รายการอ้างอิง

ภาษาไทย

การไฟฟ้านครหลวง. อัตราค่าไฟฟ้า. กรุงเทพมหานคร : การไฟฟ้านครหลวง, 2540.

ภาษาอังกฤษ

American Society of Heating, Refrigerating and Air-Conditioning Engineers. ASHRAE Handbook Fundamentals. Atlanta : American Society of Heating, Refrigerating and Air-Conditioning Engineers, 1993.

Faye C. McQuiston, P.E. and Jeffrey D. Spitler, P.E. Cooling and Heating Load Calculation Manual. Atlanta : American Society of Heating, Refrigerating and Air-Conditioning Engineers, 1994.

Frank P. Incropera and David P. Dewitt. Fundamentals of Heat and Mass Transfer. New York : John Wiley & Son, 1990.

Frank Kreith and Jan F. Kreider. Principle of Solar Engineering. New York : McGraw-Hill, 1978

Jan F. Kreider and Ari Rabl. Heating and Cooling of Buildings. New York : McGraw-Hill, 1994.

John A.Duffie and William A.Beckman. Solar Energy Thermal Process. New York : John Wiley & Son, 1974

ภาคผนวก

ก. Source Code โปรแกรม TFM

```
/*****************************************************************************
*** COOLING LOAD CALCULATION BY TRANSFER FUNCTION METHOD ***
*****************************************************************************

*************************************************************
*****        INPUT AND OUTPUT FILES IN PROGRAM        *****
*************************************************************
*****  INPUT FILE                    *****
*****  IN.DAT  = BUILDING DATA       *****
*****  DATA.DAT = AA,BB,CC AND E COEFFIENCE DATA     *****
*****  TEMP.DAT = OUTSIDE TEMPERATURE DATA....       *****
*****  RH.DAT  = RELATIVE HUMIDITY DATA              *****
*****  I.DATA  = SOLAR IRRIADIENCE DATA              *****
*****  POWC.DAT = POWER CONSUMPTION OF CHILLER       *****

*****  OUTPUT FILE                   *****
*****  IT.TXT  =  THEORY SOLAR RADIATION             *****
*****  SHGF.TXT  =  SOLAR HEAT GAIN FACTOR           *****
*****  QT.TXT  = TOTAL HEAT GAIN IN YEAR             *****
*****  LOAD.TXT  = OUTPUT OF CALCUTION               *****
*************************************************************

*****************************************************
*****        DEFINITION OF SYMBOL           *****
*****************************************************

*****  J  = INTERATION NUMBER                  *****
*****  K  = INTERATION NUMBER                  *****
*****  L  = INTERATION NUMBER                  *****
*****  I  = INTERATION NUMBER OF TIME          *****
*****  M  = INTERATION NUMBER OF MONTH         *****
*****  N  = INTERATION NUMBER OF DIRECTION     *****

*****  UCI  = THERMAL CONDUCTIVITY (U) OF CEILING    *****
*****  UFL  = THERMAL CONDUCTIVITY (U) OF FLOOR      *****
*****  UP1  = THERMAL CONDUCTIVITY (U) OF PARTITION 1 *****
*****  UP2  = THERMAL CONDUCTIVITY (U) OF PARTITION 2 *****
*****  ACI  = CEILING AREA (M2)                *****
*****  AFL  = FLOOR AREA (M2)                  *****
*****  AP1  = PARTITION 1 AREA (M2)            *****
*****  AP2  = PARTITION 2 AREA (M2)            *****
```

```
*****  TCI  = OUTSIDE CEILING TEMPERATURE ('C)          *****
*****  TFL  = OUTSIDE FLOOR TEMPERATURE ('C)            *****
*****  TP1  = OUTSIDE PARTITION 1 TEMPERATURE ('C)      *****
*****  TP2  = OUTSIDE PARTITION 2 TEMPERATURE ('C)      *****
*****  NOP  = NUMBER OF PEOPLE IN BUILDING              *****
*****  SHF  = SENSIBLE HEAT FACTOR OF PEOPLE (W)        *****
*****  LHF  = LATENT HEAT FACTOR OF PEOPLE (W)          *****
*****  LD   = LIGHTING DENSITY (W/M2)                   *****
*****  A    = AIR-CONDITIONING AREA (M2)                *****
*****  AW   = EXTERNAL WALL AREA (M2)                   *****
*****  AG   = EXTERNAL GLASS AREA (M2)                  *****
*****  UG   = EXTERNAL GLASS THERMAL CONDUCTIVITY (U)   *****
*****  SC   = EXTERNAL GLASS SHADING COEFFICIENT        *****
*****  DAYMX= NUMBER OF DAYS IN EACH MONTH              *****
*****  AA   = APPARENT SOLAR IRRADIATION (W/M2)         *****
*****  BB   = ATMOSPHERIC EXTINCTION COEFFICIENT        *****
*****  CC   = SKY DIFFUSE FACTOR                        *****
*****  E    = EQUATION OF TIME (MIN)                    *****
*****  D    = DECLINATION DEGREE (DEG)                  *****
*****  T    = APPARENT SOLAR TIME (MIN)                 *****
*****  TT   = LOCAL STANDARD TIME (HOUR)                *****
*****  B    = SOLAR ATITUDE (DEG)                       *****
*****  P    = SOLAR AZIMUTHE (DEG)                      *****
*****  PP   = SOLAR AZIMUTHE (RAD)                      *****
*****  SS   = SURFACE SOLAR AZIMUTH (DEG)               *****
*****  Z    = INCIDENT ANGLE (DEG)                      *****
*****  Y    = RATIO OF VERTICAL/HORIZONTAL SKY DIFFUSE  *****
*****  ID   = DIRECT IRRADIANCE (W/M2)                  *****
*****  IDN  = DIRECT NORMAL IRRADIANCE (W/M2)           *****
*****  IDS  = DIFFUSE SKY IRRADIANCE (W/M2)             *****
*****  IDG  = DIFFUSE GROUND DEFLECTED                  *****
*****         IRRADIANCE (W/M2)                         *****
*****  IDD  = DIFFUSE IRRADIANCE (W/M2)                 *****
*****  IT   = TOTAL SOLAR IRRADIANCE (W/M2)             *****
*****  IR   = CHOICE TO USE REAL OR CALCULATE           *****
*****         SOLAR IRRADIANCE                          *****
*****          1   CALCULATE SOLAR IRRADIANCE          *****
*****          2   REAL SOLAR IRRADIANCE               *****
*****  AI   = ABSORPTION COEFFICIENT OF GLASS           *****
*****  TI   = TRANSMISSION COEFFICIENT OF GLASS         *****
*****  A1   = FIRST TERM OF ABSORBED COMPONENT          *****
*****  A2   = SECOND TERM OF ABSORBED COMPONENT         *****
*****  T1   = FIRST TERM OF TRANSMITTED COMPONENT       *****
```

