1. Write a Python program to count the total number of alphabets, digits, and special characters in a string. Here, space is considered as a special character.
   ==========
   Sample Input:
   `This is CSE110 Course.`
   Sample Output:
   The number of Alphabets in the string is: 15
   The number of Digits in the string is : 3
   The number of Special characters in the string is: 4

2. Write a Python program to find the largest and smallest word in a string.
   [you are not allowed to use max() and min()]
   Sample Input :
   It is a string with the smallest and largest word.
   Sample Output :
   The largest word is “smallest” and the smallest word is 'a'.

3. Write a Python program that takes two strings from the user. Then remove characters from the first string which are present in the second string. [you are not allowed to use **replace()**]
   ==========
   Sample Input1:
   India is great
   is
   Sample Output1:
   nda great
   ==========
   Sample Input2:
   this is cSe110 course and we love it
   lo1iva
   Sample Output2:
   ths s cse0 curse nd we e t
   ==========
   Sample Input3:
   next course is cse111
   lo1iva
   Sample Output3:
   ext coure i ce111

4. Write a Python program to Capitalize the first character of each word in a String **[You cannot use the built-in upper() function]**

   Sample Input:
   I love python programming
   Sample Output:
   I Love Python Programming

5. Write a python function that takes a string with multiple numbers separated by commas as input from the user. Extract the numbers from the string and make a list and print it. Multiply the numbers of the list and print the product. **[Hint: You can use split()]**

   ==========

   Sample Input1:

   1,2,3,4,5

   Sample output1:

   ['1', '2', '3', '4', '5']

   Product:120

   ==========

   Sample Input2:

   10,0,20,3,1

   Sample Output2:

   ['10', '0', '20', '3', '1']

   Product:0

6. Write a Python program that reads a number and finds the sum of the series of 1 +11 + 111 + 1111 + ….+N terms.

   ==========

   Sample Input1:

   5

   Sample Output1:

   1 + 11 + 111 + 1111 + 11111

   The Sum is: 1234

   ==========

   Sample Input2:

   8

   Sample Output2:

   1 + 11 + 111 + 1111 + 11111 + 111111 + 1111111 + 11111111

   The Sum is: 12345678

   ==========

7. Write a Python program that reads a number and displays the multiplication table of the given integer.

   Sample Input: 15

   Sample Output :

   15X1=15

   15X2=30

   ...

   ...

   15X10=150

LOOP RELATED PROBLEMS

Difficulty-Easy

1. Read a number (1-9) from the user and print the sequence given in the sample outputs.
You have to identify the pattern using the sample Inputs and outputs. Assume that the user will not Input any invalid number.

Sample Input 1: 4 **Output:** 1234321	**Sample Input 2:** 8 **Output:** 123456787654321
Sample Input 3: 1 **Output:** 1	**Sample Input 2:** 6 **Output:** 12345654321

2. Read the value N from the user, and print the first Nth Fibonacci numbers.

Sample Input: 9 **Output**: 0 1 1 2 3 5 8 13 21	**Sample Input:** 5 **Output**: 0 1 1 2 3

3. Read the value of n from the user and compute the value of the following series:

result = (1**1)/1 + (2**2)/2 + (3**3)/3 + (4**4)/4 + ... +(n**n)/n

(Here, ** indicates power)

==========

Sample Input: 3
Output: 12

==========

Sample Input: 5

Output: 701

4. Take two-year Inputs from the user (Lower bound and upper bound).
Print all the years that are leap years within that range (inclusive).

Three conditions are used to identify leap years:

- The year can be evenly divided by 4, is a leap year, unless
- The year can be evenly divided by 100, it is NOT a leap year, unless:
- The year is also evenly divisible by 400. Then it is a leap year.

[which implies -
if the year can be divided by 4 and NOT divided by 100, then it's a leap year.
If the year can be divided by 4, divided by 100, and divided by 400, then it's a leap year.]

==========

Sample Input:
lower_bound= 1980
upper_bound= 2020

Sample Output:
1980 1984 1988 1992 1996 2000 2004 2008 2012 2016 2020

5. Write a python program that takes integer Inputs from the user until the user gives "STOP". Print the product of all the numbers.

Sample Input1: 5 6 -02 7 2 STOP **Sample Output1:** -840	**Sample Input2:** 5 0 2 STOP **Sample Output2:** 0
Sample Input3: STOP **Sample Output3:** 0	

6. Write a Python program that reads a number and displays the multiplication table of the given integer.

Sample Input: 15 **Sample Output**: 15X1=15 15X2=30 ... ... 15X10=150	**Sample Input**: 2 **Sample Output**: 2X1=2 2X2=4 ... ... 2X10=20

Difficulty-Easy (Lengthy tasks)

7. Imagine you are trying to draw a shape. So, you are taking the lengths of each of the sides of the shape. Now, what shape you need to draw depends on the first Input (1-4).

a. If no of sides: 1 or 2, a line needs to be drawn
b. If no of sides: 3, a triangle needs to be drawn
c. If no of sides: 4, a rectangle needs to be drawn
d. For any other number of sides, print "Invalid Input"

After deciding what your shape will be, your job is to verify whether the shape can be drawn using the lengths given by the user.

