# CS 2420 Python Spring 2020 Test 1

Name:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 200 Points - closed book and notes.

**Logs and Exponents:**

1. What is (approximately) the Log base 2 of 130,000,000?

2x = 1,000,000 -> x = 20

2x = 130 -> x = 7

log2130000000 -> 2x = 130000000 -> -> x = 27

1. What is (approximately) the Log base 4 of 1,000,000?

10

1. Assume a CIT password must be exactly 9 characters. The first 3 characters must be a lowercase letter. The next 3 may be lower case letters or uppercase letters. The last 3 must be one of the ten digits. How many different password possibilities are there? (Don’t do the math – just write the formula.)

26\*26\*26\*52\*52\*52\*10\*10\*10

263\*523\*103

**Big O:**

1. What is Big O analysis?

The study of how work increases as problem size increases

1. When is Big O analysis not important?

When the sample size is small

1. Which Big O category does this code use?

For loop N times

For loop N times

Do N2 work here.

Do N3 work here.

1. N4
2. N5 answer
3. N6
4. N7
5. List 5 Big O categories, and an algorithm that fits into each.

1 – Push on a stack, Python append

log2N – Binary Search

N – Counting Search

N\*log2N – Merge Sort

N2 – Bubble Sort

N3 – Naive subsequence, card shuffling

2N – Towers of Hanoi, SAT

10(N/2) – factoring

N! – Traveling Salesman

1. Show what happens to this array after one pass of the quick sort. Use the 4 on the far left as the pivot, and use the same algorithm we studied in class.

Original Array: 4 7 2 1 8 3 9 5

A. 3 2 1 4 8 7 9 5

B. 3 2 1 4 7 8 9 5

C. 2 1 3 4 8 7 9 5

D. 1 2 3 4 5 7 8 9

4 = pivot

4 2 7 1 8 3 9 5

4 2 1 7 8 3 9 5

4 2 1 7 8 3 9 5

4 2 1 3 8 7 9 5

4 2 1 3 8 7 9 5

4 2 1 3 8 7 9 5

3 2 1 4 8 7 9 5

1. Fill out this table with Correct Big-O catagories

|  |  |  |  |
| --- | --- | --- | --- |
|  | Expected Case | Worst Case | Best Case |
| Shaker Sort | N2 | N2 | N |
| Quick Sort | N\*LogN | N2 (if mostly sorted) | N\*LogN |
| Merge | N\*LogN |  |  |

1. If you are trying to speed up code that is running slowly, where is the bottle neck code likely to be?

Where the worst Big-O is located.

1. What happens to the overall speed of execution if you make all the code that is not the bottleneck code go 1000 times faster?

It won’t change much.

1. Why is it more likely that Factoring might be solved in polynomial time than SAT?

Factoring is not NP complete.

**Sorting:**

**Data** BubbleSort ShakerSort SelectionSort MergeSort QuickSort MQuickSort HashSort

3 5.39 4.16 4.8 4.58 4.45 3.8 3.0

4 7.9 5.39 6.9 6.0 6.72 5.24 4.0

5 9.95 6.49 8.95 7.32 8.77 6.75 5.0

6 11.97 7.53 10.97 8.58 10.97 8.11 6.0

7 13.97 8.56 12.98 9.8 12.96 9.38 7.0

8 15.99 9.57 14.99 11.0 14.99 10.64 8.0

9 17.99 10.57 16.99 12.16 16.99 11.85 9.0

10 19.99 11.58 18.99 13.32 18.98 13.03 10.0

11 21.99 12.58 20.99 14.45 20.99 14.2 11.0

12 23.99 13.58 22.99 15.58 22.99 15.35 12.0

**Size**

1. Given the chart above which measures compares on mostly sorted data, approximately how many compares does the QuickSort require when the data is 512 big?

16.99 -> 17

217 = 128K

1. If you want to keep the number of compares at or below 4K on the BubbleSort, how big can your data set be?

11.97 -> at row 6

26 -> 64

**Graphing Calculator:**

1. Convert this infix expression to postfix: 2\*(3+4/5\*6)/(9-4+3/2)\*8

answer

1. Evaluate this postfix expression: 123\*+4-6\*82-/

answer

**Homework Review:**

1. (20 points) Write Python code that finds and prints the average age of all Students in the Python list named “AllStudents”. Assume “AllStudents” is ready to go. Assume the Student class has a GetAge method.

total = 0

for s in AllStudents:

total += s.getAge()

print(total/len(AllStudents))

1. (20 points) Write Python code for the Push method of your Stack class.

def push(self, item):

self.A.append(item)

Review the topics listed in the Schedule (cit.dixie.edu)

Study the Sort, Retrieve, Tranverse, Delete Functions

Study Powers of 2

Study Big-O categories