```
*****  T2  = SECOND TERM OF TRANSMITTED COMPONENT      *****
*****  ZA1 = SUM OF 1ST TERM OF ABSORBED COMPONENT     *****
*****  ZA2 = SUM OF 2ND TERM OF ABSORBED COMPONENT     *****
*****  ZT1 = SUM OF 1ST TERM OF TRANSMITTED COMPONENT  *****
*****  ZT2 = SUM OF 2ND TERM OF TRANSMITTED COMPONENT  *****
*****  TC  = ABSORBED COMPONENT                        *****
*****  AC  = TRANSMITTED COMPONENT                     *****
*****  SHGF = SOLAR HEAT GAIN FACTOR (W/M2)            *****
*****  LRF = LONG-WAVE RADIATION FACTOR ('C)           *****
*****  TO  = OUTSIDE TEMPERATURE ('C)                  *****
*****  TE  = SOL-AIR TEMPERATURE ('C)                  *****
*****  TET = SUM OF SOL-AIR TEMPERATURE ('C)           *****
*****  TEA = AVERAGE SOL-AIR TEMPERATURE ('C)          *****
*****  RH  = RELATIVE HUMIDITY (%)                     *****
*****  TK  = OUTSIDE TEMPERATURE (K)                   *****
*****  CO  = TEMPERATURE COEFFICIENT                   *****
*****  PWS1 = EXPONENTIAL OF SATURATION PRESSURE (KPA) *****
*****  PWS = SATURATION PRESSURE (KPA)                 *****
*****  PW  = PATAIL PRESSURE OF WATER VAPOR (KPA)      *****
*****  WS  = SATURATION HUMIDITY RATIO                 *****
*****  W   = HUMIDITY RATIO                            *****

*****  FOR DESIGN CONDITION                            *****
*****  TD  = DESIGN TEMPERATURE ('C)                   *****
*****  RHD = DESIGN RELATIVE HUMIDITY                  *****
*****  TKD = TEMPERATURE (K)                           *****
*****  PWS1D= EXPONENTIAL OF SATURATION PRESSURE (KPA) *****
*****  PWSD = SATURATION PRESSURE (KPA)                *****
*****  PWD = PATAIL PRESSURE OF WATER VAPOR (KPA)      *****
*****  WD  = HUMIDITY RATIO                            *****
*****  VR  = VENTILATION FLOW RATE (M2/S)              *****
*****  Q   = HEAT GAIN FROM ROOF AND WALL (W/M2)       *****
*****  QAV = AVERAGE HEAT GAIN Q BY HEAT               *****
*****        TRANSFER EQUATION (W/M2)                  *****
*****  ZQ  = SUM OF HEAT GAIN Q (W/M2)                 *****
*****  ZQAV = AVERAGE HEAT GAIN Q BY ZQ (W/M2)         *****
*****  BF  = CONDUCTION TRANSFER COEFFICIENT B         *****
*****  DF  = CONDUCTION TRANSFER COEFFICIENT D         *****
*****  CF  = CONDUCTION TRANSFER COEFFICIENT C         *****
*****  ZB  = SUM OF 1 ST TERM IN HEAT GAIN EQUATION    *****
*****  ZD  = SUM OF 2 ND TERM IN HEAT GAIN EQUATION    *****
*****  ZC  = SUM OF 3 RD TERM IN HEAT GAIN EQUATION    *****
*****  QGC = HEAT GAIN BY CONVECTION FROM GLASS        *****
```

```
***** QGR = HEAT GAIN BY RADIATION FROM GLASS *****
***** QVS = SENSIBLE HEAT GAIN FROM VENTILATION *****
***** QVL = LATENT HEAT GAIN FROM VENTILATION *****
***** QVT = TOTAL HEAT GAIN FROM VENTILATION *****

***** INTERNAL HEAT GAIN *****
***** QCI = HEAT GAIN FROM CEILING *****
***** QFL = HEAT GAIN FROM FLOOR *****
***** QP1 = HEAT GAIN FROM PARTITION 1 *****
***** QP2 = HEAT GAIN FROM PARTITION 2 *****
***** QPS = SENSIBLE HEAT GAIN FROM PEOPLE *****
***** QPL = LATENT HEAT GAIN FROM PEOPLE *****
***** QP = HEAT GAIN FROM PEOPLE *****
***** QLS = HEAT GAIN FROM LIGHT *****
***** QES = SENSIBLE HEAT GAIN FROM EQUIPMENT *****
***** QEL = LATENT HEAT GAIN FROM EQUIPMENT *****
***** QIS = TOTAL SENSIBLE INTERNAL HEAT GAIN *****
***** QIL = TOTAL LATENT INTERNAL HEAT GAIN *****
***** QIT = TOTAL INTERNAL HEAT GAIN *****
***** QW = HEAT GAIN FROM EXTERNAL WALL *****
***** QT = TOTAL HEAT GAIN IN DIRECTION *****
***** QET = TOTAL EXTERNAL HEAT GAIN *****
***** QTT = SUM OF TOTAL HEAT GAIN ALL DIRECTION *****

***** MAXIMIUN HEAT GAIN IN DAY *****
***** IDM = TIME OF APPEARANCE *****
***** QEDM = EXTERNAL HEAT GAIN *****
***** QVDM = HEAT GAIN FROM VENTILATION *****
***** QIDM = INTERNAL HEAT GAIN *****
***** QDM = TOTAL HEAT GAIN *****

***** MAXIMIUN HEAT GAIN IN YEAR *****
***** QWM = HEAT GAIN FROM EXTERNAL WALL *****
***** QGCM = HEAT GAIN BY CONVECTION FROM GLASS *****
***** QGRM = HEAT GAIN BY RADIATION FROM GLASS *****
***** QVTM = HEAT GAIN FROM VENTILATION *****
***** QITM = INTERNAL HEAT GAIN *****
***** QMAX = MAXIMUM COOLING LOAD IN A YEAR *****
***** QMIN = MINIMUM COOLING LOAD IN A YEAR *****
***** MMAX = APPEARANCE MONTH OF MAXIMUM COOLING LOAD *****
***** MMIN = APPEARANCE MONTH OF MINIMUM COOLING LOAD *****
***** TMAX = APPEARANCE TIME OF MAXIMUM COOLING LOAD *****
***** TMIN = APPEARANCE TIME OF MINIMUM COOLING LOAD *****
```

```
*****    DMAX = APPEARANCE DAY OF MAXIMUM COOLING LOAD  *****
*****    DMIN = APPEARANCE DAY OF MINIMUM COOLING LOAD  *****
*************************************************************************/

   #include<stdio.h>
   #include<math.h>
   float PI=3.14159;
   main ()
   {
