bartext);
refreshbar(strip,1,strip->Nrecord);

        /*** axis line ***/
Vline(1,1+strip->Nrecord,37,0);
gotoxy(37,1+strip->Nrecord); putch(192);
Hline(38,79,1+strip->Nrecord,0);

        /*** X-axis unit ***/
gotoxy(38+(41-len)/2,+strip->Nrecord+2);
        cputs(strip->legend);

        /*** min max x-value ***/
gotoxy(38,strip->Nrecord+2);
if(strip->decimal)
        Tprintf(&strip->min);
else
        cprintf("%-3.0f",strip->min);

gotoxy(strip->decimal ? 70 : 78,strip->Nrecord+2);
if(strip->decimal)
        Tprintf(&strip->max);
else
        cprintf("%3.0f",strip->max);
```

```
hiliattr4(strip,1,1,1);

while((gk=getkey())!=F10 && gk!=ESC)
        {
        oldatt=attvalue;
        oldvalue=*(strip->value[attvalue-1]);
        if(decimal(gk))
                {
                float v,max=9999999.0;
                buff_key=gk;
                getfloat(&v,&max,27,attvalue,strip->decimal?10:3);
                *(strip->value[attvalue-1])=v;
                }
        switch(gk)
                {
                case UP: attvalue--; break;
                case DOWN : attvalue++;break;
                case RIGHT:
                case '+' : *(strip->value[attvalue-1]) +=5*block;
                                        break;
                case LEFT :
                case '-' : *(strip->value[attvalue-1]) -=5*block;
                                        break;
                }
        adjust2(1,strip->Allrec,&attvalue);

        if (scrollpage(&line1,line1+strip->Nrecord-1,attvalue))
                refreshbar(strip,line1,strip->Nrecord+line1-1);

        if (*(strip->value[attvalue-1])> strip->max) (*strip->value)[attvalue-1]=strip->max;
        if (*(strip->value[attvalue-1])< strip->min) (*strip->value)[attvalue-1]=strip->min;
        if (*(strip->value[attvalue-1]) != oldvalue) drawbar(strip,attvalue,line1);
        hiliattr4(strip,attvalue,line1,1);
        if(oldatt!=attvalue) hiliattr4(strip,oldatt,line1,0);
        }
return(gk);
}
/*****************/
refreshbar(struct BAR * strip,int line1,int Lline)
{
int i;
for(i=line1;i<=strip->Allrec && i<=Lline;i++)
        {
        gotoxy(1,i-line1+1);
        cprintf("%25s",strip->name[i-1]);
        gotoxy(27,i-line1+1);
        if(strip->decimal)
                Tprintf(&(*(strip->value[i-1])));
        else
                cprintf("%3.0f",*(strip->value[i-1]));
        drawbar(strip,i,line1);
        }
}
```

```
/*****************/
drawbar(struct BAR *strip,int i,int line1)
{
int k;
gotoxy(27,i-line1+1);
if(strip->decimal)
        Tprintf(&(*strip->value[i-1]));
else
        cprintf("%3.0f",*strip->value[i-1]);
gotoxy(38,i-line1+1);
k=cut(strip,i-1);
Hline(38,37+k,i-line1+1,4); clreol();
}
/*********************/

hiliattr4(struct BAR *strip,int i,int line1,int hi)
{
if (hi) txinv(&bartext);
gotoxy(27,i-line1+1);
if(strip->decimal)
        Tprintf(&(*strip->value[i-1]));
else
        cprintf("%3.0f",*strip->value[i-1]);
if (hi) txcolor(&bartext);
}
/*******************/

int cut(struct BAR *strip,int i)
{
return(((*strip->value[i]-strip->min)/(strip->max-strip->min)*42.));
}
```

FUSIINSU GRAPH.C

```
#include <c:\bc\data\include.h>
#include <c:\bc\data\define2.h>
#include "c:\bc\data\extern.h"
extern win boardtext;
extern int Nproject;
extern int Nattrselect;
extern int attrindex[];
graphwriter()
{
int attr=0,i;
int cl[]={51};
ASST asst3610[]={{3,0,""},
                                {1,3611,"F10-Main menu N/A = NO Total Utility Values;
ESC-Attribute Selection"},
                                {2,3621,"Use ↕+ENTER to select; ESC-previous"}};

assist(asst3610,1,&boardtext);
say("GRAPH","F10-menu ESC-select attribute","CUMAUM");
boxclr(&boardtext);
column(&boardtext,1,cl);

speak(53,1,"ALTERNATIVES");
for(i=0;i<Nproject;i++)
        {
        gotoxy(51,i+2);
        cprintf("%2d %-25s",i+1,project[i]->name);
        }
axiswriter();
refreshgraph(0);
while(TRUE)
        {
        switch(getkey())
                {
                case UP : attr--;break;
                case DOWN: attr++;break;
                case ESC : graphpopup(&attr);
                                        break;
                case F10 : if(newprogram()) return;
                                        tx(&boardtext);
                                        break;
                default :continue;
                }
        adjust2(0,Nattrselect-1,&attr);
        refreshgraph(attr);
        assist(asst3610,1,&boardtext);
        }
}
win graphpoptext={40,10,67,15,WHITE,BLUE,FALSE,0,RED};
/*****************/
graphpopup(int *attr)
{
char *(charpt[ATTRMAX]);
void far **bin=NULL;
int legal[]={0};
```

```
int i,ch;
menubar graphmenu={20,1,1,BLACK,WHITE,NULL};
ASST asst3620[]={{1,3621,"Use ↑↓+ENTER to select; ESC-previous"}};

graphmenu.Nchoice=Nattrselect;
graphmenu.lineM=min(Nattrselect,20);
graphpoptext.wt=13-graphmenu.lineM/2;
graphpoptext.wbt=graphmenu.lineM+graphpoptext.wt-1;
for(i=0;i<Nattrselect;i++)
        charpt[i]=useattr[i]->name;
graphmenu.choice=charpt;
graphmenu.bar=*attr+1;
textmalloc(&graphpoptext,bin);

if((popup(&graphpoptext,&graphmenu,legal,asst3620))!=ESC)
        *attr=graphmenu.bar-1;
pushdown(&graphpoptext,*bin);
tx(&boardtext);
}
/*****************/
win ingraphtext={5,5,44,19,WHITE,BLUE,FALSE,0,RED};
refreshgraph(int attr)
{
int i;
float maxutil=0;
txclr(&ingraphtext);
tx(&boardtext);
gotoxy(4,1);
cprintf("ATTRIBUTE : %25s",useattr[attr]->name);
speak(17,21,"ALTERNATIVE");

/* find the maximum utility value */
for(i=0;i<Nproject;i++)
        if(project[i]->util[attrindex[attr]] > maxutil)
                maxutil=project[i]->util[attrindex[attr]];

for(i=0;i<Nproject;i++)
        stackwriter(i,useattr[attr]->type <1 ? NULL : &(project[i]-
>util[attrindex[attr]]),project[i]->util[attrindex[attr]]==maxutil ? 1 : 0);
}
/***************/
stackwriter(int proj,float *scoreutil,int hi) /* hi for hili-color */
{
int slack,d1,d2;
int height;
int x=proj*2+4;
if(*scoreutil <0 || scoreutil==NULL)
        {
        speak(x,16,"N");
        speak(x,17,"A");
        return;
        }
height=(int)(*scoreutil*10); /* -1 for not write graph */
slack=(int)(*scoreutil*10-(float)height+0.5);
```

```
if(hi) txcolorchange(&boardtext,RED);
Vline(18-height,17,proj*2+4,3);
if(slack)
        {
        gotoxy(x,17-height);
        putch(220);
        }
gotoxy(x,13-height); cprintf("%1d",d1=(int)*scoreutil);
gotoxy(x,14-height); cputs(".");
gotoxy(x,15-height); cprintf("%1d",d2=(int)((*scoreutil-d1)*10));
gotoxy(x,16-height); cprintf("%1d",(int)((*scoreutil-d1-(float)d2/10)*100));

txcolor(&boardtext);
}
/***************/
axiswriter()
{
int i,d;
Hline(4,44,18,0);
gotoxy(3,18); putch(192);
Vline(7,17,3,0);
speak(1,7,"1");
gotoxy(1,9); putch('U');
gotoxy(1,10); putch('T');
gotoxy(1,11); putch('I');
gotoxy(1,12); putch('L');
gotoxy(1,13); putch('I');
gotoxy(1,14); putch('T');
gotoxy(1,15); putch('Y');
for(i=0;i<Nproject;i++)
        {
        gotoxy(4+i*2,19); cprintf("%1d",d=(i+1)/10);
        gotoxy(4+i*2,20); cprintf("%1d",(i+1)-d*10);
        }
}
```

fUsiinsu HARDERR.C

```
/* HARDERR.C */
/*  CRITICAL ERROR HANDLING MODULE
                !T'S MUST NOT USE INTERRUPT DOS SERVICE ROUTINE THAT
                OVER FUNCTION 0CH BECAUSE OF THE DEFINITION TO CHANGE
                THE ROUTINE
        THIS MODULE ALSO STAND ALONE.
        This file provide the example of how to detect and manage ctrl-break.
        To look at this view 'ctrlhandler'.

        To use, please insert below statements in your main routine.

                harderr(handler);
                ctrlbrk(ctrlhandler);

*/

#include "C:\bc\data\include.h"

static char    *mes[]={"WRITE-PROTECT ERROR",
                                "INVALID DRIVE NUMBER",
                                "DEVICE NOT READY",
                                "UNKNOWN COMMAND REQUESTED",
                                "DATA ERROR (BAD CRC)",
                                "BAD REQUEST STRUCTURE LENGTH",
                                "SEEK ERROR",
                                "UNKNOWN MEDIA TYPE",
                                "SECTOR NOT FOUND",
                                "PRINTER OUT OF PAPER",
                                "WRITE FAULT ERROR",
                                "READ FAULT ERROR",
                                "GENERAL FAILURE"};

win texterr={25,10,55,12,RED,YELLOW,FALSE,0,GREEN};
char errbuff[330];
#pragma warn -par


/********************************** handler */
int    handler(int errvalue,int ax,int bp,int si)
{
                char      c;

                savewin(&texterr,errbuff);
                boxclr(&texterr);
                printcenter(&texterr,1,mes[errvalue]);
                textcolor(texterr.tc+BLINK);
                printcenter(&texterr,2,"A-bort or R-etry");

                while ( (c=toupper(getkey())) != 'A' && c!='R');
                if (c=='R') hardresume(1);
                loadwin(&texterr,errbuff);
        hardretn(errvalue);
}
#pragma warn +par
```

```
int ctrlhandler()
{
        return 1;
}

/********************************* checkprinter */

int checkprinter(void)
{
int c;
union REGS r;
r.h.ah=2;
r.x.dx=0;
int86(0x17,&r,&r);
while((r.h.ah&0x20) || (r.h.ah&0x8))
                {
                savewin(&texterr,errbuff);
                boxclr(&texterr);
                printcenter(&texterr,1,"ERROR PRINTER NOT READY");
                textcolor(texterr.tc | BLINK);
                printcenter(&texterr,2,"A-bort or R-etry");

                while ( (c=toupper(getkey())) != 'A' && c!='R');
                loadwin(&texterr,errbuff);
                if (c=='A') return(FALSE);
                r.h.ah=2;
                r.x.dx=0;
                int86(0x17,&r,&r);
                }
return(TRUE);
}

/********************************* error */
void error(int i)
{
char *c[]={"",
                "CAN'T SAVE ERROR",
                "CAN'T READ ERROR",
                "MEMMORY FULL ERROR",
                "TOO MUCH CONSTRAINTS"};
savewin(&texterr,errbuff);
boxclr(&texterr);
printcenter(&texterr,1,c[i]);
textcolor(texterr.tc+BLINK);
gotoxy(7,2); cputs ("ANY KEY");
getkey();
loadwin(&texterr,errbuff);
}

