

**Afrah Abdulmajid**

**COMP2004**

**201158243**

**Assignment 3**

## Table of Contents

1.	User Guide .....	3
1.1	System Requirements .....	3
1.2	Format.....	4
1.3	How To Run.....	4
1.4	Sample Execution.....	4
2.	Top – Down Design .....	7
3.	Dictionary of Variables.....	8
4.	Dictionary of Function and Methods .....	9
5.	The Code .....	10
6.	How the code works .....	13
7.	Limitations.....	13

## 1. User Guide

The following program is coded in Java to simulate how processes change states as consequence of System calls such as create, resume, and kill. This program also displays what changes happen in the queue associated with processes and depending on each process's priority. The code also demonstrates how management of a processes as consequence of the changes in the state (i.e. A change from suspended state to a ready state). The system calls given are create, creates a process with the name given and priority given from the User input. Once a process is created, it is stored in a suspended queue until the process is moved to a ready queue, the change of states from suspended to ready is done using the system call resume. This is also done through User input, where the User provides the process name that the User would like to make ready. The third system call is the kill call, this is when you would like to kill a process, the User can call kill and provide the process name that the User would like to kill. As the Systems calls are taking place, the code will display the changes that are taking place in each call that updates the ready queue.

### 1.1 System Requirements

The .java files contains the program specified, the following program can be edited with any IDE (Visual Studio, Eclipse, IntelliJ, etc.) or any text editor (Notepad++, XCode, Sublime Text, etc.). The program could be run on any operating System from the command prompt, provided that JDK (Java Development Kit) is installed on the machine the program is being run on.

## 1.2 Format

The input format of the program is to correctly spell the words “create”, “resume”, and “kill” when the program as the program prompts, and so the program will run based on the user input and produce the result accordingly. As well as correctly spelling “yes” or “no” when the program prompts you on the next action.

## 1.3 How To Run

The file can be accessed from the USB file within the directory Assignment 3. Within the Assignment 3 folder there is a file called Processes.java. The steps to run the program are as follows:

**Step 1:** Open the Command Prompt (cmd), also known as the terminal

**Step 2:** Navigate to the directory that the program is saved in using the command **cd** */absolutepathname*, for example, */Users/Afrah/Desktop/cs2004*.

**Step 3:** Type in; Javac Processes.java

**Step 4:** Type in; Java Processes

The program will now execute.

## 1.4 Sample Execution

Below I have attached an example of how the program is run using the command prompt as well as the output corresponding to system calls.