/*****    SET VARIABLE AND CONSTANT VALUE    *****/
   char *MONTH[13],*DIRECT[10];
   char IN[30];
   int I,J,K,M,L,N,DAY,NDAY,IR,IDM;
   int MMAX,TMAX,MMIN,TMIN,DMAX,DMIN;
   int DAYMX[13];
   float AA[400],BB[400],CC[400],E[400],D[400];
   float T,TT,B,P,PP,SS,Z,Y,IDN,IDS,IDG,IDD,IT;
   float AI[6]={0.01154,0.77674,-3.94657,8.57881,-8.38135,3.01188};
   float TI[6]={-0.00885,2.71235,-0.62062,-7.07329,9.75995,-3.89922};
   float A1,A2,T1,T2,ZA1,ZA2,ZT1,ZT2,TC,AC;
   float LRF,ZB,ZD,ZC;
   float TD,RHD;
   float QDM,QEDM,QVDM,QIDM,QSM,QLM;
   float TOM,RHM,QAV,QMAX,QMIN,QVTM,QITM,QWM[10],QGCM[10],QGRM[10];
   float SHGF[25][10],TO[25],RH[25],ID[25],IDR[25],TE[25][10],TET[10],TEA[10];
   float QWT[25],QCT[25],QRT[25],QTT[25][35],QET[25],QT[25][10],Q[25][10],QTC[24][360];
   float QW[25][35],QGC[25][10],QGR[25][10],ZQ[10],ZQAV[10],QST[25][35],QLT[25][35];
   float TK[25],QVS[25],QVL[25],QVT[25];
   float PWS,PWS1,PW,WS,W,U,V,H,VR,TKD,PWSD,PWS1D,PWD,WD;
   float CO[6]={-5.8002206E+3,-5.5162560,-4.8640239E-2,4.1764768E-5,-1.4452093
         E- 8,6.5459673};
   float UCI,UFL,UP1,UP2,ACI,AFL,AP1,AP2,TCI,TFL,TP1,TP2,NOP,SHF,LHF,LD,A;
   float QCI,QFL,QP1,QP2,QPS,QPL,QP,QLS,QES,QEL,LCT,AFR;
   float QIS[25],QIL[25],QIT[25];
   float AW[10],AG[10],UG[10],UW[10],SC[10],BF[10][10],DF[10][10],CF[10];
   float CCC,ZZ,INTER;
   float C(float T);
   float S(float T);

   FILE *fp1,*fp2,*fp3,*fp4,*fp5;
   FILE *fp6,*fp7,*fp8,*fp9,*fp10,*fp0;
   FILE *fp11,*fp12,*fp13;
   clrscr();
```

```
/*****   OPEN FILE TO READ AND WRITE DATA    *****/

/*--  OPEN FILE TO READ INPUT DATA  --*/
  if((fp1=fopen("IN03.DAT","r"))==NULL) {
   printf("CANNOT OPEN FILE 1 \n");
   exit(0);              }

/*--  OPEN FILE TO READ SOURCE DATA  --*/
  if((fp2=fopen("DATA.DAT","r"))==NULL) {
   printf("CANNOT OPEN FILE 2 \n");
   exit(0);                  }

/*--  OPEN FILE TO READ OUTSIDE TEMPERATURE  --*/
  if((fp3=fopen("TEMP.DAT","r"))==NULL) {
   printf("CANNOT OPEN FILE 3 \n");
   exit(0);                  }

/*--  OPEN FILE TO READ RELATIVE HUMIDITY  --*/
  if((fp4=fopen("RH.DAT","r"))==NULL) {
   printf("CANNOT OPEN FILE 4 \n");
   exit(0);                  }

/*--  OPEN FILE TO WRITE SOLAR IRRADIANCE  --*/
  if((fp5=fopen("I.DAT","r"))==NULL) {
   printf("CANNOT OPEN FILE 5 \n");
   exit(0);                  }

/*--  OPEN FILE TO WRITE SOLAR HEAT GAIN FACTOR  --*/
  if((fp6=fopen("IT.TXT","w"))==NULL) {
   printf("CANNOT OPEN FILE 8 \n");
   exit(0);                  }

/*--  OPEN FILE TO WRITE SOL-TEMPERATURE  --*/
  if((fp7=fopen("SHGF.TXT","w"))==NULL) {
   printf("CANNOT OPEN FILE 9 \n");
   exit(0);              }

/*--  OPEN FILE TO WRITE SOL-TEMPERATURE  --*/
  if((fp8=fopen("Q.TXT","w"))==NULL) {
   printf("CANNOT OPEN FILE 9 \n");
   exit(0);              }
```

```
/*—  OPEN FILE WRITE TOTAL HEAT GAIN  —*/
   if((fp11=fopen("QTT3.TXT","w"))==NULL) {
    printf("CANNOT OPEN FILE 11 \n");
    exit(0);                    }

/*—  OPEN FILE TO WRITE MAXIMIUM COOLING LOAD  —*/
   if((fp13=fopen("OUT3.TXT","w"))==NULL) {
    printf("CANNOT OPEN FILE 13 \n");
    exit(0);                    }

/*****    READ INPUT DATA    *****/

/*—  READ CONDUCTION TRANSFER COEFFICIENT  —*/
   fscanf(fp1,"%s",IN);
   for(N=1;N<=9;N++) {
   fscanf(fp1,"%s",IN);
   fscanf(fp1,"%s",IN);
    for(K=0;K<=6;K++) {
       fscanf(fp1,"%f",&BF[N][K]);
            }
   fscanf(fp1,"%s",IN);
    for(K=0;K<=6;K++) {
       fscanf(fp1,"%f",&DF[N][K]);
            }
   fscanf(fp1,"%s %f",IN,&CF[N]);

 /*—  READ EXTERNAL WALL AREA,GLASS AREA,U-FACTOR AND SC OF GLASS  —*/
    fscanf(fp1,"%s %f",IN,&AW[N]);
    fscanf(fp1,"%s %f",IN,&UW[N]);
    fscanf(fp1,"%s %f",IN,&AG[N]);
    fscanf(fp1,"%s %f",IN,&UG[N]);
    fscanf(fp1,"%s %f",IN,&SC[N]);
            }
 /*—  READ TEMPERATURE AND RH DESIGN CONDITION  —*/
    fscanf(fp1,"%s %f",IN,&TD);
    fscanf(fp1,"%s %f",IN,&RHD);

 /*—  READ U-FACTOR,AREA AND TEMPERATURE OF INTERNAL CEILING,FLOOR,PARTITION
—*/
    fscanf(fp1,"%s",IN);
```

```
fscanf(fp1,"%s %f",IN,&UCI);
fscanf(fp1,"%s %f",IN,&ACI);
fscanf(fp1,"%s %f",IN,&TCI);
fscanf(fp1,"%s %f",IN,&UFL);
fscanf(fp1,"%s %f",IN,&AFL);
fscanf(fp1,"%s %f",IN,&TFL);
fscanf(fp1,"%s %f",IN,&UP1);
fscanf(fp1,"%s %f",IN,&AP1);
fscanf(fp1,"%s %f",IN,&TP1);
fscanf(fp1,"%s %f",IN,&UP2);
fscanf(fp1,"%s %f",IN,&AP2);
fscanf(fp1,"%s %f",IN,&TP2);
fscanf(fp1,"%s %f",IN,&NOP);
fscanf(fp1,"%s %f",IN,&SHF);
fscanf(fp1,"%s %f",IN,&LHF);
fscanf(fp1,"%s %f",IN,&LD);
fscanf(fp1,"%s %f",IN,&A);
fscanf(fp1,"%s %f",IN,&VR);
fscanf(fp1,"%s %f",IN,&QES);
fscanf(fp1,"%s %f",IN,&QEL);
fscanf(fp1,"%s %f",IN,&INTER);
fprintf(fp11,"\n %f \n",INTER);

/*****    SET INITAIL VALUE OF VARIABLE    *****/
  QMAX=0.;
  QSM=0;
  QLM=0;
  QMIN=1.0E+10;
  TMAX=TMIN=0;
  MMAX=MMIN=0;
  DMAX=DMIN=0;
  NDAY=0;
  DAYMX[1]=DAYMX[3]=DAYMX[5]=DAYMX[7]=DAYMX[8]=DAYMX[10]=DAYMX[12]=31;
  DAYMX[4]=DAYMX[6]=DAYMX[9]=DAYMX[11]=30;
  DAYMX[2]=28;
  fscanf(fp1,"%s %d",IN,&IR);

/*****    CALCULATE SOLAR IRRADIANCE    *****/

  /*- START MONTH -*/
  for(M=1;M<=12;M++) {

   switch(M) {
```

```
case 1:
  MONTH[1]="JANUARY";
  break;
case 2:
  MONTH[2]="FEBRUARY";
  break;
case 3:
  MONTH[3]="MARCH";
  break;
case 4:
  MONTH[4]="APRIL";