/********************************* atquit */
void atquit(void)
{
savewin(&texterr,errbuff);
```

```
boxclr(&texterr);
printcenter(&texterr,1,"QUIT CUMIX ?");
textcolor(texterr.tc | BLINK);
printcenter(&texterr,2,"Y/N");
if(getyn())
        exit(0);
else
        {
        loadwin(&texterr,errbuff);
        return;
        }
}

/********************************** memerr
void memerr(void)
{
message("INSUFFICIENT","MEMORY");
getkey();
quit();
}*/

/********************************** pready
int pready(void)
{
savewin(&texterr,e);
boxclr(&texterr);
printcenter(&texterr,1,"ANY KEY WHEN READY");
printcenter(&texterr,2,"ESC to abort");
gk=getkey();
loadwin(&texterr,e);
if (gk==ESC) return(FALSE);
else return (TRUE);
}*/
/********************************** getyn */
int getyn(void)
{
int i;
while ( (i=toupper(getch())) != 'N' && i != 'Y' );
if (i=='Y') return (TRUE);
else return (FALSE);
}
```

fusionsu INFO.C

```
#include "c:\bc\data\include.h"
extern win boardtext;
extern int progrid;
infowriter()
{
FILE *fp;
char far * docbuff;
char *datafile[]={"nc.doc","mis.doc","mrp.doc","fms.doc","cad.doc","cims.doc"};
char *header[]={"NC/CNC","MIS","MRP/MRP II","FMS","CAD","CIMS"};
int number=progrid-51;
ASST asst5000[]={{1,5000,"F10-Main menu"}};
say(header[number],"F10-menu","CUMAUM");
boxclr(&boardtext);
assist(asst5000,1,&boardtext);
if((docbuff = farcalloc (70,20))!=NULL)
        {
        if((fp=fopen(datafile[number],"rb")) != NULL)
                {
                fread(docbuff,1,70*20,fp);
                fclose(fp);
                cputs(docbuff);
                farfree(docbuff);
                }
        }
while(TRUE)
        {
        switch(getkey())
                {
                case F10: if(newprogram()) return;
                }
        assist(asst5000,1,&boardtext);
        }
}
```

fUsIInsU LOAD.C

```
#include "c:\bc\data\include.h"
#include "c:\bc\data\define2.h"
#include "c:\bc\data\extern.h"
extern int projectNO;

ATTR far * attralloc();

char altfile[26]="alter.dat";
char attrfile[26]="attr.dat";

extern float shiftlower,shiftupper;
extern win boardtext;
char newname1[26];
char newname2[26];


int loadfile();
win filetext={26,11,54,14,WHITE,BLACK,FALSE,0,WHITE};
win textfilename={26,13,54,13,BLACK,WHITE,TRUE,0,BLACK};
int prjsize; /* temp */
int attrsize;
int attrfixsize;

/************************/
load()
{
void far **bin=NULL;
if(textmalloc(&filetext,bin)!=FALSE)
        {
        boxclr(&filetext);
        cputs(altfile);
        editstring (altfile,1,2,25,"");
        if(getkey()!=ESC)
                {
                loadalt();
/*              loadattr();*/
                }
/*      clrscr(); */
        pushdown(&filetext,*bin);
        }
return;
}

/***************************/
int loadalt()
{
int i,j,oldattravi;
FILE *fp=NULL;
if((fp=fopen(altfile,"rb"))!=NULL)
        {
        oldattravi=Nattravi;
        fread(attrfile,sizeof(char)*26,1,fp);
        /*hold*/
        fread(&Nproject,sizeof(int),1,fp);
```

```
        fread(&Nattravi,sizeof(int),1,fp);
        fread(&shiftupper,sizeof(float),1,fp);
        fread(&shiftlower,sizeof(float),1,fp);
        for(i=0;i<Nproject;i++)
                {
                fread(project[i],prjsize,1,fp);
                project[i]->change=0;
                }
        /* hold */

        for(i=0;i<ATTRFIX;i++)
                {
                fread(attrnode[i],attrfixsize,1,fp);
                }
        for(i=ATTRFIX;i<oldattravi;i++)
                {
                attrclr(i);
                }
        for(i=ATTRFIX;i<Nattravi;i++)
                {
                initattr(i);
                fread(attrnode[i],attrsize,1,fp);
                for(j=0;j<5;j++)
                        {
                        fread((attrnode[i]->doc[j]),sizeof(char)*45,1,fp);
                        }
                }

        fclose(fp);
        endload();
        return TRUE;
        }
return FALSE;
}

/*********************/
/* hold
int loadattr()
{
int i,j,oldattravi;
FILE *fp=NULL;
if((fp=fopen(attrfile,"rb"))!=NULL)
        {
        oldattravi=Nattravi;

        fread(&Nattravi,sizeof(int),1,fp);
        for(i=0;i<ATTRFIX;i++)
                {
                fread(attrnode[i],attrfixsize,1,fp);
                }
        for(i=ATTRFIX;i<oldattravi;i++)
                {
                attrclr(i);
                }
```

```
        for(i=ATTRFIX;i<Nattravi;i++)
                {
                initattr(i);

                fread(attrnode[i],attrsize,1,fp);

                for(j=0;j<5;j++)
                        {
                        fread((attrnode[i]->doc[j]),sizeof(char)*45,1,fp);
                        }
                }
        fclose(fp);
        return TRUE;
        }
return FALSE;
}

*/
/**********************/
ATTR far * attralloc()
{
return ( farcalloc(1,sizeof(ATTR)));
}



/**********************/
save()
{
void far **bin=NULL;
if(textmalloc(&filetext,bin) !=FALSE)
        {
        boxclr(&filetext);
        cprintf("SAVE AS : \n\r%s",altfile);
        editstring(altfile,1,2,25,"");
        if(getkey() !=ESC)
                {
                savealt();
/*              saveattr();*/
                }
        pushdown(&filetext,*bin);
        }
}

/**************************/
int savealt()
{
int i,j;
FILE *fp;
if((fp=fopen(altfile,"wb")) !=NULL)
        {
        fwrite(attrfile,sizeof(char)*26,1,fp);
        fwrite(&Nproject,sizeof(int),1,fp); /* how many record */
        /* hold */
```

```
        fwrite(&Nattravi,sizeof(int),1,fp);
        fwrite(&shiftupper,sizeof(float),1,fp);
        fwrite(&shiftlower,sizeof(float),1,fp);
        for(i=0;i<Nproject;i++)
                {
                fwrite(project[i]->name,prjsize,1,fp);
                }
        /* hold */
        for(i=0;i<ATTRFIX;i++)
                {
                fwrite(attrnode[i],attrfixsize,1,fp);
                }
        for(i=ATTRFIX;i<Nattravi;i++)
                {
                fwrite(attrnode[i],attrsize,1,fp);
                for(j=0;j<5;j++)
                        {
                        fwrite((attrnode[i]->doc[j]),sizeof(char)*45,1,fp);
                        }
                }

        fclose(fp);
        return TRUE;
        }
return FALSE
}

/********** **********/
/* hold
int saveattr )
{
int i,j;
FILE * fp;
if((fp=fopen(attrfile,"wb"))!=NULL)
        {
        fwrite(&Nattravi,sizeof(int),1,fp);
        for(i=0;i<ATTRFIX;i++)
                {
                fwrite(attrnode[i],attrfixsize,1,fp);
                }
        for(i=ATTRFIX;i<Nattravi;i++)
                {
                fwrite(attrnode[i],attrsize,1,fp);
                for(j=0;j<5;j++)
                        {
                        fwrite((attrnode[i]->doc[j]),sizeof(char)*45,1,fp);
                        }
                }
        fclose(fp);
        return TRUE;
        }
return FALSE;
}     */
```

```
/*************/
clrresident()
{
int i;
for(i=Nproject;i<PROJECTMAX;i++)
        memset(project[i],0,sizeof(struct PRJ));
        /* reset resident project */
for(i=Nproject;i<PROJECTMAX;i++)
        {
        trim(project[i]->name,namelength-1);
        }
/* and any identifier */
projectNO=1;
}

/*************************/
endload()
{
clrresident();
/*      checklastfield();*/
recalcall();
checklastyear();
monetcalc();

makeuseattr();
attrratio();
anacalc();
}

/************************/
setupsize()
{
prjsize=(int)((char*)project[0]->calc[0] - (char *)project[0]->name);

attrfixsize=(int)((char*)&(attrnode[0]->doc[0])-(char *)(attrnode[0]->name));

attrsize=(int)((char*) &(attrnode[0]->doc[0]) -(char *)(attrnode[0]->name));
}

/***************/
loaddisp()
{
ASST asst1110[]={{1,1111,"F2-load; F10-Main menu"}};
say("LOAD","","");
boxclr(&boardtext);
refreshload();
assist(asst1110,1,&boardtext);
while(TRUE)
        {
        switch(getkey())
                {
                case F10: if(newprogram()) return;
                                        break;
                case RET:
```

```
                case F2 : load();
                                        break;
                }
        assist(asst1110,1,&boardtext);
        refreshload();
        }
}
/**************/
refreshload()
{
gotoxy(1,1); cprintf("ALTERNATIVE DATA : %-25s",altfile);
/*gotoxy(1,2); cprintf("ATTRIBUTE   DATA : %-25s",attrfile);*/
gotoxy(1,3); cprintf("Number of Current Alternative : %2d",Nproject);
gotoxy(1,4); cprintf("Number of Available Attribute : %2d",Nattravi);
}
/**************/
savedisp()
{
ASST asst1210[]={{1,1211,"F2-save; F10-Main menu"}};
say("SAVE","","");
boxclr(&boardtext);
refreshload();
assist(asst1210,1,&boardtext);
while(TRUE)
        {
        switch(getkey())
                {
                case F10: if(newprogram()) return;
                                        break;
                case RET:
                case F2 : save();
                                        break;
                }
        assist(asst1210,1,&boardtext);
        refreshload();
        }
}
```

Fujitsu MONET.C

```
#include "c:\bc\data\include.h"
#include "c:\bc\data\define2.h"
#include "c:\bc\data\extern.h"

extern win boardtext,fulltext;
extern int gk,buff_key;
extern char *period[2];
float shiftupper,shiftlower;
int prjmonet=1;
/*******************/
monetwriter()
{
int cl[]={27,53};
int i,oldmonet;
ASST asst3110[]={{1,3111,"F10-Main menu; Use ↕ to select Alternative"}};
tx(&fulltext);
say("MONETARY","F10-menu","CUMAUM");
boxclr(&boardtext);
assist(asst3110,1,&boardtext);
column(&boardtext,2,cl);
speak(9,1,"CASHFLOW");
speak(31,2,"NET PRESENT VALUE");
/*speak(28,9,"INTERNAL RATE OF RETURN");*/
speak(33,16,"PAYBACK PERIOD");
speak(38,21,"PERIOD");
speak(55,1,"ALTERNATIVES");
if(prjmonet>Nproject) prjmonet=Nproject;
for(i=0;i<Nproject;i++)
        speak(54,2+i,project[i]->name);
checklastyear();/* TRANSIENT */
monetcalc();   /*                                    */
refreshmonet();
hilimonet(prjmonet,1);
while((gk=getkey()) != -1)
        {
        oldmonet = prjmonet;
        switch(gk) {
                case UP: prjmonet--; break;
                case DOWN: prjmonet++; break;
                case F10 : if(newprogram()) return;
                                        tx(&boardtext);
                                        break;
                default : continue;
                }
        if(prjmonet<1) prjmonet =1;
        if(prjmonet>Nproject) prjmonet=Nproject;
        hilimonet(oldmonet,0);
        hilimonet(prjmonet,1);
        if(oldmonet!=prjmonet)refreshmonet();
        assist(asst3110,1,&boardtext);
        }
}
/*****************/
hilimonet(int monet,int hi)
```

