Indian Institute of Information Technology Vadodara

CS266: Operating Systems Lab

Lab 6

Roll No. 201951105 **Name: Nishant Andoriya**

Problem 1

In this assignment, you have to implement the round-robin scheduling strategy to schedule the processes. You have to create two programs.

Program 1:

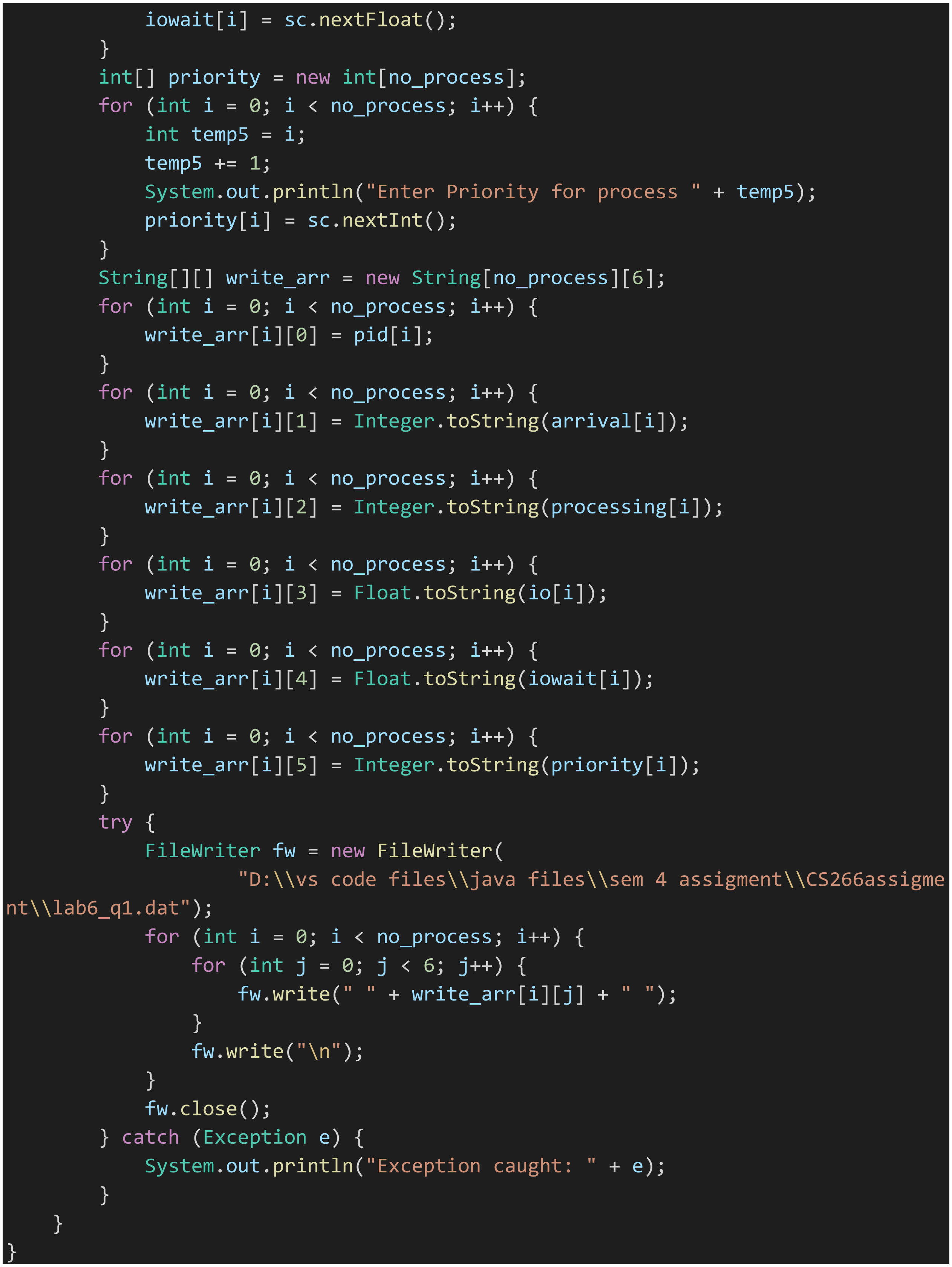
- This will create a file that will be input for the second program.
- Save this program as program1.
- You have to write the code such that it will ask the user to enter
  - Number of processes (maximum 5)
  - Process id (as String)
  - Process arrival of each process (0-5)
  - Processing time for each process (15-30) (integer)
  - The elapsed time between I/O interrupts (system calls) (1-3) (float),
  - Time spent in waiting and processing the I/O (1-5) (float)
  - Priority for each task (same or different) (1-5)
  - The created input file should be saved in test.dat.

