Practical Number: 01

TITLE: **Implementation of Multiprogramming operating system Stage 1:**

**a. CPU/Machine Simulation**

**b.Supervisor Call through Interrupt.**

NAME: **Rohan Doshi** ROLLNO: **41**

CLASS: **TY C** BRANCH: **COMPUTER** **SCIENCE** BATCH: **3**

DATE OF PERFORMANCE: **15** **January 2015**

/\*OS PHASE 1 – JOB 1

Rohan Doshi C-41

Prasad Ghangal C-46

Shubham Dod C-37

Vaibhav Dod C-38

PROGRAM:\*/

#include<iostream>

#include<string.h>

#include<fstream>

#include<string>

using namespace std;

ifstream fin;

ofstream fout;

int SI;

class memory

{

private:

char mem[100][4];

char ch;

string IR,R;

int IC;

bool C;

public:

void reset()

{

memset(mem,'$',sizeof(char)\*100\*4);

IR=R="";

IC=0;

C=false;

}

string get\_mem(int pos)

{

string temp="";

for(int i=0;i<4;i++)

temp+=mem[pos][i];

return temp;

}

void set\_mem(string s, int pos)

{

for(int i=0;i<4;i++)

mem[pos][i]=s[i];

}

void get\_program\_cards()

{

int flag=0;

for(int i=0;i<100;i++)

{

for(int j=0;j<=3;j++)

{

fin>>ch;

mem[i][j]=ch;

if(mem[i][0]=='H')

{

flag=1;

break;

}

}

if(flag)

break;

}

}

void set\_IC()

{

IC=0;

}

void set\_IC(int pos)

{

IC=pos;

}

int get\_IC()

{

return (IC++);

}

void set\_IR(int IC)

{

IR="";

for(int i=0;i<4;i++)

{

IR+=mem[IC][i];

}

}

string get\_IR()

{

return IR;

}

void set\_R(int pos)

{

R="";

for(int i=0;i<4;i++)

R+=mem[pos][i];

}

string get\_R()

{

return R;

}

void set\_C(bool value)

{

C=value;

}

bool get\_C()

{

return C;

}

}m\_obj;

class cpu

{

private:

int fetched\_IC;

bool terminate,fetched\_C;

string fetched\_IR,operand,opreator,fetched\_R,compare\_string;

public:

int s\_to\_i(string operand)

{

return ((int)operand[0]-48)\*10+((int)operand[1]-48);

}

void startexe()

{

m\_obj.set\_IC();

PROGRAM();

}

void PROGRAM()

{

terminate=false;

while(!terminate)

{

fetched\_IC = m\_obj.get\_IC();

m\_obj.set\_IR(fetched\_IC);

fetched\_IR=m\_obj.get\_IR();

opreator=fetched\_IR.substr(0,2);

operand=fetched\_IR.substr(2,2);

if(!(opreator.compare("LR")))

{

int pos=s\_to\_i(operand);

m\_obj.set\_R(pos);

}

else if (!(opreator.compare("SR")))

{

fetched\_R=m\_obj.get\_R();

int pos=s\_to\_i(operand);

m\_obj.set\_mem(fetched\_R, pos);

}

else if (!(opreator.compare("CR")))

{

fetched\_R=m\_obj.get\_R();

int pos=s\_to\_i(operand);

compare\_string=m\_obj.get\_mem(pos);

if(fetched\_R.compare(compare\_string)==0)

m\_obj.set\_C(true);

else

m\_obj.set\_C(false);

}

else if (!(opreator.compare("BT")))

{

fetched\_C=m\_obj.get\_C();

if(fetched\_C)

{

int pos=s\_to\_i(operand);

m\_obj.set\_IC(pos);

}

}

else if (!(opreator.compare("GD")))

{

SI=1;

MOS();

}

else if (!(opreator.compare("PD")))

{

SI=2;

MOS();

}

else

{

SI=3;

MOS();

}

}

}

void MOS()

{

if(SI==1)

{

string s;

int pos=s\_to\_i(operand);

pos=(pos/10)\*10;

getline(fin,s);

if(!s.empty() && s[s.size()-1]=='\r')

s.erase(s.size()-1);

int len=s.length(),start=0,i;

string s1;

for(i=pos;start<len;i++)

{

if((len-start)<4)

s1=s.substr(start,(len-start));

else

s1=s.substr(start,4);

start+=4;

m\_obj.set\_mem(s1,i);

}

}

else if(SI==2)

{

int pos=s\_to\_i(operand),flag=0;

pos=(pos/10)\*10;

string ans="",temp="";

for(int i=pos;i<pos+10;i++)

{

temp=m\_obj.get\_mem(i);

for(int j=0;j<4;j++)

{

if(temp[j]=='\0' || temp[j]=='$')

{

break;

flag=1;

}

ans+=temp[j];

}

if(flag)

break;

