Beihang University Beijing, Nov. 14th, 2018

Computer Science and Programming
Lab Class 6

Task 0. ***Get execution time*** *(15 minutes)*

First of all, we will introduce a new python package **time**. This module provides various time-related functions.

time.**time**() returns the time in seconds since the epoch as a floating point number. On Windows and most Unix systems, the epoch is January 1, 1970, 00:00:00 (UTC). To find out what the epoch is on a given platform, look at time.**gmtime**(0).

```
>>> import time
>>> print(time.time())
1510226673.87
```

Assmue that you want to get the execution time of the your program that calls the function $max\_of\_3()$, you can store the start time and end time of your program in variable **start_time**, **end_time** respectively and print the difference between them. An example code is shown below.

```
import time

def max_of_3(a,b,c):
    maxi = a
    if b > maxi:
        maxi = b
    if c > maxi:
        maxi = c
    return maxi

start_time = time.time()    #time starts a function call
max_of_3(4,5,6)
end_time = time.time()    #time ends a function call
print("Execution time is %es" %(end_time - start_time))

Execution time is 1.192093e-06s
```

Task 1. ***Sort algorithm – Insertion sort*** *(25 minutes)*

Write a function *insert_sort*() to sort an integer list in ascending order with insertion sort algorithm. Test your function with the several test cases and please compare the execution time with an increasing length of input lists: 100, 1000, 10000 elements.

Task 2. ***Recursion – Dichotomy*** *(20 minutes)*

(1) Implement a function $f(x) = 2x^2 + x - 2$ with python. For an input x, *f(x)* should be output.

(2) Please use dichotomy and recursion to obtain the approximate root with the accuracy 0.001 in the range *(0,1)*.

Hint: In dichotomy, if $f(x_1) * f(x_2) < 0$ for different x_1, x_2, there must be at least one root between x_1 and x_2 for the equation $f(x) = 0$. Therefore, we can explore $\frac{x_1+x_2}{2}$. If $f(x_1) * f(\frac{x_1+x_2}{2}) < 0$, the root is between x_1 and $\frac{x_1+x_2}{2}$; Else if $f(\frac{x_1+x_2}{2}) = 0$, the root is just $\frac{x_1+x_2}{2}$; Else, the root is between $\frac{x_1+x_2}{2}$ and x_2.

Task 3. ***Recursion – Pascal's triangle*** *(15 minutes)*

For a given number n, please implement a function with Python that prints an n-index Pascal's triangle recursively. Note that each line of Pascal's triangle is managed as a list. For instance, a Pascal's triangle with n=*8* is:

```
[1]
[1, 1]
[1, 2, 1]
[1, 3, 3, 1]
[1, 4, 6, 4, 1]
[1, 5, 10, 10, 5, 1]
[1, 6, 15, 20, 15, 6, 1]
[1, 7, 21, 35, 35, 21, 7, 1]
```

Hint: Assume that the list of i-th line of Pascal's triangle is tr_i. Its length is equal to i. After printing tr_i, we can compute tr_{i+1} as follows:

$$tr_{i+1}[0] = 1$$
$$tr_{i+1}[j] = tr_i[j-1] + tr_i[j],\ \forall 1 \leq j \leq len(tr_i) - 1$$
$$tr_{i+1}[len(tr_i)] = 1$$

Then, you can compute tr_{i+2} with the same procedure based on tr_{i+1}.

Task 4. ***Classes*** *(40 minutes)*

Implement the Rational Number class from Lecture 5 in Python. You should implement a constructor (which received a nominator and denominator), a print function, and all four standard operators (+,-,*,/). Make sure that your results are always normalized, i.e., $\frac{3}{12}$ should become $\frac{1}{4}$ after *any* constructor call. The member functions for overloading operators are:

- $\_\_add\_\_(self, other)$
- $\_\_sub\_\_(self, other)$
- $\_\_mul\_\_(self, other)$
- $\_\_div\_\_(self, other)$