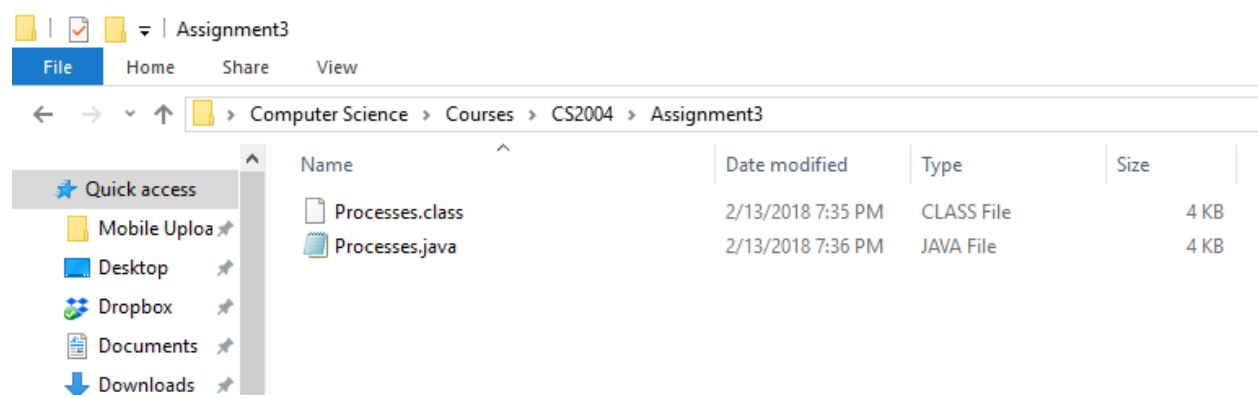
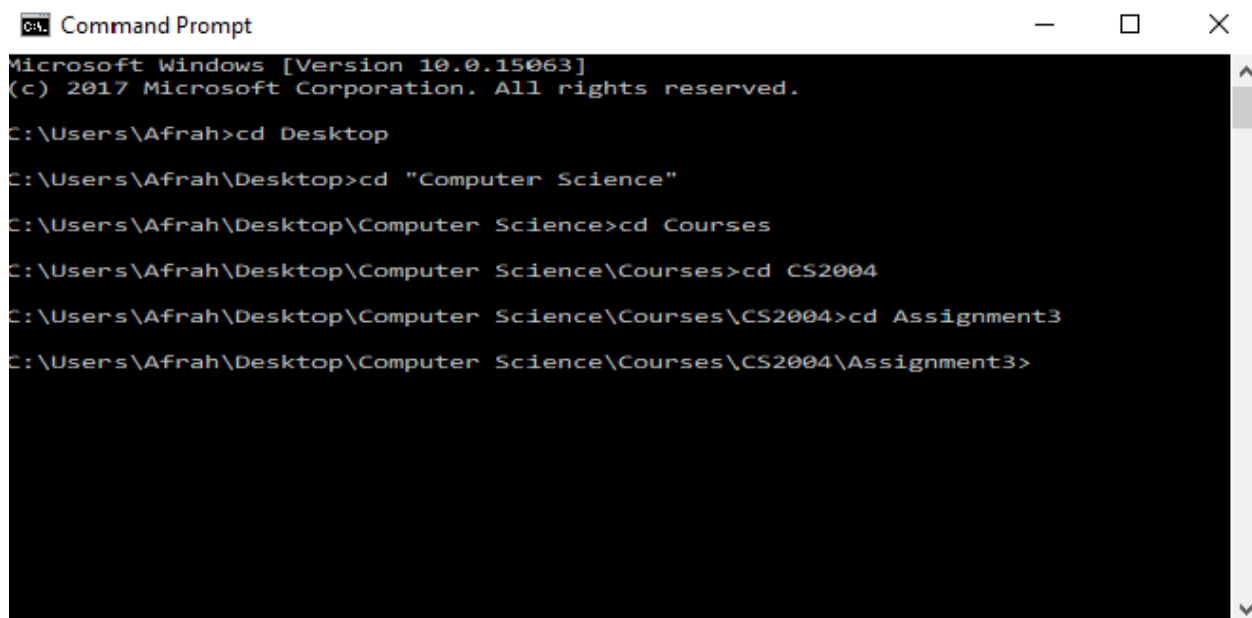


Figure 1.0: The directories and the files stored in CS2004 folder



```
Command Prompt
Microsoft Windows [Version 10.0.15063]
(c) 2017 Microsoft Corporation. All rights reserved.

C:\Users\Afrah>cd Desktop
C:\Users\Afrah\Desktop>cd "Computer Science"
C:\Users\Afrah\Desktop\Computer Science>cd Courses
C:\Users\Afrah\Desktop\Computer Science\Courses>cd CS2004
C:\Users\Afrah\Desktop\Computer Science\Courses\CS2004>cd Assignment3
C:\Users\Afrah\Desktop\Computer Science\Courses\CS2004\Assignment3>
```

Figure 2.0: The following are the commands used to navigate to the directory the file is saved in.

```
C:\Users\Afrah\Desktop\Computer Science\Courses\CS2004\Assignment3>java Processes
Would you like to create, resume or kill?
create
What is process name?
proc1
What is its priority?
10
Do you want to continue? yes/no
yes
Would you like to create, resume or kill?
create
What is process name?
proc2
What is its priority?
15
Do you want to continue? yes/no
yes
Would you like to create, resume or kill?
create
What is process name?
proc3
What is its priority?
20
Do you want to continue? yes/no
yes
Would you like to create, resume or kill?
create
What is process name?
proc4
What is its priority?
25
Do you want to continue? yes/no
yes
Would you like to create, resume or kill?
resume
What is the name of the process?
proc1
Contents of r_queue:
```

Figure 3.0: Sequence of inputs to create processes and starting resume call, and as can be seen the contents of the ready queue(r\_queue) is empty to start with.

```
Would you like to create, resume or kill?
resume
What is the name of the process?
proc2
Contents of r_queue:
Process name: proc1
Do you want to continue? yes/no
yes
Would you like to create, resume or kill?
resume
What is the name of the process?
proc3
Contents of r_queue:
Process name: proc1
Process name: proc2
Do you want to continue? yes/no
```