The input file should look like this:

```
P1 0 20 1.5 2.0 1
P2 2 15 1.0 2.5 2
P3 4 18 2.3 3.0 1
P4 2 27 2.5 2.5 2
P5 5 24 1.0 4.0 3
```

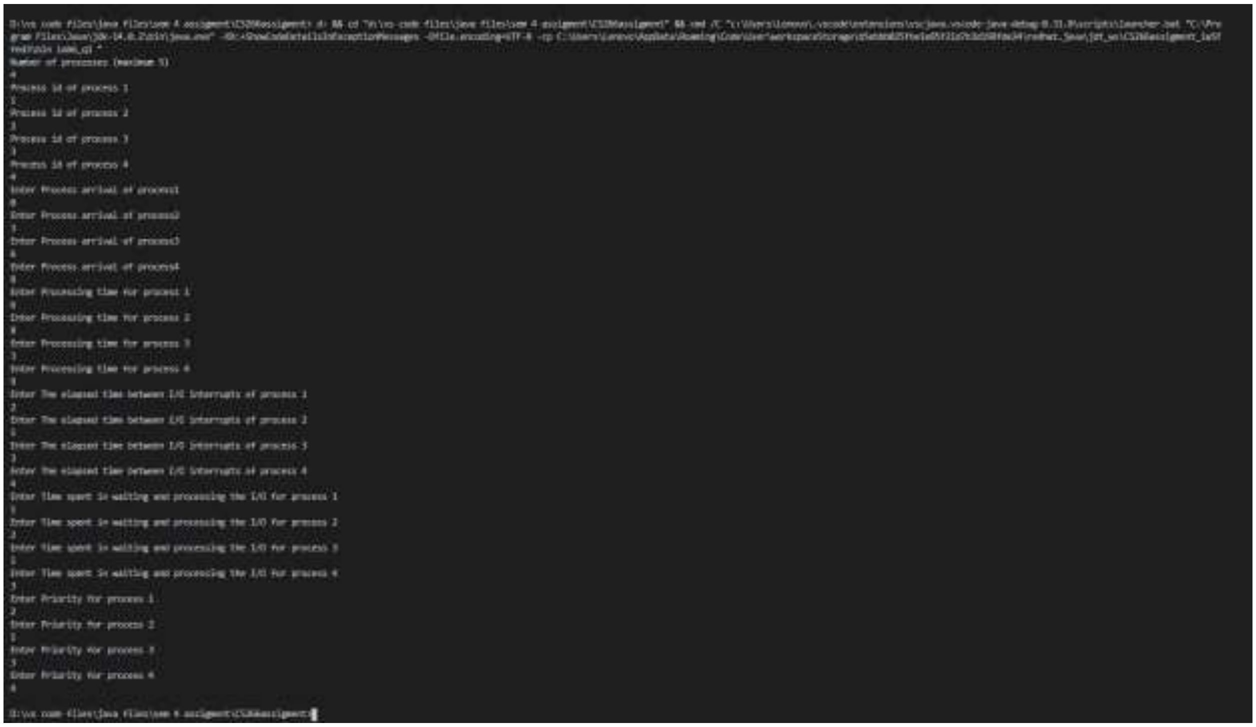
Code:-

```
import java.io.FileOutputStream;
import java.io.FileWriter;
import java.io.OutputStream;
import java.util.*;

public class lab6_q1 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int no_process = 0;
        try {
            OutputStream out = new FileOutputStream(
                    "D:\\vs code files\\java files\\sem 4 assigment\\CS266assigme
nt\\lab6_q1.dat");
        } catch (Exception e) {
            System.out.println("Exception caught " + e);
        }
        System.out.println("Number of processes (maximum 5)");
        no_process = sc.nextInt();
        String[] pid = new String[no_process];
        for (int i = 0; i < no_process; i++) {
            int temp = i;
            temp += 1;
            System.out.println("Process id of process " + temp);
            pid[i] = sc.next();
        }
        int[] arrival = new int[no_process];
        for (int i = 0; i < no_process; i++) {
            int temp1 = i;
            temp1 += 1;
            System.out.println("Enter Process arrival of process" + temp1);
            arrival[i] = sc.nextInt();
        }
        int[] processing = new int[no_process];
        for (int i = 0; i < no_process; i++) {
            int temp2 = i;
            temp2 += 1;
            System.out.println("Enter Processing time for process " + temp2);
            processing[i] = sc.nextInt();
        }
        float[] io = new float[no_process];
        for (int i = 0; i < no_process; i++) {
            int temp3 = i;
            temp3 += 1;
            System.out.println("Enter The elapsed time between I/O interrupts of
process " + temp3);
            io[i] = sc.nextFloat();
        }
        float[] iowait = new float[no_process];
        for (int i = 0; i < no_process; i++) {
            int temp4 = i;
            temp4 += 1;
            System.out.println("Enter Time spent in waiting and processing the I/
O for process " + temp4);
```



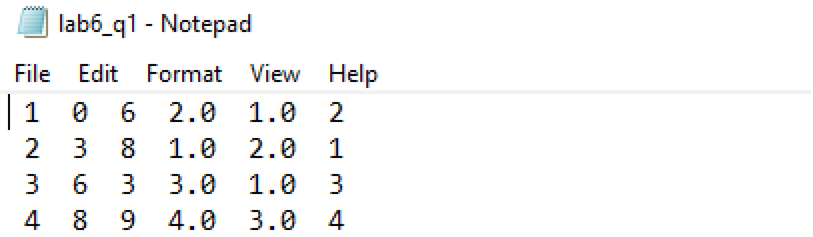
```
            iowait[i] = sc.nextFloat();
        }
        int[] priority = new int[no_process];
        for (int i = 0; i < no_process; i++) {
            int temp5 = i;
            temp5 += 1;
            System.out.println("Enter Priority for process " + temp5);
            priority[i] = sc.nextInt();
        }
        String[][] write_arr = new String[no_process][6];
        for (int i = 0; i < no_process; i++) {
            write_arr[i][0] = pid[i];
        }
        for (int i = 0; i < no_process; i++) {
            write_arr[i][1] = Integer.toString(arrival[i]);
        }
        for (int i = 0; i < no_process; i++) {
            write_arr[i][2] = Integer.toString(processing[i]);
        }
        for (int i = 0; i < no_process; i++) {
            write_arr[i][3] = Float.toString(io[i]);
        }
        for (int i = 0; i < no_process; i++) {
            write_arr[i][4] = Float.toString(iowait[i]);
        }
        for (int i = 0; i < no_process; i++) {
            write_arr[i][5] = Integer.toString(priority[i]);
        }
        try {
            FileWriter fw = new FileWriter(
                    "D:\\vs code files\\java files\\sem 4 assigment\\CS266assignment\\lab6_q1.dat");
            for (int i = 0; i < no_process; i++) {
                for (int j = 0; j < 6; j++) {
                    fw.write(" " + write_arr[i][j] + " ");
                }
                fw.write("\n");
            }
            fw.close();
        } catch (Exception e) {
            System.out.println("Exception caught: " + e);
        }
    }
}
```

