OPERATING SYSTEM JOURNAL

Name: **Rashi Sawardekar** Roll No: **31010924093** SYIT- B

all files

PRACTICAL 1

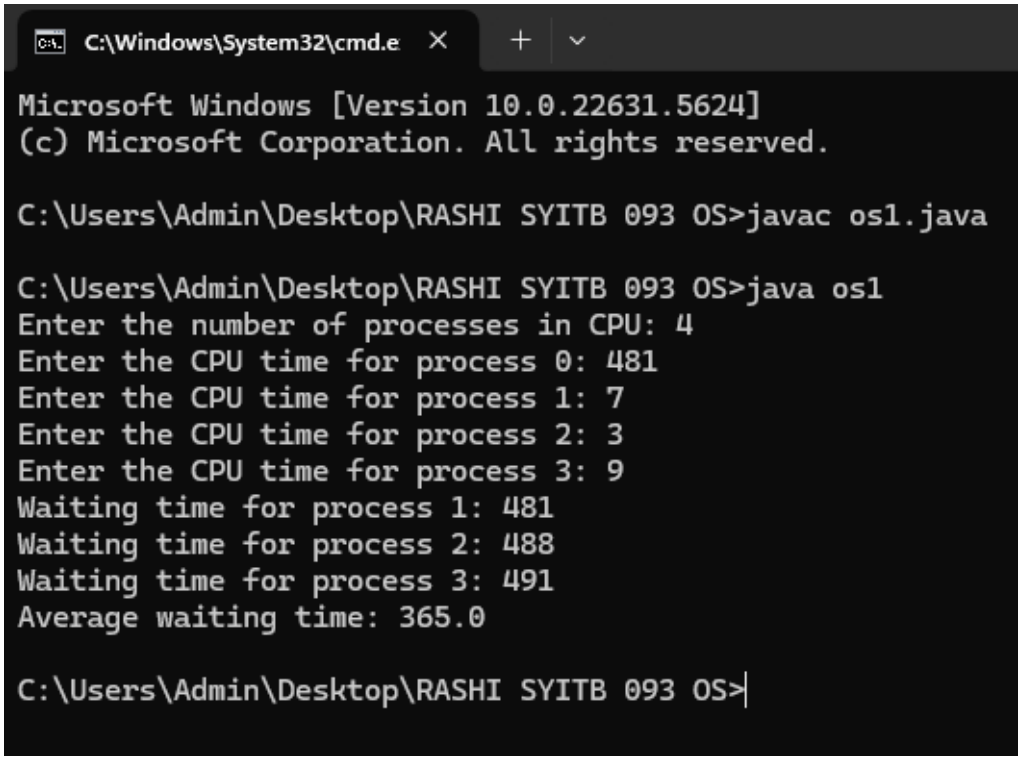
Q1. To implement Scheduling Algorithm: FCFS

Source Code:

```
import java.util.*;

class Proc {
    Proc() {
        Scanner sc = new Scanner(System.in);

        int n, sum = 0;
        float avg;
        int[] p = new int[20];
        int[] wt = new int[20];

        System.out.print("Enter the number of processes in CPU: ");
        n = sc.nextInt();

        for (int i = 0; i < n; i++) {
            System.out.print("Enter the CPU time for process " + i + ": ");
            p[i] = sc.nextInt();
        }
```

```
        wt[0] = 0;

        for (int i = 1; i < n; i++) {
            wt[i] = wt[i - 1] + p[i - 1];
            System.out.println("Waiting time for process " + i + ": " + wt[i]);
            sum += wt[i];
        }

        avg = (float) sum / n;
        System.out.println("Average waiting time: " + avg);
    }
}

public class FCFS {
    public static void main(String[] args) {
        new Proc();
    }
}
```

Output:


C:\Windows\System32\cmd.e
Microsoft Windows [Version 10.0.22631.5624]
(c) Microsoft Corporation. All rights reserved.
C:\Users\Admin\Desktop\RASHI SYITB 093 OS>javac os1.java
C:\Users\Admin\Desktop\RASHI SYITB 093 OS>java os1
Enter the number of processes in CPU: 4
Enter the CPU time for process 0: 481
Enter the CPU time for process 1: 7
Enter the CPU time for process 2: 3
Enter the CPU time for process 3: 9
Waiting time for process 1: 481
Waiting time for process 2: 488
Waiting time for process 3: 491
Average waiting time: 365.0
C:\Users\Admin\Desktop\RASHI SYITB 093 OS>

PRACTICAL 2

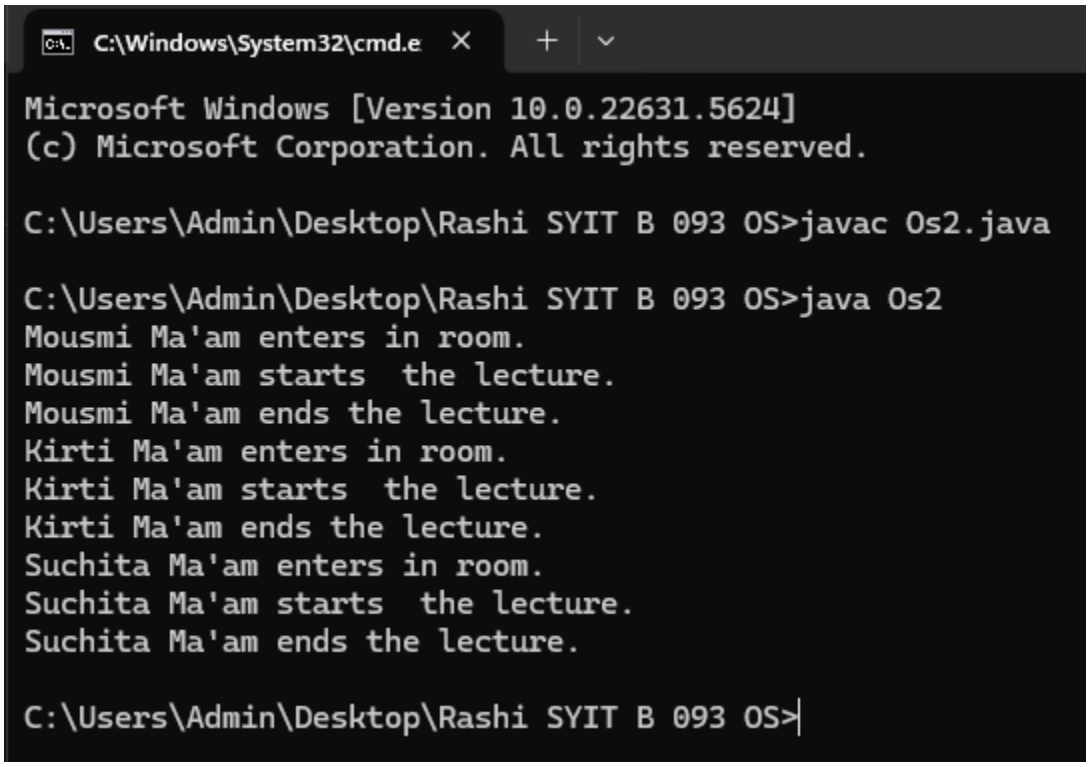
Q1.Demonstrate the Concept of Synchronized Method

Source Code:

```
class Room {
   public synchronized void lectures_Taken(String name)throws Exception
{
       System.out.println(name + " enters in room.");
       Thread.sleep(1000);
       System.out.println(name + " starts  the lecture.");
       Thread.sleep(1000);
       System.out.println(name + " ends the lecture.");
       Thread.sleep(1000);

   }
}

class Lecture implements Runnable
{
     String name;
     Room r;
     Thread t;
Lecture(String tname,Room r)
{
   name=tname;
   t=new Thread(this,tname);
   this.r=r;
```

```
    t.start();
}
public void run()
{
    try
    {
       r.lectures_Taken(name);
    }catch(Exception e)
     {
                System.out.println(e);
          }
    }
}

class Os2 {
    public static void main(String args[]) {
        Room r1 = new Room();
        Lecture l1 = new Lecture("Mousmi Ma'am", r1);
        Lecture l2 = new Lecture("Kirti Ma'am", r1);
        Lecture l3 = new Lecture("Suchita Ma'am", r1);
    }
}
```

Output:


C:\Windows\System32\cmd.e
Microsoft Windows [Version 10.0.22631.5624]
(c) Microsoft Corporation. All rights reserved.
C:\Users\Admin\Desktop\Rashi SYIT B 093 OS>javac Os2.java
C:\Users\Admin\Desktop\Rashi SYIT B 093 OS>java Os2
Mousmi Ma'am enters in room.
Mousmi Ma'am starts the lecture.
Mousmi Ma'am ends the lecture.
Kirti Ma'am enters in room.
Kirti Ma'am starts the lecture.
Kirti Ma'am ends the lecture.
Suchita Ma'am enters in room.
Suchita Ma'am starts the lecture.
Suchita Ma'am ends the lecture.
C:\Users\Admin\Desktop\Rashi SYIT B 093 OS>

