1. Create a function to perform basic arithmetic operations that includes addition, subtraction, multiplication and division on a string number (e.g. "12 + 24" or "23 - 21" or "12 // 12" or "12 \* 21").

Here, we have 1 followed by a space, operator followed by another space and 2. For the challenge, we are going to have only two numbers between 1 valid operator. The return value should be a number.

eval() is not allowed. In case of division, whenever the second number equals "0" return -1.

For example:

"15 // 0" ➞ -1

**Examples**

arithmetic\_operation("12 + 12") ➞ 24 // 12 + 12 = 24

arithmetic\_operation("12 - 12") ➞ 24 // 12 - 12 = 0

arithmetic\_operation("12 \* 12") ➞ 144 // 12 \* 12 = 144

arithmetic\_operation("12 // 0") ➞ -1 // 12 / 0 = -1

**Ans:**

def arithmetic\_operation(string):

sp=string.split(" ")

if(sp[1]=="+"):

print(int(sp[0])+int(sp[2]))

elif(sp[1]=="-"):

print(int(sp[0])-int(sp[2]))

elif(sp[1]=="\*"):

print(int(sp[0])\*int(sp[2]))

elif(sp[1]=="//"):

if(sp[2]=="0"):

print("-1")

else:

print(int(sp[0])//int(sp[2]))

2. Write a function that takes the coordinates of three points in the form of a 2d array and returns the perimeter of the triangle. The given points are the vertices of a triangle on a two-dimensional plane.

**Examples**

perimeter( [ [15, 7], [5, 22], [11, 1] ] ) ➞ 47.08

perimeter( [ [0, 0], [0, 1], [1, 0] ] ) ➞ 3.42

perimeter( [ [-10, -10], [10, 10 ], [-10, 10] ] ) ➞ 68.28

**Ans:**

def perimeter(l):

print(round(((l[0][0]-l[1][0])\*\*2+(l[0][1]-l[1][1])\*\*2)\*\*0.5+((l[1][0]-l[2][0])\*\*2+(l[1][1]-l[2][1])\*\*2)\*\*0.5+

((l[2][0]-l[0][0])\*\*2+(l[2][1]-l[0][1])\*\*2)\*\*0.5, 2))

3. A city skyline can be represented as a 2-D list with 1s representing buildings. In the example below, the height of the tallest building is 4 (second-most right column).

[[0, 0, 0, 0, 0, 0],

[0, 0, 0, 0, 1, 0],

[0, 0, 1, 0, 1, 0],

[0, 1, 1, 1, 1, 0],

[1, 1, 1, 1, 1, 1]]

Create a function that takes a skyline (2-D list of 0's and 1's) and returns the height of the tallest skyscraper.

**Examples**

tallest\_skyscraper([

[0, 0, 0, 0],

[0, 1, 0, 0],

[0, 1, 1, 0],

[1, 1, 1, 1]

]) ➞ 3

tallest\_skyscraper([

[0, 1, 0, 0],

[0, 1, 0, 0],

[0, 1, 1, 0],

[1, 1, 1, 1]

]) ➞ 4

tallest\_skyscraper([

[0, 0, 0, 0],

[0, 0, 0, 0],

[1, 1, 1, 0],

[1, 1, 1, 1]

]) ➞ 2

**Ans:**

def tallest\_skyscraper(l):

max=0

cols=len(l[0])

rows=len(l)

for i in range(0, cols):

sum=0

for j in range(0, rows):

sum+=l[j][i]

if sum>max:

max=sum

print(max)

4. A financial institution provides professional services to banks and claims charges from the customers based on the number of man-days provided. Internally, it has set a scheme to motivate and reward staff to meet and exceed targeted billable utilization and revenues by paying a bonus for each day claimed from customers in excess of a threshold target.

This quarterly scheme is calculated with a threshold target of 32 days per quarter, and the incentive payment for each billable day in excess of such threshold target is shown as follows:

Days Bonus

0 to 32 days Zero

33 to 40 days SGD$325 per billable day

41 to 48 days SGD$550 per billable day

Greater than 48 days SGD$600 per billable day

Please note that incentive payment is calculated progressively. As an example, if an employee reached total billable days of 45 in a quarter, his/her incentive payment is computed as follows:

32\*0 + 8\*325 + 5\*550 = 5350

Write a function to read the billable days of an employee and return the bonus he/she has obtained in that quarter.

**Examples**

bonus(15) ➞ 0

bonus(37) ➞ 1625

bonus(50) ➞ 8200

**Ans:**

def bonus(days):

if days<=32:

sum=0

if days >32 and days<=40:

sum=0\*32+(days-32)\*325

if days >40 and days<=48:

sum=0\*32+8\*325+(days-40)\*550

if days>48:

sum=0\*32+8\*325+8\*550+(days-48)\*600

print(sum)

5. A number is said to be Disarium if the sum of its digits raised to their respective positions is the number itself.

Create a function that determines whether a number is a Disarium or not.

**Examples**

is\_disarium(75) ➞ False

# 7^1 + 5^2 = 7 + 25 = 32

is\_disarium(135) ➞ True

# 1^1 + 3^2 + 5^3 = 1 + 9 + 125 = 135

is\_disarium(544) ➞ False

is\_disarium(518) ➞ True

is\_disarium(466) ➞ False

is\_disarium(8) ➞ True

**Ans:**

def is\_disarium(num):

sum=0

o\_num=num

for i in range(len(str(num)), 0, -1):

d=num%10

sum+=d\*\*i

num=num//10

if sum==o\_num:

print(True)

else:

print(False)