```
{
if(hi) txinv(&boardtext);
speak(54,1+monet,project[monet-1]->name);
if(hi) txcolor(&boardtext);
}
/*****************/
refreshmonet()
{
int i;
for(i=0;i<PROJECTMAX;i++)
        {
        gotoxy(2,i+2);
        if(     i<project[prjmonet-1]->yearuse)
                {
                cprintf("%-5s %2d    ",period[project[prjmonet-1]->period],i);
                Tprintf(&project[prjmonet-1]->calc[i][7]);
                }
        else cputs("                    ");
        }
gotoxy(36,3);cprintf("(%5.2f\%)",project[prjmonet-1]->interest);
gotoxy(35,5); Tprintf(&project[prjmonet-1]->NPV);
/*gotoxy(33,12); project[prjmonet-1]->IRR >=0 ? cprintf("%12.2f   ",project[prjmonet-1]-
>IRR)\

cputs("NON-SIGNIFICANT");
*/
gotoxy(35,19); Tprintf(&project[prjmonet-1]->PBP);
}
/****************/
monetcalc()
{
int i;
for(i=0;i<Nproject;i++)
        {
        if(project[i]->change==1) continue;
        NPVcalc(i);
/*      IRRcalc(i);  for more modification */
        PBPcalc(i);
        project[i]->change=1;
        }
rangeNPV();
anacalc();
}
/***************/
NPVcalc(int prj)
{
int j;
double v;
float rate;
project[prj]->NPV=0;
rate=   project[prj]->comp==0 ? project[prj]->interest/100 : /* normal rate */
                (float)exp((double)project[prj]->interest/100)-1;   /* compound rate*/
for(j=0;j<project[prj]->yearuse;j++)
        {
```

```
        v=pow(1+rate,j);
        project[prj]->NPV += project[prj]->calc[j][7]/v;
        }
/* project[prj]->score[0]=project[prj]->NPV;  automatic set value to alternative score */
}
/************/
PBPcalc(int prj)
{
int i=0;
float net=0;
do      {
        net+=project[prj]->calc[i][7];
        i++;
        }
        while (net<=0 && i<project[prj]->yearuse);
project[prj]->PBP=i-1;
project[prj]->score[1]=project[prj]->PBP / (project[prj]->period==1 /* month */ ? 12 : 1);
 /* set value to alternative */
}
/************/
IRRcalc(int prj)
{
int i;
double net0=0;
double net=1;
double maxIRR=1000,minIRR=0,midIRR=0;
for(i=0;i<project[prj]->yearuse;i++)
        net0+=(double)project[prj]->calc[i][7];
if(net0<0)
        {
        project[prj]->IRR=-1;
        return;
        }
midIRR =maxIRR/2;
while(fabs(net)>0.05 )
        {
        net=0;
        for(i=0;i<project[prj]->yearuse;i++)
                {
                double v;
                v=pow(fabs(pow(1+midIRR,(double)i)),midIRR/fabs(midIRR));
                net += project[prj]->calc[i][7]/v;
                }
        if( net * midIRR >0)
                {
                if(fabs(midIRR)>99.) break;
                minIRR=midIRR;
                midIRR=(maxIRR+midIRR)/2;
                }
        else
                {
                maxIRR=midIRR;
                midIRR = (minIRR+midIRR)/2;
                }
```

```
        }
project[prj]->IRR=midIRR*100;
}
/**************/
rangeNPV() /* find the maximum and minimum NPV of each alternative and assign to NPV
attribute range */
{
float maxNPV=MAXFLOAT* -1.0,minNPV=MAXFLOAT;
int i;

for(i=0;i<Nproject;i++)
        {
        maxfloat(&maxNPV,&maxNPV,&project[i]->NPV);
        minfloat(&minNPV,&minNPV,&project[i]->NPV);
        }
if(maxNPV != attrnode[0]->upper||minNPV != attrnode[0]->lower)
        {
        attrnode[0]->type=0;
        attrnode[0]->upper=maxNPV;
        attrnode[0]->lower=minNPV;

        shiftlower = minNPV>0 ? minNPV : 0;
        shiftupper = minNPV>0 ? maxNPV : maxNPV - minNPV;
        /* reset the value for curve fitting */

        for(i=0;i<Nproject;i++)
                project[i]->score[0]=project[i]->NPV + (shiftlower-minNPV);/* shift the axis
*/
        }
}
/***************/
maxfloat(float *output,float *float1,float *float2)
{
*output=*float1>*float2 ? *float1 : *float2 ;
}
minfloat(float *output,float *float1,float *float2)
{
*output=*float1<*float2 ? *float1 : *float2 ;
}
```

fusiinsu RANK.C

```
#include "c:\bc\data\include.h"
#include "c:\bc\data\define2.h"
#include "c:\bc\data\extern.h"
extern win boardtext;
extern int buff_key;
extern int attrindex[];

ranking()
{
int i,j,k,x,y,gk;
int cl[]={6,33,40,46,73};
int rw[]={2};
ASST asst3410[]={{1,3411,"F10-Main menu; N/A = No Total Utility Values"}};

say("RANKING","F10-menu äU = Total Utility","CUMAUM");
boxclr(&boardtext);
assist(asst3410,1,&boardtext);
column(&boardtext,5,cl);
row(&boardtext,1,rw);
net(&boardtext,1,rw,5,cl);
speak(1,1,"RANK"); speak(40,1,"RANK");
speak(11,1,"ALTERNATIVE"); speak(51,1,"ALTERNATIVE");
speak(35,1,"äU");       speak(75,1,"äU");

j=1;
for(k=1;k<=Nproject;k++)
        for (i=0;i<Nproject;i++)
                {
                if(project[i]->rank!=k) continue;
                if(j<=10) { y=j+2; x=1;   }
                else { y= j-8; x=41; };
                gotoxy(x+1,y); cprintf("%2d",k);
                gotoxy(x+5,y); cprintf("%25s",project[i]->name);
                gotoxy(x+33,y);
                        if(project[i]->total<0) cputs("N/A");
                        else cprintf("%4.2f",project[i]->total);
                j++;
                }
while(TRUE)
        {
        gk=getkey();
        switch(gk){
                case F10 : if(newprogram()) return;
/*              case F1 : help (".341") ; tx(&boardtext);
                                break;*/
                }
        assist(asst3410,1,&boardtext);
        }
}
```

fUsIInsU REPORT.C

```
#include "c:\bc\data\include.h"
#include "c:\bc\data\define2.h"
#include "c:\bc\data\extern.h"
extern float shiftlower,shiftupper;
extern win boardtext;
extern int Nattrselect;
extern int buff_key;
extern int attrindex[];
extern char *cashlabel[];
extern char *period[],*compound[];
int detail[PROJECTMAX];
int summary[PROJECTMAX];

int flag(int x) {if(x) return (int)'X' ;else return (int)' ';};
FILE *fpt=NULL;


/**********************/
altrep()
{
int projectN=1,i,col=1,oldproject,oldcol,gk;
ASST asst4210[]={{3,1,"F2-print; F10-Main menu; Use \x1B\x1A+ENTER to select "},
                                        {1,4211,"Complete Report"},
                                        {2,4212,"Conclusion Report"}};
say("MONETARY REPORT","F10-menu","CUMAUM");
boxclr(&boardtext);
assist(asst4210,1,&boardtext);
speak(5,1,"ALTERNATIVE");
speak(40,1,"DETAIL");
speak(60,1,"SUMMARY");
for(i=0;i<Nproject;i++)
        {
        gotoxy(5,i+2); cputs(project[i]->name);
        gotoxy(40,i+2); cprintf("[%c]",detail[i] ? 'X' : ' ');
        gotoxy(60,i+2); cprintf("[%c]",flag(summary[i]));
        }
hilialtrep(1,1,1);

while(TRUE)
        {
        oldcol=col;oldproject=projectN;
        gk=getkey();
        switch(gk){
                case DOWN : projectN+=1;break;
                case UP         : projectN-=1; break;
                case LEFT : col-=1; break;
                case RIGHT: col+=1;break;
                case RET :
                case SPACE : if(col==1) detail[projectN-1]=!detail[projectN-1];
                                        else summary[projectN-
1]=!summary[projectN-1];
                                        break;
                case F10 : if(newprogram()) return;
                                        tx(&boardtext);
```

```
                                        break;
                case F2: altrep2(); altrep3(); break;
                }
        adjust(1,Nproject,&projectN);
        adjust(1,2,&col);
        tx(&boardtext);
        hilialtrep(oldproject,oldcol,0);
        hilialtrep(projectN,col,1);
        assist(asst4210,col,&boardtext);
        }
}
/************************/
hilialtrep(int projectN,int col,int hi)
{
if(hi) txinv(&boardtext);
gotoxy(col==1? 41 : 61 ,projectN+1);
cprintf("%c",flag(col==1? detail[projectN-1]:summary[projectN-1]));
if(hi) txcolor(&boardtext);
}
/************************/
altrep2()
{
int l,i,j,period1,periodl;
int Nline=28;
int form[]={0,1,2,3,-1,4,-2,0,5,6,7,8,-3,-4,0,9,10,11,12,13,14,-5,0,15,-6,16,-7,-8};
/*char tab[]={27,'\x26','\x36','\x46','\x0'};*/


fpt=NULL;
if(!initprn())return;
/***/
compress(1);
for(j=0;j<Nproject;j++)
        {
        if(detail[j]==FALSE || project[j]->yearuse<1) continue;
        period1=1;
        fprintf(fpt,"\nALTERNATIVE DETAIL REPORT : %25s\n",project[j]->name);
        fprintf(fpt,"RATE  : %6.2f \%\n",project[j]->interest);
        fprintf(fpt,"COMPOUNDING TYPE : %10s\n\n",compound[project[j]->comp]);
        do      {
                /* section seperator */
                periodl=period1+6;
                adjust2(0,project[j]->yearuse,&period1);
                adjust2(0,project[j]->yearuse,&periodl);

                /* report header */
                blank(30);
                for(i=period1-1;i<periodl;i++)
                        fprintf(fpt,"    %-5s%2d",period[project[j]->period],i);

                for(l=0;l<Nline;l++)
                        {
                        fprintf(fpt,"\n%-30s",cashlabel[l]);
                        if(form[l]==0) continue; /* be header */
```

```
                        for(i=period1-1;i<periodl;i++)
                                Pprintf(form[l]>0 ? &project[j]->value[i][form[l]-1] :
&project[j]->calc[i][form[l]*-1-1]);
                        }
                period1=periodl+1;
                lfeed(0);
                }       while(periodl<project[j]->yearuse);
        }
compress(0);
fclose(fpt);
return;
}
/*****************/
Pprintf(float *x)
{
fprintf(fpt,fabs((double)*x) > 999999.99
                        ? (fabs((double)*x) > 9999999.9 ? "%10.0f" : "%10.1f")
                        : "%10.2f",*x);
}
/*****************/
altrep3()
{
int j,i;
char tab[]={27,30,55,0};
if(!initprn()) return;
for(j=0;j<Nproject;j++)
        {
        if(summary[j]==FALSE || project[j]->yearuse <1 ) continue;
        fprintf(fpt,"\nALTERNATIVE   %2d : %25s",attrindex[j]+1,project[j]->name);
        fputs("\nNet Present Value : ",fpt);
                Pprintf(&project[j]->NPV);
                fprintf(fpt," %5s",attrnode[0]->unit);
        fprintf(fpt,"\nPayBack Period    : %10d %5s",project[j]->PBP,attrnode[1]->unit);
        fprintf(fpt,"\nInterest Rate     : %6.2f \%\n\n",project[j]->interest);

        fputs("CASHFLOW OF ALTERNATIVE\n",fpt);
        fputs(tab,fpt);
        for(i=0;i<project[j]->yearuse;i++)
                {
                fprintf(fpt,"\n\t%10s %2d\t",period[project[j]->period],i);
                Pprintf(&project[j]->calc[i][7]);
                }
        lfeed(0);
        }
fclose(fpt);
}