  break;
case 5:
  MONTH[5]="MAY";
  break;
case 6:
  MONTH[6]="JUNE";
  break;
case 7:
  MONTH[7]="JULY";
  break;
case 8:
  MONTH[8]="AUGUST";
  break;
case 9:
  MONTH[9]="SEPTEMBER";
  break;
case 10:
  MONTH[10]="OCTOBER";
  break;
case 11:
  MONTH[11]="NOVEMBER";
  break;
case 12:
  MONTH[12]="DECEMBER";
  break;
 }

/*---  WRITE MONTH IN OUTPUT FILE  ---*/

  /*fprintf(fp7,"\n Month   = \t %s \n ",MONTH[M]);*/
  fprintf(fp6,"\n Month   = \t %s \n ",MONTH[M]);
  fprintf(fp7,"\n \n Month   = \t %s \n ",MONTH[M]);
```

```
fprintf(fp8,"\n Month  = \t %s \n \n",MONTH[M]);
/*  fprintf(fp10,"\n Month  = \t %s \n ",MONTH[M]);
/*  fprintf(fp11,"\n Month  = \t %s  \n ",MONTH[M]);*/

fprintf(fp8,"TIME    EXTERNAL     VENTILATION    INTERNAL     TOTAL  \n");
fprintf(fp8,"(HR)    HEAT GAIN.   HEAT GAIN.    HEAT GAIN    HEAT GAIN. \n.\n");

/*- START DAY -*/
for(DAY=1;DAY<=DAYMX[M];DAY++) {
/*  fprintf(fp7,"\n DAY  = \t %d \n \n",DAY);*/
fprintf(fp6,"\n DAY  = \t %d \n \n",DAY);
fprintf(fp7,"\n \n DAY  = \t %d \n \n",DAY);
fprintf(fp8,"\n DAY  = \t %s \n \n",DAY);
/* fprintf(fp11,"\n DAY  = \t %d \n \n",DAY);*/
/*   fprintf(fp10,"\n DAY  = \t %d  \n \n",DAY);*/

QDM=QEDM=QVDM=QIDM=0;

/*--  READ AA,BB,CC,E AND D FOR CALCULATION  --*/
fscanf(fp2,"%f",&AA[DAY]);
fscanf(fp2,"%f",&BB[DAY]);
fscanf(fp2,"%f",&CC[DAY]);
fscanf(fp2,"%f",&E[DAY]);
NDAY=NDAY+1;
D[DAY]=23.45*S((284+NDAY)/365);

/*--  SET INITIAL TET AT ALL DIRECTION = 0 AT ALL MONTH  --*/
for(N=1;N<=9;N++) {
  TET[N]=0;
      }

/*--  WRITE DIRECTION IN OUTPUT FILE  --*/
/*  fprintf(fp7,"TIME   S    SW   W    NW   N    NE   E    SE   HORZ \n \n");*/
fprintf(fp6,"TIME   S    SW   W    NW   N    NE   E    SE   HORZ \n \n");
fprintf(fp7,"TIME   S    SW   W    NW   N    NE   E    SE   HORZ \n \n");

/*- START HOUR -*/
for(I=1;I<=24;I++) {
  IDN=IDS=IDG=0.;

/*--  FIND SOLAR TIME  --*/
  T=I*60-20+E[DAY]/60;
  TT=(T+20-E[DAY]/60)/60;
```

```
/*--- WRITE TIME IN OUTPUT FILE ---*/
      fprintf(fp6,"%.2f \t",TT);
      fprintf(fp7,"%.2f \t",TT);
      fprintf(fp8,"%.2f \t",TT);

/*--- READ OUTSIDE TEMPERATURE AND RELATIVE HUMIDITY ---*/
      fscanf(fp3,"%f",&TO[I]);
      fscanf(fp4,"%f",&RH[I]);

/*--- SET CONDITION TO USE REAL OR CALCULATE SOLAR IRRADIENCE ---*/

      if(IR==2) fscanf(fp5,"%f",&IDR[I]);

      /*- START DIRECTION -*/
      for(J=180;J>=-180;J-=45) {
       N=(180-J)/45+1;

/*--- FIND SOLAR ATITUDE ---*/
       B=asin(C(13.73)*C(D[DAY])*C(0.25*(720.-T))+S(13.73)*S(D[DAY]));
       if(B<0.) B=0.;

/*--- FIND SOLAR AZIMUTHE ---*/
       P=acos((sin(B)*S(13.73)-S(D[DAY]))/(cos(B)*C(13.73)));
       if(T<720.) P=-P;
         PP=P*180/PI;

/*--- FIND INCIDENT ANGLE ---*/
       SS=PP-J;
       if(((SS>90)&&(SS<270))||((SS<-90)&&(SS>-270))) SS=90;
          ZZ=acos(sin(B));
          if(J==-180)
          Z=acos(sin(B));
          else
          Z=acos(cos(B)*C(SS));

/*--- FIND DIRECT SOLAR IRRADIANCE ---*/
       if(BI=0.) IDN=AA[DAY]*exp(-BB[DAY]/sin(B));
       if(IR==2){
          ID[I]=IDR[I];
          CCC=cos(ZZ);
          IDN=ID[I]/CCC;
          if(((I>16)||(I<9)))
```

```
            if(BI=0)
            IDN=AA[DAY]*exp(-BB[DAY]/sin(B));
            else
            IDN=0;
          /*  printf("%f \t %f \t %f \n",ID[I],CCC,IDN);*/
            }
        ID[I]=IDN*cos(Z);
        /*printf("\t %f \t %f \t %f \n",ID[I],CCC,IDN);   */


/*—  FIND DIFFUSE SOLAR IRRADIANCE  —*/
        if(cos(Z)>-0.2)
          Y=0.55+0.437*cos(Z)+0.313*pow(cos(Z),2);
        else
          Y=0.45;
        IDS=CC[DAY]*Y*IDN;
        IDG=IDN*(CC[DAY]+sin(B))*0.1;

/*—  FIND TOTAL SOLAR IRRADIANCE  —*/
        if(J==-180)
          IDD=0;
        else
          IDD=IDS+IDG;
        IT=ID[I]+IDD;

/*****   FIND SOLAR HEAT GAIN FACTOR   *****/
        A1=A2=T1=T2=ZA1=ZA2=ZT1=ZT2=TC=AC=0;
        SHGF[I][N]=0;
        for(K=0;K<=5;K++) {
          T1=TI[K]*pow(cos(Z),K);
          T2=TI[K]/(K+2);
          A1=AI[K]*pow(cos(Z),K);
          A2=AI[K]/(K+2);
          ZT1=ZT1+T1;
          ZT2=ZT2+T2;
          ZA1=ZA1+A1;
          ZA2=ZA2+A2;
              }
        TC=ID[I]*ZT1+2*IDD*ZT2;
        AC=ID[I]*ZA1+2*IDD*ZA2;
        SHGF[I][N]=TC+0.268*AC;

/*****   FIND HEAT GAIN BY CONVECTION AND RADIATION FROM GLASS   *****/
```

```
        QGC[I][N]=(UG[N]*AG[N]*(TO[I]-TD))/1000;
        QGR[I][N]=AG[N]*SC[N]*SHGF[I][N]/1000;

  /*****   FIND SOL-TEMPARATURE   *****/
        if(J==-180)
          LRF=-3.9;
        else
          LRF=0;
        TE[I][N]=TO[I]+0.026*IT-LRF;
        TET[N]=TET[N]+TE[I][N];
        fprintf(fp6,"%.2f \t",LT);
        fprintf(fp7,"%.2f \t",SHGF[I][N]);
      /* fprintf(fp7,"%.2f \t",TE[I][N]);*/
                }
        /*- END OF DIRECTION -*/

  /******   FIND VENTILATION COOLING LOAD   *****/
        TK[I]=TO[I]+273.15;

PWS1=CO[0]/TK[I]+CO[1]+CO[2]*TK[I]+CO[3]*pow(TK[I],2)+CO[4]*pow(TK[I],3)+CO[5]*log(TK[I]);
        PWS=exp(PWS1);
        PW=PWS*RH[I]/100;
        TKD=TD+273.15;

PWS1D=CO[0]/TKD+CO[1]+CO[2]*TKD+CO[3]*pow(TKD,2)+CO[4]*pow(TKD,3)+CO[5]*log(TKD);
        PWSD=exp(PWS1D);
        PWD=PWSD*RHD/100;
        WS=0.62198*PWS/(101.325-PWS);
        W=0.62198*PW/(101.325-PW);
        WD=0.62198*PWD/(101.325-PWD);
