Question 1

What data type is each of the following (evaluate where necessary)? 5 5.0 5 > 1 '5' 5 * 2 '5' * 2 '5' + '2' 5 / 2 5 % 2 {5, 2, 1} 5 == 3 Pi (the number)

```
#Importing library
import math

#Displaying datatype of each question
print(type(5))
print(type(5.0))
print(type(5 > 1))
print(type('5'))
print(type(5 * 2))
print(type('5' * 2))
print(type('5' + '2'))
print(type(5 / 2))
print(type(5 % 2))
print(type({5, 2, 1}))
print(type(5 == 3))
print(type(math.pi))
```

```
    <class 'int'>
    <class 'float'>
    <class 'bool'>
    <class 'str'>
    <class 'int'>
```

Saved successfully!

```
    <class 'int'>
    <class 'set'>
    <class 'bool'>
    <class 'float'>
```

Question 2

Write (and evaluate) python expressions that answer these questions:

a. How many letters are there in 'Supercalifragilisticexpialidocious'?

```
str = "Supercalifragilisticexpialidocious"
print("Total number of letters in string are ", len(str))
```

```
     Total number of letters in string are  34
```

b. Does 'Supercalifragilisticexpialidocious' contain 'ice' as a substring?

```
str = "Supercalifragilisticexpialidocious"

#Checking for substring available in string
if "ice" in str:
  print("Yes, it contains.")
else:
  print("No, it doesn't contain")
```

```
     Yes, it contains.
```

c. Which of the following words is the longest: Supercalifragilisticexpialidocious, Honorificabilitudinitatibus, or Bababadalgharaghtakamminarronnkonn?

```
strs = ["Supercalifragilisticexpialidocious", "Honorificabilitudinitatibus", "Bababadalgharag

#number of elements in lists
len_strs = len(strs)

#List of index having maximum length of string
max_index = []

#Storing maximum length of string
max_len = len(strs[0])
```

Saved successfully!

```
                                             string
length = 0

#Finding index of string having maximum length
for i in range(len_strs):
  length = len(strs[i])
  if(length >= max_len):
    max_len = length
    max_index.append(i)

#Displaying each string having maximum length
for i in range(len(max_index)):
  print(strs[max_index[i]])
```

```
     Supercalifragilisticexpialidocious
     Bababadalgharaghtakamminarronnkonn
```

d. Which composer comes first in the dictionary: 'Berlioz', 'Borodin', 'Brian', 'Bartok', 'Bellini',

```
a = [ "Berlioz", "Borodin", "Brian", "Bartok", "Bellini", "Buxtehude", "Bernstein"]

#Initializing first and last element of dictionary
first = a[0]
last = a[0]

#Finding first and last elements from given list 'a'
for i in range(len(a)):
  if(a[i] < first):
    first = a[i]
  if(a[i] > last):
    last = a[i]

#Displaying first and last elements
print("First is", first)
print("Last is", last)
```

```
     First is Bartok
     Last is Buxtehude
```

Question 3

Implement function triangleArea(a,b,c) that takes as input the lengths of the 3 sides of a triangle and returns the area of the triangle. By Heron's formula, the area of a triangle with side lengths a, b, and c is s(s - a)(s -b)(s -c) , where s = (a+b+c)/2.

Saved successfully! ×

20508075688772

```
#Defining function trianglearea to find area of triangle where a, b and c are length three si
def trianglearea(a, b, c):
  s = (a + b + c) / 2
  square = s * (s - a) * (s - b) * (s - c)
  area = math.sqrt(square)
  return area


#Calling function trianglearea
trianglearea(2, 2, 2)
```

```
     1.7320508075688772
```

