

**ECU178 Computer Science:  
210CT - Programming, Algorithms and Data  
Structures Portfolio**

Due on Monday, December 15th, 2014

*Dr James Shuttleworth*

**Robert Rigler : 4939377**

## Contents

<b>Item 1: Week 3 - Linear Search and Duplicate Finder</b>	<b>3</b>
Pre-Homework 1: Write a Program that displays your name 10 times . . . . .	3
Pre-Homework 2: Write a function that draws a square of stars given as a parameter	4
Pre-Homework 3: Write a program to open a file and display it's contents in capitals	5
1. Pseudocode for linear search . . . . .	6
2. Pseudocode for finding duplicates in a list . . . . .	6
<b>Item 2: Week 4 - Time complexities and Big-O notation</b>	<b>7</b>
1. Describe the runtime bounds of the linear search algorithm . . . . .	7
2. Describe the runtime bounds of the duplicate finder algorithm . . . . .	7
Additional work: Critical values of relative runtimes . . . . .	7
<b>Item 3: Week 6 - Harmonic Series or Pivot Selection</b>	<b>9</b>
<b>Item 4: Week 7 - Heapworksheet or RPN Calculator</b>	<b>10</b>
<b>Item 5: Week 8 - Linked List Delete function or Linked List Sortings</b>	<b>11</b>


## Item 1: Week 3 - Linear Search and Duplicate Finder

### Pre-Homework 1: Write a Program that displays your name 10 times

Listing 1: NameRepeat class JAVA code

```
1  /**
2   * Created by Rob on 23/10/2014.
3   */
4  public class NameRepeat {
5
6      public static void main(String[] args){
7
8          NameRepeat myObject = new NameRepeat(); /*Create Object*/
9          myObject.PrintName("Rob"); /*Use object to call PrintName() method*/
10     }
11
12     public void PrintName(String _name){
13
14         for (int i = 0; i<10;i++){ /* Loop 10 times*/
15             System.out.println((i+1) + " " + _name);
16             /*, print the number && _name parameter each time.*/
17
18         }
19
20     }
21 }
22 }
```

### Evidence



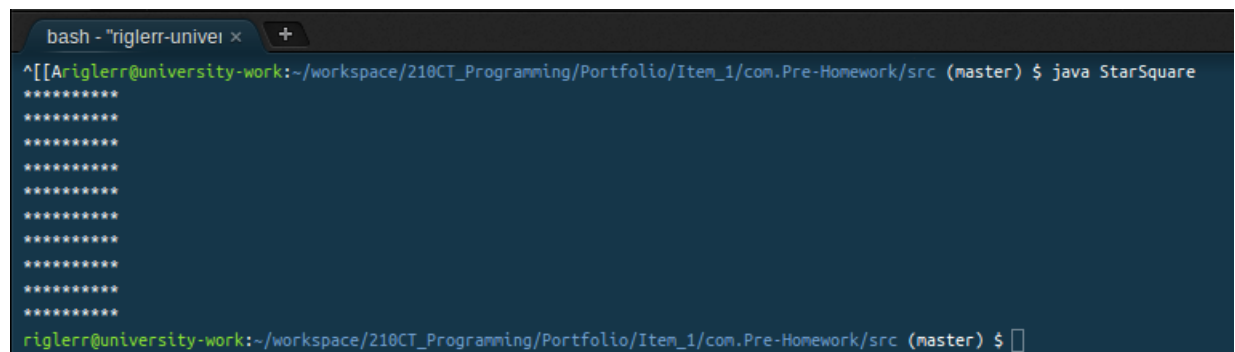
The screenshot shows a terminal window titled "bash - 'riglerr-univer x" with a plus icon and window controls. The prompt is "riglerr@university-work:~/workspace/218CT\_Programming/Portfolio/Item\_1/com.Pre-Homework/src (master) \$". The command "java NameRepeat" has been executed, resulting in ten lines of output: "1 Rob", "2 Rob", "3 Rob", "4 Rob", "5 Rob", "6 Rob", "7 Rob", "8 Rob", "9 Rob", and "10 Rob". The prompt is now "riglerr@university-work:~/workspace/218CT\_Programming/Portfolio/Item\_1/com.Pre-Homework/src (master) \$". The terminal has a dark blue background with green text. A green status bar at the bottom shows "[riglerr@u 0: bash]" and "riglerr-university-wor 17:23 24-Nov-14".