- To draw a triangle, the sum of the lengths of any two sides must be greater than the third.
- To draw a rectangle, the sum of the lengths of any three sides must be greater than the fourth.
- To draw a line, no verification is required.

Sample Input 1: No of sides: 1 **Sample Output:** A line can be drawn	**Sample Input 2:** No of sides: 2 **Sample Output:** A line can be drawn
Sample Input 3: No of sides: 3 Length of sides: 2 3 1 **Sample Output:** A triangle cannot be drawn	**Sample Input 4:** No of sides: 3 Length of sides: 2 3 2 **Sample Output:** A triangle can be drawn
Sample Input 5: No of sides: 4 Length of sides: 2 3 1 4 **Sample Output:** A rectangle can be drawn	**Sample Input 5:** No of sides: 4 Length of sides: 2 3 1 7 **Sample Output:** A rectangle cannot be drawn

8. Imagine you are trying to draw a shape. So, you are taking the angles of the shape. Now, what shape you need to draw depends on the first Input (3 or 4).

a. If no of angles is 3, a triangle needs to be drawn
b. If no of angles is 4, a rectangle needs to be drawn
c. For any other Inputs, print “Invalid Input”

After deciding what your shape will be, your job is to verify whether the shape can be drawn using the angles given by the user.

- To draw a triangle, the sum of the angles must be equal to 180 degrees.
- To draw a rectangle, the sum of the angles must be equal to 360 degrees.

Sample Input 1: No of angles: 1 **Sample Output:** Invalid number of angles	**Sample Input 2:** No of angles: 2 **Sample Output:** Invalid number of angles
Sample Input 3: No of angles: 3 Angles: 60 90 50 **Sample Output:** Triangle cannot be drawn	**Sample Input 4:** No of sides: 3 Angles: 60 90 30 **Sample Output:** A triangle can be drawn
Sample Input 5: No of angles: 4 Angles: 60 80 70 150 **Sample Output:** A rectangle can be drawn	**Sample Input 6:** No of angles: 4 Angles: 60 80 70 130 **Sample Output:** A rectangle cannot be drawn

Difficulty-Medium

9. Imagine you are playing a game with your friend which is quite similar to ludo. The player who crosses throughout the board of exactly 25 boxes wins. Each of you gets to roll the dice and can get from 1-6.

If a player crosses 24 boxes, he/she has to get exactly 1 after rolling the dice in order to win. Otherwise, he/she will stay in the same place. The one who gets to the 25th box **first** will win.

Until a winner is decided, the dice will be rolled between the players and the game will continue.

Sample Input 1: Player 1: 6 Player 2: 4 Player 1: 5 Player 2: 5 Player 1: 6 Player 2: 4 Player 1: 4 Player 2: 6 Player 1: 3 Player 2: 6 **Sample Output:** Player 2 wins	**Sample Input 2:** Player 1: 6 Player 2: 3 Player 1: 6 Player 2: 2 Player 1: 6 Player 2: 6 Player 1: 5 Player 2: 6 Player 1: 3 Player 2: 4 Player 1: 5 Player 2: 5 Player 1: 1 Player 2: 6 Player 1: 3 Player 2: 2 Player 1: 1 **Sample Output:** Player 1 wins

Difficulty-Hard

10. Imagine you are playing a game with your friend which is quite similar to ludo. The player who crosses throughout the board of 25 boxes wins. Each of you gets to roll the dice and can get from 1-6.

If both players converge on the **same box,** the player's avatar coming to this box previously will get eaten and the player will lose.

There are certain checkboxes in the board where the avatars cannot be eaten. The checkpoints are 10th and 20th.

Until a player's avatar gets eaten or he/she crosses the 25th box, the dice will be rolled between the players and the game will continue.

Sample Input 1: Player 1: 6 Player 2: 4 Player 1: 3 Player 2: 5 **Sample Output:** Player 1 died Player 2 wins	**Sample Input 2:** Player 1: 2 Player 2: 3 Player 1: 1 **Sample Output:** Player 2 died Player 1 wins
Sample Input 3: Player 1: 5 Player 2: 4 Player 1: 5 Player 2: 6 Player 1: 6 Player 2: 3 Player 1: 4 Player 2: 3 Player 1: 5 **Sample Output:** Player 1 wins	

11. You will be given three Inputs **M, N,** and **P. M** indicates the number of chocolates you have. And there is a shop where you can get **P** new chocolates if you return **N** empty chocolate packets. You need to find out the total number of chocolates that you can have and print it.