Input for this program:-



```
D:\vs code files\java files\sem 4 assigment\CS206assigment> d: && cd "d:\vs code files\java files\sem 4 assigment\CS206assigment" && cmd /C "c:\Users\Lenovo\.vscode\extensions\vscjava.vscode-java-debug-0.31.0\scripts\launcher.bat "C:\Program Files\Java\jdk-14.0.2\bin\java.exe" -XX:+ShowCodeDetailsInExceptionMessages -Dfile.encoding=UTF-8 -cp C:\Users\Lenovo\AppData\Roaming\Code\User\workspaceStorage\[illegible]\redhat.java\jdt_ws\CS206assigment_[illegible] lab6_q1 "
Number of processes (maximum 5)
4
Process id of process 1
1
Process id of process 2
2
Process id of process 3
3
Process id of process 4
4
Enter Process arrival of process1
0
Enter Process arrival of process2
3
Enter Process arrival of process3
6
Enter Process arrival of process4
8
Enter Processing time for process 1
6
Enter Processing time for process 2
8
Enter Processing time for process 3
3
Enter Processing time for process 4
9
Enter The elapsed time between I/O interrupts of process 1
2
Enter The elapsed time between I/O interrupts of process 2
1
Enter The elapsed time between I/O interrupts of process 3
3
Enter The elapsed time between I/O interrupts of process 4
4
Enter Time spent in waiting and processing the I/O for process 1
1
Enter Time spent in waiting and processing the I/O for process 2
2
Enter Time spent in waiting and processing the I/O for process 3
1
Enter Time spent in waiting and processing the I/O for process 4
3
Enter Priority for process 1
2
Enter Priority for process 2
1
Enter Priority for process 3
3
Enter Priority for process 4
4

D:\vs code files\java files\sem 4 assigment\CS206assigment>
```

Output:-



```
1  0  6  2.0  1.0  2
2  3  8  1.0  2.0  1
3  6  3  3.0  1.0  3
4  8  9  4.0  3.0  4
```

Program 2:

- Save this program as program2

- Second program will read the file (test.dat file) that consists of a list of processes with the desired parameters for each process.

- Bonus marks (+1) if you pass the test.dat file as a command-line argument, otherwise you can read the test.dat file in your second program (But NO bonus marks for this).

- This program will implement the RR algorithm.

- The program will simulate the execution of the processes and provide the following: – turnaround time of each process,

