MITx: 6.00.1x Introduction to Computer Science and Programming U.. **Help**

Bookmarks

Week 6: Algorithmic Complexity > 11. Computational Complexity > Exercise 3

Exercise 3

Bookmark this page

Exercise 3

9 points possible (graded)

ESTIMATED TIME TO COMPLETE: 12 minutes

For the following programs, fill in the best-case and the worst-case number of steps it will take to run each program.

For these questions, you'll be asked to write a mathematical expression. Use +, -, / signs to indicate addition, subtraction, and division. Explicitly indicate multiplication with a * (ie say "6*n" rather than "6n"). Indicate exponentiation with a caret (^) (ie "n^4" for n^4). Indicate base-2 logarithms with the word log2 followed by parenthesis (ie "log2(n)").

1. Program 1:

```
def program1(L):
    multiples = []
    for x in L:
        for y in L:
            multiples.append(x*y)
    return multiples
```

What is the number of steps it will take to run Program 1 in the best case? Express your answer in terms of *n*, the number of elements in the list `L`. You can assume list appending takes 1 step.

What is the number of steps it will take to run Program 1 in the worst case? Express your answer in terms of *n*, the number of elements in the list `L`. You can assume list appending takes 1 step.

2. Program 2:

```
def program2(L):
    squares = []
    for x in L:
        for y in L:
            if x == y:
                squares.append(x*y)
    return squares
```

What is the number of steps it will take to run Program 2 in the best case? Express your answer in terms of *n*, the number of elements in the list `L`.

What is the number of steps it will take to run Program 2 in the worst case? Express your answer in terms of *n*, the number of elements in the list `L`.

3. Program 3:

```
def program3(L1, L2):
    intersection = []
    for elt in L1:
        if elt in L2:
            intersection.append(elt)
    return intersection
```

What is the number of steps it will take to run Program 3 in the best case? Express your answer in terms of *n*, the number of elements in the list `L1` (assume `len(L1) == len(L2)`).

What is the number of steps it will take to run Program 3 in the worst case? Express your answer in terms of *n*, the number of elements in the list `L1` (assume `len(L1) == len(L2)`).

4. In the last video, Professor Grimson introduced the idea of "asymptotic complexity", which means we describe running time in terms of number of basic steps. We've described the best- and worst-case running times in terms number of basic steps for the three programs above. Now, we'd like you to give the complexity order (ie, "Big O" running time) of each of the above programs.

   Recall that "Big O" notation gives an upper bound on asymptotic growth of a function. So, should you use the best-case or the worst-case running times for each program? Review the video again if you're unsure of what to put for the following boxes.

   Note: Your answer should be expressed with a capital letter O, then a mathematical term similar to one described in the introduction to this problem - for example, O(n^5).

   1. What is the complexity order of Program 1?

2. What is the complexity order of Program 2?

3. What is the complexity order of Program 3?

Reminder: You do not lose points for trying a problem multiple times, nor do you lose points if you hit "Show Answer". If this problem has you stumped after you've tried it a few times, feel free to reveal the solution.

Click the "Reset" button to clear your answers.

Submit

Exercise 3

Topic: Lecture 11 / Exercise 3

Show Discussion