==========

Sample Input 1:
100
4
1

Sample Output 1:
133

Explanation 1:
In this Input, the shop will return 1 new chocolate for every 4 empty packets. So, if you have 100 chocolates in the beginning, you can have more chocolates through the following steps.

1. **Eat 100 chocolates** -> 100 empty packs.
2. 100 empty packs -> 25 new chocolates. 125
3. **Eat 25 chocolates** -> 25 empty packets.
4. 25 empty packets -> 6 new chocolates and 1 empty packet.
5. **Eat 6 chocolates** -> (6 + 1 remaining) = 7 empty packs.
6. 7 empty packets -> 1 new chocolate and 3 empty packets.
7. **Eat 1 chocolate** -> (1 + 3 remaining) = 4 empty packs.
8. 4 empty packs -> 1 new chocolate.
9. **Eat 1 chocolate** -> 1 empty packet.

No new chocolate can be obtained using only 1 packet.
Now counting all the chocolates, you have eaten you'll get (100 + 25 + 6 + 1 + 1) = 133 chocolates in total.
So, the output is 133.

==========

Sample Input 2:
100
10
3

Sample Output 2:
139

Explanation 2:
In this Input, the shop will return 3 new chocolates for every 10 empty packets. So, if you have 100 chocolates in the beginning, you can have more chocolates through the following steps.

1. **Eat 100 chocolates** -> 100 empty packs.
2. 100 empty packets -> 30 new chocolates. (3 for each 10)
3. **Eat 30 chocolates** -> 30 empty packs.
4. 30 empty packs -> 9 new chocolates. (3 for each 10)

5. **Eat 9 chocolates** -> 9 empty packs.

You can’t get any more new chocolates with 9 empty packs since 9 is less than 10.

Now counting all the chocolates, you have eaten. You’ll get (100 + 30 + 9) = 139 chocolates in total.

So, the output is 139.

Task 1

Write a Python function that takes the "amount" of money as user input. Then splits that money into 500, 100, 50, 20, 10, 5, 2, and 1 taka notes and print the final result.

==========

Hints:

This task's calculation is similar to Assignment-1's seconds to hours, minutes conversion.

==========

Example1:

Sample Input: 1234
Sample Output:
500 Taka: 2 note(s)
100 Taka: 2 note(s)
20 Taka: 1 note(s)
10 Taka: 1 note(s)
2 Taka: 2 note(s)

==========

Example2:

Sample Input: 151
Sample Output:
100 Taka: 1 note(s)
50 Taka: 1 note(s)
1 Taka: 1 note(s)

==========

Task 2

Write Python code of a program that reads three sides of a triangle and check whether the triangle is valid or not.

==========

Sample Input1:

7, 10, 5

Sample Output1:

Valid triangle

==========

Sample Input2:

2, 2, 4

Sample Output2:

Not a valid triangle

==========

Sample Input3:

7, 9, 12

Sample Output3:

Valid triangle

Task 3

Write a python code of a program that reads the values for the three sides x, y, and z of a triangle, and then calculates its area. The area is calculated as follows:

$$\textbf{Area} = s\sqrt{(s-x)(s-y)(s-z)} \qquad \textbf{where } s = \frac{x+y+z}{2}$$

==========

Sample Input1:

7, 10, 5

Sample Output1:

53.88877434122991

==========

Sample Input2:

7, 9, 12

Sample Output2:

117.13240371477058

==========

Task 4

Write Python code of a program that reads a number as a year and determines whether it is a leap year or not.

Three conditions are used to identify leap years:

- The year can be evenly divided by 4, is a leap year, unless
- The year can be evenly divided by 100, it is NOT a leap year, unless:
- The year is also evenly divisible by 400. Then it is a leap year.
  [which implies -
  if the year can be divided by 4 and NOT divided by 100, then it's a leap year.
  If the year can be divided by 4, divided by 100, and divided by 400, then it's a leap year.]

For example, the years 2000, 2024, 2028, and 2400 are leap years, while 1800, 1900, 2100, 2200, 2300, and 2500 are NOT leap years.

Here, 2000, 2400 are divided by 4, 100, and 400. Thus, leap year.

Here, 2024, 2028 are divided by 4 but not divided by 100. Thus, leap year.

==========

Sample Input1:

2500

Sample Output1:

Not a leap year

Explanation:

Here, 2500 is evenly divisible by 4 and 100. But it is not divisible by 400. So, 2500 is not a leap year.

==========

Sample Input2:
2024
Sample Output2:
A leap year
Explanation:
Here, 2024 is evenly divisible by 4 but not divisible by 100. So, 2024 is a leap year.
==========
Sample Input3:
2400
Sample Output3:
A leap year
Explanation:
Here, 2400 is evenly divisible by 4, 100, and 400. So, 2400 is a leap year.