Question 4

Write a program in python to separate odd and even integers in separate arrays. Go to the editor Test Data : Input the number of elements to be stored in the array :5 Input 5 elements in the array : element - 0 : 25 element - 1 : 47 element - 2 : 42 element - 3 : 56 element - 4 : 32 Expected Output: The Even elements are: 42 56 32 The Odd elements are : 25 47

```
#Defining function to separate odd nd even numbers in different lists
def Sep_odd_even(data):

  #Initializing odd and even lists
  odd = []
  even = []

  #Separating each element
  for i in range(len(data)):
    if(data[i]%2==1):
      odd.append(data[i])
    else:
      even.append(data[i])

  return odd, even

#Initializing list of conbined numbers
nums = [25, 47, 42, 56, 32]

#Calling function to separate
odd_nums, even_nums = Sep_odd_even(nums)

#Displaying odd and even numbers
```

Saved successfully!

```
     Even numbers : [42, 56, 32]
     Odd numbers : [25, 47]
```

Question 5

a. Write a function inside(x,y,x1,y1,x2,y2) that returns True or False depending on whether the point (x,y) lies in the rectangle with lower left corner (x1,y1) and upper right corner (x2,y2).

```
#Defining function inside
def inside(x, y, x1, y1, x2, y2):

  #Taking exact points of lower left and upper right corners from users
  while(1):
    if(x1>x2 or y1>y2):

      print("Enter lower left anf upper right coordinators correctly...!")

      print("Enter coordinates of lower left corner of rectangle :")
      x1 = input(print("x coordinate :"))
      y1 = input(print("y coordinate :"))
      print()

      print("Enter coordinates of upper right corner of rectangle :")
      x2 = input(print("x coordinate :"))
      y2 = input(print("y coordinate :"))
      print()

    else:
      break;

  #Checking wheather point(x,y) lies in the triangle
  if(x>=x1 and x<=x2 and y>=y1 and y<=y2):
    return True
  else:
    return False

#Getting three required points from users
print("Enter coordinates of the point which you want to track:")
x = input(print("x coordinate :"))
y = input(print("y coordinate :"))
print()

print("Enter coordinates of lower left corner of rectangle :")
```

Saved successfully!

```
print("Enter coordinates of upper right corner of rectangle :")
x2 = input(print("x coordinate :"))
y2 = input(print("y coordinate :"))
print()

#Initializing result variable which stores return from the function
result = False

#Calling function inside
result = inside(x, y, x1, y1, x2, y2)

#Displaying result
print(result)
```

```
    Enter coordinates of the point which you want to track:
    x coordinate :
```

```
    None1
    y coordinate :
    None1

    Enter coordinates of lower left corner of rectangle :
    x coordinate :
    None0
    y coordinate :
    None0

    Enter coordinates of upper right corner of rectangle :
    x coordinate :
    None2
    y coordinate :
    None3

    True
```

1b. Use function inside() from part a. to write an expression that tests whether the point (1,1) lies in both of the following rectangles: one with lower left corner (0.3, 0.5) and upper right corner (1.1, 0.7) and the other with lower left corner (0.5, 0.2) and upper right corner (1.1, 2).

```
#Checking points which are given as above
result = inside(1, 1, 0.3, 0.5, 1.1, 0.7)
result
```

```
    False
```

```
result = inside(1, 1, 0.5, 0.2, 1.1, 2)
result
```

Saved successfully!

Question 6

You can turn a word into pig-Latin using the following two rules (simplified):

• If the word starts with a consonant, move that letter to the end and append 'ay'. For example, 'happy' becomes 'appyhay' and 'pencil' becomes 'encilpay'.

• If the word starts with a vowel, simply append 'way' to the end of the word. For example, 'enter' becomes 'enterway' and 'other' becomes 'otherway'.

For our purposes, there are 5 vowels: a, e, i, o, u (so we count y as a consonant).

