SMAC0 SIMULATOR

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>

char fname[20];
FILE* fp;

int pc,lc,flag;

long mem[1000];
int reg[4];
int cc[6]={0,0,0,0,0,1};

int address;
long content;
int opcode,r,operand;

void load()
{
 fp=fopen(fname,"r");
 if(fp==NULL)
    printf("\n%s file is not found",fname);
 else
   {
    while(!feof(fp))
           {
            fscanf(fp,"%d %ld\n",&address,&content);

            if(address==-1)
                pc=content;
            else
               {
                lc=address;
                mem[lc]=content;
               }
           }
    fclose(fp);
   }
}
```

```
void print()
{
 int i;
 for(i=pc;i<=lc;i++)
     printf("\n%ld",mem[i]);
}

void accept()
{
 fp=fopen(fname,"w");
 printf("\nWrite Smaco code");

 do
 {
  printf("\nEnter address:");
  scanf("%d",&address);
  printf("\nEnter content:");
  scanf("%ld",&content);

  fprintf(fp,"%d %ld\n",address,content);

  if(address==-1)
       pc=content;
  else
      {
       lc=address;
       mem[lc]=content;
      }
 }while(address!=-1);
 fclose(fp);
}
```

```
void execute()
{
int i;
while(pc)
      {
       opcode=mem[pc]/10000;
       r=(mem[pc]%10000)/1000-1;
       operand=(mem[pc]%10000)%1000;

       switch(opcode)
              {
                case 0 :   pc=-1;
                           break;

                case 1 :   reg[r]=reg[r]+mem[operand];
                           break;

                case 2 :   reg[r]=reg[r]-mem[operand];
                           break;

                case 3 :   reg[r]=reg[r]*mem[operand];
                           break;

                case 4 :   reg[r]=mem[operand];
                           break;

                case 5 :   mem[operand]=reg[r];
                           break;

                case 6 :   if(reg[r]<mem[operand])
                              cc[0]=1;
                           if(reg[r]<=mem[operand])
                              cc[1]=1;
                           if(reg[r]==mem[operand])
                              cc[2]=1;
                           if(reg[r]>mem[operand])
                              cc[3]=1;
                           if(reg[r]>=mem[operand])
                              cc[4]=1;
                           break;
                case 7 :    if(cc[r]==1)
                               pc=operand-1;
                            for(i=0;i<5;i++)
                                cc[i]=0;
                            break;

                case 8 :    reg[r]=reg[r]/mem[operand];
                            break;

                case 9 :    printf("\nEnter value:");
                            scanf("%ld",&mem[operand]);
                            break;

                case 10:    printf("\nValue is %ld",mem[operand]);
                            break;
            }
        if(flag==1)
          {
           printf("\nConditional Register");
           printf("\nLT LE EQ GT GE ANY\n");
           for(i=0;i<6;i++)
               printf("%d  ",cc[i]);

           printf("\nRegisters");
           printf("\nAREG\tBREG\tCREG\tDREG\n");
           for(i=0;i<4;i++)
              printf("%d\t",reg[i]);
           getch();
          }
        pc++;
        }
}
```

```
void main(int argc,char* argv[])
{
 int ch;

 strcpy(fname,argv[1]);

 do
 {
  printf("\n1: Load");
  printf("\n2: Print");
  printf("\n3: Accept");
  printf("\n4: Run");
  printf("\n5: Trace");
  printf("\n6: Quit");

  printf("\nEnter your choice:");
  scanf("%d",&ch);

  switch(ch)
         {
           case 1 :  load();
                     break;
           case 2 :  print();
                     break;
           case 3 :  accept();
                     break;
           case 4 :  execute();
                     break;
           case 5 :  flag=1;
                     execute();
                     break;
          }
 }while(ch!=6);
}
```

USE FOLLOWING CODES FOR SMAC0

OPCODE	MNEMONIC
00	STOP
01	ADD
02	SUB
03	MULT
04	MOVER
05	MOVEM
06	COMP
07	BC
08	DIV
09	READ
10	PRINT

NO	REGISTER
1	AREG
2	BREG
3	CREG
4	DREG

CONDITIONAL-CODE	MNEMONIC
1	LT
2	LE
3	EQ
4	GT
5	GE
6	ANY

```
/*Sum of Two Numbers*/

ADDRESS         INSTRUCTION
100             READ       A
101             MOVER AREG  A
102             READ        B
103             ADD   AREG  B
104             MOVEM AREG  SUM
105             PRINT       SUM
106             STOP
107             A     DC    0
108             B     DC    0
109             SUM   DC    0


ADDRESS         INSTRUCTION
100             090107
101             041107
102             090108
103             011108
104             051109
105             100109
106             000000
107             0
108             0
109             0
 -1             100
```

```
/*Maximum of two numbers*/

ADDRESS         INSTRUCTION
100             READ        A
101             MOVER AREG  A
102             READ        B
103             COMP  AREG  B
104             BC    GT    FIRST
105             PRINT       B
106             STOP
107   FIRST     PRINT       A
108             STOP
109             A     DC    0
110             B     DC    0


ADDRESS         INSTRUCTION
100             090109
101             041109
102             090110
103             061110
104             074107
105             100110
106             000000
107             100109
108             000000
109             0
110             0
 -1             100
```

```
/* Minimum of two numbers*/

ADDRESS          INSTRUCTION
100              READ       A
101              MOVER AREG A
102              READ       B
103              COMP  AREG B
104              BC    LT   FIRST
105              PRINT      B
106              STOP
107   FIRST      PRINT      A
108              STOP
109              A     DC   0
110              B     DC   0


ADDRESS          INSTRUCTION
100              090109
101              041109
102              090110
103              061110
104              071107
105              100110
106              000000
107              100109
108              000000
109              0
110              0
 -1              100
```

```
/*Factorial of a number*/
ADDRESS          INSTRUCTION
100              READ        N
101              MOVER AREG  N
102              COMP  AREG  ZERO
103              BC    EQ    OUT
104   LOOP       MOVER AREG  PROD
105              MULT  AREG  N
106              MOVEM AREG  PROD
107              MOVER AREG  N
108              SUB   AREG  ONE
109              COMP  AREG  ZERO
110              BC    LE    OUT
111              MOVEM AREG  N
112              BC    ANY   LOOP
113   OUT        PRINT       PROD
114              STOP
115              N     DS    1
116              ZERO  DC    0
117              PROD  DC    1
118              ONE   DC    1

ADDRESS          INSTRUCTION
100              090115
101              041115
102              061116
103              073113
104              041117
105              031115
106              051117
107              041115
108              021118
109              061116
110              072113
111              051115
112              076104
113              100117
114              000000
115              1
116              0
117              1
118              1
 -1              100
```