***OS PHASE 2***

**Name- Purva Nartam**

**TY C B1 8**

**161439**

**import java.io.\*;**

**import java.util.\*;**

**import java.math.\*;**

**public class PHASE2**

{

int IC; int jobid=0; int PTR;

char M[][];

char IR[]; char R[];

boolean C = false;

int SI=3; int TI=0; int PI=0;

int TTL=0; int TLL=0;

int TTC=0; int LLC=0;

public void TERMINATE(Integer a,Integer b)

{

switch(a)

{

case 0:

System.out.println("No error");

break;

case 1:

System.out.println("Out of Data");

break;

case 2:

System.out.println("Line Limit Exceeded");

break;

case 3:

System.out.println("Time Limit Exceeded");

break;

case 4:

System.out.println("Operation Code Error"); this.PI=0;

break;

case 5:

System.out.println("Operand Error"); this.PI=0;

break;

case 6:

System.out.println("Invalid Page Fault"); this.PI = 0;

break;

default:

System.out.println("Something went wrong.");

}

switch(b)

{

case 4:

System.out.println("Operation Code Error");

break;

case 5:

System.out.println("Operand Error");

break;

default:

{}

}

}

**public int addressMap(Integer VA,Integer RA)**

{

int num = -1;

RA = RA + VA/10;

if((String.valueOf(this.M[RA])).substring(2,4).equals("\*\*"))

{

this.PI=3; System.out.println("Page fault."); //Invalid page fault

if(this.SI == 1) //Valid Page fault

{

Random generator1 = new Random();

num = generator1.nextInt(30);

this.PI = 0;

if(Integer.toString(num).length() == 1)

this.M[RA] = ("\*\*0"+Integer.toString(num)).toCharArray();

else

this.M[RA] = ("\*\*"+Integer.toString(num)).toCharArray();

}

}

else if(RA > 300)

{this.PI = 2 ; System.out.println("Operand Error"); this.TERMINATE(5,0);}

else

{ num = Integer.parseInt((String.valueOf(this.M[RA])).substring(2,4))\*10 + VA%10;}

return num;

}

**public void MOS(Scanner sc,Integer operand)**

{

String line = new String();

Integer j=0,last=4,num=0;

if(this.TI == 0)

{

switch(this.SI)

{

case 1:

//READ();

//System.out.println("READ " + address);

this.TTC++;

this.IR[3]='0';

line = sc.nextLine();

if(line.contains("END")) this.TERMINATE(1,0);

else {

num = this.addressMap(operand,this.PTR); num \*= 10;

while(j <= line.length())

{

if(last > line.length()) last = line.length();

this.M[num++] = line.substring(j,last).toCharArray();

//System.out.println("chk"+String.valueOf(this.M[0]));

// System.out.println(String.valueOf(this.M[address-1]));

j+=4;

last = j+4;

}

j=0;

}

//address = (address/10 + 1) \* 10;

break;

case 2:

//WRITE();

this.LLC++;

if(this.LLC > this.TLL ) this.TERMINATE(2,0);

this.IR[3] = '0';

// System.out.println("WRITE" + address);

num = this.addressMap(operand,this.PTR);

if(num < 0) this.TERMINATE(6,0);

else

{

for(j=num;j<= num+9;j++)

System.out.println(j+" PRINT: " + String.valueOf(this.M[j]));

}

break;

case 3:

//TERMINATE(); return;

this.TERMINATE(0,0);

break;

default:

System.out.println("No System Interrupts");

}

switch(this.PI)

{

case 1:

this.TERMINATE(4,0);

break;

case 2:

this.TERMINATE(5,0);

break;

case 3:

if(this.PI != 0)

this.TERMINATE(6,0);

break;

}

}

else{

switch(this.SI)

{

case 1:

this.TERMINATE(3,0); break;

case 2:

this.IR[3] = '0';

// System.out.println("WRITE" + address);

num = this.addressMap(operand,this.PTR);

if(num < 0) this.TERMINATE(6,0);

else {

for(j=num;j<= num+9;j++)

System.out.println(j+" PRINT: " + String.valueOf(this.M[j]));