– waiting time of each process,

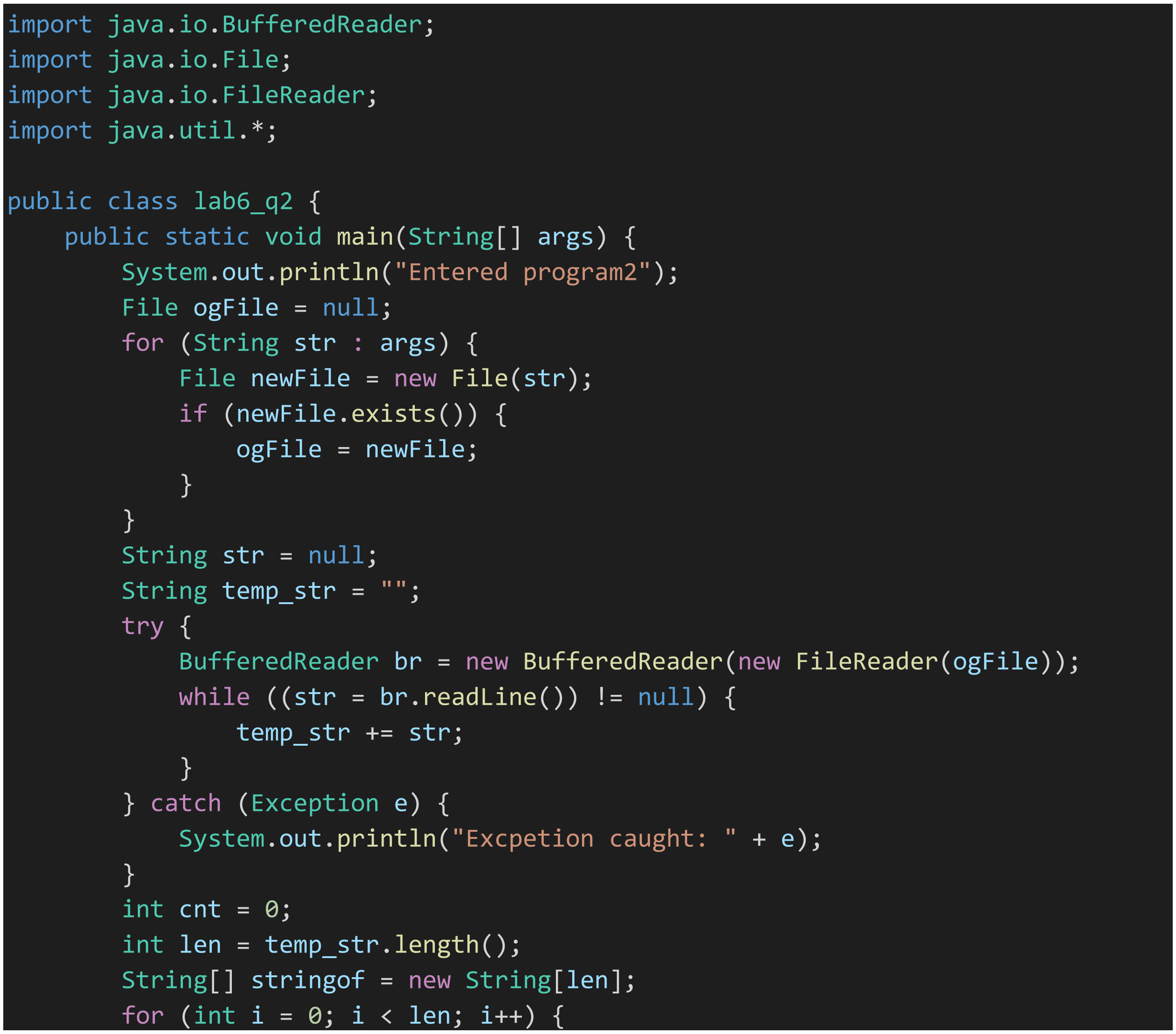
– average turnaround time,

– average waiting time.

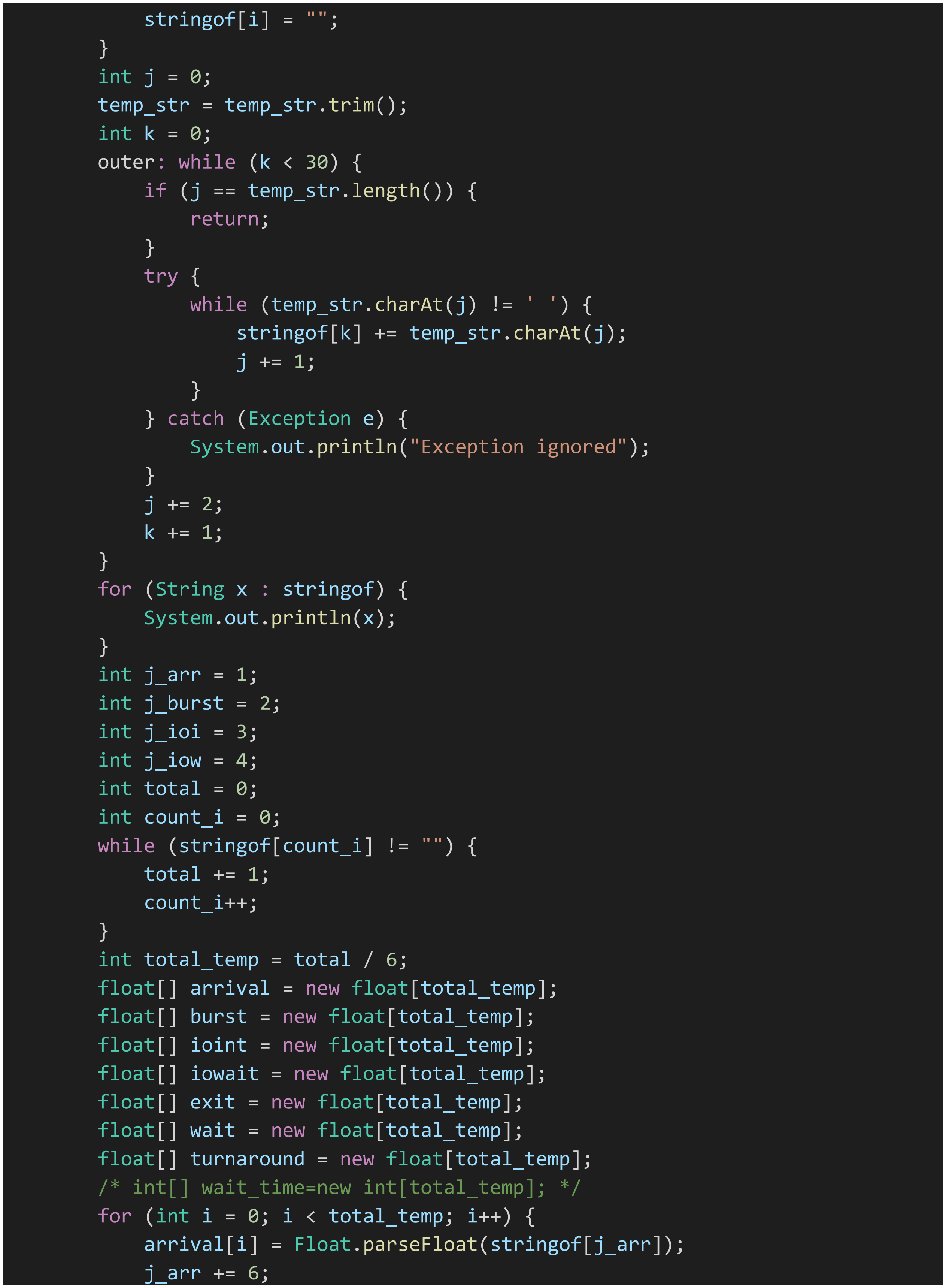
- The time slice can be varied from 1 to 10sec. Then plot the graph that will show how

– the average turnaround time for processes varies with time slice/quantum. – how the average waiting time for processes varies with time slice/quantum.

