Beihang University
School of General Engineering

Beijing, October 13th, 2020
Lin Wei and Yida Ding

Computer Science and Programming
Lab Class 3

Task 1 ***Basic loop task*** *(5 minutes)*

Use python to calculate the sum of 1∼100. You should solve this task with 'while' loop and 'for' loop, respectively.

Task 2 ***'For' loop*** *(15 minutes)*

There are 10 students in a class. Their scores are as follows: [78, 83, 95, 92, 61, 50, 83, 79, 71, 99].

(1) Please calculate and print the Grade Point (GP) for each student.

If the grade is below 60, GP = 0;
If the grade is within [60,69], GP = 1;
If the grade is within [70,79], GP = 2;
If the grade is within [80,84], GP = 3;
If the grade is within [85,100], GP = 4.

(2) Please print the number of students whose GP = 3.

Hint: You can reuse the program of evaluating GP that you created in last lab class.

Task 3 ***'For' loop*** *(15 minutes)*

Find all prime numbers in the range [2,1000] and store it in a list, then print the list.

Hint: A prime number (or a prime) is a natural number greater than 1 that has no positive divisors other than 1 and itself. For example, 5 is prime because 1 and 5 are its only positive integer factors, whereas 6 is composite because it has the divisors 2 and 3 in addition to 1 and 6. (From Wikipedia)

Task 4 ***'While' loop*** *(10 minutes)*

Given two integers m and n, their greatest common divisor (gcd) can be calculated with Euclidean algorithm (Figure 1). Please implement the algorithm with Python using while loop.

Hint: In mathematics, the greatest common divisor (gcd) of two or more integers, which are not all zero, is the largest positive integer that divides each of the integers. For example, the gcd of 8 and 12 is 4.

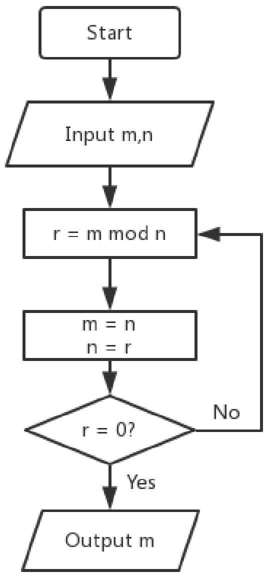


Figure 2: Euclidean algorithm

Task 5 ***Function*** *(10 minutes)*

Given a binary number, write a python function to convert this number to decimal number. (sample input: 10100 Sample output: 20)

Task 6 ***Function*** *(5 minutes)*

(1) Type the following code in python and evaluate the output. Does the add() function work?

```
a = 5
b = 6
result = 0
def add(a,b):
    result = a+b
add(a,b)
print(result)
```

(2) Please modify the code to make it work properly.

Task 7 ***Function*** *(15 minutes)*

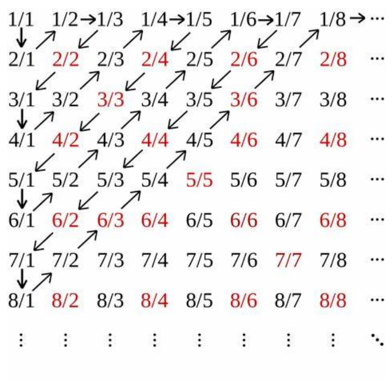
(1) Implement a function $f(x) = 2x^2 + x - 2$ with python. For an input x, *f(x)* should be output.

(2) Please use dichotomy to obtain two approximate roots 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 8 ***Cantor's table*** *(20 minutes)*

One of the famous proofs of modern mathematics was Georg Cantor's proof that rational Numbers were enumerable. He uses the following table to prove this:



1/1	1/2	1/3	1/4	1/5	1/6	1/7	1/8	···
2/1	2/2	2/3	2/4	2/5	2/6	2/7	2/8	···
3/1	3/2	3/3	3/4	3/5	3/6	3/7	3/8	···
4/1	4/2	4/3	4/4	4/5	4/6	4/7	4/8	···
5/1	5/2	5/3	5/4	5/5	5/6	5/7	5/8	···
6/1	6/2	6/3	6/4	6/5	6/6	6/7	6/8	···
7/1	7/2	7/3	7/4	7/5	7/6	7/7	7/8	···
8/1	8/2	8/3	8/4	8/5	8/6	8/7	8/8	···
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋱

Figure 3: Cantor's table

We number each item in the above table with zigzag order. The first item is 1/1, and the following are 2/1, 1/2, 1/2, 2/2, 3/1, 4/1, . . .

Write a python program which receives a integer N and print the Nth item in the table. (Sample input: 7 Sample ouput: 4/1)

Only once you are finished with all tasks above:

Task 9 ***Extra*** *(30 minutes)*

Implement a Python program which searches the largest prime number and prints the result on the screen.