}

fout<<ans<<endl;

}

else

{

terminate=true;

fout<<endl<<endl;

}

}

};

int main()

{

fin.open("job1ALL.txt");

fout.open("job1ALL\_op.txt");

string s,s1;

cpu exe;

while(!(fin.eof()))

{

getline(fin,s);

if(s.find("$AMJ")!=-1)

{

m\_obj.reset();

m\_obj.get\_program\_cards();

continue;

}

else if(s.find("$DTA")!=-1)

exe.startexe();

else if(s.find("$END")!=-1)

continue;

}

fin.close();

fout.close();

return 0;

}

**Input File 1:**

$AMJ010200210001

GD30LR36SR40LR35SR41LR34SR42LR33SR43LR32

SR44LR31SR45LR30SR46LR39SR47SR38SR49PD40

H

$DTA

AMA PANANAL A CPLANN A A MA

$END0102

$AMJ020100120003

GD20LR20GD30CR33BT07GD40PD40PD20PD30GD40

PD40H

$DTA

HOPE FOR IT

THERE IS NO HOPE

BUT STILL HOPE

$END0201

$AMJ000300170007

GD20PD20GD30PD30GD40LR40SR20PD20PD30GD40

LR40SR20PD20PD30GD50PD50H

$DTA

3 LITTLE PIGS WENT TO OS CLASS.

THE WOLF ATE ONE!

2 LI

1 LI

CLASS DISMISSED!

$END0003

$AMJ020200160005

GD20PD20LR20SR30SR31PD30SR40SR41SR42PD40

SR50SR51PD50SR60PD60H

$DTA

\*

$END0202

$AMJ000700200002

GD50LR50SR67LR51SR66LR52SR65LR53SR64LR54

SR63LR55SR62LR56SR61LR57SR60PD50PD60H

$DTA

N U T R A F I N

$END0007

$AMJ030100100003

GD20PD20GD30PD30LR20CR30BT10GD40PD40H

$DTA

VIT IS GOOD

VIIT IS GOOD

BOTH ARE SISTER INSTITUTES

$END0301

$AMJ030200080003

GD20PD20GD30PD30LR30SR20PD20H

$DTA

CAT CAN EAT RAT

RAT CAN NOT EAT CAT

$END0302

$AMJ010100040001

GD20LR22SR25PD20H

$DTA

I LIKE THIS PEN OF

$END0101

$AMJ010200070002

GD20LR26CR20BT06GD30PD30PD20H

$DTA

RAM IS OLDER THAN SHRIRAM

NOT IN EXISTANCE

$END0102

$AMJ040100120004

GD20PD20GD30PD30GD40GD50LR20CR30BT10PD40

PD50H

$DTA

ABCD

ACD

DO NOT

MATCH

$END0401

$AMJ040200130004

GD20PD20GD30LR30SR20PD20GD40PD40GD50LR50

SR20PD20H

$DTA

RAT

S

ON

M

$END0402----

**Output for Input File 1:**

A MAN A PLAN A CANAL PANAMA

HOPE FOR IT

THERE IS NO HOPE

BUT STILL HOPE

3 LITTLE PIGS WENT TO OS CLASS.

THE WOLF ATE ONE!

2 LITTLE PIGS WENT TO OS CLASS.

THE WOLF ATE ONE!

1 LITTLE PIGS WENT TO OS CLASS.

THE WOLF ATE ONE!

CLASS DISMISSED!

\*

\*\*

\*\*\*

\*\*

\*

N U T R A F I N

N I F A R T U N

VIT IS GOOD

VIIT IS GOOD

BOTH ARE SISTER INSTITUTES

CAT CAN EAT RAT

RAT CAN NOT EAT CAT

RAT CAN EAT RAT

I LIKE THIS PEN OF HIS

RAM IS OLDER THAN SHRIRAM

ABCD

ACD

DO NOT

MATCH

RAT

SAT

ON

MAT

**Input File 2:**

$AMJ050100080003

GD20GD30LR20SR23PD20PD30PD20H

$DTA

|

|--|

$END0501

$AMJ050100100005

GD20GD30GD40GD50GD60LR20CR40BT09PD50PD60

H

$DTA

14

2

1

NOT

PALLINDROME

$END0006

$AMJ040100250003

GD50GD60GD70LR50SR51SR54SR61SR65SR84LR60

SR62SR63SR80SR83SR85LR70SR50SR52SR53SR64

SR81SR82PD50PD60PD80H

$DTA

\_

|

$END0401

$AMJ040200160003

GD50GD60GD70LR50SR51SR61SR71LR60SR72LR70

SR50SR70PD50PD60PD70H

$DTA

\_

|

$END0402

**Output for Input File 2:**

| |

|--|

| |

NOT

PALLINDROME

\_ \_

|\_|| \_

| |\_|

\_

|\_

\_|