Assignment 3

Q.no -1. A list behaves like a container, and it is able to contain more than one value. Create a list with the values as shown in the example below.

Examples

```
>>> aList[0]
'Hello'
>>> aList[1:3]
[0, 20.0]
>>> aList[3]
'World'
```

Ans - aList = ['Hello',0,20.0,'World']

Q.no -2. A list can be modified, and more elements can be added to an existing list. Use the append(x) function to add some more items to the list to produce the same content shown in the sample below.

Examples

```
>>> aList[0]
'Hello'
>>> aList[1:3]
[0, 20.0]
>>> aList[3]
'World'
```

Ans - aList = ['Hello', 0]
aList.append(20.0)
aList.append('World')

Q.no -3.A list can be modified, and elements can be removed from an existing list. Use the remove(x) function to remove some items from the list shown in the sample given below so that the list is left with the following content: ['hello', 'python','programming'].

Examples

```
>>> aList
['hello', 'i', 'love', 'python', 'programming']
```

Ans - aList = ['hello', 'i', 'love', 'python', 'programming']
aList.remove('i')
aList.remove('love')

Q.no -4.Write a function addNumbersInList(numbers) to add all the numbers in a list. To access each element in a list, you can use the statement '**for** num **in** numbers:'.

Examples

```
>>> addNumbersInList([])
0
>>> addNumbersInList([10, 20, 30])
60
>>> addNumbersInList([-10, -20, 30])
0
```

Ans -
```
def addNumbersInList(numbers):
    sum=0
    for num in numbers:
        sum=sum+num
    return sum
```

Q.no -5.Write a function addOddNumbers(numbers) to add all the odd numbers in a list. To access each element in a list, you can use the statement 'for num in numbers:'.

Examples

```
>>> addOddNumbers([1, 4, 8, 9])
10
>>> addOddNumbers(range(1, 20, 3))
40
>>> addOddNumbers([])
0
```

Ans -
```
def addOddNumbers(numbers):
    sum=0
    for num in numbers:
        if num % 2 == 1 :
            sum=sum+num
    return sum
```

Q.no -6.Write a function countOddNumbers(numbers) to count the number of odd numbers in a list.

Examples

```
>>> countOddNumbers([1, 4, 8, 9])
2
>>> countOddNumbers(range(1, 20, 3))
4
>>> countOddNumbers([])
```

```
  0
```

Ans -
```
def countOddNumbers(numbers):
    count=0
    for num in numbers:
        if num%2==1:
            count=count+1
    return count
```

Q.no -7.Write a function getEvenNumbers(numbers) to return all the even numbers in a list.

Examples

```
 >>> getEvenNumbers([1, 4, 8, 9])
  [4, 8]
  >>> getEvenNumbers(range(1, 20, 3))
  [4, 10, 16]
  >>> getEvenNumbers([])
  []
```

Ans -
```
def getEvenNumbers(numbers):
    lst=[]
    for num in numbers:
        if num%2==0:
            lst.append(num)
    return lst
```

Q.no -8.Write a function removeFirstAndLast(list) that takes in a list as an argument and remove the first and last elements from the list. The function will return a list with the remaining items.

Examples

```
 >>> removeFirstAndLast([1, 4])
  []
  >>> removeFirstAndLast(range(1, 20, 3))
  [4, 7, 10, 13, 16]
  >>> removeFirstAndLast([1])
  []
```

Ans -
```
def removeFirstAndLast(numbers):
    return numbers[1:-1]
```

Q.no -9.Write a function getMaxNumber(numbers) that returns the maximum number in a list.

Examples

```
>>> getMaxNumber([1, 4, 10])
10
>>> getMaxNumber(range(1, 20, 3))
19
>>> getMaxNumber([])
'N.A'
```

Ans -
```
def getMaxNumber(numbers):
    if len(numbers)>1:
        return max(numbers)
    else:
        return 'N.A'
```

Q.no -10.Write a function getMinNumber(numbers) that returns the minimum number in a list.