PRACTICAL 3

Q1. To implement Producer Consumer Algorithm

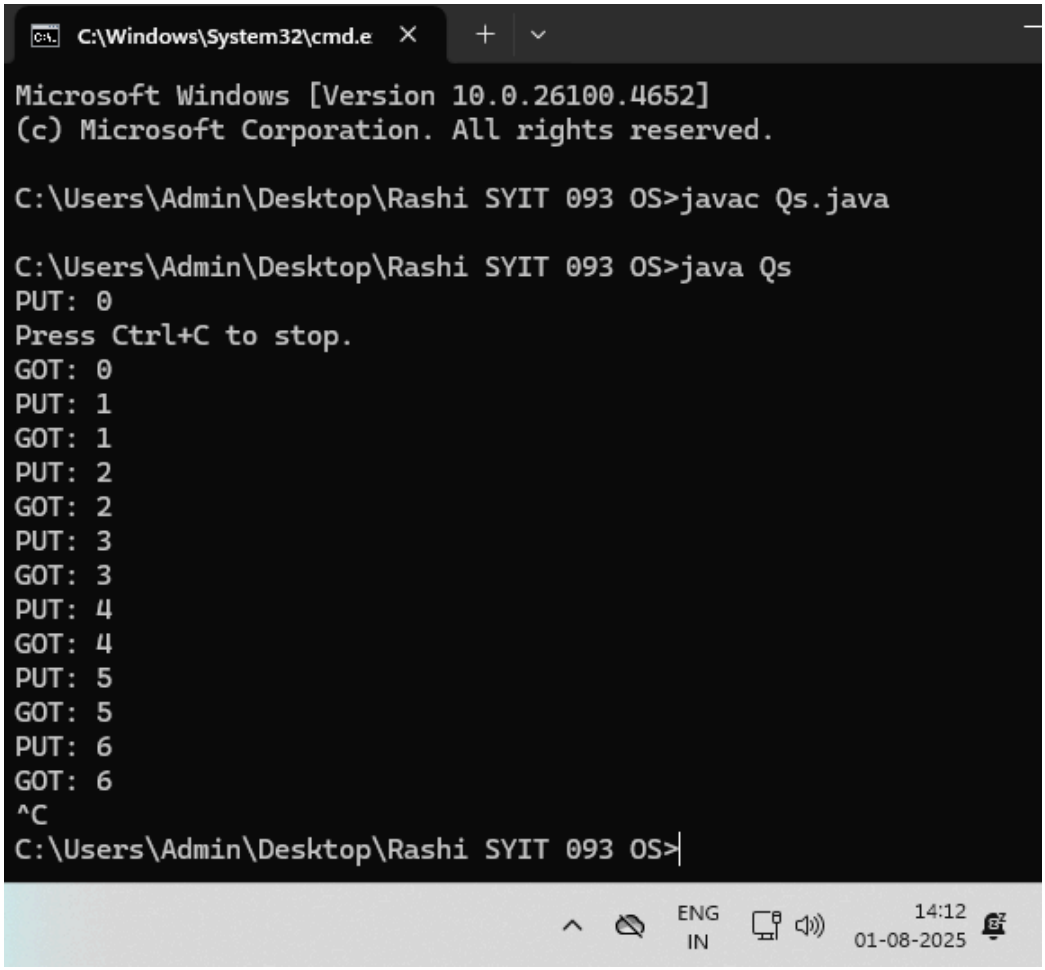
Source Code:

```
class Q {
int n;
boolean valueset = false;
synchronized int get() {
while (!valueset) {
try {
wait();
} catch (InterruptedException e) {
System.out.println("Exception Caught");
}
}
System.out.println("GOT: " + n);
valueset = false;
notify();
return n;
}
synchronized void put(int x) {
while (valueset) {
try {

wait();
} catch (InterruptedException e) {
System.out.println("Exception Caught");
}
}
this.n = x;
valueset = true;
```

```
System.out.println("PUT: " + n);
notify();
}
}
class Producer implements Runnable {
Q q;
Producer(Q q) {
this.q = q;
new Thread(this, "Producer").start();
}
public void run() {
int i = 0;
while (true) {
q.put(i++);
try {
Thread.sleep(50);
} catch (InterruptedException e) {
System.out.println("Producer interrupted");
}
}
}
}
class Consumer implements Runnable {
Q q;

Consumer(Q q) {
this.q = q;
new Thread(this, "Consumer").start();
}
public void run() {
while (true) {
int val = q.get();
try {
Thread.sleep(50);
} catch (InterruptedException e) {
```

```
System.out.println("Consumer interrupted");
}
}
}
}
public class Qs {
public static void main(String[] args) {
Q q = new Q();
new Producer(q);
new Consumer(q);
System.out.println("Press Ctrl+C to stop.");
}
}
```

Output:



PRACTICAL 4

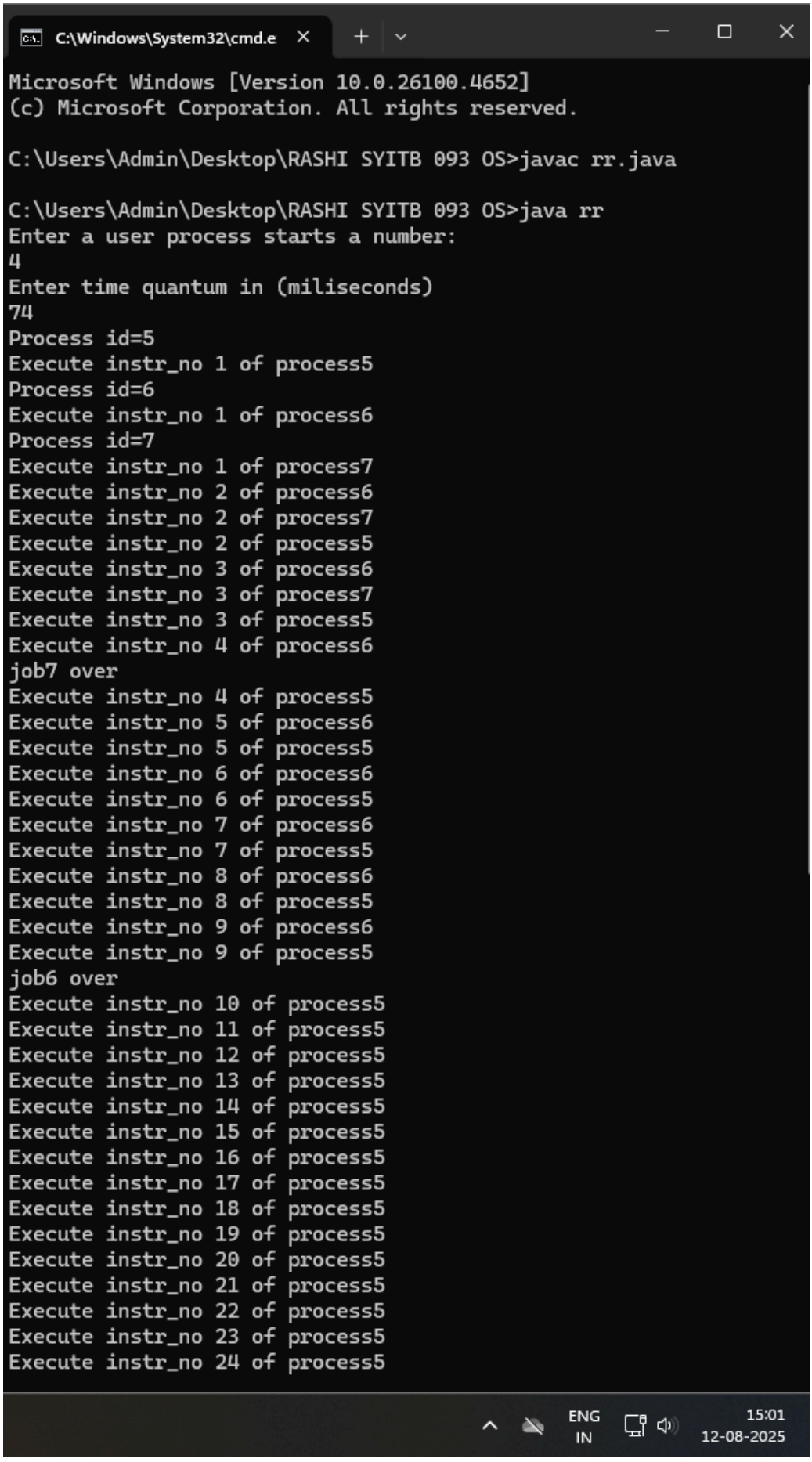
Q1. To implement Round Robin Algorithm.

