

**Department of Computer Sciences**

**College of Computer Science and Information Technology**

**CS 211 Lab Manual :**

**Lab 02 [1 mark]**

**Topic: labRunning Time Analysis (Lecture 2)**

# **Minimal Java template for all sets**

package lab02.set01 ;

import java.util.\*;

import java.io.\*;

public class Main {

// Simple stopwatch helpers

static long nowMs() { return System.nanoTime() / 1\_000\_000L; }

public static void main(String[] args) throws Exception {

// Change which problem to run by uncommenting the method call

//problem1();

//problem2();

//problem3();

//problem4();

}

// Implement problems here

static void problem1() {}

static void problem2() {}

static void problem3() {}

static void problem4() {}

}

Students will uncomment one problem at a time, run, capture outputs, then move on.

# SET 02

## Problem 1 [0.25 mark]: Tracing loop

a. Write a for loop according to the following pseudo code

For i = 1 to n  
 Print(i)  
 i = i \* 2  
EndFor

b. Manual trace for n = 20: i prints 1, 2, 4, 8, 16.  
c. Run for n = 20 and check the console matches the manual list.  
d. Change to decreasing loop as. i = n down to 1,and repeat steps b and c

Manual check for n = 20: 20, 10, 5, 2, 1.

## Problem 2 [0.25 mark]: Find function count

1. Modify to count iterations:

count = 0;

for i = 1 to n

count++

i = i + 2

print (count)

1. Run the program for n = 10, 100, 1000, 10000, 100000, … and see the count
2. Find a function for count in term of n. Is it count = log2n?
3. Take a calculator to find the value of the function (say log2n) for n = 10, 100, 1000, 10000, 100000, … and match that value with your output count
4. Capture the running time of the algorithm
5. Change the line “ i = i+ 2 “ to “i = i\*2”” and repeat the above steps (a to e)

## Problem 3 [0.25 mark]: Trace nested loops

1. Write two nested for loops as follows

n = 10

for i = 1 to n

for j = 1 to **n**

Print(i, “ “, j)

1. Write manually by **pen and paper** the output of the code.
2. Run and compare your results with the answer you got in b above.
3. Capture the running time of the algorithm

## Problem 4 [0.25 mark]: Plot run time - cubic triple loop

1. Write code according to the following pseudo code

start = currentTimeMs()  
step = 0  
For i = 1 to n  
 For j = 1 to n

For k = k to n  
 step = step + 1  
 EndFor  
 EndFor

EndFor  
end = currentTimeMs()  
Print(n, end - start)

1. Run the code for n = 1000, 5000, 10000, 50000, 100000, 500000, … so on
2. Plot n vs time taken (ns) by the code. How does this plot look like? It should be like a cubic curve. Search internet to see how a cubic curve looks like and match with your curve
3. Take an online plotter and plot a curve there with these data (n and ms). Then match that curve with your curve

# What to submit

1. One Word document containing for each problem: brief manual traces, small screenshots of console outputs, and the table of n vs time where required. Put your Name and Student ID on the first page and in the file name.
2. Java source files for each set you attempted. Filename pattern: Main.java inside package lab02.set01.
3. Comment your code with a header: name, ID, section, date, and a short note about expected complexity for each problem.

# Marking reminders

* Correct loop logic and correct manual trace for Problem 1 - 0.25
* Correct counts and justified function for Problem 2 - 0.25
* Correct nested trace and validation for Problem 3 - 0.25
* Correct timing runs and a readable plot or table for Problem 4 - 0.25
* Proper comments and student identification are required to receive full credit.