## Pre-Homework 2: Write a function that draws a square of stars given as a parameter

Listing 2: StarSquare class JAVA code

```
1  /**
2   * Created by Rob on 23/10/2014.
3   */
4  public class StarSquare {
5
6      public char ast = '*'; //Create variable to hold asterisk character.
7      public static void main(String[] args){
8
9          StarSquare sSquare = new StarSquare(); /*Create Object*/
10         sSquare.writeSquare(10); /*Use object to call writeSquare() method*/
11
12     }
13
14
15     public void writeSquare(int size){
16
17         for(int i =0; i<size;i++){ /*OuterLoop 'size' times*/
18             for(int j = 0;j<size;j++){ /*InnerLoop 'size' times*/
19
20                 System.out.print(ast); /*Print line of asterisks*/
21
22             }
23             System.out.println(); /* Start new line when inner loop finishes*/
24         }
25     }
26 }
```

### Evidence



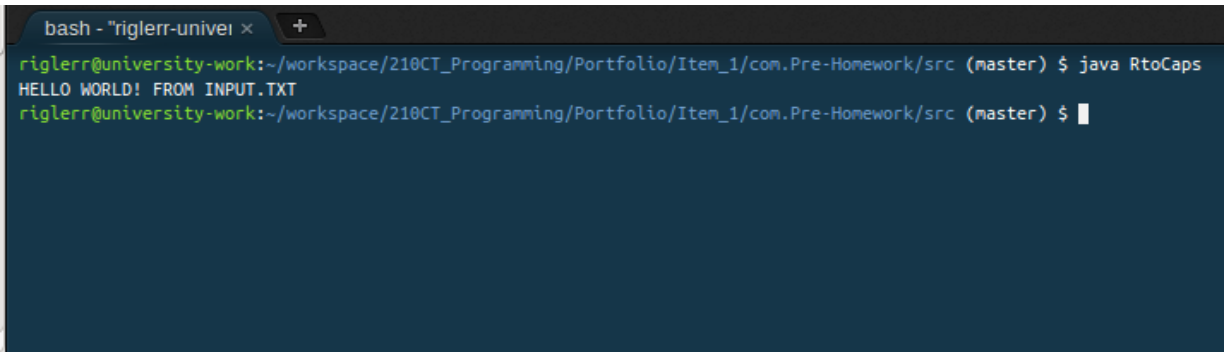
```
bash - "riglerr-univer" x +
^[[Ariglerr@university-work:~/workspace/210CT_Programming/Portfolio/Item_1/con.Pre-Homework/src (master) $ java StarSquare
*****
*****
*****
*****
*****
*****
*****
*****
*****
*****
riglerr@university-work:~/workspace/210CT_Programming/Portfolio/Item_1/con.Pre-Homework/src (master) $
```

## Pre-Homework 3: Write a program to open a file and display it's contents in capitals

Listing 3: RtoCaps class JAVA code

```
1  /**
2   * Created by Rob on 23/10/2014.
3   */
4  import java.io.File;
5  import java.io.FileNotFoundException;
6  import java.util.Scanner;
7
8  public class RtoCaps {
9
10     public static void main(String[] args) throws FileNotFoundException {
11         File inFile = new File("input.txt");
12         /*Create a file object */
13         RtoCaps obj = new RtoCaps(); /*Create class object*/
14         obj.rInput(inFile); /*Use Class object to call rInput() method*/
15     }
16
17     public void rInput(File inFile) throws FileNotFoundException{
18
19         /*Create a new scanner to read from the file*/
20         Scanner in = new Scanner(inFile);
21
22         /*Loop While there is still lines left in the document*/
23         while(in.hasNextLine())
24         {
25             /* Place the next line in a strin varibale*/
26             String line = in.nextLine();
27
28             /* Print the line in uppercase*/
29             System.out.println(line.toUpperCase());
30         }
31
32     }
33
34
35 }
```