Write a function pig() that takes a word (i.e., a string) as input and returns its pigLatin form. Your function should still work if the input word contains upper case characters. Your output should always be lower case however.

```
#Defining function pig which transform any word into pig-Latin
def pig(word):

  #Converting each charachter of word into lower case
  word = word.lower()

  char = ''
  new_word = ""

  #Checking for word starting with consonant
  if(word[0]!='a' and word[0]!='e' and word[0]!='i' and word[0]!='o' and word[0]!='u'):
    new_word = word[1:] + word[0] + "ay"

  else:
    new_word = word + "way"
  return new_word

#Getting value of word from user
str = input(print("Enter any word to convert it into pig-Latin :"))

#Calling function pig
new_str = pig(str)

#Displaying output string
new_str
```

```
    Enter any word to convert it into pig-Latin :
    NoneHappy
```

Saved successfully!

File bloodtype1.txt records blood-types of patients (A, B, AB, O or OO) at a clinic.

Write a function bldcount() that reads the file with name name and reports (i.e.,prints) how many patients there are in each bloodtype.

> > > bldcount('bloodtype.txt')

There are 10 patients of blood type A.

There is one patient of blood type B.

There are 10 patients of blood type AB.

There are 12 patients of blood type O.

There are no patients of blood type OO.

```
#Defining function bldcount to count same element of different blood types
def bldcount(data):

  #Splitting data into words
  words = data.split(" ")

  #Initializing each count of blood type by 0
  A = 0; B = 0; AB = 0; O = 0; OO = 0

  #Counting same element of different blood types
  for i in range(len(words)):
    if(words[i] == "A"):
      A += 1
    elif(words[i] == "B"):
      B += 1
    elif(words[i] == "AB"):
      AB += 1
    elif(words[i] == "O"):
      O += 1
    elif(words[i] == "OO"):
      OO += 1
    else:
      print("Something went wrong....!")
      break;

  #Defining function print_output to display output string
  def print_output(n, s):
    if(n == 1):
      print("There is one patient of blood type", s)
    elif(n == 0):
      print("There is no patient of blood type", s)

 Saved successfully!                        ts of blood type", s)

  #Calling print_output function to display output
  print_output(A, "A")
  print_output(B, "B")
  print_output(AB, "AB")
  print_output(O, "O")
  print_output(OO, "OO")

#Opening and reading text file
open_file = open("/content/sample_data/bloodType1.txt")
content = open_file.read()

#Calling bldcount function
bldcount(content)
```

```
    There are 15 patients of blood type A
    There is one patient of blood type B
    There are 13 patients of blood type AB
```

```
There are 15 patients of blood type O
There is no patient of blood type OO
```

Question 8

Write a function curconv() that takes as input:

1. a currency represented using a string (e.g., 'JPY' for the Japanese Yen or 'EUR' for the Euro)
2. an amount and then converts and returns the amount in US dollars.

   curconv('EUR', 100)

3. 96544

   curconv('JPY', 100)

4. 241401

The currency rates you will need are stored in file currencies.txt:

AUD 1.0345157 Australian Dollar

CHF 1.0237414 Swiss Franc

CNY 0.1550176 Chinese Yuan

DKK 0.1651442 Danish Krone

Saved successfully!

[illegible]

HKD 0.1270207 Hong Kong Dollar

INR 0.0177643 Indian Rupee

JPY 0.01241401 Japanese Yen

MXN 0.0751848 Mexican Peso

MYR 0.3145411 Malaysian Ringgit

NOK 0.1677063 Norwegian Krone

NZD 0.8003591 New Zealand Dollar

PHP 0.0233234 Philippine Peso

SEK 0.148269 Swedish Krona

SGD 0.788871 Singapore Dollar

THB 0.0313789 Thai Baht

```
#Defining curconv function to convert any particular currency to US Dollars
def curconv(str, amt):

  #Opening and reading file
  open_file = open("/content/sample_data/currencies.txt")
  content = open_file.read()

  #Spliting data into sentences
  sentences = content.split("\n")

  #Spliting sentences into words
  short_forms = []; values = []
  for i in range(len(sentences)):
    words = sentences[i].split("\t")
    short_forms.append(words[0])
    values.append(float(words[1]))

  #Converting amount ot US Dollars
  i = 0
  while(str != short_forms[i]):
    i += 1

  conv_curr = amt*values[i]
  return conv_curr

#Calling curconv and displaying US Dollars according to amount and amount type
usd = curconv("JPY", 100)
```

Saved successfully! ×

Question 9

Each of the following will cause an exception (an error). Identify what type of exception each will cause.