/***************/
blank(int Nblank)
{
int i;
for(i=0;i<Nblank;i++)
        fputc(' ',fpt);
```

```
/*char space[]={27,102,0,0,0};
space[3]=(char)Nblank;
fputs(space,fpt);*/
}
/*****************/
int initprn()
{
fpt=NULL;
if(!checkprinter()) return 0;
if((fpt=fopen("PRN","wb"))==NULL)return 0;
/*fputs("\x1B@\r",fpt);*/
return 1;
}
/*****************/
compress(int code)
{
char set[]={27,15,0};
char reset[]={18,0};
fputs(code ? set : reset ,fpt);
}
/*****************/
lfeed(int nline)
{
int i;
if(nline==0) fputc('\f',fpt);
else  for(i=0;i<nline;i++) fputc('\n',fpt);
}
/****************/
rankrepdisp()
{
ASST asst4410[]={{1,4411,"F2-print; F10-Main menu"}};
say("RANKING REPORT","","");
boxclr(&boardtext);
assist(asst4410,1,&boardtext);
printcenter(&boardtext,10,"PRESS F2 TO GO");
while(TRUE)
        {
        switch(getkey())
                {
                case F2: rankrepdisp(); break;
                case F10 : if(newprogram()); return;
                                        tx(&boardtext);
                                        break;
                }

        }
}

/****************/
rankrep()
{
int i,j,k;
char tab[]={20,0};
char tab1[]={5,20,60,0};
```

```
if(!initprn() || Nproject==0 ) return;

j=1;

settab(tab);
lfeed(1);
fputs("\tSUGGESTION THE WAY TO CHOOSE ALTERNATIVE\n",fpt);

settab(tab1);
fputs("\tRANK\tALTERNATIVE\tTOTAL UTILITY",fpt);
for(k=1;k<=Nproject;k++)
        for (i=0;i<Nproject;i++)
                {
                if(project[i]->rank!=k) continue; /* find the ranking(k) */
                fprintf(fpt,"\n\t%2d\t%-25s\t",k,project[i]->name);
                        if(project[i]->total<0) fputs("N/A",fpt);
                        else fprintf(fpt,"%4.2f",project[i]->total);
                j++;
                }
lfeed(0);
fclose(fpt);
while( !newprogram()) ; /* debug later */
return;
}
/*************/
int utilrepmark[PROJECTMAX];
utilrep()
{
int alt=0,oldalt; /* alt start at 1 (0 for report type )*/
int i;
ASST asst4310[]={{3,1,"F2-print; F10-Main menu; "},
                                        {1,4311,"Space Bar-toggle DETAIL/SUMMARY"},
                                        {2,4312,"↑↓+ENTER-select"}};
say("UTILITY REPORT","F10-menu","CUMAUM");
boxclr(&boardtext);
assist(asst4310,1,&boardtext);
printcenter(&boardtext,1,"UTILITY FUNCTION");
speak(4,6,"ALTERNATIVE"); speak(44,6,"ALTERNATIVE");
speak(30,6,"REPORT"); speak(70,6,"REPORT");
speak(4,3,"REPORT TYPE");

hiliutilrep(0,utilrepmark[0],1);

for(i=0;i<Nproject;i++)
        {
        gotoxy(i<10 ? 5 : 45 ,i < 10 ? i+9 :  i-1);
        cprintf("%-30s [%c]",project[i]->name,utilrepmark[i] ? 'X' : ' ');
        }
while(TRUE)
        {
        oldalt=alt;
        switch(getkey())
                {
                case HOME: alt=0; break;
```

```
                case END : alt=Nproject;break;
                case UP : alt-- ;break;
                case DOWN : alt++;break;
                case RIGHT : if (alt <= 10) alt +=10;
                                        break;
                case LEFT : if(alt>10) alt -=10;
                                        break;
                case RET:
                case SPACE :
                case 'X' : utilrepmark[alt]= !utilrepmark[alt];
                                break;
                case F10 : if(newprogram()) return;
                                        tx(&boardtext);
                                        break;
                case F2 : if (utilrepmark[0]) utilrep3();
                                        else utilrep2();
                                        break;
                }
        adjust(0,Nproject,&alt);
        tx(&boardtext);
        hiliutilrep(oldalt,utilrepmark[oldalt],0);
        hiliutilrep(alt,utilrepmark[alt],1);
        assist(asst4310,alt<1 ? 1 : 2,&boardtext);
        }
}
/*****************************/
hiliutilrep(int alt ,int repmark,int hi)
{
if(hi) txinv(&boardtext);
if(!alt) {
        gotoxy(20,3); cprintf("%10s",repmark ? "SUMMARY" : "DETAIL");
        }
else {
        gotoxy(alt > 10 ? 77 : 37 , alt >10 ? alt-1 :alt+8 );
        cputs(repmark ? "X" : " ");
        }
if(hi) txcolor(&boardtext);
}
/*********/
utilrep2()
{
int i,j;
char tab[]={2,28,35,50,60,0};
if(!initprn() || Nproject ==0 || Nattrselect==0) return;
settab(tab);
lfeed(1);
for(i=0;i<Nproject;i++)
        {
        if(!utilrepmark[i+1]) continue;
        blank(6);
        fprintf(fpt,"ALTERNATIVE : %25s",project[i]->name);
        lfeed(3);
        fputs("\tATTRIBUTE\tWEIGHT\tSCORE\tUTILITY\tUTILITY*WEIGHT\n",fpt);
        underline(80);
```

```
        lfeed(1);
        for(j=0;j<Nattrselect;j++)
                {
                fprintf(fpt,"\t%-25s\t%4.2f\t",useattr[j]->name,useattr[j]->ratio);
                        Pprintf(&project[i]->score[attrindex[j]]);
                        fputs("\t",fpt);
                if(useattr[j]->type==0)
                        fprintf(fpt,"%4s\t%4s\n","N/A","N/A");
                else
                        fprintf(fpt,"%4.2f\t%4.2f\n",
                        project[i]->util[attrindex[j]],
                        useattr[j]->ratio*project[i]->util[attrindex[j]]);
                }
        lfeed(1);
        underline(80);
        fputs("\tTOTAL\t\t\t\t",fpt);
        if(project[i]->total<0)
                fputs("N/A\n",fpt);
        else
                fprintf(fpt,"%4.2f\n",project[i]->total);
        lfeed(0);
        }
fclose(fpt);
}

/********/
utilrep3()
{
int i;
char tab[]={10,40,45,0};
if(!initprn() || Nproject ==0 || Nattrselect==0) return;
blank(35); fputs("UTILITY SUMMARY\n\n",fpt);
settab(tab);
fputs("\tALTERNATIVE\tTOTAL UTILITY\n",fpt);
for(i=0;i<Nproject;i++)
        {
        if(!utilrepmark[i+1]) continue;
        fprintf(fpt,"\t%-25s\t\t",project[i]->name);
        if(project[i]->total<0)
                fputs("N/A\n",fpt);
        else
                fprintf(fpt,"%4.2f\n",project[i]->total);
        }
lfeed(0);
fclose(fpt);
}
/********/
underline(int nline)
{
int i;
for (i=0;i<nline;i++)
fputs("-",fpt);
}
/*********/
```

```
settab(char *tab)
{
fprintf(fpt,"\x1B\x44%s\r",tab);
}
/***************/
utilfrepdisp()
{
ASST asst4110[]={{1,4111,"F2-print; F10-Main menu"}};
say("UTILITY FUNCTION REPORT","","");
boxclr(&boardtext);
assist(asst4110,1,&boardtext);
printcenter(&boardtext,10,"PRESS F2 TO GO");
while(TRUE)
        {
        switch(getkey())
                {
                case F2: utilfrep();break;
                case F10 : if(newprogram()) return;
                                        tx(&boardtext);
                }
        assist(asst4110,1,&boardtext);
        }
}
/*****************/
utilfrep()
{
int i;
char tab[]={30,60,0};
if(!initprn() || Nproject ==0 || Nattrselect==0) return;
settab(tab);
lfeed(1);
fputs("UTILITY FUNCTION\n\n",fpt);
fputs("ATTRIBUTE\tEQUATION\tRISK ATTITUDE\n",fpt);
for(i=0;i<Nattrselect;i++)
        {
        fprintf(fpt,"%-25s\t",useattr[i]->name);
        switch(useattr[i]->type)
                {
                case 0: fprintf(fpt,"N/A");
                                break;
                case 1: fprintf(fpt,"U(x) = %7.2g %+7.2gx",useattr[i-1]->a,useattr[i-1]->b);
                                break;
                case 3:
                case 4: fprintf(fpt,"U(x) = %7.2g * x^%7.2g",useattr[i-1]->a,useattr[i-1]->b);
                                break;
                case 2: fprintf(fpt,"U(x) = exp(%7.2g %+7.2g/x)",useattr[i-1]->a,useattr[i-1]-
>b);
                                break;
                }
/*      fprintf(fpt,"\t4.2f\t"); reserve for R value */
        fputc('\t',fpt);
        switch(useattr[i]->type)
                {
                case 0: fputs("N/A",fpt); break;
```

```
                case 1: fputs("Risk Indifferent",fpt); break;
                case 2: fputs("S-shape",fpt); break;
                case 3: fputs("Risk Aversion",fpt); break;
                case 4: fputs("Risk Seeking",fpt); break;
                }
        lfeed(1);
        }
lfeed(3);
fputs("\nU(X) = UTILITY",fpt);
lfeed(0);
fclose(fpt);
while( !newprogram()) ; /* debug later */
return;
}
```

fUsHnsU SCORE.C

```
#include "c:\bc\data\include.h"
#include "c:\bc\data\define2.h"
#include "c:\bc\data\extern.h"
struct BAR scorebar;
extern win boardtext,fulltext,fulltext2;
extern int buff_key;
extern attrindex[ATTRMAX];
extern char *positive[];
/**********************/
scorewriter()
{
int gk,oldattr,attr=1;
int lineF=1,Nline=18;
ASST asst2610[] = { {3,0,""},
                    {1,2611,"↑↓+ENTER to select Attribute for scoring
Alternative; F10-Main menu"},
                    {2,2612,"Use \x1B\x1A to give score for
Alternative; ESC-previous"}};
redrawscore(1,Nline,1);
assist(asst2610,1,&boardtext);
while(TRUE)
        {
        char head[40];
        gk=getkey();
        oldattr= attr;
        switch(gk)
                {
                case UP : attr--; break;
                case DOWN : attr ++; break;
                case RET :  if( attrindex[attr-1] <= 1 ) break;
                                scoremaking(attr) ;
                                sprintf(head,"SCORING OF %25s",useattr[attr-1]-
>name);
                                txclr(&fulltext2);
                                gotoxy(25,2);
                                cprintf("OBJECTIVE : %8s",positive[useattr[attr-1]-
>positive]);
                                say(head,"ESC-Previous menu","CUMAUM");
                                assist(asst2610,2,&fulltext);
                                barwriter(&scorebar);
                                redrawscore(lineF,Nline+lineF-1,attr);
                                break;
                case F10: if(newprogram())
                                        {
                                        endscore();
                                        return;
                                        }
                                tx(&boardtext);break;
/*              case F1 : help(".251"); tx(&boardtext) ;break; */
                }
        adjust2(1,Nattrselect,&attr);

        if ( scrollpage(&lineF,lineF+Nline-1,attr))
                {
```

```
                refreshscore(lineF,Nline+lineF-1,attr);
                continue;
                }

        hiliscore(lineF,attr,1);
        if(oldattr !=attr)
                hiliscore(lineF,oldattr,0);
        assist(asst2610,1,&boardtext);
        }
}
/*******************/
redrawscore(int lineF,int lineL,int attr)
{
say("SCORE","F10-menu","CUMAUM");
box(&boardtext);
refreshscore(lineF,lineL,attr);
}