  /*    U=W/WS;
        V=287.055*TO[I]*(1+1.6078*W);
        H=1.006*TO[I]+W*(2501+1.805*TO[I]);*/
        QVS[I]=1.23*VR*(TO[I]-TD);
        if(QVS[I]<0.0) QVS[I]=0;
        QVL[I]=3010*VR*(W-WD);
        QVT[I]=(QVS[I]+QVL[I])/1000;

  /*****   FIND INTERNAL COOLING LOAD   *****/

  /*--  HEAT GAIN BY ROOF FLOOR AND PARTITION  --*/
        if(TCI==0) TCI=TO[I];
        if(TFL==0) TFL=TO[I];
```

```
        if(TP1==0) TP1=TO[I];
        if(TP2==0) TP2=TO[I];
        QCI=UCI*ACI*(TCI-TD)/1000;
        QFL=UFL*AFL*(TFL-TD)/1000;
        QP1=UP1*AP1*(TP1-TD)/1000;
        QP2=UP2*AP2*(TP2-TD)/1000;

/*--- HEAT GAIN BY PEOPLE ---*/
        QPS=NOP*SHF/1000;
        QPL=NOP*LHF/1000;
        QP=(QPS+QPL);

/*--- HEAT GAIN BY ELECTRICAL AND EQUIPMENT ---*/
        QLS=LD*A/1000;

/*--- TOTAL INTERNAL HEAT GAIN ---*/
        QIS[I]=QCI+QFL+QP1+QP2+QPS+QLS+QES;
        QIL[I]=QPL+QEL;
        QIT[I]=(QIS[I]+QIL[I]);

        fprintf(fp7,"\n");
        fprintf(fp6,"\n");
        fprintf(fp7,"\n");

            }
        /*- END OF HOUR -*/

/*****  FIND AVERAGE SOL-TEMPARATURE  *****/
    /*  fprintf(fp7,"\n TEAV  ");*/
    for(N=1;N<=9;N++) {
        TEA[N]=TET[N]/24;
    /*  fprintf(fp7,"%.2f \t",TEA[N]);*/
            }

/*****  FIND HEAT GAIN BY EXTERNAL WALL  *****/
    for(I=1;I<=24;I++) {
        QWT[I]=0;
        QRT[I]=0;
        QCT[I]=0;
        QET[I]=0;
            }

    /*- START DIRECTION 2 -*/
```

```
for(N=1;N<=9;N++) {
   switch(N) {
      case 1:
       DIRECT[1]="SOUTHERN";
       break;
      case 2:
       DIRECT[2]="SOUTH WEST";
       break;
      case 3:
       DIRECT[3]="WEST     ";
       break;
      case 4:
       DIRECT[4]="NORTH WEST";
       break;
      case 5:
       DIRECT[5]="NORTHERN";
       break;
      case 6:
       DIRECT[6]="NORTH EAST";
       break;
      case 7:
       DIRECT[7]="EAST     ";
       break;
      case 8:
       DIRECT[8]="SOUTH EAST";
       break;
      case 9:
       DIRECT[9]="HORIZONTAL";
       break;
        }
   /*  fprintf(fp10," DIRECTION   =   %s \n \n",DIRECT[N]);*/
     L=0;
     ZQAV[N]=0;
     for(I=1;I<=24;I++) {
      Q[I][N]=0;
            }

/*--- CHECK CONDITION TO STOP LOOP TO FIND HEAT GAIN  ---*/
      QAV=UW[N]*(TEA[N]-TD);
      while(fabs(QAV-ZQAV[N])>=0.01&&(L<10)) {
        ZQ[N]=0;

        /*- START HOUR 2 -*/
```

```
        for(I=1;I<=24;I++) {
          ZB=ZD=0;
          for(K=0;K<=6;K++) {
           J=I-K;
           if(J<=0) J=24+I-K;
           ZB=ZB+BF[N][K]*TE[J][N];
      /*     fprintf(fp6,"%d \t %f \t %f \t %f \n",I,ZB,BF[N][K],TE[J][N]);*/
                }
          for(K=1;K<=6;K++) {
           J=I-K;
           if(J<=0) J=24+I-K;
           ZD=ZD+DF[N][K]*Q[J][N];
      /*     fprintf(fp0,"%d \t %f \t %f \t %f \n",I,ZD,DF[N][K],Q[J][N]); */
                }
          ZC=TD*CF[N];
          Q[I][N]=(ZB-ZD-ZC);
          if(DAY==2) fprintf(fp0,"%d \t %f \n",I,Q[I][N]);
          ZQ[N]=ZQ[N]+Q[I][N];
                }
          /*- END OF HOUR 2 -*/

        L++;
        ZQAV[N]=ZQ[N]/24;
                  }
          if(DAY==2){
          fprintf(fp0,"\n %s \n",DIRECT[N]);
          fprintf(fp0,"\n %f \t %f \n",QAV,ZQAV[N]);}

 /*****   FIND EXTERNAL COOLING LOAD   *****/
        for(I=1;I<=24;I++) {
        QW[I][N]=Q[I][N]*AW[N]/1000;
        QT[I][N]=QW[I][N]+QGC[I][N]+QGR[I][N];
        QWT[I]=QWT[I]+QW[I][N];
        QCT[I]=QCT[I]+QGC[I][N];
        QRT[I]=QRT[I]+QGR[I][N];
        QET[I]=QET[I]+QT[I][N];
              }
            }
        /*- END OF DIRERTION 2 -*/

  /*--  WRITE HEADER FOR OUTPUT FILE  --*/
   /* fprintf(fp10,"TIME  EXTERNAL WALL  GLASS CONDUCTION  GLASS RADIATION  TOTAL
EXTERNAL  \n");
```

```
    fprintf(fp10,"(HR)   HEAT GAIN    HEAT GAIN    HEAT GAIN    HEAT GAIN \n\n");

/******   FIND TOTAL,MAXIMUM AND MINIMUM COOLING LOAD   *****/

/*--  FIND TOTAL COOLING LOAD  --*/

   /*- START HOUR 3 -*/

   for(I=1;I<=24;I++) {
      QTT[I][DAY]=(QET[I]+QVT[I]+QLT[I]);
      QLT[I][DAY]=QVL[I]/1000+QIL[I];
      QST[I][DAY]=QTT[I][DAY]-QLT[I][DAY];
      if(QTT[I][DAY]<=0) QTT[I][DAY]=0;
    /* QTC[I][I1]=QTT[I][DAY];*/

        if(QTT[I][DAY]>QDM) {
          IDM=I;
          QDM=QTT[I][DAY];
          QEDM=QET[I];
          QVDM=QVT[I];
          QIDM=QIT[I];
              }
/*--  FIND MAXIMUM OF COOLING LOAD  --*/
        if(QTT[I][DAY]>QMAX) {
          QMAX=QTT[I][DAY];
          QSM=QST[I][DAY];
          QLM=QLT[I][DAY];
          MMAX=M;
          TMAX=I;
          DMAX=DAY;
          TOM=TO[I];
          RHM=RH[I];
    for(N=1;N<=9;N++) {
     QWM[N]=QW[I][N];
     QGCM[N]=QGC[I][N];
     QGRM[N]=QGR[I][N];
         }
     QVTM=QVT[I];
     QITM=QIT[I];
               }

/*--  FIND MINIMUM OF COOLING LOAD  --*/
```

```
        if(QTT[I][DAY]<QMIN) {
          QMIN=QTT[I][DAY];
          MMIN=M;
          TMIN=I;
          DMIN=DAY;
                }

   /*       fprintf(fp10,"%d \t %f \t %f \t %f \t %f",I,QWT[I],QCT[I],QRT[I],QET[I]);
         fprintf(fp10,"\n");*/
          /* QTC[I][II]=QTT[I][DAY];  */
         fprintf(fp11,"%7.2f \n",QTT[I][DAY]);
         fprintf(fp8,"%d \t %7.2f \t %7.2f \t %7.2f \t %7.2f \n",I,QET[I],QVT[I],QIT[I],QTT[I][DAY]);

                }
