Bansilal Ramnath Agarwal Charitable Trust's

Vishwakarma Institute of Technology, Pune-37

(Anautonomous Institute of Savitribai Phule Pune University)

Department of Computer Engineering

Division	**CS**
Batch	**B1**
Roll no.	**90**
Name	**Aditya Shrinivas Kurapati**
PRN No	**12320184**

Phase 1:

Program Code:

Main.java

```
import java.io.BufferedReader;

import java.io.FileReader;

import java.io.FileWriter;

import java.io.IOException;


public class Main {

    static BufferedReader fread;

    private static FileReader fr;

    FileWriter fw;

    static int memory_used;

    static int IC;

    static int T;

    static String line;

    static char[][] memory = new char[100][4];

    static char[] buffer = new char[40];

    static char[] IR = new char[4];

    static char[] R = new char[4];
    public Main() {
    }


    public void load() {
        try {
            this.fw = new FileWriter("output.txt");
            fr = new FileReader("/Users/Ronak/IdeaProjects/phase1/src/input.txt");
            fread = new BufferedReader(fr);


            while(true) {
                while((line = fread.readLine()) != null) {
                    buffer = line.toCharArray();
                    if (buffer[0] == '$' && buffer[1] == 'A' && buffer[2] == 'M' && buffer[3] == 'J') {
                        System.out.println("program card detected");
                        init();
                        this.pcb(buffer);
                    } else if (buffer[0] == '$' && buffer[1] == 'D' && buffer[2] == 'T' && buffer[3] == 'A') {
                        System.out.println("DATA card detected");
                        this.execute();
                    } else if (buffer[0] == '$' && buffer[1] == 'E' && buffer[2]
    == 'N' && buffer[3] == 'D') {
                        System.out.println("END card detected");
                        System.out.println();
                        this.fw.write("\n");
                    } else {
                        if (memory_used == 100) {
                            System.out.println("Abort due to exceed memory
    usage");
                        }

                        int i = 0;

                        while(i < line.length()) {
                            memory[memory_used][i % 4] = buffer[i];
                            ++i;
                            if (i % 4 == 0) {
                                ++memory_used;
                            }
                        }
                    }
                    }

                    this.fw.close();
                    break;
                }
            } catch (Exception var2) {
                System.out.println("All Jobs Executed Successfully....");
            }

        }

        public static void init() {
            memory_used = 0;
            memory = new char[100][4];
            T = 0;
            IC = 0;
        }

        public void execute() throws IOException {
            while(true) {
                if (IC != 100) {
```

```
IR[0] = memory[IC][0];
IR[1] = memory[IC][1];
IR[2] = memory[IC][2];
IR[3] = memory[IC][3];
++IC;
String LINE;
int num;
if (IR[0] == 'L' && IR[1] == 'R') {
    LINE = new String(IR);
    num = Integer.parseInt(LINE.substring(2));
    R[0] = memory[num][0];
    R[1] = memory[num][1];
    R[2] = memory[num][2];
    R[3] = memory[num][3];
    continue;
}

if (IR[0] == 'S' && IR[1] == 'R') {
    LINE = new String(IR);
    num = Integer.parseInt(LINE.substring(2));
    memory[num][0] = R[0];
```

```
                memory[num][1] = R[1];

                memory[num][2] = R[2];

                memory[num][3] = R[3];

                continue;

            }


            if (IR[0] == 'C' && IR[1] == 'R') {

                LINE = new String(IR);

                num = Integer.parseInt(LINE.substring(2));

                if (memory[num][0] == R[0] && memory[num][1] == R[1]
&& memory[num][2] == R[2] && memory[num][3] == R[3]) {

                    T = 1;

                }

                continue;

            }


            if (IR[0] == 'B' && IR[1] == 'T') {

                if (T == 1) {

                    LINE = new String(IR);

                    num = Integer.parseInt(LINE.substring(2));

                    IC = num;
            T = 0;
        }
        continue;
    }

    if (IR[0] == 'G' && IR[1] == 'D') {
        this.masterMode(1);
        continue;
    }

    if (IR[0] == 'P' && IR[1] == 'D') {
        this.masterMode(2);
        continue;
    }

    if (IR[0] != 'H' && IR[3] != 'H') {
        continue;
    }
}

return;
```

```
    }
}

private void masterMode(int i) throws IOException {
    if (i == 1) {
        this.Read();
    } else if (i == 2) {
        this.Write();
    }

}

public void Write() throws IOException {
    String Line = new String(IR);
    int num = Integer.parseInt(Line.substring(2));
    String total = "";

    for(int i = 0; i < 10; ++i) {
        String t = new String(memory[num + i]);
        t = t.trim();
        if (!t.isEmpty()) {
            total = total.concat(t);
        }
    }

    System.out.println(total);
    this.fw.write("\n" + total);
    this.fw.flush();
}

public void Read() {
    String Line = new String(IR);
    int num = Integer.parseInt(Line.substring(2));

    try {
        Line = fread.readLine();
    } catch (IOException var4) {
        var4.printStackTrace();
    }

    buffer = Line.toCharArray();
    int i = 0;
```

```
        while(i < Line.length()) {
            memory[num][i % 4] = buffer[i];
            ++i;
            if (i % 4 == 0) {
                ++num;
            }
        }

    }

    public void print_memory() {
        for(int i = 0; i < 100; ++i) {
            System.out.println("memory[" + i + "] " + new
String(memory[i]));
        }

    }

    public void pcb(char[] buffer) {
        System.out.println(buffer);
    }

    public static void main(String[] arg) throws IOException {
        Main ph = new Main();
        ph.load();
    }
}
```

