OS Phase 1

Group: TY 3

Group Members:

Roll no NAME GR. NO.

3 Aarya Tiwari 12010588

5 Harshal Abak 12011089

9 Abu Ansari 12010599

26 Kaivalya Aole 12011351

Code was written in java

# Code:-

import java.io.\*;

import java.nio.file.Files;

import java.nio.file.Paths;

import java.nio.file.StandardOpenOption;

import java.util.Scanner;

import java.io.FileWriter;

public class Phase\_1 {

    public static void main(String[] args) throws FileNotFoundException {

        OS object = new OS();

        object.load();

        object.printer();

    }

}

class OS{

    char [] buffer;

    char [][] memory;

    int mem\_ptr;

    //CPU

    char [] IR;

    char [] R;

    int IC;

    boolean toggle;

    int SI; //system Interrupt

    File input;

    FileReader fr;

    public OS() throws FileNotFoundException {

        //size allocation

        buffer = new char[40];

        memory = new char[100][4];

        IR = new char[4];

        R = new char [4];

        value\_allocator();

    }

    private void value\_allocator() throws FileNotFoundException {

        //value allocation

        for(int i = 0;i<4;i++){

            IR[i] = '@';

            R[i] = '@';

        }

        buffer\_reset();

        for(int i = 0;i < 100; i++){

            for(int j = 0; j< 4;j++){

                memory[i][j] = '@';

            }

        }

        IC = 0;

        toggle = false;

        mem\_ptr = 0;

        SI = 0;

    }

    public void MOS(){

        switch (SI){

            case 1:

                try {

                    read();

                } catch (IOException e) {

                    e.printStackTrace();

                }

                break;

            case 2:

                write();

                break;

            case 3:

                halt();

                break;

        }

    }

    public void read() throws IOException {

        System.out.println("Read executed");

        IR[3] = '0';

        int buffer\_ptr;

        char [] temp;

        if(reader.ready()){

            temp = reader.readLine().toCharArray();

            for (buffer\_ptr = 0; buffer\_ptr < temp.length; buffer\_ptr++) {

                buffer[buffer\_ptr] = temp[buffer\_ptr];

            }

            System.out.println(buffer);

        }

        if (buffer[0] == '$' && buffer[1] == 'E' && buffer[2] == 'N' && buffer[3] == 'D') {

            System.out.println("Program Over");

            return;

        }

        load\_memory\_data((IR[2]-'0')\*10);

        buffer\_reset();

    }

    public void write(){

        System.out.println("Write executed");

        IR[3] = '0';

        int buffer\_ptr;

        char [] temp;

        try{

            IR[3] = '0';

            int memory\_ptr = (IR[2]-'0')\*10;

            int buffer\_ptr4 = 0;

            int limit = memory\_ptr+10;

            for(memory\_ptr = (IR[2]-'0')\*10;memory\_ptr<limit;memory\_ptr++){

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

                    if(memory[memory\_ptr][i] == '@')

                        continue;

                    buffer[buffer\_ptr4] = memory[memory\_ptr][i];

                    buffer\_ptr4++;

                }

            }

            StringBuffer sb = new StringBuffer();

            buffer\_ptr4 = 0;

            while(buffer\_ptr4 < 40 && buffer[buffer\_ptr4]!='@'){

                sb.append(buffer[buffer\_ptr4]);

                buffer\_ptr4++;

            }

            sb.append('\n');

            Files.write(Paths.get("out.txt"), String.valueOf(sb).getBytes(), StandardOpenOption.APPEND);

            buffer\_reset();

        }

        catch(IOException e){

            System.out.println("Exception occured at write()");

            e.printStackTrace();

        }

    }

    public void halt(){

        try{

            System.out.println("Halt called");

            Files.write(Paths.get("out.txt"), "\n\n".getBytes(), StandardOpenOption.APPEND);

            IC = 100;

            return;

        }

        catch(IOException e){

            System.out.println("Exception occured at halt()");

            e.printStackTrace();

        }

    }

    public void MOSstartExec(){

        System.out.println("Exec started");

        IC = 0;

        executeUserProgram();   //slave mode

    }

    public void executeUserProgram(){

        //loading IR

        while (IC<90 && memory[IC][0] != '@') {

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

                IR[i] = memory[IC][i];

            }

            IC++;

            switch (IR[0]) {

                case 'L':

                    if(IR[1] == 'R'){

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

                            R[i] = memory[(IR[2]-'0')\*10 + (IR[3] -'0')][i];

                        }

                    }

                    break;

                case 'S':

                    if(IR[1] == 'R'){

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

                            memory[(IR[2]-'0')\*10 + (IR[3] -'0')][i] = R[i];

                        }

                    }

                    break;

                case 'C':

                    if (IR[1] == 'R') {

                        char a =IR[2];

                        char b =IR[3];

                        comparing(a,b);

                    }

                    break;

                case 'B':

                    if(IR[1] == 'T'){

                        if(toggle == true){

                            IC = (IR[2] - '0') \*10 + (IR[3] - '0');

