**CS\_6560\_02**

**Execute.java-------------**

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

**import** java.util.ArrayList;

**import** java.util.Scanner;

**public** **class** execute {

**private** **final** String FILENAME = "src/file1.txt";

**private** **final** String FILENAME1 = "src/file2.txt";

**int**[] array = **new** **int**[]{1,5,4,7,6,8,9,2,3,2};

ArrayList<String> m= **new** ArrayList<String>(40);

**int** accum, mar=0, pc=0, F, R, E=0;

**int** mbr1, jumpAddress = -1;

String address;

String OPR;

String I;

String mbr;

**boolean** processSwitched = **true**;

**char**[] ch;

**int** turn=0;

**int** count=3;

**boolean** flag = **false**;

ArrayList<Processblock> processes;

execute()

{

F=0;

R=0;

}

**public** **void** Execute1(String OPR,String address) {

**switch** (OPR) {

**case** "AND":

System.***out***.println("Befor AND operation:" +accum);

mar = Integer.*parseInt*(address);

mbr1 = array[mar];

accum = accum & mbr1;

F = 0;

System.***out***.println("After AND operation:" +accum);

processSwitching();

**break**;

**case** "ADD":

System.***out***.println("Befor ADD operation:" +accum);

mar = Integer.*parseInt*(address);

mbr1 = array[mar];

accum = accum + mbr1;

F = 0;

System.***out***.println("After ADD operation:" +accum);

processSwitching();

**break**;

**case** "LDA":

System.***out***.println("Load Operation");

System.***out***.println("Before LDA: "+ accum);

mar = Integer.*parseInt*(address);

mbr1 = array[mar];

accum = mbr1;

F = 0;

System.***out***.println("After Loaded to accumcumulator: "+ accum);

processSwitching();

**break**;

**case** "STA":

System.***out***.println("Store");

mar = Integer.*parseInt*(address);

mbr1 = accum;

array[mar] = mbr1;

F = 0;

System.***out***.println("pc value:"+pc);

System.***out***.println(array[mar] + "stored in accumcumulator"+ mar);

processSwitching();

**break**;

**case** "BUN":

System.***out***.println("before BUN:"+pc);

pc = Integer.*parseInt*(address);

F=0;

// Fetch(pc);

System.***out***.println(pc);

processSwitching();

**break**;

**case** "BSA":

System.***out***.println("bsa :" + pc + " "+ mar);

**int** mar =Integer.*parseInt*(address);

jumpAddress = pc;

**int** temp = pc;

pc = mar;

mar = temp;

pc=pc+1;

F=0;

//Fetch(pc);

System.***out***.println("After bsa :" + pc + " " + mar);

processSwitching();

**break**;

**case** "ISZ":

System.***out***.println("isz");

mar=Integer.*parseInt*(address);

mbr1=array[mar];

mbr1 = mbr1+1;

**if**(mbr1 == 0)

pc = pc+1;

processSwitching();

flag = **true**;

**break**;

**case** "REG" :

regRef(address);

processSwitching();

}

}

**int** rightRotate(**int** accum) {

**int** INT\_BITS = 3;

**int** shiftbits=1;

**return** (accum >> shiftbits) | (accum << (INT\_BITS - shiftbits));

}

**int** leftRotate(**int** accum) {

**int** INT\_BITS = 3;

**int** shiftbits=1;

**return** (accum << shiftbits) | (accum >> (INT\_BITS - shiftbits));

}

**public** **void** regRef(String address)

{

**switch**(address){

**case** "CLA":

System.***out***.println("Before CLA:" + accum);

accum=0;

System.***out***.println("After CLA:" + accum);

**break**;

**case** "CLE":

System.***out***.println("Before CLE:" + E);

E=0;

System.***out***.println("Before CLE:" + E);

**break**;

**case** "CMA":

**int** a =-accum;

accum = a;

System.***out***.println(accum);

**break**;

**case** "CME":

**int** Ec = -E;

System.***out***.println(Ec);

**break**;

**case** "CIR":

//int cir = accum/2;

System.***out***.println("Before CIR: "+accum);

**int** cir =rightRotate(accum);

System.***out***.println("After CIR: "+cir);

**break**;

**case** "CIL":

// int cil = accum\*2;

System.***out***.println("Before CIL: "+accum);

**int** cil =leftRotate(accum);

System.***out***.println("After CIL: "+cil);

**break**;

**case** "INC":

System.***out***.println("Befor INC: "+accum);

accum=accum+1;

System.***out***.println("After INC: "+accum);

**break**;

**case** "SPA":

System.***out***.println("SPA");