### Evidence

A terminal window with a dark background and light-colored text. The title bar shows 'bash - "riglerr-univeri x' and a '+' icon. The terminal content shows a user prompt 'riglerr@university-work:~/workspace/210CT\_Programming/Portfolio/Item\_1/com.Pre-Homework/src (master) \$' followed by the command 'java RtoCaps'. The output is 'HELLO WORLD! FROM INPUT.TXT'. The prompt is repeated on the next line with a cursor at the end.

```
bash - "riglerr-univeri x +
riglerr@university-work:~/workspace/210CT_Programming/Portfolio/Item_1/com.Pre-Homework/src (master) $ java RtoCaps
HELLO WORLD! FROM INPUT.TXT
riglerr@university-work:~/workspace/210CT_Programming/Portfolio/Item_1/com.Pre-Homework/src (master) $
```

## 1. Pseudocode for linear search

---

### Algorithm 1 LinearSearch

---

```
procedure BOOL LINEARSEARCH(item, list[])
  for each element i in list do
    if list[i] = list then
      return true
    end if
  end for
  return false
end procedure
```

---

## 2. Pseudocode for finding duplicates in a list

---

### Algorithm 2 Examining for duplicates

---

```
procedure BOOL EXFORDUPES(list[])
  for each element i in list[] do
    for each element j in list[] do
      if list[i] = list[j] then
        return true
      end if
    end for
  end for
end procedure
```

---

## Item 2: Week 4 - Time complexities and Big-O notation

### 1. Describe the runtime bounds of the linear search algorithm

---

**Algorithm 3** LinearSearch

---

```
procedure BOOL LINEARSEARCH(item, list[ ])

    for each element i in list do           (n)
        if list[i] = item then t           (n)
            return true                      (n)
        end if
    end for
    return false                             (1)
end procedure
```

---

The time complexity of the algorithm is  $O(n)$

### 2. Describe the runtime bounds of the duplicate finder algorithm

---

**Algorithm 4** Examining for duplicates

---

```
procedure BOOL EXFORDUPES(list[ ])

    for each element i in list[ ] do         (n)
        for each element j in list[ ] do     (n*n)
            if list[i] = list[j] then       (n*n)
                return true                  (n*n)
            end if
        end for
    end for
    return false                             (1)
end procedure
```

---

The time complexity of the algorithm is  $O(n^2)$

### Additional work: Critical values of relative runtimes

Write a function that determines the critical value at which the relative runtime of two linear algorithms swap.

---

**Algorithm 5** Relative runtime comparison algorithm

---

```
procedure CRITVAL( $m1, k1, m2, k2$ )  
   $switch \leftarrow false$   
   $n \leftarrow 0$   
  if  $((m1 * n) + k1) > ((m2 * n) + k2)$  then  
    while ! $switch$  do  
      if  $((m1 * n) + k1) < ((m2 * n) + k2)$  then  
         $switch \leftarrow true$   
      else  
         $n++$   
      end if  
    end while  
  else  
    while ! $switch$  do  
      if  $((m1 * n) + k1) > ((m2 * n) + k2)$  then  
         $switch \leftarrow true$   
      else  
         $n++$   
      end if  
    end while  
  end if  
  return  $n$   
end procedure
```

---



## **Item 3: Week 6 - Harmonic Series or Pivot Selection**

## **Item 4: Week 7 - Heapworksheet or RPN Calculator**

## **Item 5: Week 8 - Linked List Delete function or Linked List Sortings**