/*******************/
refreshscore(int lineF,int lineL,int attr)
{
int i;
txclr(&boardtext);
speak(1,1,"ATTRIBUTE");
printcenter(&boardtext,21,"Use II to choose attribute for scoring");
for(i=lineF;i<=Nattrselect && i<=lineL;i++)
        {
        gotoxy(1,2+i-lineF);
        cputs(useattr[i-1]->name);
        gotoxy(40,2+i-lineF);
        if(attrindex[i-1] <=1 ) cputs("ALREADY INSTALLED");
        else clreol();
        }
hiliscore(lineF,attr,1);
}

/*********************/
hiliscore (int lineF,int attr,int hi)
{
if (hi) txinv(&boardtext);
gotoxy(1,2+attr-lineF);
cputs(useattr[attr-1]->name);
if(hi) txcolor(&boardtext);
}

/*******************/
scoremaking(int attr)
{
int i;
for(i=0;i<Nproject;i++)
        {
        scorebar.name[i]=project[i]->name;
        scorebar.value[i] =&project[i]->score[attrindex[attr-1]];
        }
```

```
scorebar.Allrec=Nproject;
scorebar.Nrecord=min(15,Nproject);
scorebar.min=useattr[attr-1]->lower;
scorebar.max=useattr[attr-1]->upper;
scorebar.namesize=25;
scorebar.decimal=TRUE;
scorebar.legend=useattr[attr-1]->unit;
}
endscore()
{
anacalc();
}
```

fUsiinsu SCREEN.C

```
/* SCREEN.C */
#include "c:\bc\data\include.h"
void printcenter(win *x,int row,char *line);
void box(win *x);
void boxclr(win *x);
void tx(win *x);
void txd(win *x);
void txclr(win *x);
void txdclr(win *x);
void di(win *x,char *ax);
void invdi (win *x,char *ax);
void speak(int x,int y,char *s);
char *fullline(char *,int);

win headtext={1,1,80,1,WHITE,RED,TRUE,0,BLACK};
win usertext={1,25,80,25,WHITE,RED,TRUE,0,BLACK};
/******************************** say */
void say(char *line1,char *btline1,char *btline2)
{
if(strlen(line1))
        {
        txclr(&headtext);
        printcenter(&headtext,1,line1);
        }
/*txclr(&usertext);
if(strlen(btline1))
        {
        cputs(btline1);
        }
if(strlen(btline2))
        {
        gotoxy(80-strlen(btline2),1);
        cputs(btline2);
        }*/
}


/******************************** checkcon */
BIOSDATA far *bios = MK_FP (0x0040, 0);
int color = 0;
char linebin[80];
win fulltext={1,1,80,25,WHITE,BLACK,FALSE,0,BLACK};

void checkcon(void)
{
switch (bios->videoMode) {
        case 0:
        case 2:
        case 6:
        case 7:color=FALSE;break;
        default:color =TRUE;break;
        }
}
```

```
/********************************** box */
void box(win *x)
{
int i,l=x->wl-1,t=x->wt-1,r=x->wr+1,bt=x->wbt+1;
int bord[][6] = {              /*      border characters  */
                 { 196, 179, 218, 191, 217, 192 },
                 { 205, 186, 201, 187, 188, 200 },
                 { 178, 178, 178, 178, 178, 178},
                 };
int type=x->brd;
tx(&fulltext);
txborder(x);
gotoxy(l,t);              putch(bord[type][2]);
for(i=1;i<r-l;i++)        putch(bord[type][0]);
                                  putch(bord[type][3]);
gotoxy(l,bt);             putch(bord[type][5]);
for(i=1;i<r-l;i++)        putch(bord[type][0]);
                                  putch(bord[type][4]);
for(i=1;i<bt-t;i++)
        {
        gotoxy(l,t+i);    putch(bord[type][1]);
        gotoxy(r,t+i);    putch(bord[type][1]);
        }
}

/********************************** boxclr */
void boxclr(win *x)
{
box(x);
txclr(x);
}

/********************************** tx */
void tx(win *x)
{
window(x->wl,x->wt,x->wr,x->wbt);
txcolor(x);
gotoxy(1,1);
}

/********************************** txd */
void txd(win *x)
{
window(x->wl,x->wt,x->wr,x->wbt);
txinv(x);
gotoxy(1,1);
}
/********************/
txinv(win *x)
{
textcolor( color ? x->tb : (x->inv ?  WHITE : BLACK ));
textbackground(          color ? x->tc : (x->inv ? BLACK : WHITE));
}
/********************/
```

```
txcolorchange(win *x,int colorchange)
{
textcolor( color ? colorchange : (x->inv ?  BLACK : WHITE));
}
/*********************/
txcolor(win *x)
{
textcolor          ( color ? x->tc : (x->inv ?  BLACK : WHITE));
textbackground(          color ? x->tb : (x->inv ? WHITE : BLACK));
}
/*********************/
txborder(win *x)
{
textcolor( color ? x->bc : (x->inv ?  BLACK : WHITE));
textbackground(          color ? x->tb : (x->inv ? WHITE : BLACK));
}
/********************************** txclr */
void txclr(win *x)
{
tx(x);
clrscr();
}

/********************************** txdclr */
void txdclr(win *x)
{
txd(x);
clrscr();
}


/********************************** invdi */
void invdi (win *x,char *ax)
{
txdclr(x);
cprintf("%s",ax);
}

/********************************** speak */
void speak(int x,int y,char *s)
{
gotoxy(x,y);
cputs(s);
}

/********************************** textsize */
int textsize(win *x)
{
return(x->wr-x->wl+3)*(x->wbt-x->wt+3)*2*sizeof(char);
}


/********************************** loadwin */
void loadwin(win*x,void far *e)
```

```
{
puttext(x->wl-1,x->wt-1,x->wr+1,x->wbt+1,e);
}

/********************************* savewin */
void savewin(win*x,void far *e)
{
gettext(x->wl-1,x->wt-1,x->wr+1,x->wbt+1,e);
}

/********************************* printcenter */
void printcenter(win *x,int row,char *line)
{
gotoxy((x->wr-x->wl+1-strlen(line))/2+1,row);
cputs(line);
}

/********************************* getkey */
int buff_key =0;
int   getkey(void)
{
        char   c,h=0;
        int      l;
        if (!buff_key)
        {
                c=getch();
                if (!c) h=getch();
                return ((h<<8)¦c);
        }
        else
        {
                l=buff_key;
                buff_key=0;
                return(l);
        }
}

void * textalloc(win *pwin)
{
void * bin=NULL;
if((bin=(void*)malloc(textsize(pwin)))!=NULL)
        savewin(pwin,bin);
return(bin);
}
/********************************* trim */
trim(char *s,int maxlen)
{
int len;
if((len=strlen(s))<maxlen)
        memset(s+len,' ',maxlen-len);
s[maxlen]=0;
}
/************************/
column(win *pwin,int Nx,int x[])/* column of fullscreen */
```

```
{
int j;
window(1,1,80,25);
txborder(pwin);
for(j=0;j<Nx;j++)
        {
        Vline(pwin->wt,pwin->wbt,x[j],0);
/*      for(i=pwin->wt;i<=pwin->wbt;i++)
                {
                gotoxy(x[j],i);
                putch(179);
                }*/
        gotoxy(x[j],pwin->wt-1);
        putch(194);
        gotoxy(x[j],pwin->wbt+1);
        putch(193);
        }
tx(pwin);
}
/*******************/
row(win *pwin,int Nx,int x[])/*row of local window*/
{
int j;
window(1,1,80,25);
txborder(pwin);
for(j=0;j<Nx;j++)
        {
        Hline(pwin->wl,pwin->wr,x[j]+pwin->wt-1,0);
/*      for(i=pwin->wl;i<=pwin->wr;i++)
                {
                gotoxy(i,x[j]+pwin->wt-1);
                putch(196);
                }*/
        gotoxy(pwin->wl-1,x[j]+pwin->wt-1);
        putch(195);
        gotoxy(pwin->wr+1,x[j]+pwin->wt-1);
        putch(180);
        }
tx(pwin);
}
/**********************/
Hline(int left,int right,int line,int type)
{
int style[]={196,205,178,219,223},i;
gotoxy(left,line);
for(i=left;i<=right;i++)
        putch(style[type]);
}
/***********************/
Vline(int top,int bottom,int line,int type)
{
int style[]={179,186,178,219},i;
for(i=top;i<=bottom;i++)
        {
```

```
        gotoxy(line,i);
        putch(style[type]);
        }
}
/**********************/
net(win *pwin,int nrow,int *row,int ncol,int *col)
{
int i,j;
window(1,1,80,25);
txborder(pwin);
for(i=0;i<ncol;i++)
        for(j=0;j<nrow;j++)
                {
                gotoxy(col[i],row[j]+pwin->wt-1);
                putch(197);
                }
tx(pwin);
}


/*******************/
int textmalloc(win *pwin,void far **bin)
{
if((*bin=farmalloc(textsize(pwin))) ==NULL ) return FALSE;
savewin(pwin,*bin);
return TRUE;
}
```

fUsiinsu SHOW.C

```
#include "c:\bc\data\include.h"
win bloomtext={1,3,1,23,WHITE,LIGHTRED,FALSE,1,BLUE};
extern win fulltext;

bloom()
{
int i,j;
txclr(&fulltext);
for(i=1;i<=10;i++)
        {
        for(j=0;j<10000;j++);
        bloomtext.wl=40-(float)(3.8*(float)i);
        bloomtext.wr=41+(float)(3.8*(float)i);
        boxclr(&bloomtext);
        }
printcenter(&bloomtext,6,"A DECISION SUPPORT SYSTEM");
printcenter(&bloomtext,7,"FOR");
printcenter(&bloomtext,8,"EVALUATING AUTOMATION PROJECTS");
printcenter(&bloomtext,10,"BY : DAVID SAMANYAPORN C215718");
printcenter(&bloomtext,11,"ADVISE BY : Asst. Prof. CHUVEJ CHANSANGAVEJ, Ph.D.");
printcenter(&bloomtext,12,"CO-ADVISE BY : Asst. Prof. MANOP REODECHA, Ph.D.");
printcenter(&bloomtext,14,"DEPARTMENT OF INDUSTRIAL ENGINEERING ");
printcenter(&bloomtext,15,"FACULTY OF ENGINEERING");
printcenter(&bloomtext,17,"CHULALONGKORN UNIVERSITY");
}
```

โปรแกรม TEXT.C

```
#include "c:\bc\data\include.h"
int Nproject=1;
int    handler(int errvalue,int ax,int bp,int si);
int ctrlhandler();
extern int Nproject;
extern int Nattrselect;
extern char altfile[26];
extern char attrfile[26];
extern win fulltext;
win headwin={1,1,80,1,WHITE,BLACK,FALSE,0,BLACK};
extern menubar headmenu;
extern win boardtext,fulltext;
extern char *period[2];
int progrid=0;
int gk;
int group;
win atext[]={{2,3,12,5,WHITE,BLUE,FALSE,1,MAGENTA},
                        {18,3,38,8,WHITE,BLUE,FALSE,1,MAGENTA},
                        {34,3,54,7,WHITE,BLUE,FALSE,1,MAGENTA},
                        {50,3,68,6,WHITE,BLUE,FALSE,1,MAGENTA},
                        {65,3,77,8,WHITE,BLUE,FALSE,1,MAGENTA}};
menubar amenu[]={{3,3,1,BLACK,WHITE,NULL},
                                {6,6,1,BLACK,WHITE,NULL},
                                {5,5,1,BLACK,WHITE,NULL},
                                {4,4,1,BLACK,WHITE,NULL},
                                {6,6,1,BLACK,WHITE,NULL}};

char *achoice[]={"LOAD","SAVE","QUIT"};
char *bchoice[]={"ALTERNATIVES","CASHFLOW","ATTRIBUTES","ATTRIBUTE
WEIGHTING","UTILITIES","ALTERNATIVE SCORING"};
char *cchoice[]={"MONETARY","UTILITY","UTILITY FUNCTIONS","RANKING","GRAPH"};
char *dchoice[]={"UTILITY FUNCTIONS","MONETARY","UTILITY","RANKING"};
char *echoice[]={"NC/CNC","MIS","MRP/MRP II","FMS","CAD","CIMS"};