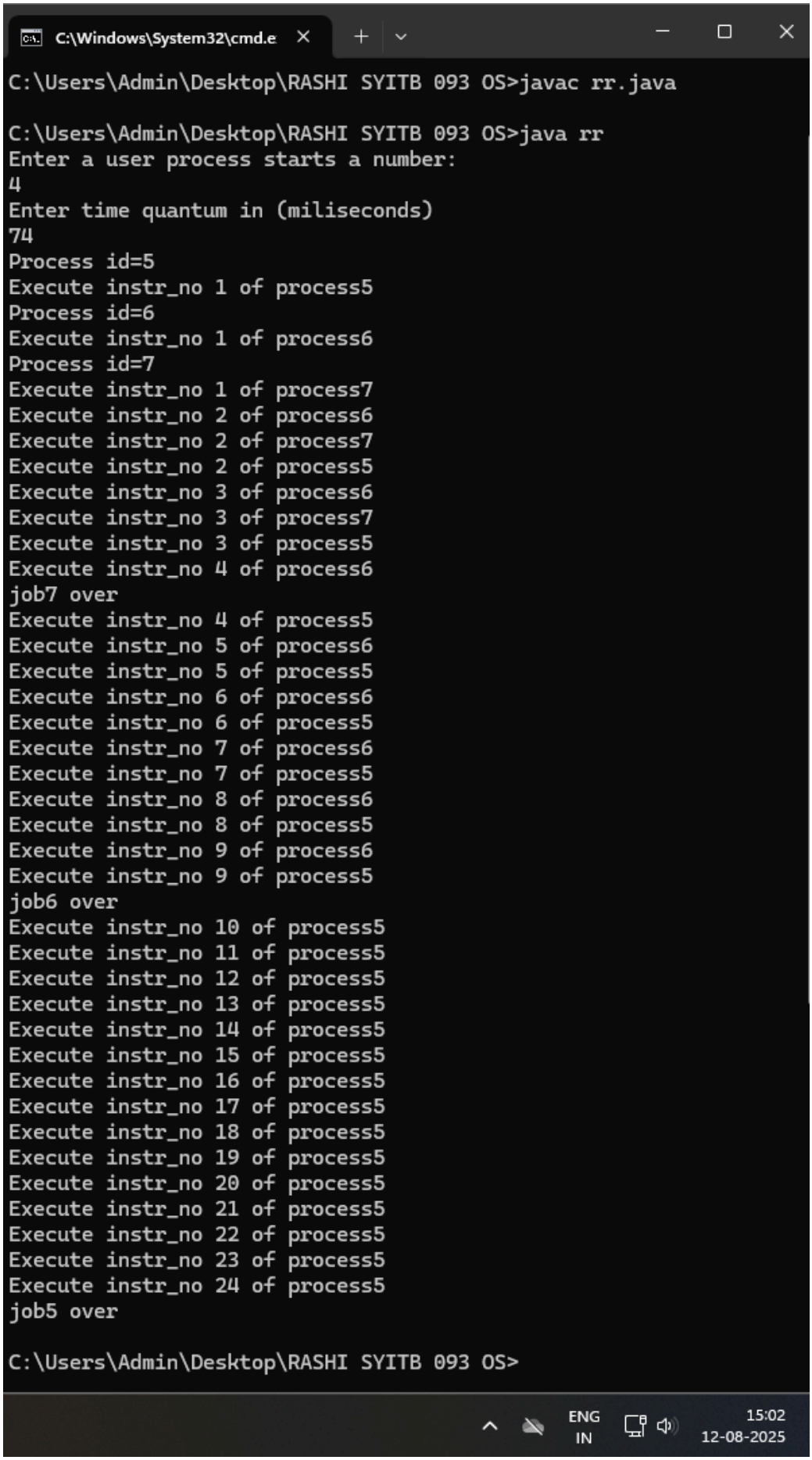
Source Code:

```
import java.util.*;
class job implements Runnable
{
int process_id,no_of_instr,time_quantum;
Thread t;
job(int pid,int instr,int tq)
{
process_id=pid;
no_of_instr=instr;
time_quantum=tq;
t=new Thread(this);
t.start();
}
public void run()
{
try
{
System.out.println("Process id="+process_id);
for(int i=1;i<=no_of_instr;i++)
{
System.out.println("Execute instr_no "+i+" of process"+process_id);
Thread.sleep(time_quantum);
}
System.out.println("job"+process_id+" over");
}
catch(Exception e)
```

```
{
System.out.println("job has interrupted");
}
}
}
class rr
{
public static void main(String args[])
{
try
{
int process_id,time_quantum;
Scanner sc=new Scanner (System.in);
System.out.println("Enter a user process starts a number:");
process_id=sc.nextInt();
System.out.println("Enter time quantum in (miliseconds)");
time_quantum=sc.nextInt();
job j1=new job(++process_id,24,time_quantum);
job j2=new job(++process_id,9,time_quantum);
job j3=new job(++process_id,3,time_quantum);
}
catch(Exception e)
{
System.out.println("process is failed");
System.out.println("please contact Admin");
}
}
}
```

Output:


C:\Windows\System32\cmd.e
Microsoft Windows [Version 10.0.26100.4652]
(c) Microsoft Corporation. All rights reserved.
C:\Users\Admin\Desktop\RASHI SYITB 093 OS>javac rr.java
C:\Users\Admin\Desktop\RASHI SYITB 093 OS>java rr
Enter a user process starts a number:
4
Enter time quantum in (miliseconds)
74
Process id=5
Execute instr_no 1 of process5
Process id=6
Execute instr_no 1 of process6
Process id=7
Execute instr_no 1 of process7
Execute instr_no 2 of process6
Execute instr_no 2 of process7
Execute instr_no 2 of process5
Execute instr_no 3 of process6
Execute instr_no 3 of process7
Execute instr_no 3 of process5
Execute instr_no 4 of process6
job7 over
Execute instr_no 4 of process5
Execute instr_no 5 of process6
Execute instr_no 5 of process5
Execute instr_no 6 of process6
Execute instr_no 6 of process5
Execute instr_no 7 of process6
Execute instr_no 7 of process5
Execute instr_no 8 of process6
Execute instr_no 8 of process5
Execute instr_no 9 of process6
Execute instr_no 9 of process5
job6 over
Execute instr_no 10 of process5
Execute instr_no 11 of process5
Execute instr_no 12 of process5
Execute instr_no 13 of process5
Execute instr_no 14 of process5
Execute instr_no 15 of process5
Execute instr_no 16 of process5
Execute instr_no 17 of process5
Execute instr_no 18 of process5
Execute instr_no 19 of process5
Execute instr_no 20 of process5
Execute instr_no 21 of process5
Execute instr_no 22 of process5
Execute instr_no 23 of process5
Execute instr_no 24 of process5
ENG
IN
15:01
12-08-2025

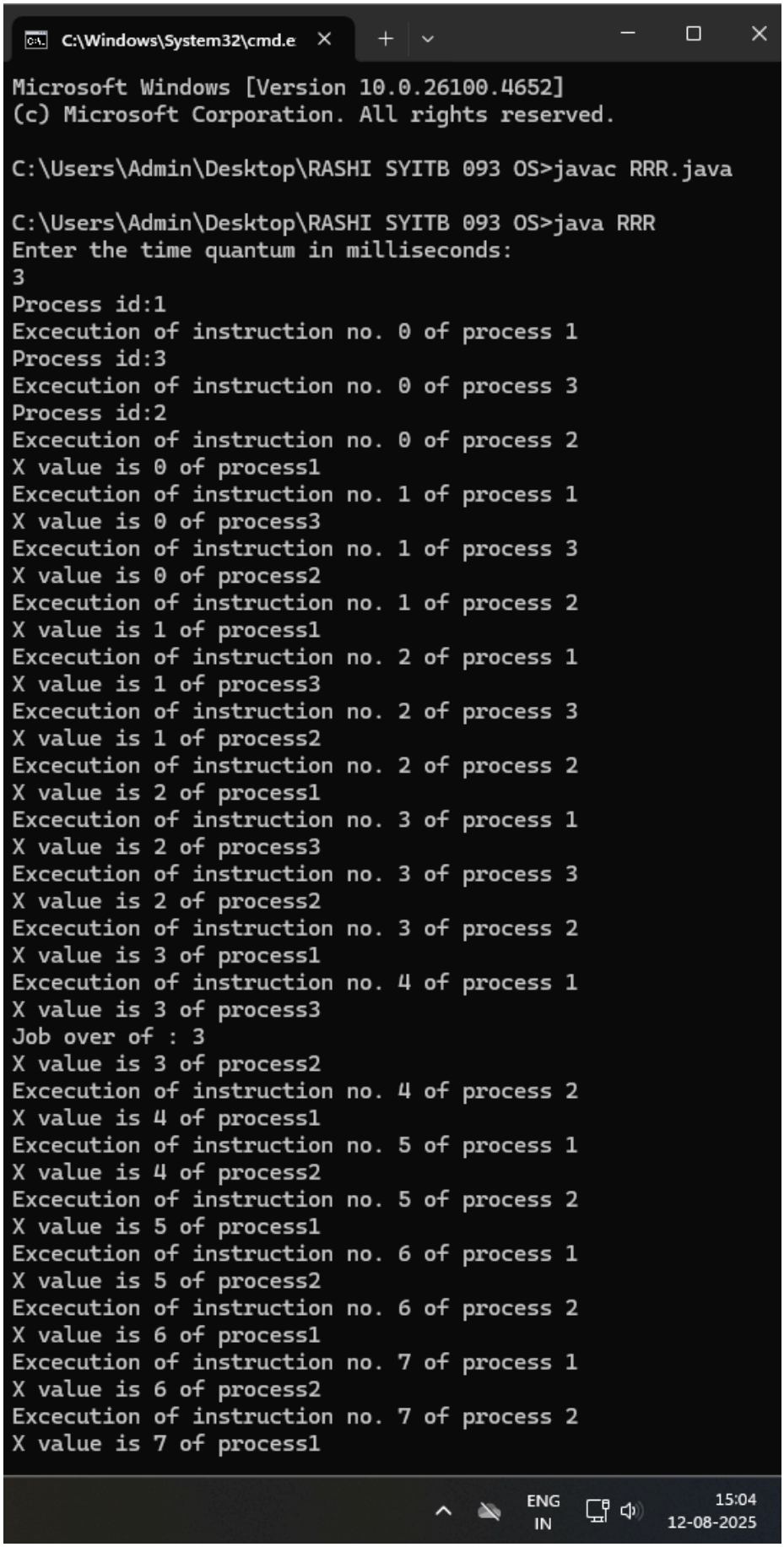

C:\Windows\System32\cmd.e
C:\Users\Admin\Desktop\RASHI SYITB 093 OS>javac rr.java
C:\Users\Admin\Desktop\RASHI SYITB 093 OS>java rr
Enter a user process starts a number:
4
Enter time quantum in (miliseconds)
74
Process id=5
Execute instr_no 1 of process5
Process id=6
Execute instr_no 1 of process6
Process id=7
Execute instr_no 1 of process7
Execute instr_no 2 of process6
Execute instr_no 2 of process7
Execute instr_no 2 of process5
Execute instr_no 3 of process6
Execute instr_no 3 of process7
Execute instr_no 3 of process5
Execute instr_no 4 of process6
job7 over
Execute instr_no 4 of process5
Execute instr_no 5 of process6
Execute instr_no 5 of process5
Execute instr_no 6 of process6
Execute instr_no 6 of process5
Execute instr_no 7 of process6
Execute instr_no 7 of process5
Execute instr_no 8 of process6
Execute instr_no 8 of process5
Execute instr_no 9 of process6
Execute instr_no 9 of process5
job6 over
Execute instr_no 10 of process5
Execute instr_no 11 of process5
Execute instr_no 12 of process5
Execute instr_no 13 of process5
Execute instr_no 14 of process5
Execute instr_no 15 of process5
Execute instr_no 16 of process5
Execute instr_no 17 of process5
Execute instr_no 18 of process5
Execute instr_no 19 of process5
Execute instr_no 20 of process5
Execute instr_no 21 of process5
Execute instr_no 22 of process5
Execute instr_no 23 of process5
Execute instr_no 24 of process5
job5 over
C:\Users\Admin\Desktop\RASHI SYITB 093 OS>
ENG
IN
15:02
12-08-2025