         /*- END OF HOUR 3 -*/
                }
         /*- END OF DAY -*/
                }
       /*- END OF MONTH -*/

   /*—  WRITE MAX,MIN COOLING LOAD TO OUTPUT FILE  —*/
       fprintf(fp11,"\n %f",QMAX);
       fprintf(fp13,"DESIGN TEMP  =  %4.2f  DESIGN RH  =  %4.2f \n\n",TD,RHD);
       fprintf(fp13,"OUTSIDE TEMP =  %4.2f  OUTSIDE RH =  %4.2f \n\n",TOM,RHM);
       fprintf(fp13,"EXTERNAL WALL HEAT GAIN \n \n");
       fprintf(fp13,"DIRECTION        AREA(M^2)    HEAT GAIN(KW) \n \n");
       for(N=1;N<=9;N++) {
        fprintf(fp13,"%s \t %7.0f \t %7.2f \n",DIRECT[N],AW[N],QWM[N]);
            }
       fprintf(fp13,"\n EXTERNAL GLASS HEAT GAIN \n \n");
       fprintf(fp13,"DIRECTION      SC    AREA(M^2)   RAD. HEAT(KW)   CON. HEAT(KW) \n \n");
       for(N=1;N<=9;N++) {
        fprintf(fp13,"%s \t %4.2f \t %4.0f \t \t %6.2f \t %5.2f
\n",DIRECT[N],SC[N],AG[N],QGRM[N],QGCM[N]);
            }
       fprintf(fp13,"\n VENTILATION FLOW RATE (L/S) = \t %8.0f \n",VR);
       fprintf(fp13," VENTILATION HEAT GAIN (KW) = \t %8.2f \n",QVTM);
       fprintf(fp13,"\n INTERNAL HEAT GAIN \n \n \t     U(KW/M^2-C)  AREA(M^2)  OUT-TEMP(C)
HEAT GAIN(KW)");
       fprintf(fp13,"\n CEILING \t %3.2f \t %8.2f \t %3.1f \t %8.2f",UCI,ACI,TCI,QCI);
       fprintf(fp13,"\n FLOOR   \t %3.2f \t %8.2f \t %3.1f \t %8.2f",UFL,AFL,TFL,QFL);
       fprintf(fp13,"\n PARTITION 1 \t %3.2f \t %8.2f \t %3.1f \t %8.2f",UP1,AP1,TP1,QP1);
       fprintf(fp13,"\n PARTITION 2 \t %3.2f \t %8.2f \t %3.1f \t %8.2f \n",UP2,AP2,TP2,QP2);
```

```
    fprintf(fp13,"\n NUMBER OF PEOPLE = \t %5.0f",NOP);
    fprintf(fp13,"\n SENSIBLE HEAT GAIN (W/PERSON) = %5.2f LATENT HEAT GAIN
(W/PERSON) = %5.2f",SHF,LHF);
    fprintf(fp13,"\n TOTAL HEAT GAIN FROM PEOPLE (KW) = %5.2f\n",QP);
    fprintf(fp13,"\n LIGHT DENSITY (W/M^2) = %5.2f \t AREA(M^2) = %8.0f ",LD,A);
    fprintf(fp13,"\n HEAT GAIN FROM LIGHT (KW) =%5.2f.\n",QLS);
    fprintf(fp13,"\n TOTAL INTERNAL HEAT GAIN (KW) = \t %f \n",QITM);
    fprintf(fp13,"\n TOTAL SENSIBLE HEAT GAIN (KW) = \t %f \n",QSM);
    fprintf(fp13,"\n TOTAL LATENT HEAT GAIN (KW) = \t %f \n",QLM);
    fprintf(fp13,"\n MONTH = %s \t DAY = %d \t TIME = %d \t MAXIMIUM LOAD = %f
\n",MONTH[MMAX],DMAX,TMAX,QMAX);
    fprintf(fp13,"\n MONTH = %s \t DAY = %d \tTIME = %d \t MINIMIUM LOAD = %f
\n",MONTH[MMIN],DMIN,TMIN,QMIN);

    fscanf(fp1,"%s %f",IN,&LCT);
    AFR=QSM/1.23/(TD-LCT);
    fprintf(fp13,"\n LEAVING COIL TEMPERATURE(C) \t %f",LCT);
    fprintf(fp13,"\n AIR FLOW RATE \t %f \t M^3/S",AFR);

  /***** FIND POWER USED OF CHILLER AND NUMBER OF CHILLER OPERATE *****/

    /*— CLOSE WIITING FILE —*/
    fclose(fp6);
    fclose(fp7);
    fclose(fp7);
    fclose(fp10);
    fclose(fp11);
    fclose(fp12);
    fclose(fp13);
    }

  /***** FUNCTION TO FIND SIN(Z) IN DEGREE *****/
    float S(float N) {
      N=sin(PI/180*N);
      return(N);
        }

  /***** FUNCTION TO FIND COSIN(Z) IN DEGREE *****/
    float C(float N) {
      N=cos(PI/180*N);
      return(N);
            }
```

¶ Source Code โปรแกรม COST.C

```
/*****************************************************************************
*** COOLING LOAD CALCULATION BY TRANSFER FUNCTION METHOD ***
*****************************************************************************
*************************************************************
*****        INPUT AND OUTPUT FILES IN PROGRAM        *****
*************************************************************

*****  INPUT FILE                    *****
*****   ELEC.DAT = ELECTRICAL PAYMENT RATE          *****
*****   COST.DAT = PRICE OF CHILLER                 *****

*****  OUTPUT FILE                   *****
*****   ESTIMATE.TXT  = PAYMENT IN A YEAR                 *****
*****   POW.TXT  = POWER USED AND ELECTRICAL PAYMENT      *****
*****   COST.TXT  = OUTPUT OF CALCUTION                   *****
*************************************************************

*************************************************************
*****        DEFINITION OF SYMBOL             *****
*************************************************************
*****   J   = INTERATION NUMBER                 *****
*****   K   = INTERATION NUMBER                 *****
*****   L   = INTERATION NUMBER                 *****
*****   I   = INTERATION NUMBER OF TIME         *****
*****   M   = INTERATION NUMBER OF MONTH        *****
*****   N   = INTERATION NUMBER OF DIRECTION    *****

*****   QMAX = MAXIMUM COOLING LOAD IN A YEAR   *****
*****   CH1  = NUMBER OF FULL LOAD CHILLER      *****
*****   CH   = NUMBER OF CHILLER                *****
*****   CHM  = NUMBER OF CHILLER USED IN MONTH  *****
*****   CHU  = NUMBER OF CHILLER USED           *****
*****   CON  = CHILLER CONTROL METHODS          *****
*****    1   BY FULL-LOAD RUNNING               *****
*****    2   BY AVERAGE LOAD RUNNING            *****
*****   LOAD = LOAD STEP OF CHILLER             *****
*****   PCC  = CHILLER SIZE TO CALCULATE        *****
*****   POWC = POWER CONSUMPTION OF CHILLER     *****
*****   PC   = POWER CONSUMPTION OF CHILLER     *****
*****   PLOAD = PART LOAD OF CHILLER            *****
*****   ALOAD = AVERAGE LOAD OF CHILLER         *****
```

```