}

this.TERMINATE(3,0); break;

case 3:

this.TERMINATE(0,0);

}

switch(this.PI)

{

case 1:

this.TERMINATE(3,4); break;

case 2:

this.TERMINATE(3,5); break;

case 3:

this.TERMINATE(3,0); break;

}

}

}

**public void exec\_user\_prog(Scanner sc)**

{

String opcode="";

Integer operand=0;

System.out.println("EXEC\_USER\_PROG");

int RA=0;

do

{

RA = this.addressMap(this.IC,this.PTR);

/\* if( this.PI != 0 )

{

//System.out.println("Error hai.");

//System.exit(0);

} \*/

this.IR = this.M[RA]; //IR <- M[IC]

this.IC += 1; //IC ++

this.TTC++; if(this.TTC == this.TTL) {this.TI=2;}

if(String.valueOf(this.IR).length() > 1)

{

opcode = String.valueOf(this.IR).substring(0,2);

operand = Integer.parseInt(String.valueOf(this.IR).substring(2,4));

}

else

{opcode = String.valueOf(this.IR); }

switch(opcode)

{

case "GD":

this.SI = 1;

this.MOS(sc,operand);

this.SI = 0;

break;

case "PD":

this.SI = 2;

this.MOS(sc,operand);

this.SI=0;

break;

case "H":

this.SI = 3;

this.MOS(sc,operand);

break;

case "LR":

/\* System.out.println("");

System.out.println(operand); \*/

operand = this.addressMap(operand,this.PTR);

if(operand < 0) this.TERMINATE(6,0);

else this.R = this.M[operand];

break;

case "CR":

operand = this.addressMap(operand,this.PTR);

if(String.valueOf(this.R).equals(String.valueOf(this.M[operand])))

this.C = true;

else

this.C = false;

break;

case "SR":

operand = this.addressMap(operand,this.PTR);

this.M[operand] = this.R;

break;

case "BT":

if(this.C) IC = operand;

break;

default:

this.PI = 1;

}

}while(this.SI != 3);

}

**public void loadIntoMem(String line,Random generator)**

{

int m=0,i=0,j=0,k=0,last=4,pgmcard=0,end=0;

while( j <= line.length() )

{

i = generator.nextInt(30); //generate the random number for program cards

System.out.println(i);

if(Integer.toString(i).length() == 1)

this.M[this.PTR + (pgmcard)] = ("\*\*0"+Integer.toString(i)).toCharArray();

else

this.M[this.PTR + (pgmcard)] = ("\*\*"+Integer.toString(i)).toCharArray();

for(m=0;m<10;m++)

{

this.M[i\*10+m] = line.substring(j,last).toCharArray();

if(line.substring(j,last).equals("H")) {end=1; break;}

j+=4;

last = j+4;

if(last > line.length()) last = line.length();

}

k=Integer.parseInt((String.valueOf(this.M[this.PTR+pgmcard])).substring(2,4)); //the page table number

for(m=k\*10;m<k\*10+10;m++)

System.out.println("MEM: " + m + String.valueOf(this.M[m])); //to print the program cards

if(end == 1) break;

pgmcard+= 1;

}

j=0;

}

**private void initPageTable(Integer Ptr)**

{

int k=0; int j=0;

for(k=0;k<10;k++){

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

this.M[Ptr + k][j]='\*';

}

for(k=0;k<10;k++)

System.out.println(String.valueOf(this.M[Ptr+k]));

}

public void init(String line,Random generator)

{

System.out.println("INIT");

this.R = new char[4];

this.IR = new char[4];

this.M = new char[300][4];

this.jobid = Integer.parseInt(line.substring(4,8));

this.TTL = Integer.parseInt(line.substring(8,12));

this.TLL = Integer.parseInt(line.substring(12,16));

this.PTR = generator.nextInt(30) \* 10;

this.initPageTable(this.PTR);

}

**public static void main(String args[]) throws FileNotFoundException, StringIndexOutOfBoundsException**

{

PHASE2 p = new PHASE2();

int j=0,pgmcard=0;

String line="";

String sub = "";

**File file = new File("job2.txt");**

Scanner sc = new Scanner(file);