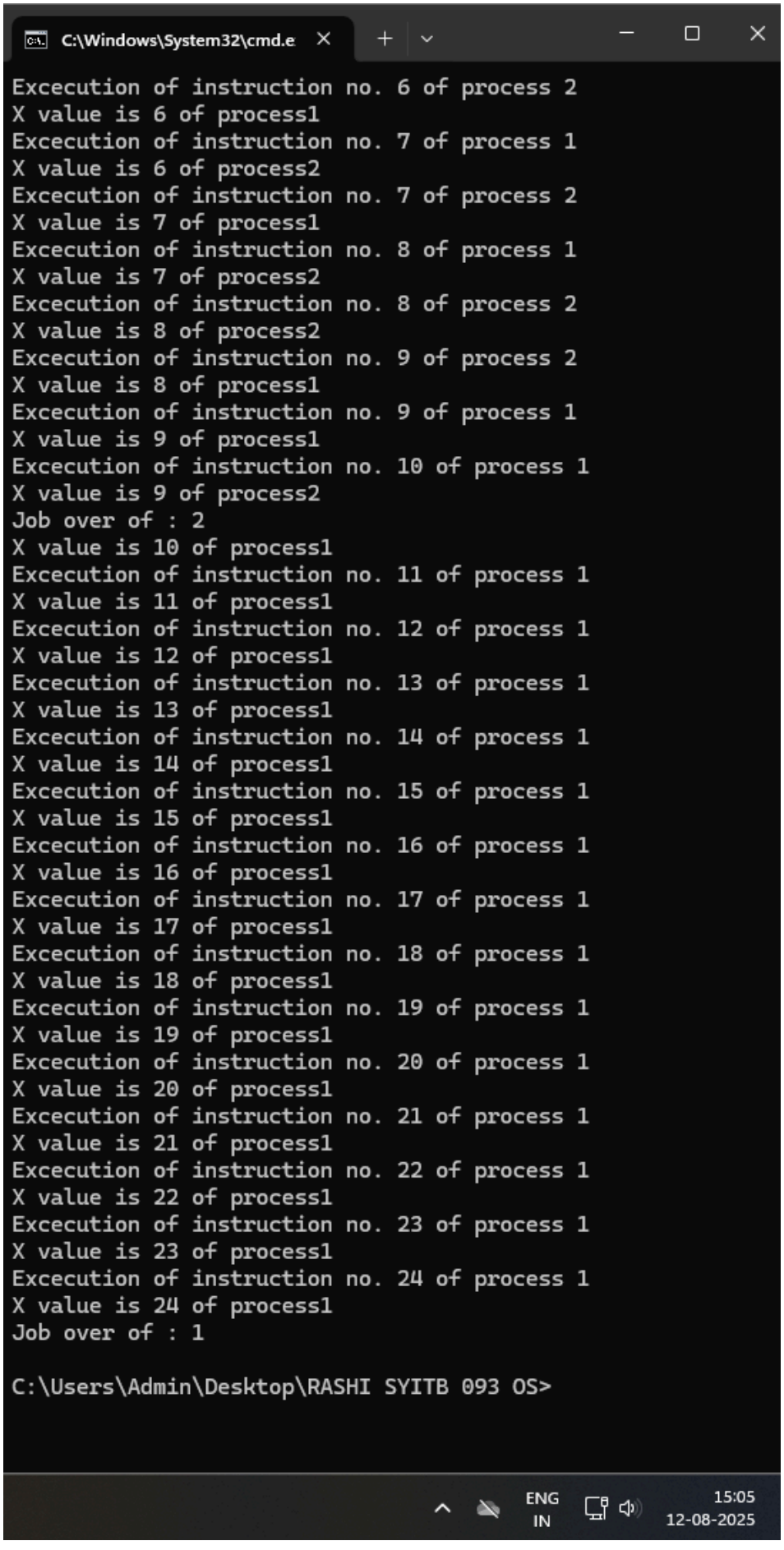
Source Code:

```
import java.util.*;
class job implements Runnable
{
int process_id,no_of_instructions,time_quantum;
Thread t;
job(int pid, int instr, int tq) {
process_id = pid;
no_of_instructions = instr;
time_quantum = tq;
t = new Thread(this);
t.start();
}
public void run(){
try{
int x=0;
System.out.println("Process id:" +process_id);
for (int i=0;i<=no_of_instructions;i++){
System.out.println("Excecution of instruction no. "+i+" of process
"+process_id);
Thread.sleep(time_quantum);
x=i;
System.out.println("X value is "+x+ " of process" + process_id);
}

System.out.println("Job over of : "+process_id);
}
catch(Exception e){
System.out.println("Exception occurred!!" +e);
}
}
}
class RRR
{
```

```
public static void main(String arr[])
{
try{
int process_id, time_quantum;
Scanner sc=new Scanner(System.in);
System.out.println("Enter the time quantum in milliseconds:");
time_quantum=sc.nextInt();
job j1=new job(1,24,time_quantum);
job j2=new job(2,9,time_quantum);
job j3=new job(3,3,time_quantum);
}
catch(Exception e){
System.out.println("Error occurred!!" +e);

}
}
}
```

Output:


C:\Windows\System32\cmd.e
Microsoft Windows [Version 10.0.26100.4652]
(c) Microsoft Corporation. All rights reserved.
C:\Users\Admin\Desktop\RASHI SYITB 093 OS>javac RRR.java
C:\Users\Admin\Desktop\RASHI SYITB 093 OS>java RRR
Enter the time quantum in milliseconds:
3
Process id:1
Excecution of instruction no. 0 of process 1
Process id:3
Excecution of instruction no. 0 of process 3
Process id:2
Excecution of instruction no. 0 of process 2
X value is 0 of process1
Excecution of instruction no. 1 of process 1
X value is 0 of process3
Excecution of instruction no. 1 of process 3
X value is 0 of process2
Excecution of instruction no. 1 of process 2
X value is 1 of process1
Excecution of instruction no. 2 of process 1
X value is 1 of process3
Excecution of instruction no. 2 of process 3
X value is 1 of process2
Excecution of instruction no. 2 of process 2
X value is 2 of process1
Excecution of instruction no. 3 of process 1
X value is 2 of process3
Excecution of instruction no. 3 of process 3
X value is 2 of process2
Excecution of instruction no. 3 of process 2
X value is 3 of process1
Excecution of instruction no. 4 of process 1
X value is 3 of process3
Job over of : 3
X value is 3 of process2
Excecution of instruction no. 4 of process 2
X value is 4 of process1
Excecution of instruction no. 5 of process 1
X value is 4 of process2
Excecution of instruction no. 5 of process 2
X value is 5 of process1
Excecution of instruction no. 6 of process 1
X value is 5 of process2
Excecution of instruction no. 6 of process 2
X value is 6 of process1
Excecution of instruction no. 7 of process 1
X value is 6 of process2
Excecution of instruction no. 7 of process 2
X value is 7 of process1
ENG
IN
15:04
12-08-2025


C:\Windows\System32\cmd.e
Excecution of instruction no. 6 of process 2
X value is 6 of process1
Excecution of instruction no. 7 of process 1
X value is 6 of process2
Excecution of instruction no. 7 of process 2
X value is 7 of process1
Excecution of instruction no. 8 of process 1
X value is 7 of process2
Excecution of instruction no. 8 of process 2
X value is 8 of process2
Excecution of instruction no. 9 of process 2
X value is 8 of process1
Excecution of instruction no. 9 of process 1
X value is 9 of process1
Excecution of instruction no. 10 of process 1
X value is 9 of process2
Job over of : 2
X value is 10 of process1
Excecution of instruction no. 11 of process 1
X value is 11 of process1
Excecution of instruction no. 12 of process 1
X value is 12 of process1
Excecution of instruction no. 13 of process 1
X value is 13 of process1
Excecution of instruction no. 14 of process 1
X value is 14 of process1
Excecution of instruction no. 15 of process 1
X value is 15 of process1
Excecution of instruction no. 16 of process 1
X value is 16 of process1
Excecution of instruction no. 17 of process 1
X value is 17 of process1
Excecution of instruction no. 18 of process 1
X value is 18 of process1
Excecution of instruction no. 19 of process 1
X value is 19 of process1
Excecution of instruction no. 20 of process 1
X value is 20 of process1
Excecution of instruction no. 21 of process 1
X value is 21 of process1
Excecution of instruction no. 22 of process 1
X value is 22 of process1
Excecution of instruction no. 23 of process 1
X value is 23 of process1
Excecution of instruction no. 24 of process 1
X value is 24 of process1
Job over of : 1
C:\Users\Admin\Desktop\RASHI SYITB 093 OS>
ENG
IN
15:05
12-08-2025

