**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
   1. ***Task’* Create a package** (project if needed) and create a class named **“Countit”**
   2. ***Task’*** Cut the code from each section into a class
   3. ***Execute the program with different N values record the different times.***
2. ***Task’*** For each problem, provide the order (Big O) of the execution speed for that program. For problems 1 - 3, also determine the exact execution speed.
3. ***Task’*** Using the space provided, graph the execution speed for each problem below, use open office spread sheet (or similar program) to do so
4. Share you screen (if online) to show your answers (exact execution speed, Big O, graph)

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

public class CountIt {

public int linearLoop(int N) {

int x = 0;

int i = 0; x++;

while (i < N) { x++; // while loop

x++; // stmt //

System.out.println("."); // slow things do, does not count

i++; x++;

}

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., 200000, 400000, 600000) 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: O(n^2)**  **Exact Execution speed: 3n^2 + 5n + 2**  **5636** |

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(100000));

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., 100000, 200000, 400000, 800000) 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  SomeStatement  i = i \* 2;  } | **Big O: O(log(n))**  **Exact Execution speed: 4\*log(n) + 2**  (This one is tough)! |

public class CountIt {

public long SnippetLog(long n) {

long i, j, x = 0;

i = 1; x++;

while (i < n) { x++;

x++;  // SomeStatement

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 (1000000));

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., 1000000, 2000000, 8000000, 10000000)**

**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: O(2^N)** |

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 = 20;

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., 20, 30,40,50, 60)**

**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)