                        }

                    }

                    break;

                case 'G':

                    if (IR[1] == 'D') {

                        SI = 1;

                        MOS();

                    }

                    break;

                case 'P':

                    if (IR[1] == 'D') {

                        SI = 2;

                        MOS();

                    }

                    break;

                case 'H':

                    SI = 3;

                    MOS();

                    break;

            }

        }

    }

    public void comparing( char a,char b) {

        int c = (a-'0') \* 10 + (b-'0') ;

        for (int i = 0; i < 4; i++) {

            if (R[i] == memory[c][i]) {

                continue;

            } else {

                toggle = false;

                return;

            }

        }

        toggle = true;

        return ;

    }

    private static java.io.File file;

    private static BufferedReader reader;

    static {

        try {

            file = new java.io.File("input.txt");

            reader = new BufferedReader(new FileReader(String.valueOf(file)));

        } catch (FileNotFoundException e) {

            e.printStackTrace();

        }

    }

    public void load(){

        //loading from card to buffer

        try {

            do {

                int buffer\_ptr = 0;

                char [] temp;

                temp = reader.readLine().toCharArray();

                for (buffer\_ptr = 0; buffer\_ptr < temp.length; buffer\_ptr++) {

                    buffer[buffer\_ptr] = temp[buffer\_ptr];

                }

                System.out.println(buffer);

                if (buffer[0] == '$' && buffer[1] == 'A' && buffer[2] == 'M' && buffer[3] == 'J') {

                    buffer\_reset();

                    continue;

                }

                else if (buffer[0] == '$' && buffer[1] == 'D' && buffer[2] == 'T' && buffer[3] == 'A') {

                    buffer\_reset();

                    mem\_ptr = (mem\_ptr % 10 == 0) ? mem\_ptr : ((mem\_ptr / 10 + 1) \* 10);

                    MOSstartExec();

                }

                else if (buffer[0] == '$' && buffer[1] == 'E' && buffer[2] == 'N' && buffer[3] == 'D') {

                    System.out.println("Program Over");

                    printer();

                    value\_allocator();

                }

                else{

                    load\_memory\_instructions();

                    buffer\_reset();

                }

                if (buffer\_ptr > 40) {

                    System.out.println("Buffer Overload, quitting...");

                    return;

                }

            }while(reader.ready());

        } catch (IOException e) {

            e.printStackTrace();

        }

    }

    public void load\_memory\_instructions(){

        int buffer\_ptr2 = 0;

        while(buffer\_ptr2<40 && buffer[buffer\_ptr2]!='@'){

            for(int i = 0;i<4;i++){

                if(buffer[buffer\_ptr2] == '@')

                    break;

                memory[mem\_ptr][i] = buffer[buffer\_ptr2];

                if(buffer[buffer\_ptr2]=='H'){

                    buffer\_ptr2++;

                    break;

                }

                buffer\_ptr2++;

            }

            mem\_ptr++;

        }

    }

    public void load\_memory\_data(int location){

        mem\_ptr = location;

        if(mem\_ptr>=100){

            System.out.println("Memory Overload, quitting...");

            return;

        }

        int buffer\_ptr1 = 0;

        while(buffer\_ptr1<40 && buffer[buffer\_ptr1]!='@'){

            for(int i = 0;i<4;i++){

                if(buffer[buffer\_ptr1] == '@')

                    break;

                memory[mem\_ptr][i] = buffer[buffer\_ptr1];

                buffer\_ptr1++;

            }

            mem\_ptr++;

        }

    }

    public void buffer\_reset(){

        for(int i = 0;i< 40;i++){

            buffer[i] = '@';

        }

    }

    public void printer(){

        System.out.println("Buffer is: ");

        System.out.println(buffer);

        System.out.println("IC is:");

        System.out.println(IC);

        System.out.println("Main memory");

        for(int i = 0; i< 100;i++){

            System.out.print(i+" : ");

            for(int j = 0;j<4;j++){

                System.out.print(memory[i][j]);

            }

            System.out.println();

        }

        System.out.println("IR:");

        System.out.println(IR);

        System.out.println("R:");

        System.out.println(R);

        System.out.println("Toggle:");

        System.out.println(toggle);

    }

}

# Input File:

$AMJ000200160003

GD20PD20GD30PD30GD40GD50LR20CR30BT11

PD40HPD50H

$DTA

VIIT

VIT

NOT SAME

SAME

$END0002

$AMJ000400230001

GD30GD40GD50GD60LR40CR30BT17CR31BT17

CR32BT17CR33BT17CR34BT17PD60HPD50H

$DTA

a e i o u

b

IS VOWEL

NOT VOWEL

$END0004

$AMJ000600300007

GD20LR20SR60PD60SR30SR31PD30SR40SR41

SR42PD40SR50SR51SR52SR53PD50PD40PD30

PD60H

$DTA

\*

$END0006

# Output File:

VIIT

VIT

NOT SAME

NOT VOWEL

\*

\* \*

\* \* \*

\* \* \* \*

\* \* \*

\* \*

\*