PRACTICAL 5A

Q1. To implement Page Replacement Algorithm -FIFO

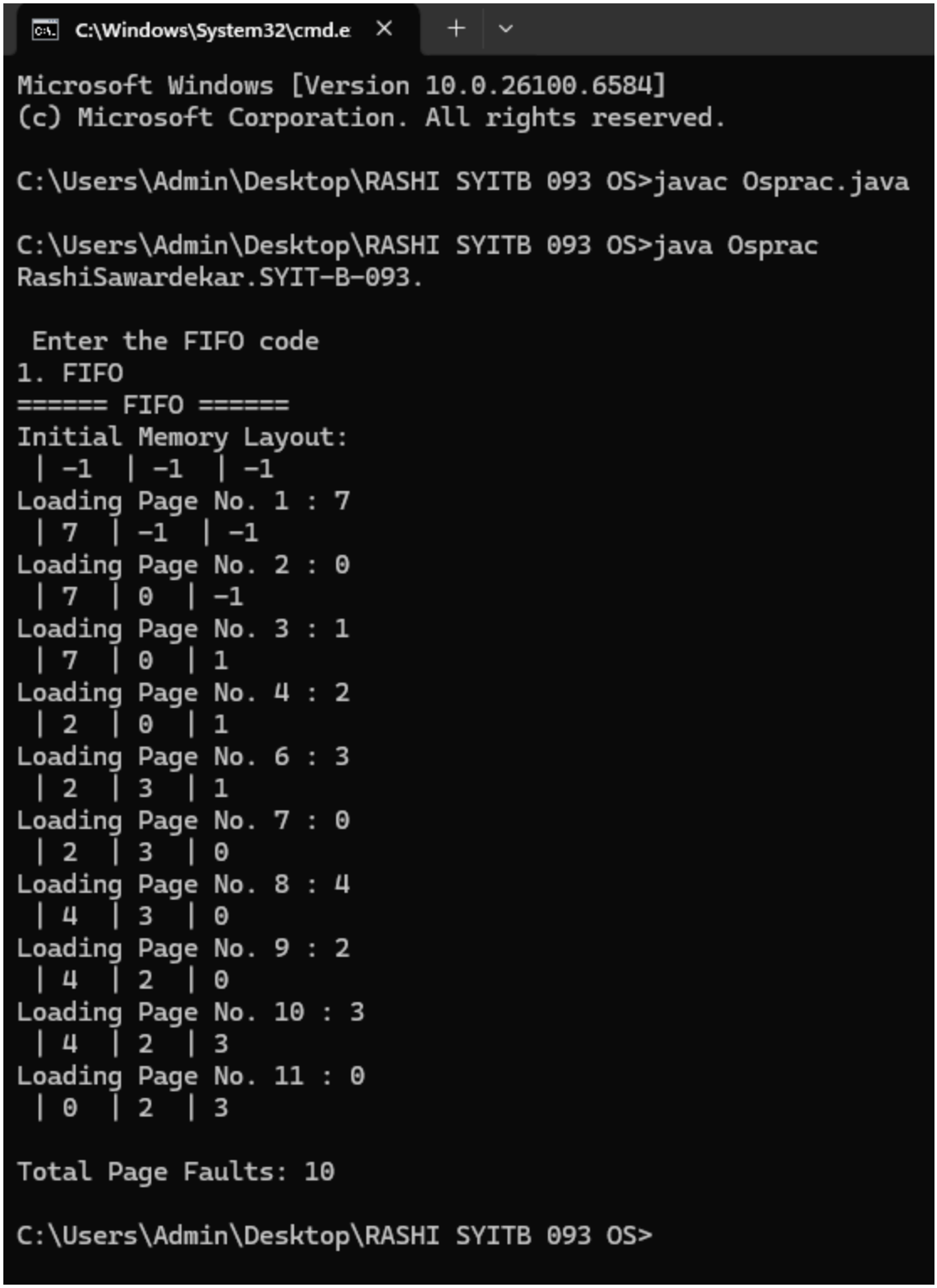
Source Code:

```
class Mem_mgmt
{
int pages[];
int mem_block[];
int pages_faults;
Mem_mgmt(int fs[], int n)
{
pages = new int[fs.length];
for (int i = 0; i < fs.length; i++)
{
pages[i] = fs[i];
}
mem_block = new int[n];
for (int i = 0; i < n; i++)
{
mem_block[i] = -1;
}
}
void disMemBlock()
{
for (int i = 0; i < mem_block.length; i++)
{
System.out.print(" | " + mem_block[i] + " ");
}

System.out.println();
}
```

```
void fifo()
{
System.out.println("====== FIFO ======");
System.out.println("Initial Memory Layout:");
disMemBlock();
int mem_block_num = 0;
pages_faults = 0;
for (int i = 0; i < pages.length; i++)
{
boolean present = false;
for (int j = 0; j < mem_block.length; j++)
{
if (mem_block[j] == pages[i])
{
present = true;
break;
}
}
if (!present)
{
System.out.println("Loading Page No. " + (i + 1) + " : " + pages[i]);
mem_block[mem_block_num] = pages[i];
mem_block_num++;
pages_faults++;

if (mem_block_num == mem_block.length)
{
mem_block_num = 0;
}
disMemBlock();
}
}
System.out.println("\nTotal Page Faults: " + pages_faults);
}
}
```

```
class Osprac
{
public static void main(String [] args)
{
System.out.println("RashiSawardekar.SYIT-B-093.");
int frame_sequence[] = {7,0,1,2,0,3,0,4,2,3,0,3,2,3};
System.out.println("\n Enter the FIFO code");
System.out.println("1. FIFO");
Mem_mgmt m = new Mem_mgmt(frame_sequence, 3);
m.fifo();
}
}
```

Output:


C:\Windows\System32\cmd.e
Microsoft Windows [Version 10.0.26100.6584]
(c) Microsoft Corporation. All rights reserved.
C:\Users\Admin\Desktop\RASHI SYITB 093 OS>javac Osprac.java
C:\Users\Admin\Desktop\RASHI SYITB 093 OS>java Osprac
RashiSawardekar.SYIT-B-093.
Enter the FIFO code
1. FIFO
====== FIFO ======
Initial Memory Layout:
| -1 | -1 | -1
Loading Page No. 1 : 7
| 7 | -1 | -1
Loading Page No. 2 : 0
| 7 | 0 | -1
Loading Page No. 3 : 1
| 7 | 0 | 1
Loading Page No. 4 : 2
| 2 | 0 | 1
Loading Page No. 6 : 3
| 2 | 3 | 1
Loading Page No. 7 : 0
| 2 | 3 | 0
Loading Page No. 8 : 4
| 4 | 3 | 0
Loading Page No. 9 : 2
| 4 | 2 | 0
Loading Page No. 10 : 3
| 4 | 2 | 3
Loading Page No. 11 : 0
| 0 | 2 | 3
Total Page Faults: 10
C:\Users\Admin\Desktop\RASHI SYITB 093 OS>

PRACTICAL 5B

Q1.The aim of Optimal Page Replacement (OPR) is to manage memory by replacing the page that will not be used for the longest time in the future, thereby minimizing the total number of page faults and achieving the best possible page replacement performance.

