Advanced Topics in Comp Sci — Activity—Stack Comparison

ASSIGNMENT OVERVIEW

In this assignment you'll compare the performance of a list-based `Stack` class and an `UnorderedList`-based stack (`UnorderedListStack`).

This assignment is worth 30 points and is due on the *crashwhite.polytechnic.org* server at 23:59:59 on the date given in class.

BACKGROUND

You are familiar with the abstract data type (ADT) "stack", which includes the operations:

- `push(item)`
- `pop()`
- `peek()`
- `size()`
- `isEmpty()`

Because we've used two different strategies for implementing the Stack ADT, we'd expect there might be differences in the performance of each one. Which one do you think will be faster? and why?

PROGRAM SPECIFICATION

Write the program `stack_comparison.py` that compares performance of your `UnorderedList/Node` stack and your list-based `Stack`. After collecting your data, write a paragraph in a comment at the beginning of that program relating the outcome of running that program and your analysis of the possible reasons why.

DELIVERABLES

`stack_comparison.py`

This single file will import from your `atds.py` class as needed to collect the data for this analysis.

To submit your assignment for grading, copy your file to your directory in `/home/studentID/forInstructor/` at *crashwhite.polytechnic.org* before the deadline.

ASSIGNMENT NOTES

- The `stack_comparison.py` program relies on a functioning `atds.py` package. Make sure your `atds.py` program uses the standard methods that we've developed in class to ensure that your program will run with the instructors.
- Use the `time` module to collect data on how long it takes to perform a given number of `push()` operations with each strategy. Also, collect data on how long it takes to perform a given number of `pop()` operations.
- Your analysis paragraph should be well written, in standard English, with an explanation of what you did and what kind of results you got. You may refer to the "expense" of certain operations in each algorithm in terms of *time* or *memory* used. You do not need to perform a detailed Big-O performance analysis for this assignment.

GETTING STARTED

1. Ensure that your `atds.py` file is working as it should, especially:
   - `Stack`
   - `Node`
   - `UnorderedList` (which uses `Node`)
   - `UnorderedListStack` (which uses `UnorderedList`)

2. In the new file `stack_comparison.py`, import the classes for your `atds.py` and set up the tests as indicated.

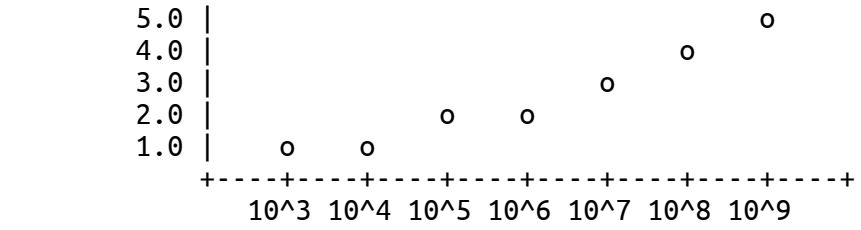
3. Once you've collected your data, write as a large multi-line comment at the beginning of the program the paragraph reporting those results and your analysis.

4. When your program is completed (but before the deadline), copy `atds.py` to the server as indicated above.

EXTENSIONS

1. Perform a more extensive analysis to determine the efficiency of the algorithms. Change the size of the input systematically and identify how the time to run the algorithm changes as a result. Do the `push()` operations for the two different stack implementations have the same efficiency? Do the `pop()` operations?

2. As an interesting graphics exercise, take a set of data that you've collected for either one of the operations and create a text-based graph of that information such as the one shown here.



QUESTIONS FOR YOU TO CONSIDER (NOT HAND IN)

1. If the two different algorithms have greatly different efficiencies, why have we even bother to learn about the less efficient one?

2. Does the algorithm that runs more quickly with a fixed size of data have a better Big-O efficiency? Do you think this will always be the case? Why?