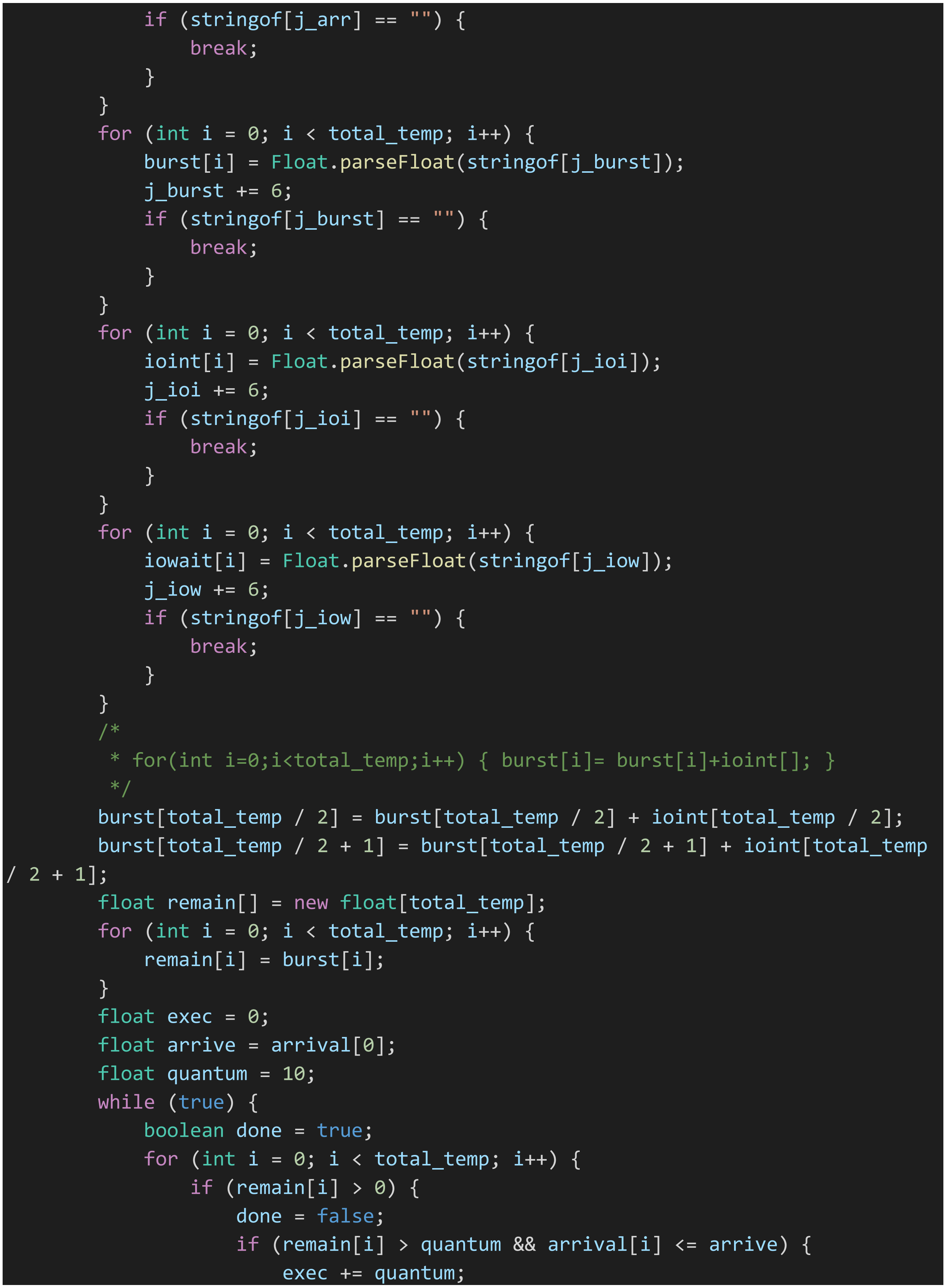
Code:-



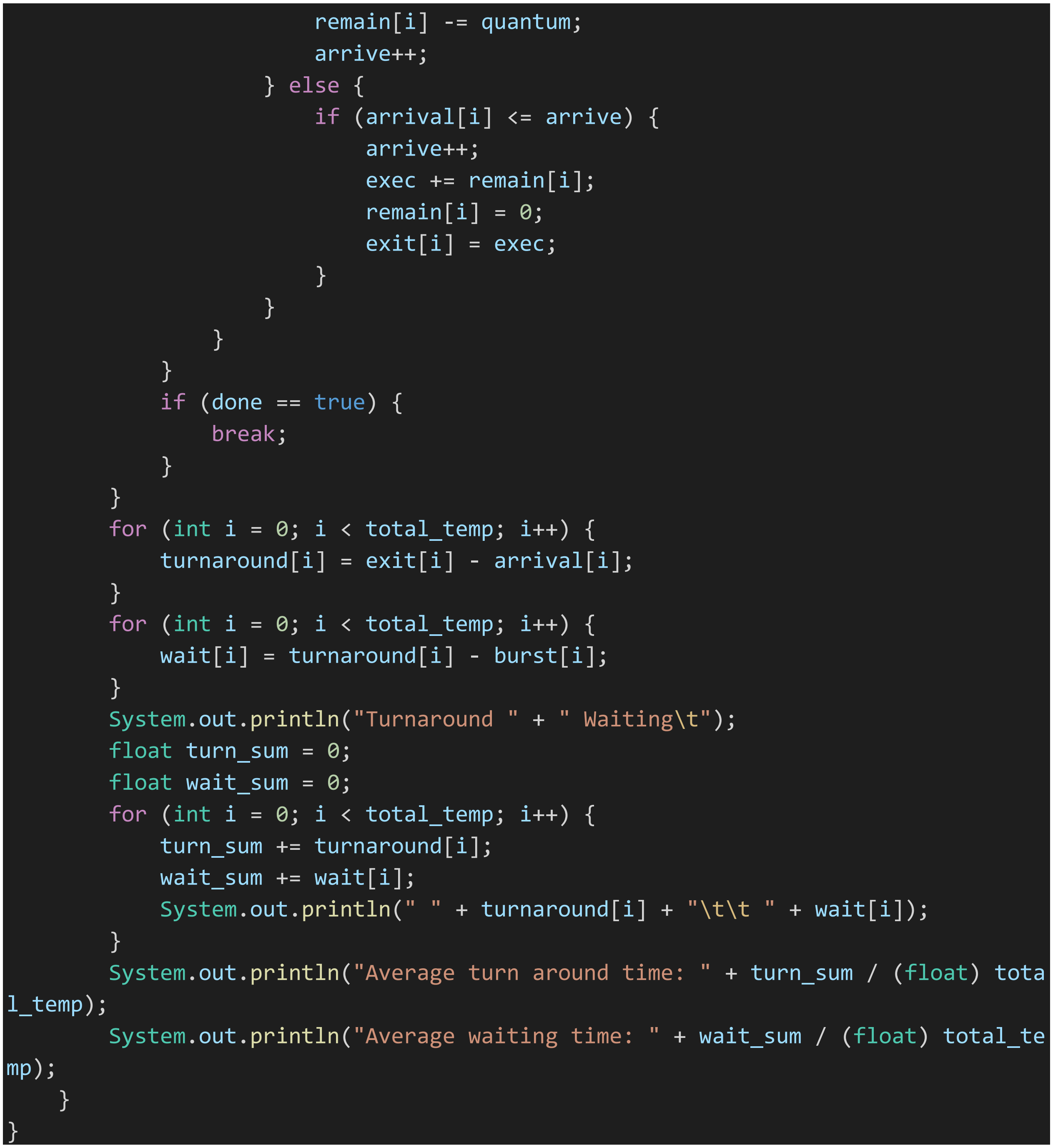
```java
import java.io.BufferedReader;
import java.io.File;
import java.io.FileReader;
import java.util.*;

public class lab6_q2 {
    public static void main(String[] args) {
        System.out.println("Entered program2");
        File ogFile = null;
        for (String str : args) {
            File newFile = new File(str);
            if (newFile.exists()) {
                ogFile = newFile;
            }
        }
        String str = null;
        String temp_str = "";
        try {
            BufferedReader br = new BufferedReader(new FileReader(ogFile));
            while ((str = br.readLine()) != null) {
                temp_str += str;
            }
        } catch (Exception e) {
            System.out.println("Excpetion caught: " + e);
        }
        int cnt = 0;
        int len = temp_str.length();
        String[] stringof = new String[len];
        for (int i = 0; i < len; i++) {
```



```
            stringof[i] = "";
        }
        int j = 0;
        temp_str = temp_str.trim();
        int k = 0;
        outer: while (k < 30) {
            if (j == temp_str.length()) {
                return;
            }
            try {
                while (temp_str.charAt(j) != ' ') {
                    stringof[k] += temp_str.charAt(j);
                    j += 1;
                }
            } catch (Exception e) {
                System.out.println("Exception ignored");
            }
            j += 2;
            k += 1;
        }
        for (String x : stringof) {
            System.out.println(x);
        }
        int j_arr = 1;
        int j_burst = 2;
        int j_ioi = 3;
        int j_iow = 4;
        int total = 0;
        int count_i = 0;
        while (stringof[count_i] != "") {
            total += 1;
            count_i++;
        }
        int total_temp = total / 6;
        float[] arrival = new float[total_temp];
        float[] burst = new float[total_temp];
        float[] ioint = new float[total_temp];
        float[] iowait = new float[total_temp];
        float[] exit = new float[total_temp];
        float[] wait = new float[total_temp];
        float[] turnaround = new float[total_temp];
        /* int[] wait_time=new int[total_temp]; */
        for (int i = 0; i < total_temp; i++) {
            arrival[i] = Float.parseFloat(stringof[j_arr]);
            j_arr += 6;
```