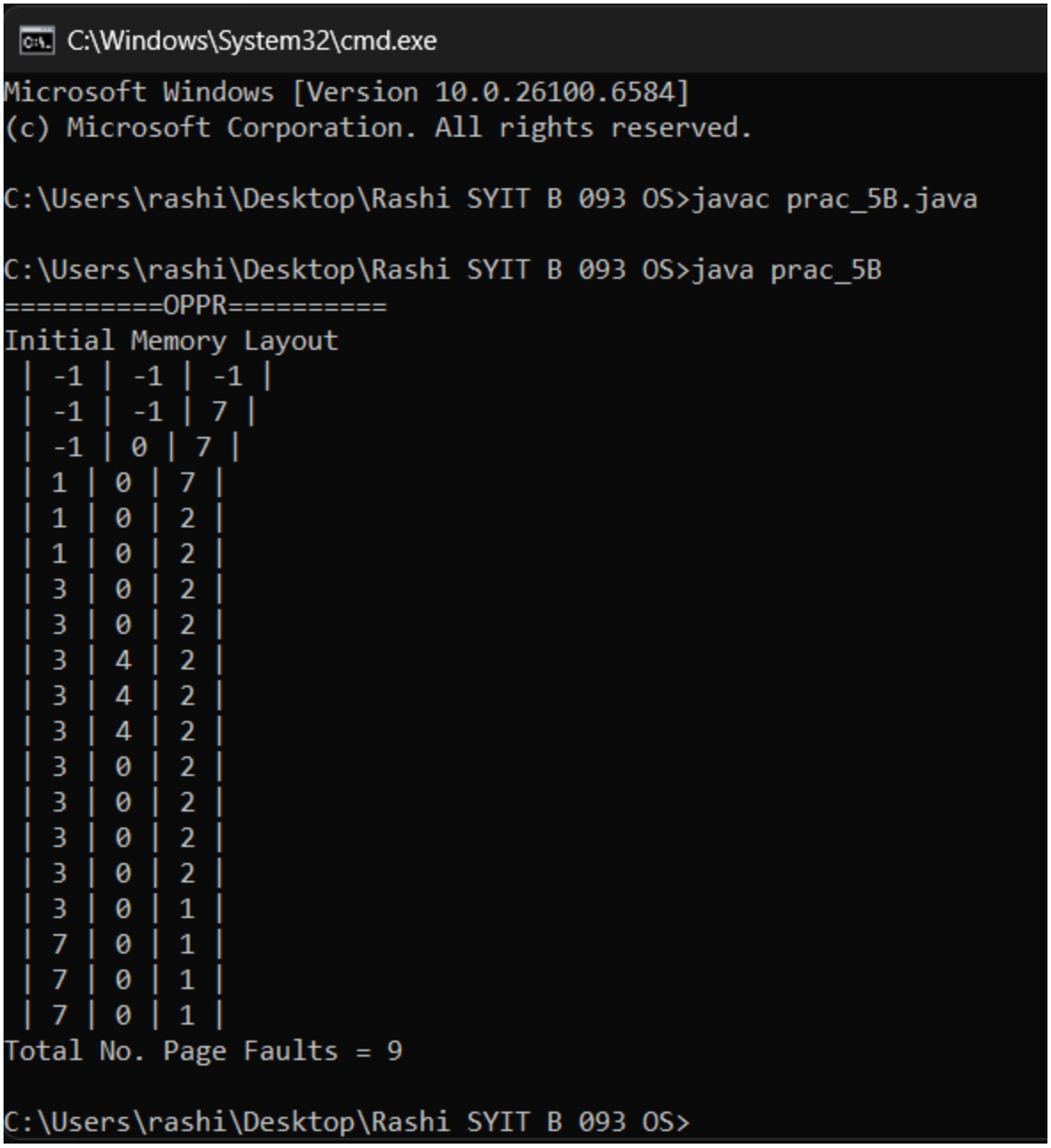
Source Code:

```
class mem_mange
{
int pages[], mem_block[];
mem_mange(int fs[], int n)
{
pages = new int[fs.length];
for(int i=0; i<fs.length; i++)
pages[i] = fs[i];
mem_block = new int[n];
for(int i=0; i<n; i++)
mem_block[i] = -1;

}
void disp_mem_block()
{
System.out.print(" | ");
for(int i=0; i<mem_block.length; i++)
System.out.print( mem_block[i] + " | ");
System.out.println();

}
void oppr()
{
```

```
int mem_block_num = 0, page_fault = 0;
System.out.println("==========OPPR==========");
System.out.println("Initial Memory Layout");
disp_mem_block();
for(int i=0; i<pages.length; i++)
{
boolean present = false;
for(int j=0; j<mem_block.length; j++)
{
if(pages[i] == mem_block[j])
{
present = true;
break;
}
}
if(!present)
{
mem_block_num = -1;
int longest_page = -1;
for(int j=0; j<mem_block.length; j++)
{
int k =0;
for(k=i+1; k<pages.length; k++)
{
if(mem_block[j] == pages[k])
{

if(k > longest_page)
{
longest_page = k;
mem_block_num = j;
}
break;
}
}
```

```
if(k == pages.length)
{
longest_page = pages.length;
mem_block_num = j;
}
}
mem_block[mem_block_num] = pages[i];
page_fault++;
}
disp_mem_block();
}
System.out.println("Total No. Page Faults = " + page_fault);
}
}

class prac_5B
{
public static void main(String args[])
{
int frame_seq[] = {7, 0, 1, 2, 0, 3, 0, 4, 2, 3, 0, 3, 2, 0, 1, 7, 0, 1};
mem_mange m = new mem_mange(frame_seq, 3);

m.oppr();
}
}
```

Output:


C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.26100.6584]
(c) Microsoft Corporation. All rights reserved.
C:\Users\rashi\Desktop\Rashi SYIT B 093 OS>javac prac_5B.java
C:\Users\rashi\Desktop\Rashi SYIT B 093 OS>java prac_5B
==========OPPR==========
Initial Memory Layout
-1	-1	-1
-1	-1	7
-1	0	7
1	0	7
1	0	2
1	0	2
3	0	2
3	0	2
3	4	2
3	4	2
3	4	2
3	0	2
3	0	2
3	0	2
3	0	2
3	0	1
7	0	1
7	0	1
7	0	1
Total No. Page Faults = 9
C:\Users\rashi\Desktop\Rashi SYIT B 093 OS>

PRACTICAL 6

Q1. To implement Banker's Algorithm

Source Code:

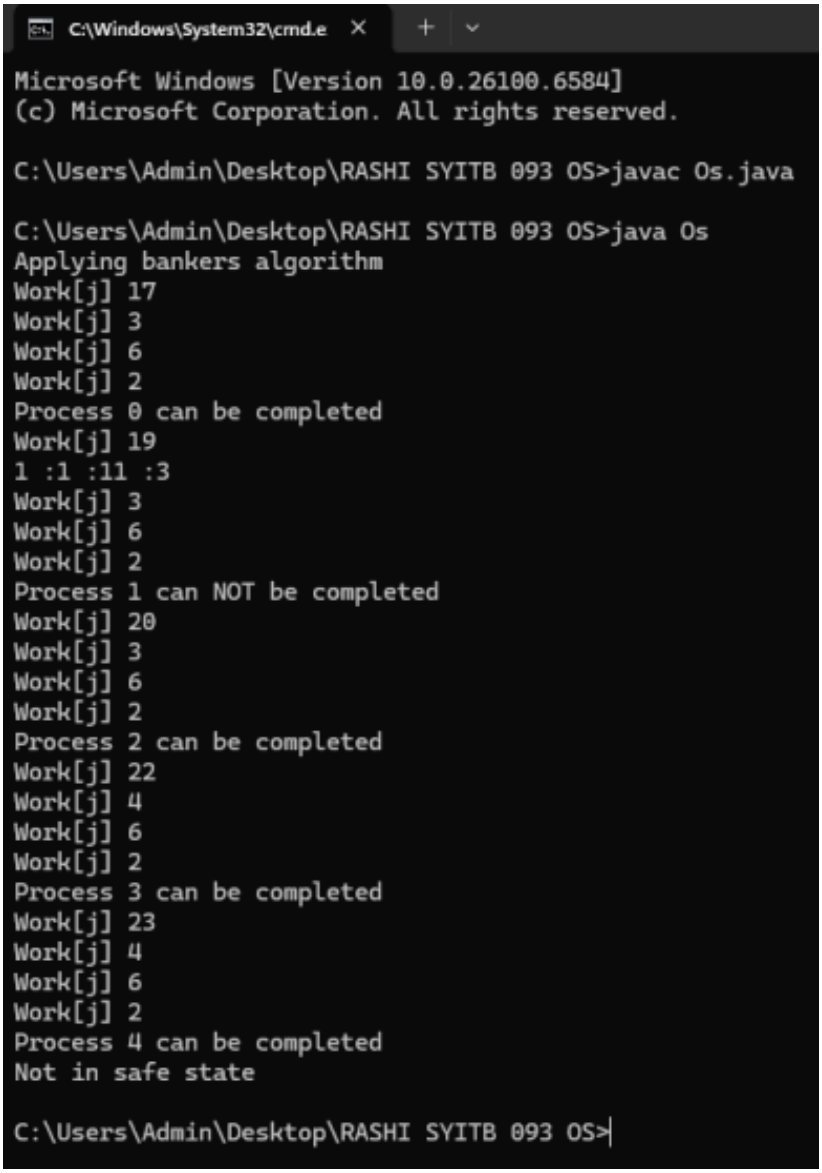
```
class Banker
{
    int av[]= {16,2,5,2};
    int max[][]= {
        {3,2,2,1},
        {8,12,0,0},
        {2,1,0,0},
        {4,3,0,0},
        {2,0,3,1}
    };
    int alloc[][]= {
        {1,1,1,0},
        {2,1,0,0},
        {1,0,0,0},
        {2,1,0,0},
        {1,0,0,0}
    };
    int need[][];
    int m=4,n=5;

    Banker()
    {
        need=new int[n][m];
        for(int i=0;i<n;i++)
        {
            for(int j=0;j<m;j++)
            {
                need[i][j]=max[i][j]-alloc[i][j];
```

```
        }
    }
}
boolean isSafe()
{
    int work[]=new int[m];
    boolean finish[]=new boolean[n];
     boolean incomplete=false;

    for(int i=0;i<m;i++)
    {
        work[i]= av[i];
    }
    for(int i=0;i<n;i++)
    {
        finish[i]=false;
    }
    for(int i=0;i<n;i++)
    {
            incomplete=false;
        if(finish[i] == false)
        {
            for(int j=0;j<m;j++)
            {
                if(need[i][j]>work[j])
                {
                    incomplete=true;
                    System.out.println(i+" :"+j+" :" +need[i][j] +" :" +work[j]);
                }
                if(need[i][j]!=0 && need[i][j]<=work[j])
                {
                    work[j]=work[j]-need[i][j];
                    work[j]=work[j]+max[i][j];
                }
                System.out.println("Work[j] " +work[j]);
```

```
            }
        }
            if(!incomplete)
            {
               finish[i]=true;
            }

            if(finish[i])
            {
               System.out.println("Process " +i+ " can be completed");
            }
            else
            {
               System.out.println("Process " +i+ " can NOT be completed");
            }
        }
          for(int i=0;i<n;i++)
          {
                if(!finish[i])
                {
                   return false;
                }
          }
           return false;

     }
}

class Os
{
    public static void main(String ar[])
    {
        Banker b=new Banker();
        System.out.println("Applying bankers algorithm");
        if(b.isSafe())
```

```
        {
            System.out.println("In safe state");
        }
        else
        {
            System.out.println("Not in safe state");
        }
    }
}
```

Output:



PRACTICAL 7A

Q1.To implement the First Fit memory allocation algorithm in Java that allocates memory blocks to processes by placing each process in the first available contiguous free memory segment large enough to accommodate it, and to display the memory layout after each allocation.