Figure 4.0: Sequence of resumes and the output of the ready queue

```
Process name: proc1
Process name: proc2
Contents of r_queue:
Do you want to continue? yes/no
yes
Would you like to create, resume or kill?
kill
What is the name of the process?
proc2
Process name: proc1
Contents of r_queue:
Do you want to continue? yes/no
yes
Would you like to create, resume or kill?
create
What is process name?
proc4
```

Figure 5.0: Sequence of kill calls, and the resulting ready queue.

```
Command Prompt - java Processes
C:\Users\Afrah\Desktop\Computer Science\Courses\CS2004\Assignment3>java Processes
Would you like to create, resume or kill?
create
What is process name?
proc1
What is its priority?
10
Do you want to continue? yes/no
yes
Would you like to create, resume or kill?
resume
What is the name of the process?
proc2
Error: The Process name is not in the Suspended Queue
Contents of r_queue:
Do you want to continue? yes/no
```

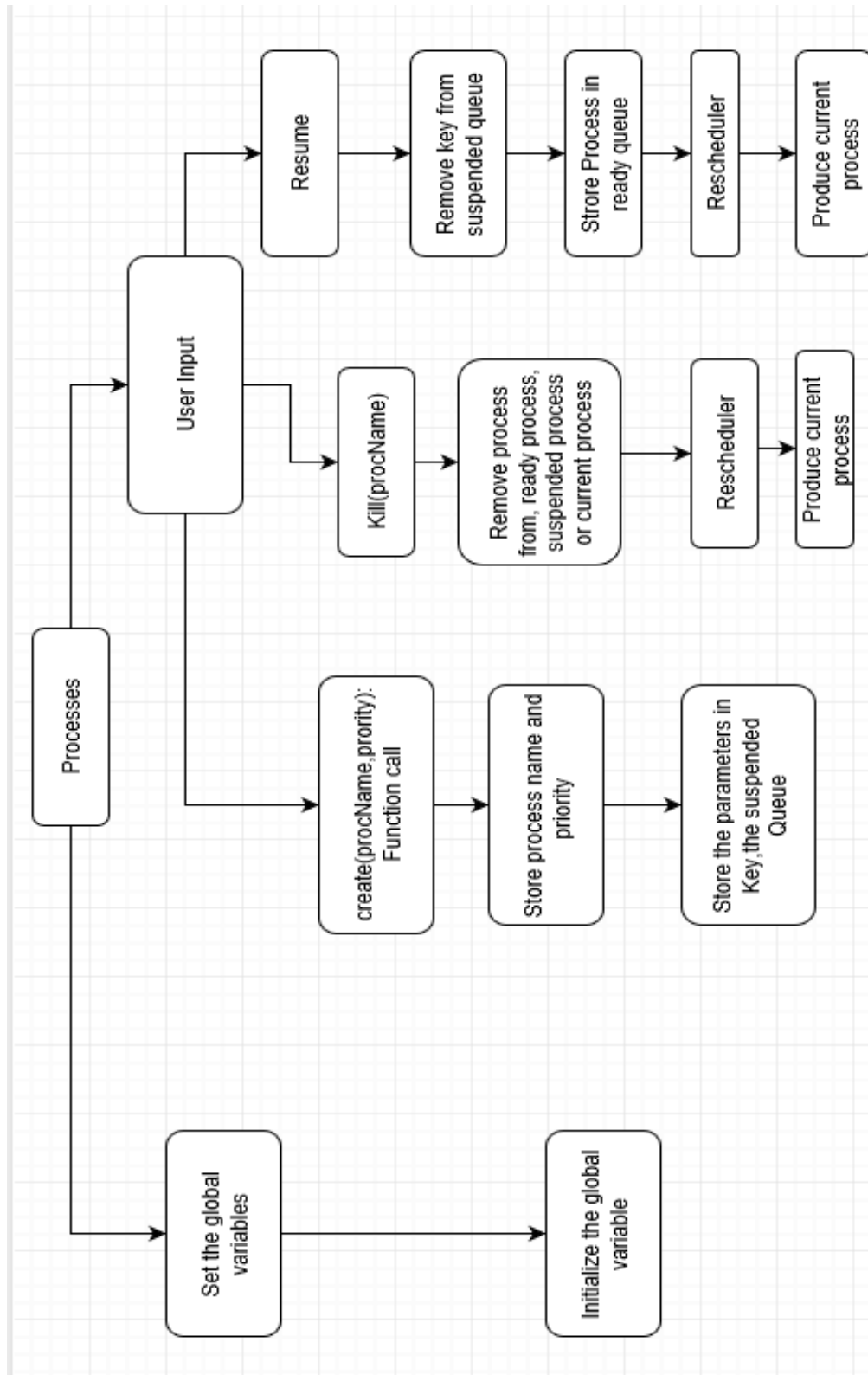
Figure 6.0: Result of when User tries resume a process that does not exist

```
C:\Users\Afrah\Desktop\Computer Science\Courses\CS2004\Assignment3>java Processes
Would you like to create, resume or kill?
kill
What is the name of the process?
proc1
The Process does not exist
Contents of r_queue:
Do you want to continue? yes/no
```