Random generator = new Random();

while(sc.hasNextLine())

{

line = sc.nextLine();

if(line.length() > 3) sub = line.substring(0,4);

else sub = line;

switch(sub)

{

case "$AMJ":

System.out.println("Here i begin");

p.init(line,generator);

System.out.println("jobid:"+p.jobid+" TTL:"+p.TTL+" TLL:"+p.TLL+ " PTR:"+p.PTR);

break;

case "$DTA":

p.IC = 0;

p.exec\_user\_prog(sc);

break;

case "$END":

break;

default:

if(sub.substring(0,2).equals("GD") || sub.substring(0,2).equals("PD") || sub.substring(0,2).equals("LR") || sub.substring(0,2).equals("SR") || sub.substring(0,2).equals("CR") ||sub.substring(0,2).equals("BT") || sub.substring(0,2).equals("H"))

p.loadIntoMem(line,generator);

}

}

System.out.println("@@@@@@@@@@@@@@@@@@@@@@@@@@@@@");

System.out.println(String.valueOf(p.M[p.PTR]));

for(j=p.PTR;j<p.PTR+10;j++)

{

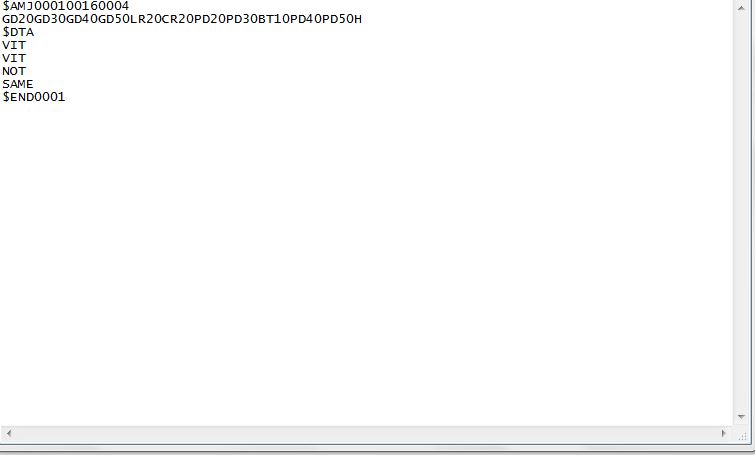
System.out.println("MEM: " + j + String.valueOf(p.M[j])); //to print the page table

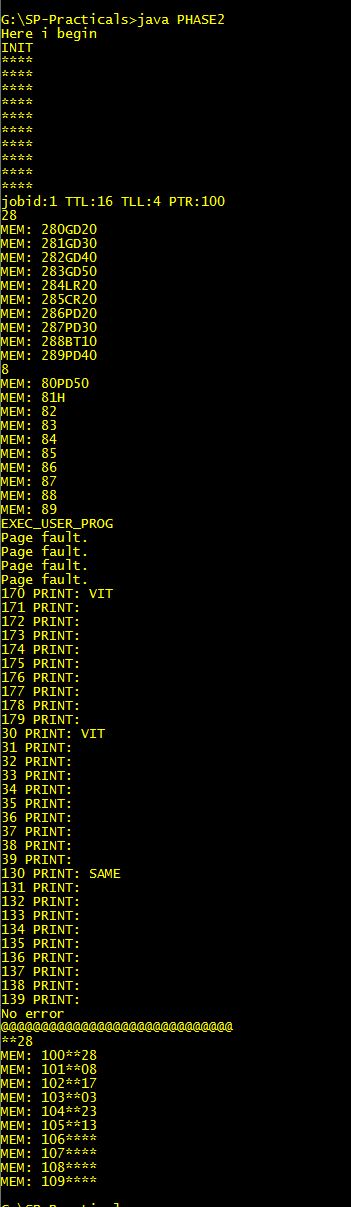
}

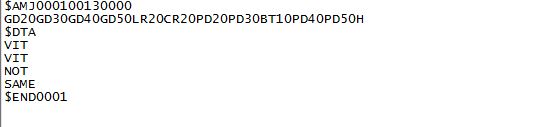
}

}

For job.txt as



Output is 

For job file as:

Output is: 