**if** (accum>0){

pc=pc+1;

}

**break**;

**case** "SNA":

**if** (accum<0){

pc=pc+1;

}

**break**;

**case** "SZA":

**if** (accum==0){

pc=pc+1;

}

**break**;

**case** "SZE":

**if** (E==0)

pc=pc+1;

**break**;

**case** "HLT":

System.*exit*(0);

**break**;

}

}

**void** Indirect(String OPR,String address)

{

**if**( F == 0 && R ==1) {

mar = Integer.*parseInt*(address);

R = 0;

**if**(OPR.equals("4") && jumpAddress>-1){

Execute1(OPR, Integer.*toString*(jumpAddress));

}

**if**(OPR.equals("4") && jumpAddress>-1) {

Execute1(OPR, Integer.*toString*(mar));

}

**else**

{

mar = array[mar];

Execute1(OPR, Integer.*toString*(mar));

}

}

}

**public** **void** Fetch(**int** Al)

{

**if**(Al < m.size()){

**if**(F==0 && R ==0)

{

mar = Al;

pc=pc+1;

System.***out***.println(turn);

m= processes.get(turn).instructions;

mbr = m.get(mar);

String parts[] = mbr.split(" ");

I = parts[0];

OPR = parts[1];

address = parts[2];

**if**(I.equals("0"))

{

F = 1;

R = 0;

Execute1(OPR,address);

}

**else** **if**(I.equals("1"))

{

F=0;

R=1;

Indirect(OPR,address);

}

}

}**else**{

processes.remove(turn);

processSwitching();

}

}

**public** **void** processSwitching()

{

**if**(processes.size()>1){

**if**(count == 0)

{

pcb b = processes.get(turn).getPcb();

b.setaccumulator(accum);

b.setprogramCounter(pc);

b.setmar(mar);

turn = (turn+1)%2;

b = processes.get(turn).getPcb();

m= processes.get(turn).getInstructions();

// System.out.println("Process Switching"+m.get(0));

accum = b.getaccumulator();

mar = b.getmar();

pc = b.getprogramCounter();

count =3;

}

}**else** **if**(processes.size() == 1 && processSwitched == **true**){

turn = 0;

pcb b = processes.get(turn).getPcb();

b = processes.get(turn).getPcb();

m = processes.get(turn).getInstructions();

// System.out.println("Process Switcing"+m.get(0));

accum = b.getaccumulator();

mar = b.getmar();

pc = b.getprogramCounter();

processSwitched = **false**;

count = 10000;

}**else**{

System.*exit*(0);

}

count--;

Fetch(pc);

}

**public** **void** initializeProg(){

BufferedReader br = **null**;

FileReader fr = **null**;

BufferedReader br1=**null**;

FileReader fr1 = **null**;

execute Obj = **new** execute();

processes = **new** ArrayList<>();

**try** {

//1st file

fr = **new** FileReader(FILENAME);

br = **new** BufferedReader(fr);

String sCurrentLine;//Contents of each line in a string is stored in arraylist

**while** ((sCurrentLine = br.readLine()) != **null**) {

// System.out.println(sCurrentLine);

m.add(sCurrentLine);

}

processes.add(**new** Processblock(**new** pcb(0,0,0),m));

m.clear();

//2nd file

fr1 = **new** FileReader(FILENAME1);

br1 = **new** BufferedReader(fr1);

String sCurrentLine1;//Contents of each line in a string is stored in arraylist

**while** ((sCurrentLine1 = br1.readLine()) != **null**) {

// System.out.println(sCurrentLine1);

m.add(sCurrentLine1);

}

processes.add(**new** Processblock(**new** pcb(0,0,0),m));

m = processes.get(turn).getInstructions();

System.***out***.println("File1:"+ processes.get(0).getInstructions());

System.***out***.println("File2:"+ processes.get(1).getInstructions());

processSwitching();

} **catch** (IOException e) {

e.printStackTrace();

} **finally** {

**try** {

**if** (br != **null**)

br.close();

**if** (fr != **null**)

fr.close();

} **catch** (IOException ex) {

ex.printStackTrace();

}

}

}

**public** **static** **void** main(String[] args) {

execute e = **new** execute();

e.initializeProg();

}

}

**pcb.java-----------**

**public** **class** pcb {

**int** programCounter;

**int** accumulator;

**int** mar;

pcb(**int** programCounter,**int** accumulator,**int** mar)

{

**this**.programCounter=programCounter;

**this**.accumulator=accumulator;

**this**.mar=mar;

}

**public** **int** getprogramCounter() {

**return** programCounter;

}

**public** **void** setprogramCounter( **int** pc) {

programCounter = pc;