Figure 7.0: Result of when User tries to kill a process that does not exist.

The following program will continue to run or stop based on the User input

## 2. Top – Down Design



### 3. Dictionary of Variables

The following global variables are initialized in the program. Where they are passed in to functions



r_queue	It's a treeMap data structure that stores the processes in the ready queue.
s_queue	It is a Queue data structure that stores the suspended processes
ln	It is a Scanner object that processes the input from the command prompt
higestPriority	Integer value that stores the priority that is the highest.
val	It is the highest priority before reschedule() is called to compare new priority with the highest priority.
str	It is a String that stores the key of the highest Priority
cr	The string variable that stores the current process

#### 4. Dictionary of Function and Methods

Listed below are the list of function or methods that are used with the program. There are two types of functions in these programs, some are builtin and the other functions are ones that was written and implanted by the author.

main(String[] args)	The function that runs the program
Processes()	Constructor of the processes class
create(String procName, int priority)	This function creates a process procName and sets the priority, the two
Resume(String procName)	This function takes the process procName from within the suspended Queue to the ready queue
Kill(String procName)	This function finds the process with the given name and kills the process, i.e. deletes it.
rescheduler	This function updates the current process if needed every time change is done to the queue.

## 5. The Code

C:\Users\AfrAh\Desktop\Computer Science\Courses\CS2004\Assignment3\Processes.java - Sublime

```
Edit Selection Find View Goto Tools Project Preferences Help

Scheduler.java x Processes.java x ImageHistogram.java x

1  import java.util.LinkedList;
2  import java.util.Queue;
3  import java.util.Map;
4  import java.util.TreeMap;
5  import java.util.Iterator;
6  import java.util.Scanner;
7  import java.lang.System;
8  import java.util.Set;
9  import java.util.Map.Entry;
10 import java.util.Scanner;
11 import java.lang.System;
12
13 public class Processes {
14     TreeMap <String,Integer> r_queue;
15     Queue <String> s_queue;
16     TreeMap<String,Integer> proc;
17     int highestPriority;
18     int val;
19     String str;
20     String cr ;
21     public Processes(){
22         val = 0;
23         str = " ";
24         cr = "";
25         r_queue= new TreeMap <String,Integer>();
26         s_queue= new LinkedList <String>();
27         proc = new TreeMap<String,Integer>();
28         highestPriority = 0;
29         Scanner in = new Scanner(System.in);
30         String response = "yes";
31
32         while(response.equals("yes")){
33             System.out.println("Would you like to create, resume or kill?");
34             String cmd = in.next();
35             cmd.toLowerCase();
36
37             if(cmd.equals("create")){
38                 System.out.println("What is process name?");
39                 String para1 = in.next();
40                 String dummy = in.nextLine();
41                 System.out.println("What is its priority?");
42                 int para2 = in.nextInt();
43                 create(para1,para2);
44             }
45             else if(cmd.equals("resume")){
46                 System.out.println("What is the name of the process?");
47                 String para3 = in.next();
48                 String dummy = in.nextLine();
49                 resume(para3);
50             }
51             else if(cmd.equals("kill")){
52                 System.out.println("What is the name of the process?");
53                 String para4 = in.next();
54                 String dummy = in.nextLine();
55                 kill(para4);
56             }
57             response = in.next();
58         }
59     }
60
61     void create(String name, int priority){
62         if(priority < 0 || priority > 10){
63             System.out.println("Priority must be between 0 and 10");
64             return;
65         }
66         if(name.length() > 20){
67             System.out.println("Process name must be less than 20 characters");
68             return;
69         }
70         if(proc.containsKey(name)){
71             System.out.println("Process already exists");
72             return;
73         }
74         proc.put(name, priority);
75         r_queue.put(name, priority);
76     }
77
78     void resume(String name){
79         if(!proc.containsKey(name)){
80             System.out.println("Process does not exist");
81             return;
82         }
83         int priority = proc.get(name);
84         s_queue.add(name);
85         r_queue.remove(name);
86     }
87
88     void kill(String name){
89         if(!proc.containsKey(name)){
90             System.out.println("Process does not exist");
91             return;
92         }
93         proc.remove(name);
94         r_queue.remove(name);
95     }
96 }
```

```

46         System.out.println("What is the name of the process?");
47         String para3 = in.next();
48         String dummy = in.nextLine();
49         resume(para3);
50     }
51     else if(cmd.equals("kill")){
52         System.out.println("What is the name of the process?");
53         String para4 = in.next();
54         String dummy = in.nextLine();
55         kill(para4);
56     }
57     System.out.println("Do you want to continue? yes/no");
58     response = in.next();
59     String dummy = in.nextLine();
60     response = response.toLowerCase();
61 }
62
63 }
64
65 public static void main(String[] args){
66     new Processes();
67 }
68
69 public void create( String procName, int priority){
70     proc.put(procName,priority);
71     if (s_queue.isEmpty())
72         s_queue.add(procName);
73
74     else{
75         if(s_queue.contains(procName))
76             System.out.println("Error: The Process name already exists");
77
78         else{
79             s_queue.add(procName);