*****   POW1 = POWER USED OF FULL LOAD CHILLER          *****
*****   POW2 = POWER USED OF PATAIL LOAD CHILLER        *****
*****   POW  = TOTAL POWER USED OF CHILLER              *****
*****   POWT = TOTAL POWER USED OF CHILLER IN A YEAR    *****
*****   DC   = DEMAND CHARGE                            *****
*****   EPR1 = ELECTRICAL POWER UNIT RATING             *****
*****        DURING 9:00 AM - 10:00 PM                  *****
*****   EPR2 = ELECTRICAL POWER UNIT RATING             *****
*****        BEFORE 9:00 AM AND AFTER 10:00 PM          *****
*****   EPR  = ELECTRICAL POWER UNIT RATING             *****
*****   EP1  = DEMAND CHARGE PAYMENT                    *****
*****   EP2  = ELECTRICAL POWER RATING PAYMENT          *****
*****   TEP  = TOTAL ELECTRICAL POWER PAYMENT           *****
*****   T_ON = TIME TO TURN ON CHILLER                  *****
*****   T_OFF= TIME TO TURN OFF CHILLER                 *****
*************************************************************/
  #include<stdio.h>
  #include<math.h>
  float PI=3.14159;
  main ()
  {

/*****   SET VARIABLE AND CONSTANT VALUE    *****/
  char *MONTH[13],*DIRECT[10];
  char IN[30];
  int I,J,K,M,L,N,DAY,NDAY,YDAY,JR,JDM,JI,KK;
  int MMAX,TMAX,MMIN,TMIN,DMAX,DMIN,CH,CH1,CHM,CHU[10],T_ON,T_OFF,CON;
  int DAYMX[13],ZDAY[13],YEAR[10],MYEAR[10];
  float QMAX,LMAX,QTT[25][366],CHP[10];
  float LOAD[20][10],POWC[20][10],POW[25],POW1[25],POW2[25],PC[25],PRICE[10];
  float PCC,PLOAD,ELOAD,POWT,DC,EPR1,EPR2,EPR,EP1,EP2,TEP[10],CCC,ZZ,INTER;
  float PVT[10][110],PE[10][110],COST[10];

  FILE *fp1,*fp2,*fp3,*fp4,*fp5;
  FILE *fp6,*fp7,*fp8,*fp9,*fp10,*fp0;
  FILE *fp11,*fp12,*fp13;
  clrscr();
/*****   OPEN FILE TO READ AND WRITE DATA    *****/

/*--  OPEN FILE TO READ ELECTRICAL PAYMENT  --*/
  if((fp6=fopen("ELEC.DAT","r"))==NULL) {
   printf("CANNOT OPEN FILE 6.\n");
   exit(0);                    }
```

```
/*—  OPEN FILE TO READ POWER CONSUMPTION OF CHILLER  —*/
  if((fp7=fopen("PCA.DAT","r"))==NULL) {
   printf("CANNOT OPEN FILE 7 \n");
   exit(0);                    }

/*—  OPEN FILE TO WRITE HEAT GAIN  —*/
  if((fp10=fopen("COST.DAT","w"))==NULL) {
   printf("CANNOT OPEN FILE 10 \n");
   exit(0);                    }

/*—  OPEN FILE WRITE TOTAL HEAT GAIN  —*/
  if((fp11=fopen("QTT2.TXT","r"))==NULL) {
   printf("CANNOT OPEN FILE 11 \n");
   exit(0);                    }

/*—  OPEN FILE TO WRITE POWER USED OF CHILLER  —*/
  if((fp12=fopen("POW.TXT","w"))==NULL) {
   printf("CANNOT OPEN FILE 12 \n");
   exit(0);                    }

/*—  OPEN FILE TO WRITE MAXIMIUM COOLING LOAD  —*/
  if((fp13=fopen("OUT.TXT","w"))==NULL) {
   printf("CANNOT OPEN FILE 13 \n");
   exit(0);                    }

/*—  OPEN FILE TO READ INPUT DATA  —*/
  if((fp3=fopen("COST.TXT","w"))==NULL) {
   printf("CANNOT OPEN FILE 3 \n");
   exit(0);                    }

/*—  OPEN FILE TO READ INPUT DATA  —*/
  if((fp5=fopen("EST.TXT","w"))==NULL) {
   printf("CANNOT OPEN FILE 5 \n");
   exit(0);                    }

/*****   READ INPUT DATA   *****/

  for(N=1;N<=7;N++) {
  for(I=2;I<=10;I++) {
   fscanf(fp7,"%f",&LOAD[I][N]);
   fscanf(fp7,"%f",&POWC[I][N]);
   printf("\n %f \t %f ",LOAD[I][N],POWC[I][N]);
```

```
            }
            }

    for(N=1;N<=7;N++) {
    fscanf(fp7,"%f",&PRICE[N]);
            }

    fscanf(fp6,"%s %d %s %d",IN,&T_ON,IN,&T_OFF);
    fscanf(fp6,"%s %f %s %f %s %f %s %d %s %d",IN,&DC,IN,&EPR1,IN,&IR,IN,&CON);
/*  fscanf(fp6,"%s %d",IN,&IR);
    fscanf(fp6,"%s %d",IN,&CON);*/
    fscanf(fp11,"%f",&INTER);
    fprintf(fp10,"%f",INTER);
    for(N=1;N<=365;N++) {
    for(I=1;I<=24;I++) {
    fscanf(fp11,"%f \n",&QTT[I][N]);
/*  fprintf(fp12,"%f \n",QTT[I][N]); */
             }
            }
    fscanf(fp11,"%f",&QMAX);

    DAYMX[1]=DAYMX[3]=DAYMX[5]=DAYMX[7]=DAYMX[8]=DAYMX[10]=DAYMX[12]=31;
    DAYMX[4]=DAYMX[6]=DAYMX[9]=DAYMX[11]=30;
    DAYMX[2]=28;
    ZDAY[1]=0;
    ZDAY[2]=31;
    ZDAY[3]=59;
    ZDAY[4]=90;
    ZDAY[5]=120;
    ZDAY[6]=151;
    ZDAY[7]=181;
    ZDAY[8]=212;
    ZDAY[9]=243;
    ZDAY[10]=273;
    ZDAY[11]=304;
    ZDAY[12]=334;

/*****   CALCULATE SOLAR IRRADIANCE    *****/

    /*- START MONTH -*/
    for(M=1;M<=12;M++) {

     switch(M) {
```

```
        case 1:
          MONTH[1]="JANUARY";
          break;
        case 2:
          MONTH[2]="FEBRUARY";
          break;
        case 3:
          MONTH[3]="MARCH";
          break;
        case 4:
          MONTH[4]="APRIL";
          break;
        case 5:
          MONTH[5]="MAY";
          break;
        case 6:
          MONTH[6]="JUNE";
          break;
        case 7:
          MONTH[7]="JULY";
          break;
        case 8:
          MONTH[8]="AUGUST";
          break;
        case 9:
          MONTH[9]="SEPTEMBER";
          break;
        case 10:
          MONTH[10]="OCTOBER";
          break;
        case 11:
          MONTH[11]="NOVEMBER";
          break;
        case 12:
          MONTH[12]="DECEMBER";
          break;
        }
          }

/*****   FIND POWER USED OF CHILLER AND NUMBER OF CHILLER OPERATE   *****/
    K=0;
    LMAX=QMAX/12/.293;
    fprintf(fp3,"\n MAXIMUN LOAD  =  %4.0f  TONS\n", LMAX);
```

```
fprintf(fp3,"\n CHILLER SIZE \t NO. \t POWER USED \t ELECTRICAL \t CHILLER COST");
