Lab 10 Problems

Problem 1: Vocabulary. In computer science, in addition to developing programming and problem solving skills, it's also necessary to learn the vocabulary of common terms used in the field. For this problem, write out the definitions of the following terms. You can refer to the slides for help.

special method

abstract data type (also name at least two examples)

a recursive function

binary search tree

black box testing

white box testing

edge case

In the Python method `astr.split()`, if no argument is given, the method splits the string `astr` on whitespace. Define whitespace.

Problem 2 Complexity.

Consider the Python built-in method `alist.index(elem)` that returns the index of the first occurrence of `elem` in `alist`. For example, if `alist` is `[2,4,6,8]`, `alist.index(4)` returns `1`.

a) Even though you don't have the code for `index()`, you can reason about its worst-case complexity. Give the worst-case complexity of `index()` expressed in big-O notation and explain your reasoning.

b) Give an example of calling `index()` that demonstrates its worst-case complexity.

Problem 3 Complexity.

In the code below for a) and b), `alist` is a list of integers and has at least one element.

a)

```
min_val = alist[0]
max_val = alist[0]

for elem in alist:
    if elem < min_val:
        min_val = elem
    elif elem > max_val:
        max_val = elem
```

b)

```
min_val = alist[0]
max_val = alist[0]

for elem in alist:
    if elem < min_val:
        min_val = elem
for elem in alist:
    if elem > max_val:
        max_val = elem
```

It seems on the surface that the code in b) does twice the amount of work that a) does because it loops through the list twice.

What is the complexity of the code in a)?

What is the complexity of the code in b)?

What property of big-O complexity do you use to get your result, i.e., is the code in a) and b) sequential or nested?

Problem 4 Linked Lists

Write a method `make_even(self, value)` for the `LinkedList` class that removes the last node of a linked list if it has an odd number of elements. You may access the attributes directly or use the usual getter and setter methods. (This was on the exam!)

Notes:

- If the list has only one element, it is removed and the list becomes empty.
- The method modifies the linked list and has no return value. (Python returns `None` if there is no return statement.).

Step 1. Before writing your code, answer these questions:

a) What are the three cases that must always be considered when writing code to modify a linked list.

b) Will you need a loop to solve this problem? If so, why?

Step 2. Download the starter code `lab10_starter.py` from the class website.

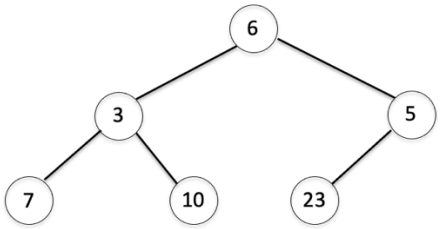
Step 3. Write your code below or in your IDE.

Step 4: What is the complexity of your solution to `make_even()`?

Problem 5 Trees.

Write a recursive function `post_order(bt)` that takes a binary tree `bt` and prints the post-order traversal of the tree node values, with each value printed on a separate line.

Step 1: Before you write the function, answer these questions:



a) What is the post-order traversal of the tree above?

b) Does your function have to return a value or not?

Step 2: Write your code below or in your IDE.

Problem 6 Trees.

Write a recursive function `post_order_v2(bt)` that takes a binary tree `bt` and returns a Python *list* of the node values in the order of the post-order traversal.

Step 1: Before you write the function, answer this question: does your function have to return a value or not?

Step 2: Write your code below or in your IDE.