}

**public** **int** getaccumulator() {

**return** accumulator;

}

**public** **void** setaccumulator( **int** accum) {

accumulator = accum;

}

**public** **int** getmar() {

**return** mar;

}

**public** **void** setmar( **int** mar1) {

mar = mar1;

}

}

**Processblock.java---------------**

**import** java.util.ArrayList;

**public** **class** Processblock {

pcb pcb;

ArrayList<String> instructions;

**public** Processblock(pcb pcb, ArrayList<String> instructions) {

**this**.pcb = pcb;

**this**.instructions = **new** ArrayList<>(instructions);

}

**public** pcb getPcb() {

**return** pcb;

}

**public** **void** setPcb(pcb pcb) {

**this**.pcb = pcb;

}

**public** ArrayList<String> getInstructions() {

**return** instructions;

}

**public** **void** setInstructions(ArrayList<String> instructions) {

**this**.instructions = instructions;

}

}

**InLineExecute.java------------**

**import** java.util.Arrays;

**import** java.util.Scanner;

**public** **class** InLineExecute {

**public** **static** **boolean** *isDirect* = **true**;

**public** **static** String *opCode*;

**public** **static** **int** *AC* = 10,*MBR* = 20,*MAR* = 12,*F*,*PC*,*Address*,*addressOfaddress*,*E*,*S*;

**public** **static** String[] *RAM* = **new** String[100];

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

**int** option;

Scanner reader = **new** Scanner(System.***in***); // Reading from System.in

**boolean** isTrue = **false**;

**do**

{

System.***out***.println("Press 1 to Execute In-Line Commands");

option = reader.nextInt();

**if**(option == 1)

isTrue = **true**;

**else**

System.***out***.println("Select correct option");

}

**while** (!isTrue);

System.***out***.println("Enter a number: "+option);

**int** i;

**for**(i = 0; i < 20;i++)

{

*RAM*[i] = Integer.*toString*((**int**)(Math.*random*()\*20 + 5));

}

**int** j=0;

**for**(i = 20; i < 30;i++)

{

*RAM*[i] = Integer.*toString*(j);

j++;

}

System.***out***.println("Address " + Arrays.*toString*(*RAM*));

//reader.close();

**if**(option == 1)

{

Scanner in = **new** Scanner(System.***in***);

String s = in.nextLine();

*setVariables*(s);

System.***out***.println(*isDirect*);

}

}

**public** **static** **void** openFiles() {

System.***out***.println("openfile function");

}

**public** **static** **void** setVariables(String s) {

String[] parts = s.split(" ");

System.***out***.println(parts.length);

**if**(parts[0].equals("I"))

{

*isDirect* = **false**;

*addressOfaddress* = Integer.*parseInt*(parts[2]);

//System.out.println("Indirect" + addressOfaddress);

*Address* = Integer.*parseInt*(*RAM*[*addressOfaddress*]);

}**else**

{

*Address* = Integer.*parseInt*(parts[2]);

}

*opCode* = parts[1];

*performOperations*(*opCode*);

}

**public** **static** **void** performOperations(String add) {

String opCode = add.toUpperCase();

System.***out***.println("Performing Operations AC is : " + *AC* +" "+ *Address*);

**switch**(add)

{

**case** "AND":

*MAR* = Integer.*parseInt*(*RAM*[*Address*]);

*MBR* = *MAR*;

*AC* = *AC* & *MAR*;

*F* = 0;

System.***out***.println("AND Operation:" + *AC*);

**break**;

**case** "ADD":

*MAR* = Integer.*parseInt*(*RAM*[*Address*]);

*MBR* = *MAR*;

*AC* = *AC* + *MBR*;

*F* = 0;

System.***out***.println("ADD Operation:" + *AC*);

**break**;

**case** "LDA":

*MAR* = Integer.*parseInt*(*RAM*[*Address*]);

*MBR* = *MAR*;

*AC* = *MBR*;

*F* = 0;

System.***out***.println("LDA Operation:" + *AC*);

**break**;

**case** "STA":

System.***out***.println(" Before STA Operation:" + *RAM*[*MAR*]);

*MAR* = Integer.*parseInt*(*RAM*[*Address*]);

*MBR* = *AC*;

*RAM*[*MAR*] = Integer.*toString*(*MBR*);

*F* = 0;

System.***out***.println("STA Operation:" + *RAM*[*MAR*]);

**break**;

**case** "BUN":

System.***out***.println("before BUN Operation:" + *PC*);