fprintf(fp3,"\n   (TONS)    \t\t  (kW) \t PAYMENT(BATH) \t  (BATH)\n");

for(N=1;N<=7;N++) {
CHU[N]=QMAX/LOAD[10][N]+1;
POWT=0;
TEP[N]=0;
YDAY=0;
J=N-1;
if(CHU[N]!=1) {
if(CHU[J]!=CHU[N]) {
K=K+1;
for(M=1;M<=12;M++) {
CHM=0;
fprintf(fp12,"\n Month  = \t %s \n ",MONTH[M]);
for(DAY=1;DAY<=DAYMX[M];DAY++) {
YDAY=YDAY+1;
fprintf(fp12,"\n DAY  = \t %d \n \n",DAY);
fprintf(fp12,"\n TIME  HEAT GAIN    USED POWER  ELECT PAYMENT  NO. OF \n");
fprintf(fp12,"(HR)   (WATT)      (WATT)    (BATH)   CHILLER \n \n");
for(I=1;I<=24;I++) {
fprintf(fp12,"\n %d \t %d ",I,YDAY);

/*--- FIND NUMBER OF CHILLER ---*/
    CH1=QTT[I][YDAY]/LOAD[10][N];
    PLOAD=QTT[I][YDAY]-CH1*LOAD[10][N];
    CH=CH1+1;
    ELOAD=QTT[I][YDAY]/CH;

/*--- SET CONTROL SYSTEM OF CHILLER ---*/
    CON=1;
    if(CON==1) {
      PLOAD=ELOAD;
      }

/*--- DETECT CHILLER USED NOT LESS THAN 20 % OF FULL-LOAD ---*/
    else  {
    if(PLOAD<=LOAD[2][N]) {
      POW1[I]=(CH1-1)*LOAD[10][N]*POWC[10][N]+LOAD[2][N]*POWC[2][N];
      PLOAD=LOAD[10][N]+PLOAD-LOAD[2][N];
          }
    else
      POW1[I]=CH1*LOAD[10][N]*POWC[10][N];
```

```
        }

/*—  FIND POWER USED  —*/

      if(PLOAD<=LOAD[2][N])
        POW2[I]=POWC[2][N]*PLOAD;
      else if(PLOAD<=LOAD[3][N])
         POW2[I]=POWC[3][N]*PLOAD;
       else if(PLOAD<=LOAD[4][N])
          POW2[I]=POWC[4][N]*PLOAD;
         else if(PLOAD<=LOAD[5][N])
            POW2[I]=POWC[5][N]*PLOAD;
           else if(PLOAD<=LOAD[6][N])
              POW2[I]=POWC[6][N]*PLOAD;
             else if(PLOAD<=LOAD[7][N])
                POW2[I]=POWC[7][N]*PLOAD;
               else if(PLOAD<=LOAD[8][N])
                  POW2[I]=POWC[8][N]*PLOAD;
                 else if(PLOAD<=LOAD[9][N])
                    POW2[I]=POWC[9][N]*PLOAD;
                   else
                    POW2[I]=POWC[10][N]*PLOAD;
      PC[I]=POW2[I]/PLOAD;
      if(CON==1)
        POW[I]=CH*POW2[I];
      else
        POW[I]=POW1[I]+POW2[I];
      POWT=POWT+POW[I];

/*****   FIND ELECTRICAL PAYMENT   *****/

/*—  FIND PAYMENT FROM UNIT RATING  —*/
      if(CH>CHM) CHM=CH;
     /*  if(CH>CHU[N]) CHU[N]=CH;
      /* if(I<9||I>22)
        EPR=EPR2;
      else
        EPR=EPR1;*/
      EPR=EPR1;
      EP2=POW[I]*EPR;
      TEP[N]=TEP[N]+EP2;
      fprintf(fp12,"%d \t %7.2f \t %f \t %7.2f \t %d \n",I,QTT[I][YDAY],POW[I],EP2,CH);
      } /* end of hour */
```

```
        } /* end of day */

  /*—  FIND PAYMENT FROM DEMAND CHARGE  —*/
      EP1=DC*CHM*POWC[10][N]*LOAD[10][N]+850;
      TEP[N]=TEP[N]+EP1;
      fprintf(fp12,"\n DEMAND CHARGE = %f.\n",EP1);
       } /* end of month */

  /*—  WRITE NUMBER OF CHILLER,POWER USED AND ELEC PAYMENT TO OUTPUT FILE  —*/
     LOAD[10][N]=LOAD[10][N]/.293/12;
     CHP[N]=PRICE[N]*CHU[N];
     fprintf(fp3,"\n %5.0f \t\t %d \t %10.2f \t %10.2f \t
%10.2f",LOAD[10][N],CHU[N],POWT,TEP[N],CHP[N]);
       }
       }
       }
     fprintf(fp10," \n \n %d",K);

     L=0;
     for(N=1;N<=7;N++) {
     J=N-1;
     if(CHU[N]!=1) {
     if(CHU[J]!=CHU[N]) {
       L=L+1;
       LOAD[10][L]=LOAD[10][N];
       CHU[L]=CHU[N];
       PRICE[L]=PRICE[N];
       TEP[L]=TEP[N];
       fprintf(fp10,"\n %4.0f \t %d \t %f \t %f",LOAD[10][N],CHU[N],PRICE[N],TEP[N]);
             }
             }
             }

     for(N=1;N<=100;N++) {
     for(I=1;I<=K;I++) {
     PVT[I][N]=0;
             }
            }
     for(N=1;N<=100;N++) {
      for(I=1;I<=K;I++) {
        YEAR[I]=0;
        MYEAR[I]=100;
        PE[I][N]=TEP[I]*(pow(1+INTER,N)-1)/(INTER*pow(1+INTER,N));
```

```
        COST[I]=CHU[I]*PRICE[I];
       PVT[I][N]=COST[I]+ PE[I][N];
              }
            }
   for(N=1;N<=100;N++) {
    fprintf(fp5,"\n YEAR =  %d \n",N);
    fprintf(fp5,"\n LOAD \t CH PRICE \t ELEC COST \t TORAL COST \n");
    for(I=1;I<=K;I++) {
      fprintf(fp5,"\n %f \t %10.2f \t %10.2f \t %10.2f",LOAD[10][I],COST[I],PE[I][N],PVT[I][N]);
    if(PVT[I][N]<PVT[K][N]){
      YEAR[I]=N;
      if(YEAR[I]<=MYEAR[I]) {
        MYEAR[I]=YEAR[I];
              }
/*  fprintf(fp3,"\n YEAR =  %d \n",YEAR[I]);
/*  fprintf(fp3,"\n LOAD \t CH PRICE \t ELEC COST \t TORAL COST \n");
    fprintf(fp3,"\n %f \t %10.2f \t %10.2f \t %10.2f",LOAD[10][I],COST[I],PE[I][N],PVT[I][N]);*/
              }
       else
    YEAR[I]=0;
     /*  fprintf(fp3,"\n NOT CONVERT");*/
          }
          }
     J=K-1;
    fprintf(fp3," \n \n CHILLER SIZE(TONS) \tPAYBACK TIME(YEARS) \n");
    for(I=1;I<=J;I++) {
      if(MYEAR[I]!=100) {
        fprintf(fp3,"\n %5.0f \t \t \t%d",LOAD[10][I],MYEAR[I]);
              }
        else
        fprintf(fp3,"\n %5.0f \t \t \tLONG TIME OR NOT PAYBACK",LOAD[10][I]);
           }
     fprintf(fp3,"\n \n COMPARE WITH \t \t %5.0f \t TONS",LOAD[10][K]);
   /*—  CLOSE WIITING FILE  —*/
   fclose(fp7);
   fclose(fp8);
   fclose(fp9);
   fclose(fp10);
   fclose(fp11);
   fclose(fp12);
   fclose(fp13);
   }
```

ประวัติผู้เขียน

นายเทพฤทธิ์ ทองชุบ เกิดที่ อำเภอ เมือง จังหวัด ปัตตานี เมื่อ วันที่ 1 สิงหาคม พ.ศ. 2516 สำเร็จการศึกษาวิศวกรรมศาสตร์บัณฑิต สาขาวิศวกรรมเครื่องกล คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย ในปีการศึกษา 2536 และเข้าศึกษาต่อ ในหลักสูตรวิศวกรรมศาสตรมหาบัณฑิต สาขาวิศวกรรมเครื่องกล จุฬาลงกรณ์มหาวิทยาลัย ในปีการศึกษา 2537