ASST Aasst[]={{4,0,""},
                                {1,11,"Load a DATA FILE from disk"},
                                {2,12,"Save current DATA to disk"},
                                {3,13,"Quit CUMAUM"}};
ASST Basst[]={{7,0,""},
                                {1,21,"Input Alternative Details"},
                                {2,22,"Input Alternatives' Cash Flow Details"},
                                {3,23,"Choose and Input Attributes"},
                                {4,24,"Weighting all chosen Attributes"},
                                {5,25,"Create Utility Function of Attributes"},
                                {6,26,"Scoring all Alternatives"}};

ASST Casst[]={{6,0,""},
                                {1,31,"Monetary factors Analysis"},
                                {2,32,"Show Utility Value of Attributes"},
                                {3,33,"Show Risk Attitude for Attributes"},
                                {4,34,"Show Rank and Total Utility Values of
Alternatives"},
                                {5,35,"Show Graph of Alternatives' Utility Values"}};
ASST Dasst[]={{5,0,""},
```

```
                        {1,41,"Print out Utility Function"},
                        {2,42,"Print out Cash Flow and Monetary Factors"},
                        {3,43,"Print out Utility Values"},
                        {4,44,"Print out Rank of Alternatives"}};
ASST Easst[]={{7,0,""},
                        {1,51,"Suggestion for NC/CNC project"},
                        {2,52,"Suggestion for MIS project"},
                        {3,53,"Suggestion for MRP/MRP II project"},
                        {4,54,"Suggestion for FMS project"},
                        {5,55,"Suggestion for CAD project"},
                        {6,56,"Suggestion for CIMS project"}};
ASST *asstgroup[5];

/*************************/
int newprogram()
{
int pr=progrid;
int st;
pr=callmenu();
st = checkprogstatus(pr); /* checkprogram status available or not */
if(pr != progrid  && pr!=0 && st==TRUE)/*not select*/
        {
        progrid=pr;
        return TRUE;
        }
if(pr==23 && progrid==23) return TRUE;
return FALSE;
}
/*********************/
int callmenu()
{
int react=ESC;
int instant=FALSE;
void *bin;
bin=malloc(160);
gettext(1,1,80,1,bin);
while(group != ESC)
        {
        switch (react)
                {
                case 0:
                case ESC:
                                if(react==0) instant=TRUE;
                                if(react==ESC) instant=FALSE;
                                group=menurow(instant);
                                if(group==ESC) break;
                                switch(group)
                                        {
                                        case 1:
                                        case 2:
                                        case 3:
                                        case 4:
                                        case 5:react=joba(group-1);break;
                                        }
```

```
                                        break;
                        default: puttext(1,1,80,1,bin);
                                        free(bin);
                                        return(react);
                        }
                }
group=1;
puttext(1,1,80,1,bin);
free(bin);
return(0);
}
/*****************************/
main()
{
harderr(handler);
ctrlbrk(ctrlhandler);
checkcon();
cursoff();

/*****/
bloom();
trim(altfile,25);
trim(attrfile,25);
projectalloc();
makeattrnode();  /* set attribute pointer */
/* attrfixdoc(); */
attrdoc();  /*load attribute document*/

amenu[0].choice=(char **)&achoice;    /* setup popup legend */
amenu[1].choice=(char **)&bchoice;
amenu[2].choice=(char **)&cchoice;
amenu[3].choice=(char **)&dchoice;
amenu[4].choice=(char **)&echoice;

asstgroup[0]=(ASST *)Aasst;
asstgroup[1]=(ASST *)Basst;
asstgroup[2]=(ASST *)Casst;
asstgroup[3]=(ASST *)Dasst;
asstgroup[4]=(ASST *)Easst;

makeuseattr();
getkey();
demo();                                 /* data for testing */
checklastfield();
 /* temporary filesize */
setupsize();

/*txclr(&fulltext);
boxclr(&boardtext);*/
while((progrid=callmenu())==0); /* for first screen */
while(TRUE)
        {
        switch(progrid)
                {
```

```
                case 11: loaddisp(); break;
                case 12: savedisp(); break;
                case 13: txclr(&fulltext);          exit(0);

                case 21: altcontrol(); break;
                case 22: boardwriter(); break;
                case 23: attrwriter() ;break;
                case 24: attrscale(); break;
                case 25: utilwriter() ;break;
                case 26: scorewriter(); break;

                case 31: monetwriter();break;
                case 32: utilana(); break;
                case 33: utilfwriter(); break;
                case 34: ranking(); break;
                case 35: graphwriter();break;

                case 41: utilfrepdisp();break;
                case 42: altrep(); break;
                case 43: utilrep();break;
                case 44: rankrepdisp();break;

                case 51:
                case 52:
                case 53:
                case 54:
                case 55:
                case 56:infowriter(); break;
                default : cputs("NON APPLICATION");
                                        break;
                }
        }
}
/*********************/
int checkprogstatus(int prog)
{
switch(prog)
        {
        case 12 :
        case 26 :
        case 32 :
        case 34 :
        case 35 :
        case 43 :
        case 44 : if(Nproject <1 || Nattrselect < 1 ) return FALSE;

        case 22 :
        case 31 :
        case 42 : if(Nproject < 1 ) return FALSE ;

        case 25 :
        case 24 :
        case 33 :
        case 41 : if(Nattrselect<1) return FALSE;
```

```
        default : return TRUE;
        }
}
```

```
/*****************************/
int joba(int title)
{
int legal[]={LEFT,RIGHT,0};
int ch;
int program;
void far *(far *bin)=NULL;
textmalloc(&atext[title],bin);
/*bin=malloc(textsize(&atext[title]));
savewin(&atext[title],bin);          */
while( (ch=popup (&atext[title],&amenu[title],legal,asstgroup[title])) !=ESC)
        {
        switch (ch)
                {
                case RIGHT: headmenu.bar++;
                                if(headmenu.bar>=6) headmenu.bar=1;
                                pushdown(&atext[title],*bin);
                                return 0;
                case LEFT:      headmenu.bar--;
                                if(headmenu.bar <= 0 ) headmenu.bar=5;
                                pushdown(&atext[title],*bin);
                                return 0;
                case RET:       switch((program=(title+1)*10+amenu[title].bar))
                                        {
                                        case 11 : break;
                                        case 12 : break;

/*                                      if(progrid != 0) save();
                                                else program=ESC;
                                                break;    may create
meassage for save nothing*/
/*                                      case 13:  helpstart(); break; */
                                        case 13:break;
                                        case 21:break;
                                        case 22:break;
                                        case 23:break;
                                        case 24:break;
                                        case 25:break;
                                        case 26:break;

                                        case 31:break;
                                        case 32:break;
                                        case 33:break;
```

```
                case 34:break;
                case 35:break;

                case 41:break;
                case 42:break;
                case 43:break;
                case 44:break;

                case 51:break;
                case 52:break;
                case 53:break;
                case 54:break;
                case 55:break;
                case 56:break;
                default : program=ESC; break;
                }
        pushdown(&atext[title],*bin);
        return(program);

                }
        }
pushdown(&atext[title],*bin);
return(ESC);
}
/*helpstart()
{
txclr(&fulltext);
help(".131");
} */
```

fUsIInsU UTiL.C

```
#include "c:\bc\data\include.h"
#include "c:\bc\data\define2.h"
#include "c:\bc\data\extern.h"

typedef struct STAT {double sumX,sumY,sumX2,sumY2,sumXY;}stat;
#define Line 10
extern win boardtext;
extern int buff_key;
extern int attrindex[ATTRMAX];
extern float shiftlower,shiftupper;
win coltext={2,11,79,20,WHITE,BLUE,FALSE,0,WHITE};
win ttext={3,10,16,21,WHITE,BLUE,FALSE,0,RED};

/*******************/
utilwriter()
{
int gk;
int utilNO=1,oldutil;
int lineF=1;
int Nline=18;
ASST asst2510[] = { {1,2511,"II+ENTER to select Attribute for making Utility Function;
F10-Main menu"}};
say("UTILITY","F10-menu","CUMAUM");
boxclr(&boardtext);
refreshutil(1,Nline,1);
assist(asst2510,1,&boardtext);
while((gk=getkey())!=ESC)
        {
        oldutil=utilNO;
       .switch(gk)
                {
                case UP:utilNO--;break;
                case DOWN: utilNO++;break;
                case RET:  if(!canutil(utilNO)) break;
                                        if(utilmake2(utilNO)!=ESC)      utilcalc(utilNO);
                                        refreshutil(lineF,Nline+lineF-1,utilNO);
                                        /*continue;*/
                                        break;
                case F10: if(newprogram())
                                                {
                                                anacalc();
                                                return;
                                                }
                                        tx(&boardtext);break;
                }
         adjust2(1,Nattrselect,&utilNO);


         if ( scrollpage(&lineF,lineF+Nline-1,utilNO))
                {
                refreshutil(lineF,Nline+lineF-1,utilNO);
                continue;
                }
         hiliutil(lineF,utilNO,1);
```

```
        if(oldutil!=utilNO)
                hiliutil(lineF,oldutil,0);
        assist(asst2510,1,&boardtext);
        }
}
/***********/
int canutil(int utilNO)/* for check that error occur */
{
if(attrindex[utilNO-1]==0 && shiftlower==shiftupper) return FALSE;
return TRUE;
}
/***********/
refreshutil(int lineF,int lineL,int utilNO)
{
int i;
txclr(&boardtext);
speak(1,1,"ATTRIBUTE");
for(i=lineF;i<=Nattrselect && i<=lineL;i++)
        {
        gotoxy(1,2+i-lineF);
        cputs(useattr[i-1]->name);
        gotoxy(40,2+i-lineF);
        switch(useattr[i-1]->type)
                {
                case 0: cprintf("N/A"); break;
                case 1: cprintf("Straight line"); break;
                case 2: cprintf("S-shape"); break;
                case 3: cprintf("Concave"); break;
                case 4: cprintf("Convex"); break;
/*              case 0: cprintf("N/A");
                                break;
                case 1: cprintf("U(x) = %10.2f %+10.2fx",useattr[i-1]->a,useattr[i-1]->b);
                                break;
                case 3:
                case 4:cprintf("U(x) = %10.2f * x^%10.2f",useattr[i-1]->a,useattr[i-1]->b);
                                break;
                case 2: cprintf("U(x) = exp(%10.2f %+10.2f/x)",useattr[i-1]->a,useattr[i-1]-
>b);
                                break;
*/              }
        }
hiliutil(lineF,utilNO,1);
}

/*****/
hiliutil(int lineF,int util,int hi)
{
if(hi) txinv(&boardtext);
gotoxy(1,2+util-lineF);
cputs(useattr[util-1]->name);
if(hi) txcolor(&boardtext);
}
```