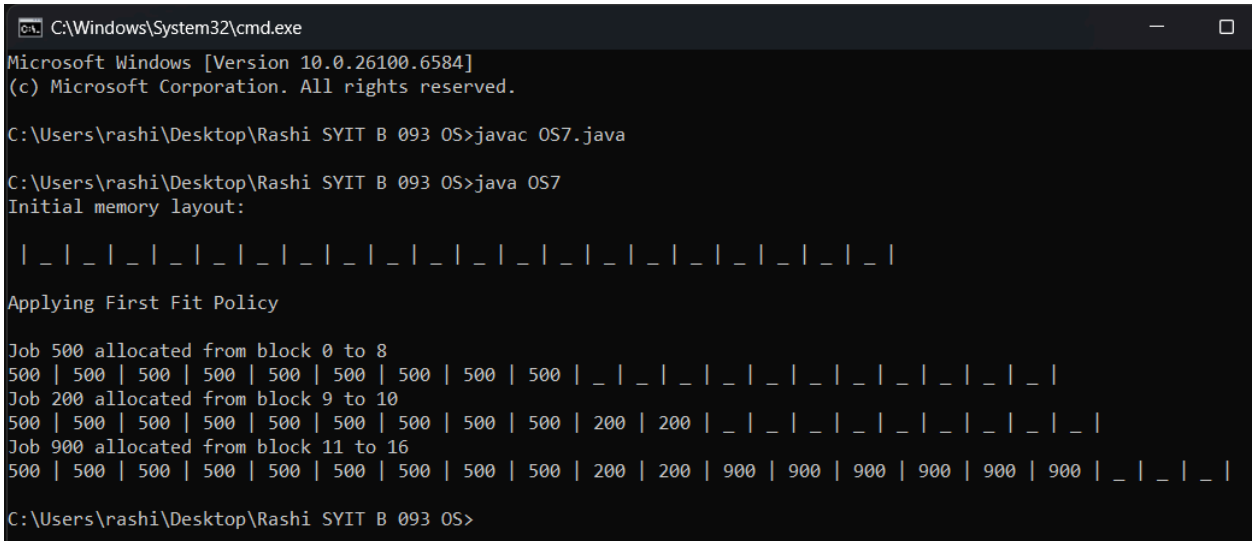
Source Code:

```
import java.util.*;

class Job {
    int Pid, size;

    Job(int Process_id, int n) {
        Pid = Process_id;
        size = n;
    }
}

class MemMgmt {
    int mem_array[];
    Vector<Job> jobqueue;

    MemMgmt(Vector<Job> v) {
        jobqueue = v;
        mem_array = new int[20];
        for (int i = 0; i < 20; i++) {
            mem_array[i] = 0;  // 0 means free block
        }
        System.out.println("Initial memory layout:");
        System.out.println();
```

```
    System.out.print(" | ");
    printArray();
}

void printArray() {
    for (int i = 0; i < mem_array.length; i++) {
        if (mem_array[i] == 0)
            System.out.print("_ | ");
        else
            System.out.print(mem_array[i] + " | ");
    }
    System.out.println();
}

void addJob(Job job) {
    jobqueue.add(job);
}

void firstfit() {
    System.out.println("\nApplying First Fit Policy\n");
    for (Job job : jobqueue) {
        boolean allocated = false;
        for (int i = 0; i <= mem_array.length - job.size; i++) {
            boolean canAllocate = true;
            for (int j = i; j < i + job.size; j++) {
                if (mem_array[j] != 0) {
                    canAllocate = false;
                    break;
                }
            }
            if (canAllocate) {
                for (int j = i; j < i + job.size; j++) {
                    mem_array[j] = job.Pid;
                }
```

```
                System.out.println("Job " + job.Pid + " allocated from block " + i
+ " to " + (i + job.size - 1));
                allocated = true;
                break;
            }
        }
        if (!allocated) {
            System.out.println("Job " + job.Pid + " cannot be allocated -
insufficient space.");
        }
        printArray();
    }
  }
}

public class OS7 {
    public static void main(String[] args) throws Exception {
        Vector<Job> v = new Vector<>()
        MemMgmt m = new MemMgmt(v);

        Job job1 = new Job(500, 9);
        Job job2 = new Job(200, 2);
        Job job3 = new Job(900, 6);

        m.addJob(job1);
        m.addJob(job2);
        m.addJob(job3);
        m.firstfit();
    }
}
```

Output:


C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.26100.6584]
(c) Microsoft Corporation. All rights reserved.
C:\Users\rashi\Desktop\Rashi SYIT B 093 OS>javac OS7.java
C:\Users\rashi\Desktop\Rashi SYIT B 093 OS>java OS7
Initial memory layout:
| _ | _ | _ | _ | _ | _ | _ | _ | _ | _ | _ | _ | _ | _ | _ | _ | _ | _ | _ | _ |
Applying First Fit Policy
Job 500 allocated from block 0 to 8
500 | 500 | 500 | 500 | 500 | 500 | 500 | 500 | 500 | _ | _ | _ | _ | _ | _ | _ | _ | _ | _ | _ |
Job 200 allocated from block 9 to 10
500 | 500 | 500 | 500 | 500 | 500 | 500 | 500 | 500 | 200 | 200 | _ | _ | _ | _ | _ | _ | _ | _ | _ |
Job 900 allocated from block 11 to 16
500 | 500 | 500 | 500 | 500 | 500 | 500 | 500 | 500 | 200 | 200 | 900 | 900 | 900 | 900 | 900 | 900 | _ | _ | _ |
C:\Users\rashi\Desktop\Rashi SYIT B 093 OS>

PRACTICAL 8

Q1.To implement the Shortest Seek Time First (SSTF) disk scheduling algorithm in Java and calculate the total seek time (cylinders traversed) for a given sequence of disk track requests starting from a specified initial head position.

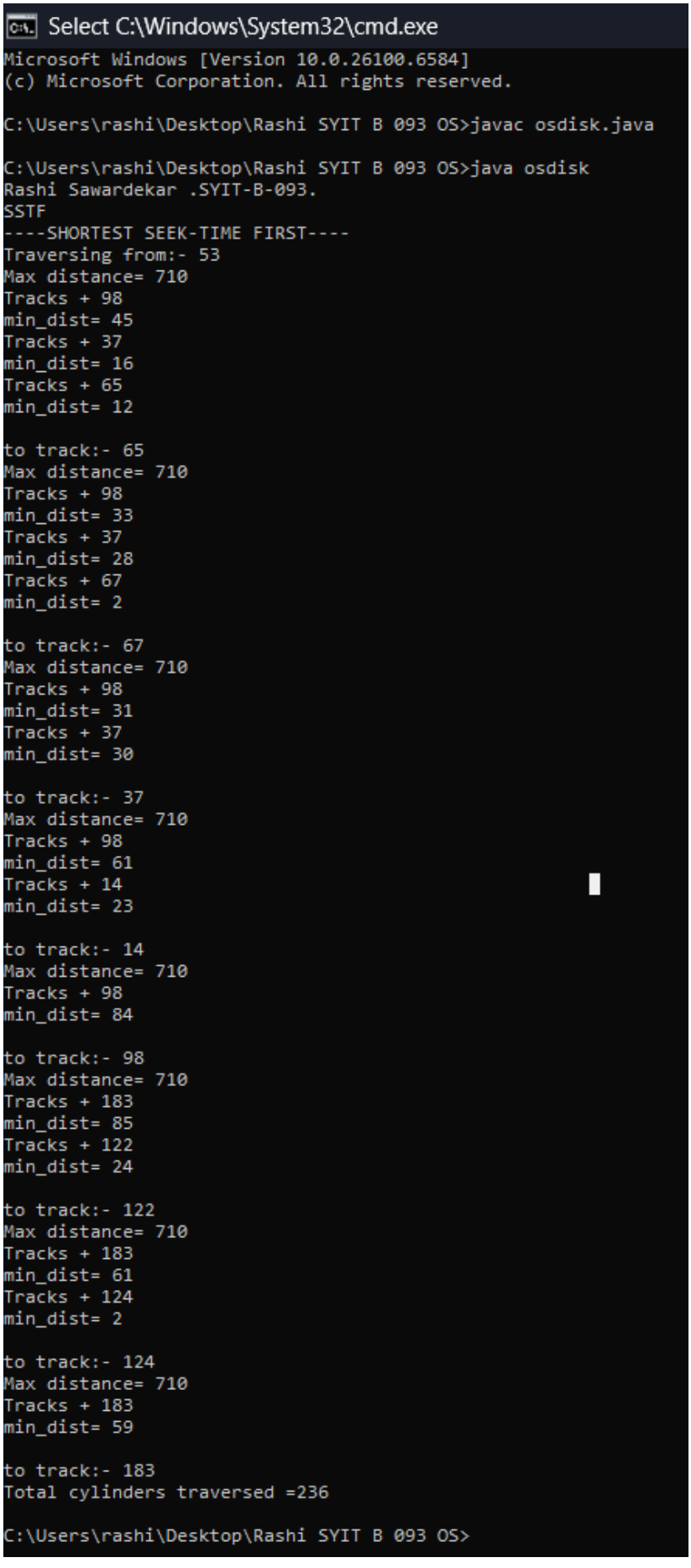
Source Code:

```
import java.io.*;
class DevMgmt
{
 int []tracks;
 DevMgmt(int []ds)
{
 tracks=new int[ds.length];
 for(int i=0;i<ds.length;i++)
{
 tracks[i]=ds[i];
}
}
void sstf(int start_track)
{
 System.out.println("----SHORTEST SEEK-TIME FIRST----");
 boolean tracks_bool[]= new boolean[tracks.length];
 int min_dist=0,max_dist=0;
for(int i=0;i<tracks.length;i++)
{
 tracks_bool[i]=false;
 max_dist=max_dist+tracks[i];
}
min_dist=max_dist;
int total_cylinders=0;
```

```
int from_track=start_track;
int current_track_index= -1;
System.out.println("Traversing from:- "+from_track);
for(int i=0;i<tracks.length;i++)
{
 System.out.println("Max distance= "+max_dist);
 int to_track_index=0;
for(int t=0;t<tracks.length;t++)
{

if((!tracks_bool[t])&&(Math.abs(from_track-tracks[t])<min_dist)&&(t!=current
_track_index
))
{
System.out.println("Tracks + "+tracks[t]);
to_track_index=t;
min_dist=Math.abs(from_track-tracks[t]);
System.out.println("min_dist= "+min_dist);
}
}
System.out.println();
System.out.println("to track:- "+tracks[to_track_index]);
total_cylinders+=Math.abs(from_track-tracks[to_track_index]);
tracks_bool[to_track_index]=true;
from_track=tracks[to_track_index];
current_track_index=to_track_index;
min_dist=max_dist;
}
System.out.println("Total cylinders traversed ="+total_cylinders);
}
}
class osdisk
{
public static void main (String []args) throws Exception
{
```

```
 System.out.println("Rashi Sawardekar .SYIT-B-093.");
 System.out.println("SSTF");
 int disk_sequence[]={98,183,37,122,14,124,65,67};
 int track=53;
 DevMgmt m= new DevMgmt(disk_sequence);
 m.sstf(track);
}
}
```