*PC* = *MBR*;

*F* = 0;

System.***out***.println("BUN Operation:" + *PC*);

**break**;

**case** "BSA":

System.***out***.println("BSA Operation:" + *PC*);

*MAR* = Integer.*parseInt*(*RAM*[*Address*]);

*RAM*[*Address*] = Integer.*toString*(*PC*);

*PC* = Integer.*parseInt*(*RAM*[*Address*]);

*MAR* = *MBR*;

*PC* = *PC* + 1;

*F* = 0;

System.***out***.println("BSA Operation:" + *PC*);

**break**;

**case** "ISZ":

System.***out***.println("ISZ Operation:" + *PC*);

*MAR* = Integer.*parseInt*(*RAM*[*Address*]);

*MBR* = *MAR*;

*MBR* = *MBR* + 1;

*MAR* = *MBR*;

**if**(*MBR* == 0)

{

*PC* = *PC* + 1;

}

*F* = 0;

System.***out***.println("ISZ Operation:" + *PC*);

**break**;

**case** "CLA":

System.***out***.println("CLA Operation:" + *AC*);

*AC* = 0;

System.***out***.println("CLA Operation:" + *AC*);

**break**;

**case** "CLE":

*E* = 0;

**break**;

**case** "CMA":

*AC* = ~*AC*;

**break**;

**case** "CME":

*E* = ~*E*;

**break**;

**case** "CIR":

*AC* = *AC* >> 1;

*E* = *E* >> 1;

**break**;

**case** "CIL":

*AC* = *AC* << 1;

*E* = *E* << 1;

**break**;

**case** "INC":

*AC* = *AC* + 1;

**break**;

**case** "SPA":

**if**(*AC* > 0)

{

*PC* = *PC* + 1;

}

**break**;

**case** "SNA":

**if**(*AC* < 0)

{

*PC* = *PC* + 1;

}

**break**;

**case** "SZA":

**if**(*AC* == 0)

{

*PC* = *PC* + 1;

}

**break**;

**case** "SZE":

**if**(*E* == 0)

{

*PC* = *PC* + 1;

}

**break**;

**case** "HLT":

System.*exit*(0);

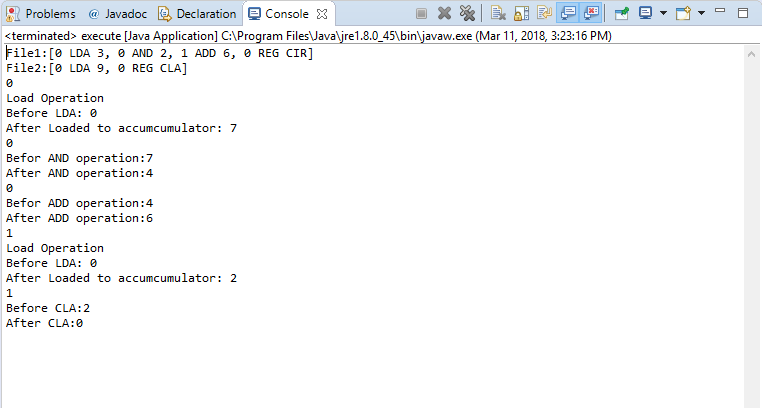
**break**;

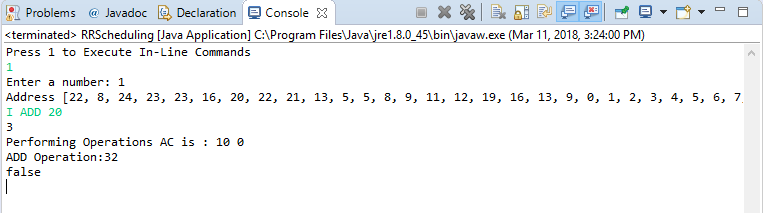
}

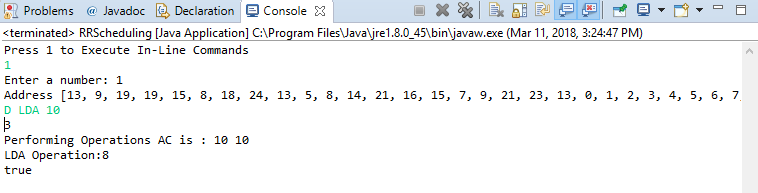
}

}

**OUTPUT**

****

****

****

**file1.txt**

0 LDA 3

0 AND 2

1 ADD 6

0 REG CIR

**file2.txt**

0 LDA 9

0 REG CLA

1 ADD 10

0 REG CIL