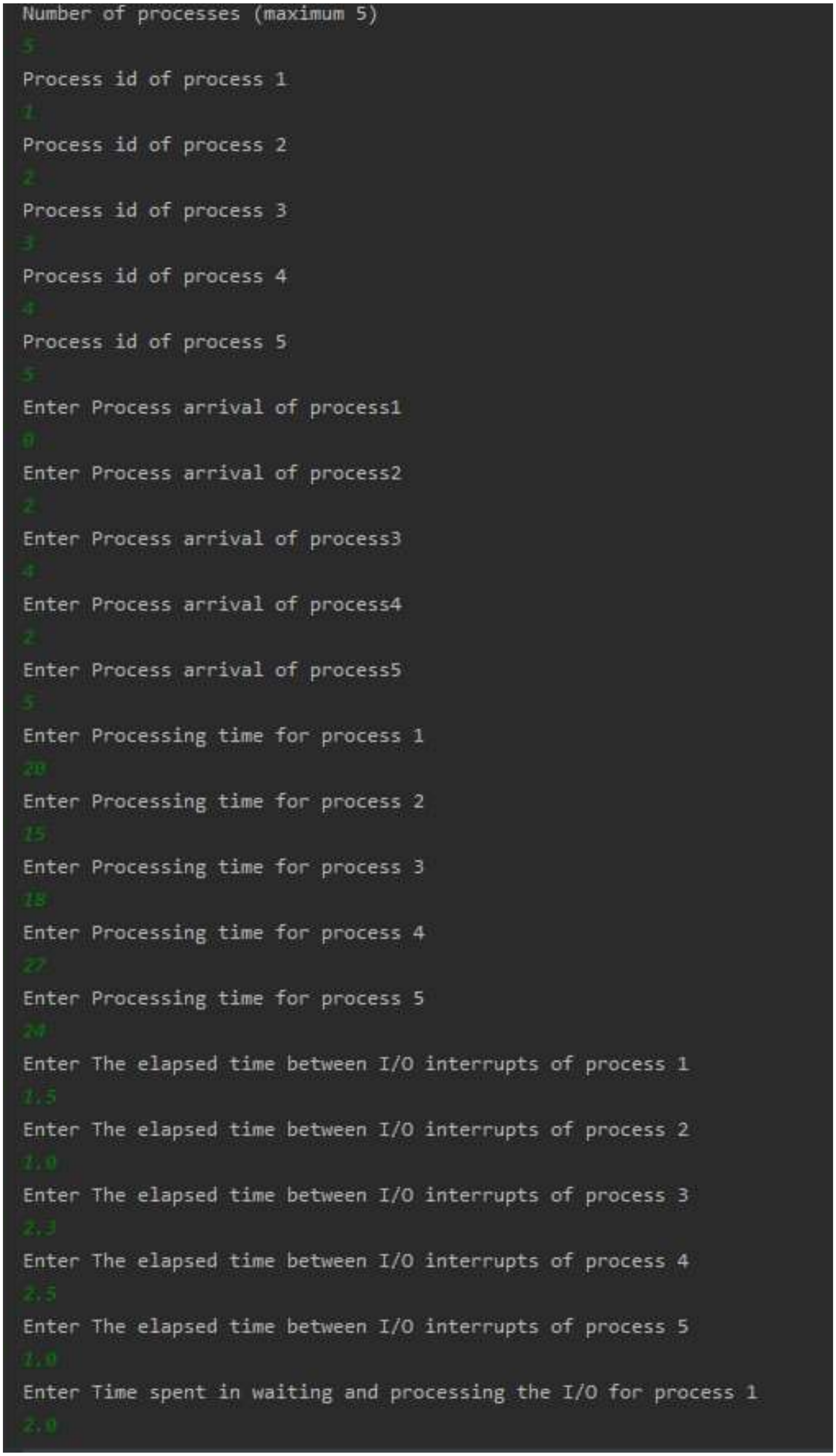


```
            if (stringof[j_arr] == "") {
                break;
            }
        }
        for (int i = 0; i < total_temp; i++) {
            burst[i] = Float.parseFloat(stringof[j_burst]);
            j_burst += 6;
            if (stringof[j_burst] == "") {
                break;
            }
        }
        for (int i = 0; i < total_temp; i++) {
            ioint[i] = Float.parseFloat(stringof[j_ioi]);
            j_ioi += 6;
            if (stringof[j_ioi] == "") {
                break;
            }
        }
        for (int i = 0; i < total_temp; i++) {
            iowait[i] = Float.parseFloat(stringof[j_iow]);
            j_iow += 6;
            if (stringof[j_iow] == "") {
                break;
            }
        }
        /*
         * for(int i=0;i<total_temp;i++) { burst[i]= burst[i]+ioint[]; }
         */
        burst[total_temp / 2] = burst[total_temp / 2] + ioint[total_temp / 2];
        burst[total_temp / 2 + 1] = burst[total_temp / 2 + 1] + ioint[total_temp
/ 2 + 1];
        float remain[] = new float[total_temp];
        for (int i = 0; i < total_temp; i++) {
            remain[i] = burst[i];
        }
        float exec = 0;
        float arrive = arrival[0];
        float quantum = 10;
        while (true) {
            boolean done = true;
            for (int i = 0; i < total_temp; i++) {
                if (remain[i] > 0) {
                    done = false;
                    if (remain[i] > quantum && arrival[i] <= arrive) {
                        exec += quantum;
```