```

```

81     }
82 }
83
84 public void resume(String procName){
85     if(s_queue.contains(procName)){
86         s_queue.remove(procName);
87         for(String item : proc.keySet()){
88             if(item.equals(procName)){
89                 int val = proc.get(procName);
90                 r_queue.put(item,val);
91             }
92         }
93     }

```

```

93     //System.out.println(readyTemp);
94 }
95 else
96     System.out.println("Error: The Process name is not in the Suspended Queue");
97
98 rescheduler();
99 for(Map.Entry<String,Integer> queue : r_queue.entrySet()){
100     if(!queue.getKey().equals(""))
101         System.out.println("Process name: " + queue.getKey());
102     }
103 }
104
105 public void kill(String procName){
106
107     if(!s_queue.isEmpty()){
108         if(s_queue.contains(procName)){
109             s_queue.remove(procName);
110         }
111     }
112     else if(r_queue.containsKey(procName)){
113         r_queue.remove(procName);
114     }
115     else if (procName.equals(cr)){
116         cr=" ";
117         highestPriority = 0;
118     }
119     else{
120         System.out.println("The Process does not exist");
121     }
122
123
124     rescheduler();
125     for(Map.Entry<String,Integer> queue : r_queue.entrySet()){
126         if(!queue.getKey().equals("")){
127             System.out.println("The Current Process is " + cr);

```

```

128         System.out.println("Process name: " + queue.getKey()); }
129     }
130 }
131 public void rescheduler(){
132     val = highestPriority;
133     str = cr;
134
135     for (Integer item : r_queue.values()){
136         if(item > highestPriority){
137             highestPriority = item;
138         }
139     }
140 }

```

```

0
1      if(highestPriority > val){
2          int oldval = val;
3          val = highestPriority;
4          r_queue.put(str,oldval);
5          for(Map.Entry<String,Integer> entry : r_queue.entrySet()){
6              if (entry.getValue() == highestPriority){
7                  cr = entry.getKey();
8                  str = cr;
9                  val = entry.getValue();
10             }
11         }
12     }
13     r_queue.remove(str);
14     System.out.println("Contents of r_queue:");
15

```

## 6. How the code works

The code is designed to *create* a process with a given name and priority, when the process is created it is inserted in a suspended queue is a Linked List Queue that holds the process names that have been created. The User will choice the sequence of steps he/she would like to follow to see states change. The other system calls the user can perform are *resume* and *kill*, with their respective arguments. The *resume* system call uses a treeMap data structure to store the { Key: Value } pairs. If a process that does not exist and is resumed or killed, the program will provide the appropriate message. As the system calls take place, the program will show the contents of the ready queue and as it displays the queue if any new process is resumed the program will iterate through the ready queue looking to see if the new current process needs to be updated. A current process gets change only if there is a process in the ready queue that has a priority higher than that of the current process.

## 7. Limitations

The limitations that come with the code is the aspect where we are not able to store our ready processes in an object queue because Queue in java does not store key: pair Entries.