Output:


Select C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.26100.6584]
(c) Microsoft Corporation. All rights reserved.
C:\Users\rashi\Desktop\Rashi SYIT B 093 OS>javac osdisk.java
C:\Users\rashi\Desktop\Rashi SYIT B 093 OS>java osdisk
Rashi Sawardekar .SYIT-B-093.
SSTF
----SHORTEST SEEK-TIME FIRST----
Traversing from:- 53
Max distance= 710
Tracks + 98
min_dist= 45
Tracks + 37
min_dist= 16
Tracks + 65
min_dist= 12
to track:- 65
Max distance= 710
Tracks + 98
min_dist= 33
Tracks + 37
min_dist= 28
Tracks + 67
min_dist= 2
to track:- 67
Max distance= 710
Tracks + 98
min_dist= 31
Tracks + 37
min_dist= 30
to track:- 37
Max distance= 710
Tracks + 98
min_dist= 61
Tracks + 14
min_dist= 23
to track:- 14
Max distance= 710
Tracks + 98
min_dist= 84
to track:- 98
Max distance= 710
Tracks + 183
min_dist= 85
Tracks + 122
min_dist= 24
to track:- 122
Max distance= 710
Tracks + 183
min_dist= 61
Tracks + 124
min_dist= 2
to track:- 124
Max distance= 710
Tracks + 183
min_dist= 59
to track:- 183
Total cylinders traversed =236
C:\Users\rashi\Desktop\Rashi SYIT B 093 OS>

PRACTICAL 9

Q1.To implement Disk Scheduling - LOOK

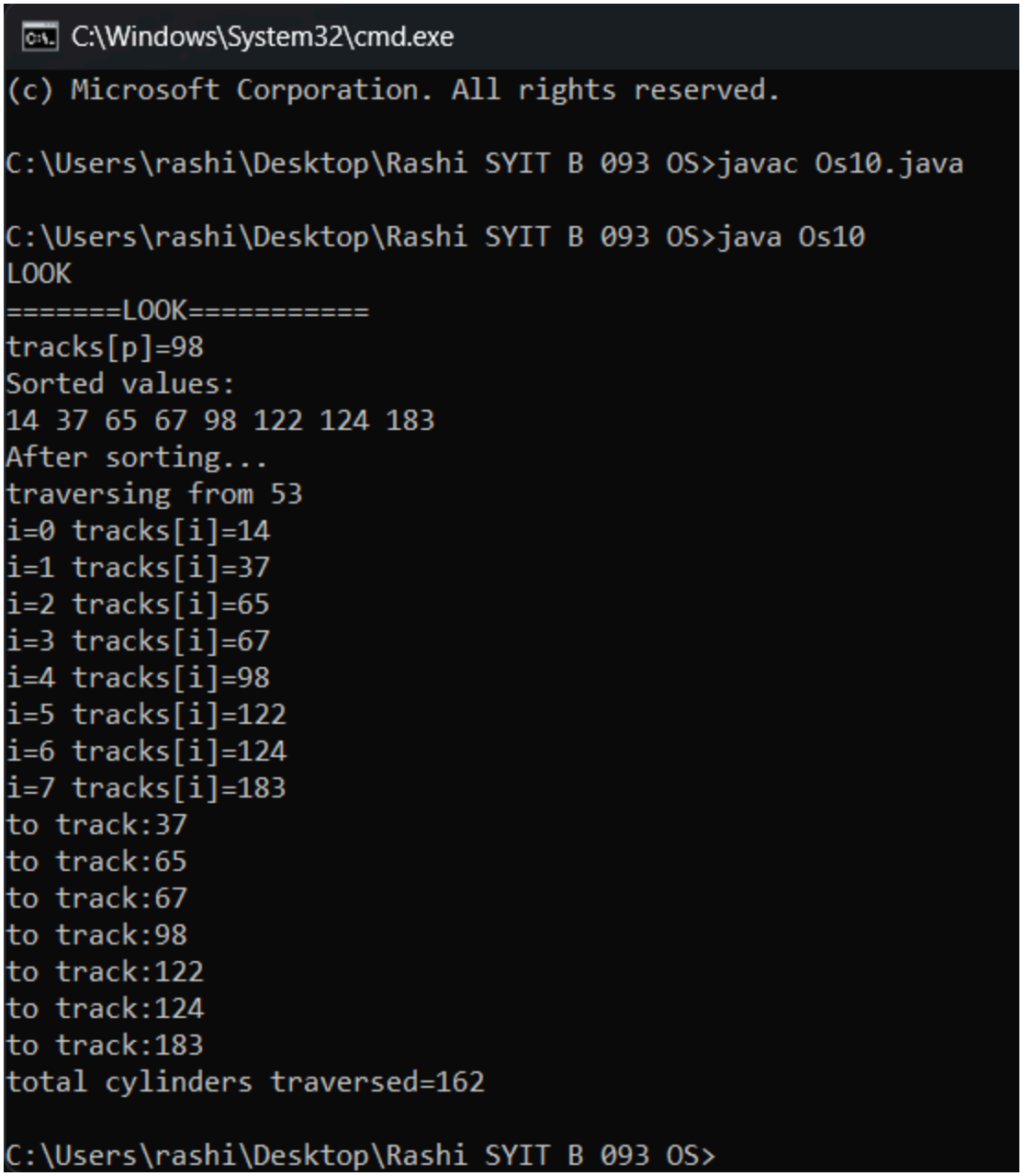
Source Code:

```
class DevMgmt
{
    String disk_sequence;
    int[] tracks;

    DevMgmt(int[] ds)
    {
        tracks = new int[ds.length];
        for (int i = 0; i < ds.length; i++)
        {
            tracks[i] = ds[i];
        }
    }

    void look(int start_track)
    {
        System.out.println("=======LOOK==========");
        int p = 0;
        System.out.println("tracks[p]=" + tracks[p]);

        for (int i = 0; i < tracks.length - 1; i++)
        {
```

```
    for (int j = 0; j < tracks.length - i - 1; j++) // Added inner loop
    {
        if (tracks[j] > tracks[j + 1])
        {
            int temp = tracks[j];
            tracks[j] = tracks[j + 1];
            tracks[j + 1] = temp;
        }
    }
}

System.out.println("Sorted values:");
for (int i = 0; i < tracks.length; i++)
{
    System.out.print(tracks[i] + " ");
}
System.out.println();

System.out.println("After sorting...");
int total_cylinder = 0, from_track = start_track, to_track = 0;
System.out.println("traversing from " + from_track);

int i = 0;
for (i = 0; i < tracks.length; i++)
{
    System.out.println("i=" + i + " tracks[i]=" + tracks[i]);
    /*
    if (tracks[i] <= start_track)
    {
        break;
    }
    */
}

for (int x = 1; x > 0; x--)
```

```
        {
            System.out.println("to track:" + tracks[x]);
            to_track = tracks[x];
            total_cylinder = total_cylinder + Math.abs(from_track - to_track);
            from_track = to_track;
        }

        for (int x = 2; x < tracks.length; x++)
        {
            System.out.println("to track:" + tracks[x]);
            to_track = tracks[x];
            total_cylinder = total_cylinder + Math.abs(from_track - to_track);
            from_track = to_track;
        }

        System.out.println("total cylinders traversed=" + total_cylinder);
    }
}

class Os10
{
    public static void main(String args[]) throws Exception
    {
        System.out.println("LOOK");
        int disk_sequence[] = {98, 183, 37, 122, 14, 124, 65, 67};
        int track = 53;
        DevMgmt m = new DevMgmt(disk_sequence);
        m.look(track);
    }
}
```

Output:


C:\Windows\System32\cmd.exe
(c) Microsoft Corporation. All rights reserved.
C:\Users\rashi\Desktop\Rashi SYIT B 093 OS>javac Os10.java
C:\Users\rashi\Desktop\Rashi SYIT B 093 OS>java Os10
LOOK
=======LOOK==========
tracks[p]=98
Sorted values:
14 37 65 67 98 122 124 183
After sorting...
traversing from 53
i=0 tracks[i]=14
i=1 tracks[i]=37
i=2 tracks[i]=65
i=3 tracks[i]=67
i=4 tracks[i]=98
i=5 tracks[i]=122
i=6 tracks[i]=124
i=7 tracks[i]=183
to track:37
to track:65
to track:67
to track:98
to track:122
to track:124
to track:183
total cylinders traversed=162
C:\Users\rashi\Desktop\Rashi SYIT B 093 OS>