Bachelor of IT (Computer Science)
Assignment Title
Course

Dane Madsen
n10983864@qut.edu.au

Contents

1 Algorithm Design

1.1 Jobs ADT

1.1.1 IsValidId Method

This method checks whether a provided job ID is valid. It achieves this by checking that the provided ID is greater than the minimum valid ID (1) and less than the maximum valid ID (999). If the ID is meets these criteria, the method returns true indicating the ID is valid, otherwise it returns false indicating the ID is invalid.

```
ALGORITHM IsValidId(v)
  // Given an integer ($v$)
  // Returns True if $v$ is a valid job ID
  // Otherwise returns False
  if $v \geq 1$ and $v \leq 999$
      return True
  else
      return False
```

1.1.2 IsValidExecutionTime Method

This method simply checks whether a provided job execution time is valid. It achieves this by simply checking whether the execution time is greater than 0. If the execution time is greater than 0, the method returns true indicating the execution time is valid, otherwise it returns false indicating the execution time is invalid.

```
ALGORITHM IsValidExecutionTime(v)
  // Given an integer ($v$)
  // Returns True if $v$ is a valid job execution time
  // Otherwise returns False
  if $v > 0$
      return True
  else
      return False
```

1.1.3 IsValidPriority Method

This method checks whether a provided job priority is valid. It achieves this by checking that the provided priority is greater than or equal to the minimum valid priority (1) and less than or equal to the maximum valid priority (9). If the priority is meets these criteria, the method returns true indicating the priority is valid, otherwise it returns false indicating the priority is invalid.

```
ALGORITHM IsValidPriority(v)
  // Given an integer ($v$)
  // Returns True if $v$ is a valid job priority
  // Otherwise returns False
  if $v \geq 1$ and $v \leq 9$
      return True
  else
      return False
```

1.1.4 IsTimeReceived Method

This method checks whether a provided job time received is valid. It achieves this by checking that the provided time received is greater than zero. If the time received is greater than zero, the method returns true indicating the time received is valid, otherwise it returns false indicating the time received is invalid.

```
ALGORITHM IsTimeReceived(v)
  // Given a job time received ($v$)
  // Returns True if $v$ is a valid time received
  // Otherwise returns False
  if $v > 0$
      return True
  else
      return False
```

1.2 JobCollection ADT

1.2.1 Add Method

This method adds a job to the job collection. It achieves this by first checking that the job doesn't already exist in the collection. If the job does already exist in the collection, the method returns false indicating the job was not added to the collection. If the job does not already exist in the collection, the method adds the job to the collection, increments the count variable and returns true.

```
ALGORITHM Add(v)
  // Let ($n$) be count
  // Given a job ($v$)
  // Returns True if $v$ was added to the jobs array ($J$)
  // Otherwise returns False
  for $i \leftarrow 0$ in $n$ - 1 do
      if $v.id = J[i].id$
          return False
  else
      $J[n] \leftarrow v$
      $n \leftarrow n + 1$
      return True
```

1.2.2 Contains Method

This method is used to check if a job exists in the job collection. It achieves this by checking if there is a job in the collection with the same ID as the provided job ID. If there is a job with the same ID the method returns true, otherwise it returns false.

```
ALGORITHM Contains(v)
  // Let ($n$) be count
  // Given an integer ($v$)
  // Returns True if a job with the ID $v$ exists in the jobs array ($J$)
  // Otherwise returns False
  for $i \leftarrow 0$ in $n$ - 1 do
      if $v = J[i].id$
          return True
  else
      return False
```

1.2.3 Find Method

This method is used to find a job in the job collection. It achieves this by using the provided job ID to find the job in the collection. If the job is found, the method returns the job, otherwise it returns null.

```
ALGORITHM Find(v)
  // Let ($n$) be count
  // Given an integer ($v$)
  // Returns the job with the ID $v$ if it exists in the jobs array ($J$)
  // Otherwise returns null
  for $i \leftarrow 0$ in $n$ - 1 do
      if $v = J[i].id$
          return J[i]
  return null
```

1.2.4 Remove Method

This method is used to remove a job from the job collection. It achieves this by using the provided job ID to find the job in the collection. If the job is found, the method removes the job from the collection, decrements the count variable and returns true. If the job is not found, the method returns false.

```
ALGORITHM Remove(v)
  // Let ($n$) be count
  // Given an integer ($v$)
  // Returns True if a job with the ID $v$ was removed
  // from the jobs array ($J$)
  // Otherwise returns False
  for $i \leftarrow 0$ in $n$ - 1 do
      if $v = J[i].id$
              for $j \leftarrow 0$ in $n$ - 2 do
                  $J[j] \leftarrow J[j + 1]$
              $n \leftarrow n - 1$
              return True
  return False
```

1.2.5 ToArray Method

This method is used to convert the job collection to an array. It achieves this by creating a new array of the same size as the job collection and then copying the jobs from the job collection to the new array. The method then returns the new array.

```
ALGORITHM ToArray()
  // Let ($n$) be count
  // Returns a new array of copied from the jobs array ($J$)
  $A \leftarrow newJob[n]$
  for $i \leftarrow 0$ in $n$ - $1$ do
        $A[i] \leftarrow J[i]$
  return $A$
```

1.3 Scheduler ADT

1.3.1 FirstComeFirstServed Method

This method is used to schedule jobs using the first come first served algorithm. It achieves this by sorting the jobs in the job collection by their arrival time and then adding them to the schedule in the order they were sorted.

```
ALGORITHM FirstComeFirstServed()
  // Let ($n$) be count
  // Returns a new array of jobs sorted by their arrival time
  $A \leftarrow J.ToArray()$
  for $i \leftarrow 0$ in $n$ - $1$ do
        for $j \leftarrow 0$ in $n$ - $2$ do
              if $A[j].arrivalTime > A[j+1].arrivalTime$
                    $temp \leftarrow A[j]$
                    $A[j] \leftarrow A[j+1]$
                    $A[j+1] \leftarrow temp$
  return $A$
```

2 Analysis

2.1 Jobs ADT

2.2 JobCollection ADT

2.3 Scheduler ADT

3 Testing

3.1 Jobs ADT

3.2 JobCollection ADT

3.3 Scheduler ADT