Trying to add incompatible variables, as in adding 6 + ‘a’ Referring to the 12th item of a list that has only 10 items Using a value that is out of range for a function’s input, such as calling math.sqrt(-1.0) Using an undeclared variable, such as print(x) when x has not been defined Trying to open a file that does not exist, such as mistyping the file name or looking in the wrong directory.

```
6 + 'a'
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-51-63a7ed804344> in <module>
----> 1 6 + 'a'

TypeError: unsupported operand type(s) for +: 'int' and 'str'
```

SEARCH STACK OVERFLOW

```
a = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
a[11]
```

```
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
<ipython-input-52-aa18925e3293> in <module>
      1 a = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
----> 2 a[11]

IndexError: list index out of range
```

SEARCH STACK OVERFLOW

```
import math
math.sqrt(-2)
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-53-da135427c2fa> in <module>
      1 import math
----> 2 math.sqrt(-2)

[illegible]
```

Saved successfully! ×

SEARCH STACK OVERFLOW

```
print(X)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-54-fa0134f942c4> in <module>
----> 1 print(X)

NameError: name 'X' is not defined
```

SEARCH STACK OVERFLOW

```
open_file = open("bloodtype.txt")
```

```
---------------------------------------------------------------------------
FileNotFoundError                         Traceback (most recent call last)
<ipython-input-55-2c3106a55277> in <module>
----> 1 open_file = open("bloodtype.txt")

FileNotFoundError: [Errno 2] No such file or directory: 'bloodtype.txt'
```

Question 10

Encryption is the process of hiding the meaning of a text by substituting letters in the message with other letters, according to some system. If the process is successful, no one but the intended recipient can understand the encrypted message. Cryptanalysis refers to attempts to undo the encryption, even if some details of the encryption are unknown (for example, if an encrypted message has been intercepted). The first step of cryptanalysis is often to build up a table of letter frequencies in the encrypted text. Assume that the string letters is already defined as 'abcdefghijklmnopqrstuvwxyz'. Write a function called frequencies() that takes a string as its only parameter, and returns a list of integers, showing the number of times each character appears in the text. Your function may ignore any characters that are not in letters.

> > > frequencies('The quick red fox got bored and went home.')

[1, 1, 1, 3, 5, 1, 1, 2, 1, 0, 1, 0, 1, 2, 4, 0, 1, 2, 0, 2, 1, 0, 1, 1, 0, 0]

> > > frequencies('apple')

Saved successfully!

```
ount frequency of each character of alphabet available in
def frequencies(str):
  str = str.upper()

  #Initializing count list
  count = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
  index = 0

  #Counting frequency of each alphabet
  for i in range(len(str)):
    if(str[i]>='A' and str[i]<='Z'):
      index = ord(str[i]) - 65
      count[index] += 1

  #Returning count list
  return count

#Initializing alphabet count list
alpha_count = []
```

```
    #Calling frequencies function and displaying output
    alpha_count = frequencies("The quick red fox got bored and went home.")
    alpha_count

         [1, 1, 1, 3, 5, 1, 1, 2, 1, 0, 1, 0, 1, 2, 4, 0, 1, 2, 0, 3, 1, 0, 1, 1, 0, 0]


    alpha_count = []
    alpha_count = frequencies("apple")
    alpha_count

         [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
```

Saved successfully! ×

Colab paid products - Cancel contracts here

✓ 0s completed at 15:44