**CIS163 Lab**

**Big O lab: (Algorithm analysis)**

**Preparation:**

Attended class on Big 0 and read Chapter on Big O / Algorithm analysis.

**Objectives** (after completing the lab you will be able to:)

* Understand how to determine the Big O of a program
* Validate your Algorithm Analysis process
* Instrument code.

**Activities:**

1. Determine the Big 0 for each problem below
2. For each problem, provide the order (Big O) of the execution speed. For problems 1 - 3, also determine the exact execution speed.
3. Using the space provided, graph the execution speed for each problem below, use open office spread sheet (or similar program) to do so

|  |  |
| --- | --- |
| **Problem 1:**  for (int i = 0; i < n; i++) {      SomeStatement  } | **Big O: N**  **Exact Execution speed:**  1, n+1, n, n  = 3n+2 |

public class CountIt {

public int linearLoop(int N) {

int x = 0;

int count = 0;

x++;

while (count < N) {

x++; // while loop

x++; // stmt

System.out.println(".");

x++; // i++

count++;

}

X++;

return x;

}

public static void main(String[] args) {

CountIt ex = new CountIt();

int N = 400000;

long start = System.currentTimeMillis();

System.out.println ("x = " + ex.linearLoop(N));

Long endTime = System.currentTimeMillis();

System.out.println(

"\nN = " + N + ". Loops ran in: " + (endTime - start));

}

}

**Your task: Cut and paste this code into IntelliJ and using different values for the parameter (e.g., 20000, 40000, 60000) graph the different times vs parameter values.**

**Your task: Compare your return value (i.e., x = …) to your “Exact Execution speed:” equation to see if they match. Using the space provided, graph the execution speed for each problem below, use open office spread sheet (or similar program) to do so.**

Time

Different parameter values (N)

|  |  |
| --- | --- |
| **Problem 2:**  for (int i = 0; i < n; i++) {      SomeStatement      for (int j = 0; j < n; j++)  SomeStatement  } | **Big O: N^2**  **Exact Execution speed:**  1, n+1, n, n, n(n+1), n(n), n(n), n,  = 3n^2 + 5n + 2 |

public class CountIt {

public long SnippetNestedLoop(long n) {

long i, j, x = 0;

i = 0;  x++;

while (i < n) {  x++;    // i < n

x++;    // SomeStatement

j = 0; x++;      // j = 0;

while (j < n) { x++;   // j < n

x++;   // SomeStatement

j++; x++;   // j++;

}

x++; // Can you explain why is this here?

i++; x++;  // i++;

}

x++; // Can you explain why is this here? Ans: i < n

return x;

}

public static void main(String[] args) {

CountIt r = new CountIt();

Long t = System.currentTimeMillis();

System.out.println("x = " + r.SnippetNestedLoop(100));

System.out.println ("Time:" + (System.currentTimeMillis() - t));

}

}

**Your task: Cut and paste this code into IntelliJ and using different values for the parameter (e.g., 100, 1000, 10000) graph the different times vs parameter values.**

**Your task: Compare your return value (i.e., x = …) to your “Exact Execution speed:” equation to see if they match. Using the space provided, graph the execution speed for each problem below, use open office spread sheet (or similar program) to do so.**

Time

Different parameter values (N)

|  |  |
| --- | --- |
| **Problem 3:**  int i = 1;  while (i < n) {  SomeStatement  i = i \* 2;  } | **Big O: log(N)**  **Exact Execution speed:**  1, ceil(log(n))+1, ceil(log(n)), ceil(log(n))  = 3\*ceil(log(n)) + 2 |

public class CountIt {

public long SnippetLog(long n) {

long i, j, x = 0;

i = 1; x++;

while (i < n) { x++;

x++;  // SomeStatement

i = i \* 2; x++;

}

x++; // Can you explain why is this here?

return x;

}

public static void main(String[] args) {

CountIt r = new CountIt();

Long t = System.currentTimeMillis();

System.out.println("x = " + r. SnippetLog (1000));

System.out.println ("Time:" + (System.currentTimeMillis() - t));

}

}

**Your task: Determine the Big O and execution speed for the code above. Next, cut and paste the code above into IntelliJ and to graph the different times vs parameter values using different values for the parameter (e.g., 1000, 10000)**

**Your task: Compare your return value (i.e., x = …) to your “Exact Execution speed:” equation to see if they match. Using the space provided, graph the execution speed for each problem below, use open office spread sheet (or similar program) to do so.**

Time

Different parameter values (N)

|  |  |
| --- | --- |
| **Problem 4: Fibonacci (see code below)** | **Big O:** |

public class CountIt {

private long fib(int n) {

if (n < 3) {

return 1;

else

return fib(n - 1) + fib (n - 2);

}

}

public static void main(String[] args) {

CountIt r = new CountIt();

Long t = System.currentTimeMillis();

int N = 10;

System.out.println("Fib of " + N + " = " + r. fib (N));

System.out.println ("Time:" + (System.currentTimeMillis() - t));

}

}

**Your task: Determine the Big O and cut and paste the code above into IntelliJ to graph the different times vs parameter values using different values for the parameter (e.g., 10, 50, 55)**

**Your task: Compare your return value (i.e.,** Fib of **…) to the expected value. Using the space provided, graph the execution speed for each problem below, use open office spread sheet (or similar program) to do so.**

Time

Different parameter values (N)

|  |  |
| --- | --- |
| **Problem 5: non-recursive form of zero-fill method in project 2** | **Big O:** |

For example:

public static void main(String[] args) {

Minesweeper game = new Minesweeper (100, 1); // board size of 100 with 1 mine

Long t = System.currentTimeMillis();

Game.select (2,3) ;

System.out.println ("Time:" + (System.currentTimeMillis() - t));

}

**Try different values of board size, I suggest 100, 200, and so on. Plot the results below. Using the space provided, graph the execution speed for each problem below, use open office spread sheet (or similar program) to do so**

Time

N

Different parameter values (N) where N is the board size