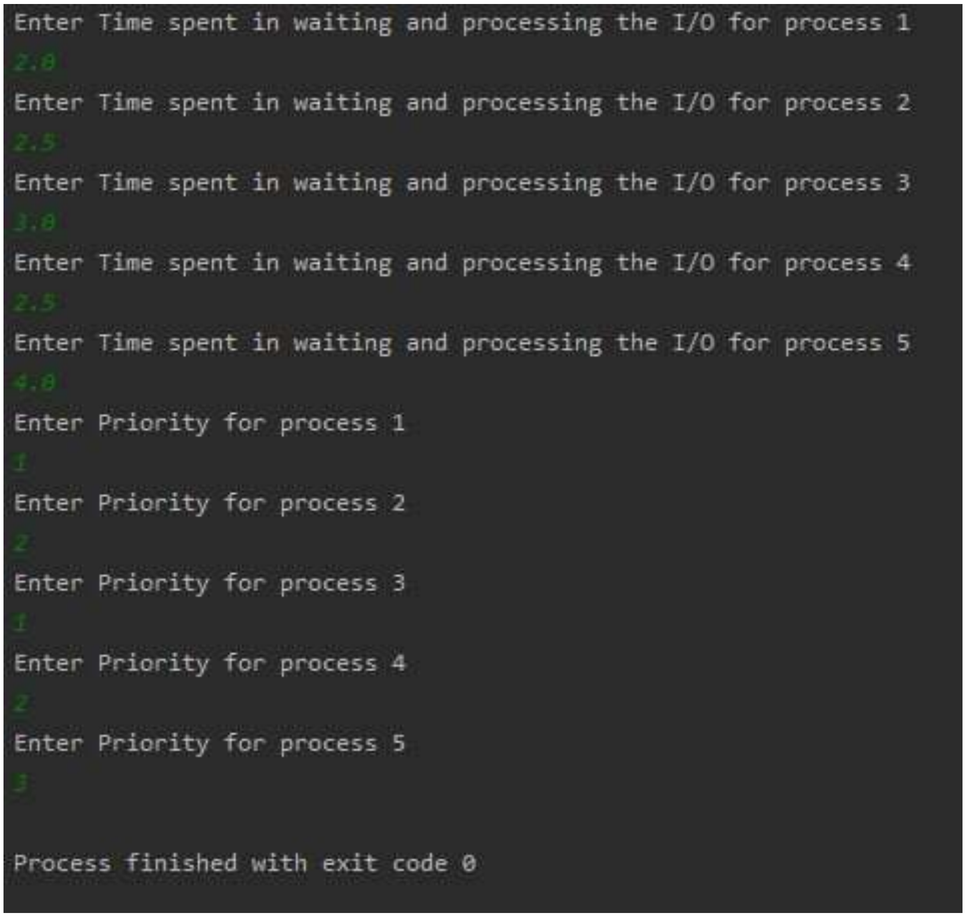


```
                        remain[i] -= quantum;
                        arrive++;
                    } else {
                        if (arrival[i] <= arrive) {
                            arrive++;
                            exec += remain[i];
                            remain[i] = 0;
                            exit[i] = exec;
                        }
                    }
                }
            }
            if (done == true) {
                break;
            }
        }
        for (int i = 0; i < total_temp; i++) {
            turnaround[i] = exit[i] - arrival[i];
        }
        for (int i = 0; i < total_temp; i++) {
            wait[i] = turnaround[i] - burst[i];
        }
        System.out.println("Turnaround " + " Waiting\t");
        float turn_sum = 0;
        float wait_sum = 0;
        for (int i = 0; i < total_temp; i++) {
            turn_sum += turnaround[i];
            wait_sum += wait[i];
            System.out.println(" " + turnaround[i] + "\t\t " + wait[i]);
        }
        System.out.println("Average turn around time: " + turn_sum / (float) tota
l_temp);
        System.out.println("Average waiting time: " + wait_sum / (float) total_te
mp);
    }
}
```

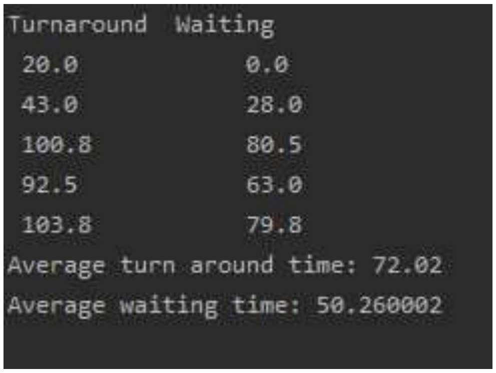
Output:-



```
Number of processes (maximum 5)
5
Process id of process 1
1
Process id of process 2
2
Process id of process 3
3
Process id of process 4
4
Process id of process 5
5
Enter Process arrival of process1
0
Enter Process arrival of process2
2
Enter Process arrival of process3
4
Enter Process arrival of process4
2
Enter Process arrival of process5
5
Enter Processing time for process 1
20
Enter Processing time for process 2
15
Enter Processing time for process 3
18
Enter Processing time for process 4
27
Enter Processing time for process 5
24
Enter The elapsed time between I/O interrupts of process 1
1.5
Enter The elapsed time between I/O interrupts of process 2
1.0
Enter The elapsed time between I/O interrupts of process 3
2.3
Enter The elapsed time between I/O interrupts of process 4
2.5
Enter The elapsed time between I/O interrupts of process 5
1.0
Enter Time spent in waiting and processing the I/O for process 1
2.0
```



```
Enter Time spent in waiting and processing the I/O for process 1
2.0
Enter Time spent in waiting and processing the I/O for process 2
2.5
Enter Time spent in waiting and processing the I/O for process 3
3.0
Enter Time spent in waiting and processing the I/O for process 4
2.5
Enter Time spent in waiting and processing the I/O for process 5
4.0
Enter Priority for process 1
1
Enter Priority for process 2
2
Enter Priority for process 3
1
Enter Priority for process 4
2
Enter Priority for process 5
3


Process finished with exit code 0
```



```
Turnaround  Waiting
 20.0          0.0
 43.0          28.0
 100.8         80.5
 92.5          63.0
 103.8         79.8
Average turn around time: 72.02
Average waiting time: 50.260002
```