```
/*************/
int utilmake2(int utilNO)
{
int i,gk;
float value,max=9999000.00;
CO cobuff[FIT];
int value1,value2;
ASST asst2520[]={{2,1,"Input Value or use II + ENTER"},
                                        {1,2521,""}};
say("UTILITY MAKING","ESC-previous","CUMAUM");
assist(asst2520,1,&boardtext);
boxclr(&boardtext);

memset(cobuff,0,FIT*sizeof(CO));
cobuff[0].x=attrindex[utilNO-1]==0 ? shiftlower : useattr[utilNO-1]->lower; /* set boundary
of utility */
cobuff[1].x=attrindex[utilNO-1]==0 ? shiftupper : useattr[utilNO-1]->upper; /* if use NPV
use the axis shifted value */
cobuff[0].y=!useattr[utilNO-1]->positive;
cobuff[1].y=useattr[utilNO-1]->positive;

speak(1,1,"Suppose you have a choice either a 50:50 chance of");
speak(53,1,useattr[utilNO-1]->name);
speak(1,2,"at _____ or _____ you will take the certain amount");
speak(1,3,"if it were _____");

for(i=2;i<FIT;i++)
        {
        float rnd=.5;
        int arrow,round,oldarrow,ok;
        float upper,lower;

        round=i-1; /*input time */
        utilsort(&cobuff,i,&value1,&value2);

        upper=cobuff[value1].x;
        lower=cobuff[value2].x;

        arrow=5;                        /* set first arrow */
        valuex(&value,&upper,&lower,arrow);   /* and value */

        colwriter(round,&upper,&lower);    /* draw column */
        hiliarrow(round,arrow,&value,1);   /* draw first arrow */

        txinv(&boardtext);
        gotoxy(4,2); Tprintf(&upper);
        gotoxy(18,2); Tprintf(&lower);
        gotoxy(12,3); /*Tprintf(&value);*/ cputs("??????????");
        tx(&boardtext);

        ok=FALSE;
        while(!ok)
                {
                curson();    /* display cursor */
```

```
                gotoxy(12,3); /* at corodinate */
                gk=getkey();
                cursoff();

                oldarrow=arrow;
                if(decimal(gk))
                        {
                        buff_key=gk;
                        do {
                                gotoxy(12,3);
                                getfloat(&value,&max,12,3,10);
                                buff_key=0;
                                }
                        while(value<=lower || value >=upper);
                        cobuff[i].x=value;
                        arrowx(&value,&upper,&lower,&arrow);
                        hiliarrow(round,oldarrow,&value,0);
                        hiliarrow(round,arrow,&value,1);
                        ok=TRUE;
                        break;
                        }
                switch(gk)
                        {
                        case UP: arrow--; break;
                        case DOWN:arrow++; break;
                        case RET: cobuff[i].x=value;ok=TRUE;break;
                        case ESC: cursoff();return ESC;
/*                      case F1 : help(".241"); tx(&boardtext); break; */
                        }
                adjust2(1,Line,&arrow);
                valuex(&value,&upper,&lower,arrow);
                hiliarrow(round,oldarrow,&value,0);
                hiliarrow(round,arrow,&value,1);
                }
        utilsort(&cobuff,i,&value1,&value2);

        cobuff[i].y=cobuff[value1].y*rnd+cobuff[value2].y*(1-rnd); /* make utility of x*/
        }
memmove(useattr[utilNO-1]->co,cobuff,FIT*sizeof(CO));
useattr[utilNO-1]->utilchange=0; /* reset to recalculate utility */
return(TRUE);
}
/*********************/
valuex(float *value,float *upper,float *lower,int arrow)
{
*value=(*upper-*lower)/(Line+2)*(Line+1-arrow)+ *lower;
}
/*********************/
arrowx(float *value,float *upper,float *lower,int *arrow)
{
*arrow=(*value-*lower) *(Line+2)/(*lower-*upper)+(Line+1);
}
/*********************/
hiliarrow(int round,int arrow,float *value,int hi)
```

```
{
if(hi)
        {
        tx(&boardtext);
        gotoxy(12,3);   Tprintf(value);
        }
tx(&ttext);
gotoxy(1,arrow+1);
        if(hi) cputs("->");
        else  cputs("  ");
gotoxy(4,arrow+1);
        if(!hi) cputs("        ");
        else Tprintf(value);
tx(&boardtext);
}
/*********************/
colwriter(int round,float *upper,float *lower)
{
int col[]={0};
/*draw border */
ttext.wl=round*11-8;
ttext.wr=round*11+5;
boxclr(&ttext);

gotoxy(4,1); Tprintf(upper);
gotoxy(4,12); Tprintf(lower);

col[0]=round*11-6;
tx(&coltext);
column(&coltext,1,col);
}

/**********************/
utilsort(CO (*pcobuff)[FIT],int item,int *value1,int *value2)
{
int i,j,maxi;
int index[FIT];
float maxone=0;
memset(index,0,FIT*sizeof(int));
for(i=0;i<item;i++)  /********** sorting **********/
        {
        for(j=0;j<FIT;j++)
                {
                if((*pcobuff)[i].x>=(*pcobuff)[index[j]].x)
                        {
                        memmove(&index[j+1],&index[j],(FIT-1-j)*sizeof(int));
                        index[j]=i;
                        break;
                        }
                }
        }
for(i=0;i<item-1;i++)   /********** select *********/
        {
        if((*pcobuff)[index[i]].x-(*pcobuff)[index[i+1]].x>maxone)
```

```
                {
                maxone=(*pcobuff)[index[i]].x-(*pcobuff)[index[i+1]].x;
                maxi=i;
                }
        }
*value1=index[maxi];
*value2=index[maxi+1];
}

/*******************/
utilcalc(int utilNO)
{
CO data[FIT],dataX[FIT],(*pdata)[FIT];
double R2,maxR=0,a,b;
int i,j,maxtype,k,nvalid,nvX;
pdata=&useattr[utilNO-1]->co;
for(i=1;i<=3;i++)
        {
        k=0; /* pointer of data */
        for(j=0;j<8;j++) /* pointer of useattr*/
                {
                switch(i)
                        {
                        case 1:data[k].y=(*pdata)[j].y;
                                data[k].x=(*pdata)[j].x;
                                        break; /* linear */
                        case 3:if((*pdata)[j].y<=0.001 || (*pdata)[j].x<=0.01) continue;
                                        data[k].y = (float)log((double)(*pdata)[j].y);
                                        data[k].x=(float)log((double)(*pdata)[j].x);
                                        break; /* popower */
                        case 2:if((*pdata)[j].y<=0.001 || (*pdata)[j].x<=0.01) continue;
                                        data[k].x = 1/(*pdata)[j].x;
                                        data[k].y = (float)log((double)(*pdata)[j].y);
                                        break;
                        }
                k++;
                }
        nvalid=k;
        cor(&data,nvalid,&R2);
        if(fabs(R2)>fabs(maxR))
                {
                maxtype=i;
                maxR=R2;
                nvX=nvalid;
                memmove(dataX,data,sizeof(data));
                }
        }
equation(&dataX,nvX,&a,&b);
if (maxtype==3 && b>1) maxtype=4;
useattr[utilNO-1]->type=maxtype;
switch(maxtype)
        {
        case 3:
        case 4:useattr[utilNO-1]->a=(float)exp(a);
```

```
                    useattr[utilNO-1]->b=(float)b;
                    break;
        case 1:
        case 2:useattr[utilNO-1]->a=(float)a;
                    useattr[utilNO-1]->b=(float)b;
                    break;
        }
}


cor(CO (*data)[FIT],int nvalid,double *R2)
{
double R21,R22;
stat para;
statdetail(data,nvalid,&para);
R21=pow((para.sumXY-para.sumX*para.sumY/(double)nvalid),(double)2);
R22=(para.sumX2-pow(para.sumX,(double)2)/(double)nvalid) * (para.sumY2-
pow(para.sumY,(double)2)/(double)nvalid);
*R2=R21/R22;
}

equation(CO (*data)[FIT],int nvX,double *a,double *b)
{
stat para;
statdetail (data,nvX,&para);
*a=(para.sumX2*para.sumY - para.sumX*para.sumXY)/
  ((double)nvX*para.sumX2-pow(para.sumX,(double)2));
*b=(FIT*para.sumXY-para.sumX*para.sumY)/
  ((double)nvX*para.sumX2-pow(para.sumX,(double)2));
}

statdetail(CO (*data)[FIT],int nvalid,stat *para)
{
int i;
memset(para,0,sizeof(stat));
for (i=0;i<nvalid;i++)
        {
        para->sumX += (double)(*data)[i].x;
        para->sumX2 += pow((double)(*data)[i].x ,(double)2);
        para->sumY += (double)(*data)[i].y;
        para->sumY2 +=  pow((double)(*data)[i].y ,(double)2);
        para->sumXY += (double)(*data)[i].x * (double)(*data)[i].y;
        }
}
```

fUsiinsu UTILF.C

```
#include "c:\bc\data\include.h"
#include "c:\bc\data\define2.h"
#include "c:\bc\data\extern.h"
extern float shiftlower,shiftupper;
extern win boardtext;
extern int buff_key;
extern int attrindex[];

utilfwriter()
{
int gk;
int lineF=1,Nline=19;
int attr=1,oldattr;
ASST asst3310[]={{1,3311,"F10-Main menu; N/A = No Utility Function"}};
say("UTILITY FUNCTIONS","F10-menu","CUMAUM");
assist(asst3310,1,&boardtext);
refreshutilf(1,Nline);
while(TRUE)
        {
        gk=getkey();
        oldattr=attr;
        switch(gk)
                {
                case UP  : attr--; break;
                case DOWN : attr++;break;

                case F10 : if(newprogram()) return;
                }
        adjust2(1,max( Nattrselect-Nline+1,1),&attr);

        if(attr != oldattr)
                refreshutilf(attr,attr+Nline-1);
        assist(asst3310,1,&boardtext);
        }
}

/************************/
refreshutilf(int lineF,int lineL)
{
int i;
boxclr(&boardtext);    /* should be make local window */
speak(15,1,"ATTRIBUTE");
speak(50,1,"RISK ATTITUDE");
for(i=lineF;i<=Nattrselect && i<=lineL;i++)
        {
        gotoxy(8,3+i-lineF);
        cputs(useattr[i-1]->name);
        gotoxy(50,3+i-lineF);
        switch(useattr[i-1]->type)
                {
                case 0: cputs("N/A");break;
                case 1: cputs("Risk Indifference"); break;
                case 2: cputs("S-curve"); break;
                case 3: cputs("Risk Aversion"); break;
```

```
                case 4: cputs("Risk Seeking"); break;
                }
        }
}

/******************************** utilana */
utilana()
{
int prj=1,startprj=1,oldprj;
int attr=1,oldattr,i;

int lineF=1,Nline=17;
int cl[]={28,34,39,44,49,54,59,64,69,74};
int rw[]={2,20};
float totalw=0;
ASST asst3210[]={{1,3211,"Use \x1B\x1A select Alternative; F10-Main menu"}};

/* anacalc() */
boxclr(&boardtext);
assist(asst3210,1,&boardtext);
column(&boardtext,10,cl);
row(&boardtext,2,rw);
net(&boardtext,2,rw,10,cl);

speak(10,1,"Attribute");
speak(28,1,"Wt.");
speak(10,21,"TOTAL");

refreshutilana(1,1,Nline);
hiliana(1,1,1);

/**** total sum */
for(i=0;i<Nattrselect;i++)
        {
        totalw+=useattr[i]->ratio;
        }

gotoxy(29,21); cprintf("%4.2f",totalw);
while(TRUE)
        {
        oldattr=attr;
        oldprj=prj;
        switch((getkey()))
                {
                case LEFT :prj--;break;
                case RIGHT: prj++; break;

                case UP : attr--;break;
                case DOWN : attr++; break;

                case TAB        : prj +=4; break;
                case 3840  : prj-=4;break;
                case HOME  : prj=1; break;
                case END   : prj=Nproject;break;
```