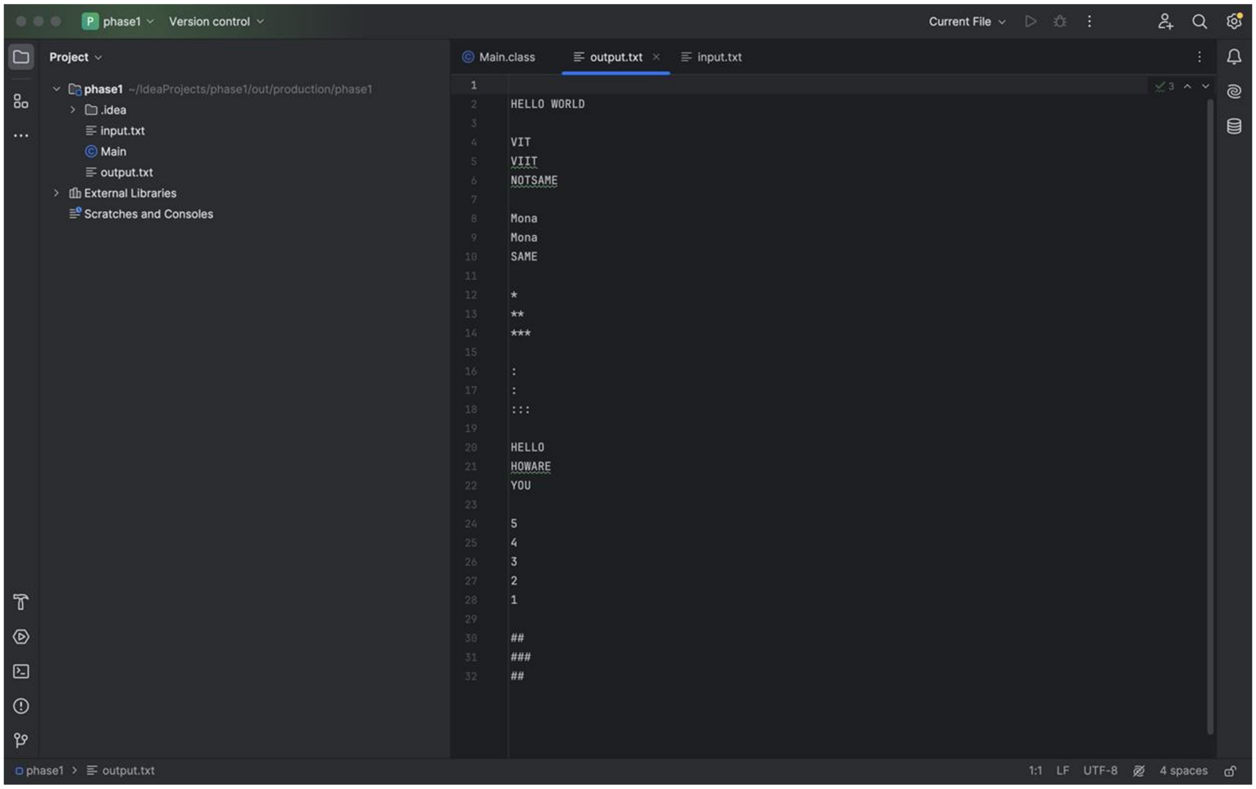
Input.text

```
$AMJ000100030015
GD10PD10H
$DTA
HELLO WORLD
$END0001
$AMJ000100130001
GD20GD30GD40GD50LR20CR30BT11PD50000HPD40H
$DTA
VIT
VIIT
SAME
NOT SAME
$END0001
```

```
$AMJ000100030001
GD20GD30GD40GD50PD20PD30LR20CR30BT11PD50000HPD40H
$DTA
Mona
Mona
SAME
NOT SAME
$END0001
$AMJ000100030003
GD20LR20SR45SR53SR57SR61SR65SR69PD40PD50PD60H
$DTA
*
$END0001
$AMJ000100030003
GD20LR20SR31SR41SR51SR52SR53PD30PD40PD50H
$DTA
:
$END0001
$AMJ000100030003
GD20GD30GD40PD20PD30PD40H
$DTA
```

```
HELLO
HOW ARE
YOU
$END0001
$AMJ000100030005
GD10GD20GD30GD40GD50PD10PD20PD30PD40PD50H
$DTA
5
4
3
2
1
$END0001
$AMJ000100030003
GD20LR20SR41SR43SR51SR52SR53SR61SR63PD40PD50PD60H
$DTA
#
$END0001
```

Output.txt



```

HELLO WORLD

VIT
VIIT
NOTSAME

Mona
Mona
SAME

*
**
***

:
:
:::

HELLO
HOWARE
YOU

5
4
3
2
1

##
###
##
```