```
                case F10        : if(newprogram()) return;
                                  else tx(&boardtext);
                                  break;
                }

        adjust2(1,max (Nattrselect-Nline+1,1),&attr);
        adjust2(1,Nproject,&prj);

        if(oldattr !=attr)
                refreshutilana(startprj,attr,attr+Nline-1);

        if (scrollpage(&startprj,startprj+9-1,prj))
                refreshutilana(startprj,attr,attr+Nline-1);

        else hiliana(startprj,oldprj,0);
        hiliana(startprj,prj,1);
        assist(asst3210,1,&boardtext);
        }
}
int clana[]={34,39,44,49,54,59,64,69,74};
/*****************/
refreshutilana(int startprj,int attr,int lineL)
{
int j,i;
int clprj;
for(i=attr;i<=Nattrselect && i<=lineL;i++)
        {
        gotoxy(2,i-attr+3); cputs(useattr[i-1]->name);
        gotoxy(29,i-attr+3); cprintf("%4.2f",useattr[i-1]->ratio);
        }

for(j=startprj;j<=Nproject && j<=startprj+8;j++)
        {
        clprj=j-startprj;
        gotoxy(clana[clprj]+1,1);
        cprintf("%2d",j);  /* No of project header */

        for(i=attr;i<=Nattrselect && i <=lineL ;i++)
                {
                gotoxy(clana[clprj],i-attr+3);
                if(useattr[i-1]->type==0 || project[j-1]->util[attrindex[i-1]]==-1)
                        cprintf("%4s","N/A");
                else
                        cprintf("%4.2f",project[j-1]->util[attrindex[i-1]]);
                }

        gotoxy(clana[clprj],21);
                if(project[j-1]->total == -1)
                        cprintf("%4s","N/A");
                else
                        cprintf("%4.2f",project[j-1]->total);
        }
}
```

```
/*************/
hiliana(int startprj,int prj,int hi)
{
char head[80];
if(hi) txinv(&boardtext);
gotoxy(clana[prj-startprj]+1,1);
cprintf("%2d",prj);
if(hi)
        {
        sprintf(head,"UTILITY OF %25s",project[prj-1]->name);
        say(head,"use arrow key","CUMAUM");
        }
tx(&boardtext);
}
/************************/
anacalc() /* not clear */
{
int y,x;
for(x=1;x<=Nattrselect;x++)
        if(TRUE)  /* temporary */
                {
                for(y=1;y<=Nproject;y++)        recalcutil(x,y);
                useattr[x-1]->utilchange=1;
                }
for(y=0;y<Nproject;y++) totalutil(y);
rankproject();
}
/****************/
recalcutil(int attrx,int projecty)
{
int i=attrx-1,j=projecty-1;
int k=attrindex[i];
float x,xbest,xworst,xutil;
if(k==0)
        {
        xbest=useattr[i]->positive ? shiftupper : shiftlower ;
        xworst=useattr[i]->positive ? shiftlower : shiftupper ;
        }
else
        {
        xbest= useattr[i]->positive ? useattr[i]->upper : useattr[i]->lower;
        xworst=useattr[i]->positive ? useattr[i]->lower : useattr[i]->upper;
        }
if((xbest-xworst <= 0 && useattr[i]->positive) || (xbest-xworst>=0 && !useattr[i]->positive))
x=0;
else x=(project[j]->score[k]-xworst)/(xbest-xworst);

if(x<0) x=0;
if(x>1)  x=1;
switch(useattr[attrx-1]->type)
        {
        case 0: xutil=-1;
                        break;
        case 1: xutil= x; /* linear */
```

```
                        break;
        case 2: if(x<0.001) xutil=0;
                        else xutil= exp(0.7 -0.7/x); /* s-shape */
                        break;
        case 3: xutil=  (float)sqrt(1-pow((double)(x-1),2)); /* concave */
                        break;
        case 4: xutil=   (float)(1-sqrt(1-pow((double)x,2))); /* convex */
                        break;
        }
if(xutil<0) xutil=0;
if(xutil>1) xutil=1;

project[j]->util[k]=xutil;
}
/**********************/
totalutii(int prj)  /* recalculate total utilities of specific project */
{
int i;

/*  if anyone if n/a */
for(i=0;i<Nattrselect;i++)
        if(useattr[i]->type==0 || project[prj]->util[attrindex[i]]==-1)
                {
                project[prj]->total=-1;
                return;
                }

project[prj]->total=0;
for(i=0;i<Nattrselect;i++)
        {
        project[prj]->total+=project[prj]->util[attrindex[i]]*useattr[i]->ratio;
        project[prj]->total=((float)((int) (project[prj]->total*100)))/100; /* cut tag */
        }
}
/********** ranking ****/
rankproject()
{
int i,j;
float maxl,lastmax=1;
int ranknow=1,themax;

for(i=0;i<Nproject;i++) project[i]->rank=-1; /* reset rank data */
for(j=1;j<=Nproject;j++)
        {
        maxl=-2;
        for(i=0;i<Nproject;i++)
                {
                if(project[i]->rank != -1 ) continue;
                if(project[i]->total >= maxl )
                        {
                        maxl = project[i]->total;
                        themax=i;
                        }
                }
```

```
        if(maxl < lastmax)
                {
                ranknow=j;
                lastmax=maxl;
                }
        project[themax]->rank=ranknow;
        }
}
```

HEADER FILE

โปรแกรม ASST.H

```
typedef struct {
                int key;
                int assid;
                char label[75];
                } ASST;
```

fUsIInsU BIOSAREA.H

```
/* biosarea.h: ROM BIOS data area at 0x0040:0 in memory */

#ifndef byte
#define byte  unsigned char
#endif

/* BIT FIELDS USED IN ROM BIOS DATA AREA */
typedef struct {
        unsigned        hasFloppies : 1,
                                nu1 : 1,
                                mbRAM : 2,
                                initVideo : 2,
                                nDisks : 2,
                                nu8 : 1,
                                nSerialPorts : 3,
                                gamePort : 1,
                                mu13 : 1,
                                nLPT : 2;
}       EQFLAGS;

typedef struct {
        unsigned        riteShiftDown : 1,
                                leftShiftDown : 1,
                                ctrlShiftDown : 1,
                                altShiftDown : 1,
                                scrollLockOn : 1,
                                numLockOn : 1,
                                capsLockOn : 1,
                                insOn : 1,
                                unused : 3,
                                ctrlNumLockOn : 1,
                                scroolLockDown : 1,
                                numLockDown : 1,
                                capsLockDown : 1,
                                insDown : 1;
}       KBDFLAGS;

typedef struct {
        unsigned        serialPortAddr[4];
        unsigned        parallelPortAddr[4];
        EQFLAGS                 eqptFlags;
        byte            mfgrTestFlags;
        unsigned        mainMem;
        unsigned        expRAM;
        KBDFLAGS        kbdStat;
        byte            keypad;
        unsigned        kbdBuffHead;
        unsigned        kbdBuffTail;
        char            kbdBuff[32];
        byte            seekStat;
        byte            motorStat;
        byte            motorCnt;
        byte            diskErr;
        byte            NECStatus[7];
```

```
    byte        videoMode;
    unsigned    scrnWidth;
    unsigned    vidBuffSz;
    unsigned    vidBuffOfs;
    byte        cursPos[8][2];
    byte        cursBottom;
    byte        cursTop;
    byte        activeDispPage;
    unsigned    activeDispPort;
    byte        CRTModeReg;
    byte        palette;
    unsigned    dataEdgeTimeCount;
    unsigned    CRCReg;
    char        lastInputValue;
    unsigned    tick;
    int             hour;
    byte        timerOverflow;
    byte        brkStat;
    unsigned    resetFlag;
    long        hardDiskStat;
    byte        parallelTimeout[4];
    byte        serialTimeout[4];
    unsigned    kbdBuffOfs;
    unsigned    kbdBuffEnd;
} BIOSDATA;
```

fusions DEFINE.H

```
/* DEFINE.H */
#define MAX   20
/************************ key code */
#define RET             13
#define UP              18432
#define DOWN            20480
#define LEFT   19200
#define RIGHT           19712
#define PgUp    18688
#define PgDn   20736
#define TAB             9
#define CTR_Y           25
#define CTR_N           14
#define ESC    27
#define BACK   8
#define SPACE           32

#define F1              15104
#define F2              15360
#define F3              15616
#define F4              15872
#define F5      16128
#define F6      16384
#define F7      16640
#define F8      16896
#define F9      17152
#define F10     17408

#define HOME  18176
#define END    20224
#define INS             20992
#define DEL             21248

#define TRUE  1
#define FALSE 0

#define PROJECTMAX 20
#define YEARMAX    20
#define valuesize 16
#define calcsize 13
#define namelength 25
#define ATTRFIX 11
#define ATTRMAX 30
typedef struct {unsigned char wl,wt,wr,wbt,
                                                    tc,tb,inv,
                                                    brd,bc;
                                } win ;
typedef struct {unsigned char lineM,
                                                Nchoice,
                                                bar,
                                                fore,
                                                back;
                                char (*(*choice));
                                } menubar ;
```

```
/*typedef struct{ char NAME[25];
                        float cash[20],
                              interest,
                              Equp_cost,
                              Access_cost,
                              Other_cap,
                              Total_cap,
                              Tax_credit,
                              Engineer,
                              Install,
                              Start,
                              Other_ex,
                              Total_ex,
                              Total_afl,
                              Direct_Lab_sav,
                              Inderect_Lab_sav,
                              Mat_sav,
                              Maint_sav,
                              Other_sav,
                              Other_cost,
                              Total_sav,
                              Depre,
                              Not_sav,
                              Nat_sav,
                              Nat_fac,
                              NPV,
                              IRR,
                              PAYBACK
                              } PRJ;
*/
struct  PRJ{ char name[namelength];
                    float interest;
                    int period;
                    int comp;
                    float value[YEARMAX][valuesize];
                    float score[ATTRMAX];
                    int rank;
                /********** not save *************/
                    float calc[YEARMAX][calcsize];
                    float NPV,IRR,PBP;
                    int yearuse;
                    int change;
                    float util[ATTRMAX];
                    float total;
                    } ;
```

fUsiinsu DEFINE2.H

```
#define FIT 8
typedef struct  {float x,y;} CO;
typedef struct  {
                char name[26];
                char unit[6];
                char fix;
                char unitchange;
                char use;

                char positive;
                float lower,upper;

                float scaling;
                char type;
                double a;
                double b;

                char far *(doc[5]);

                float ratio;
/*              char far *(*doc); */
                CO  co[FIT];
                int utilchange;
                } ATTR;
#define BARMAX 30
struct BAR { char *(name[BARMAX]);
                        int namesize;
                        int Nrecord;
                        int Allrec;
                        float min;
                        float max;
                        float *(value[BARMAX]);
                        int decimal;
                        char *legend;
                        };
```

fUsiinsU EXTERN.H

```
extern struct PRJ far * project[PROJECTMAX];
extern ATTR far *(attrnode[ATTRMAX]);
extern ATTR     attribute[ATTRMAX];
extern int Nproject;
extern int Nattravi;
extern int Nattrselect;
extern ATTR far *(useattr[ATTRMAX]);
```

fUsIInsU INCLUDE.H

```
/* INCLUDE.H */
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
#include <mem.h>
#include <alloc.h>
#include <string.h>
#include <math.h>
#include <alloc.h>
#include <dos.h>
#include <io.h>
#include <ctype.h>
#include <values.h>
#include "c:\bc\data\define.h"
#include "c:\bc\data\biosarea.h"
#include "c:\bc\data\asst.h"
```

ประวัติผู้เขียน

นายเดวิด สมัญญาภรณ์ เกิดวันที่ 10 เมษายน 2508 สำเร็จการศึกษาปริญญาวิทยาศาสตร์บัณฑิต (เทคโนโลยีชีวภาพ) จากคณะอุตสาหกรรมเกษตร มหาวิทยาลัยเกษตรศาสตร์ ในปีการศึกษา 2531 และเข้าศึกษาต่อในระดับปริญญามหาบัณฑิต ภาควิชาวิศวกรรมอุตสาหการ คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย ในปีการศึกษา 2532 ปัจจุบันทำงานเป็นวิศวกรอาวุโสประจำฝ่ายเบเกอรี่ ของ บริษัท เอสแอนด์พี ซินดิเคท จำกัด รับผิดชอบโครงการ MRPII