Examples

```
>>> getMinNumber([12, 4, 10])
4
>>> getMinNumber([])
'N.A'
```

Ans -
```
def getMinNumber(numbers):
    if len(numbers)>0:
        return min(numbers)
    else:
        return 'N.A'
```

Q.no -11.Write a function that does matrix multiplication.

The product of a **m**x**n** matrix with a **n**x**p** matrix results in a **m**x**p** matrix.

A mxn matrix, with m rows and n columns, can be represented using nested lists.

$A_{m,n}$ = [[x_{11}, x_{12}, ..., x_{1n}], ..., [x_{m1}, ..., x_{mn}]]

Examples

```
>>> A = [ [1, 3], [-5, 6], [2, 4]]
>>> B = [ [1, 4], [8, 7]]
>>> MatrixProduct(A, B)
[[25, 25], [43, 22], [34, 36]]
```

Ans -
```
def MatrixProduct(a, b):
    C = [[0 for row in range(len(b[0]))] for col in range(len(a))]
    for i in range(len(a)):
        for j in range(len(b[0])):
            for k in range(len(b)):
                C[i][j] += a[i][k]*b[k][j]
    return C
```

Q.no -12.A list can be modified. You can add more item to a list or change its existing content. Choose the appropriate method to modify a list.

Question: [MCQ] Which of the following will NOT produce an error?

1) a = [1,2,3]
   a.append[4]
2) a = "123"
   a[3] = 4
3) a = [1,2,3]
   a.append(4)
4) a=[1,2,3]
   a[3] = 4

None of the above

Ans - 3

Q.no -13. A mxn matrix, m rows and n columns, can be represented using nested lists. Write a function that returns the diminensions of a matrix.

Examples

```
>>> a = [ [1, 3], [-5, 6], [2, 4]]
  >>> matrixDimensions(a)
  'This is a 3x2 matrix.'
  >>> b = [ [1, 3, 2], [-5, 6, 0] ]
  >>> matrixDimensions(b)
  'This is a 2x3 matrix.'
  >>> c = [ [1, 3], [-5, 6, 0] ]
  >>> matrixDimensions(c)
  'This is not a valid matrix.'
```

Ans -
```
def matrixDimensions(m):
    column_list = []
    a = len(m)
    b = len(m[0])
    for i in range(0, a):
```

```
            if(b != len(m[i])):
                return 'This is not a valid matrix.'
    return "This is a %dx%d matrix." % (a, b)
```

Q.no -14.In Python, variables are linked to objects by references.

Examples

```
>>> a = [1, 2, 3]
>>> b = a
>>> id(a)
23355480
>>> id(b)
23355480
```

Question:

[MCQ] What are the final values of 'a' and 'b' in the code below?

```
>>> a = [4, 5, 6 ]
>>> b = a
>>> b[0] = 1
>>> a[2] = 3
```

1)a = [1, 5, 3]
b = [1, 5, 3]

2) a = [4, 5, 6]
b = [1, 5, 6]

3) a = [4, 5, 3]
b = [4, 5, 6]

4) a = [4, 5, 3]
b = [1, 5, 6]

5) a = [4, 5, 6]
b = [4, 5, 6]

Ans - 1

Q.no -15. In Python, variables are linked to objects by references.

Examples

```
>>> a = [1, 2, 3]
>>> b = a[:]
>>> id(a)
34901728
>>> id(b)
23355480
```

Question:
[MCQ] What are the final values of 'a' and 'b' in the code below?

```
>>> a = [4, 5, 6 ]
>>> b = a[:]
>>> b[0] = 1
>>> a[2] = 3
```

1)a = [1, 5, 3]
b = [1, 5, 3]

2) a = [4, 5, 6]
b = [1, 5, 6]

3) a = [4, 5, 3]
b = [4, 5, 6]

4) a = [4, 5, 3]
b = [1, 5, 6]

5) a = [4, 5, 6]
b = [4, 5, 6]

Ans - 4

Q.no -16. Write a function combine(la, lb) that takes in two lists and return a list with the contents of both list sorted in ascending order.

Examples

```
>>> combine(['a', 'p', 'l'], ['g','o','l'])
['a', 'g', 'l', 'l', 'o', 'p']
>>> combine(range(10, 2, -2), range(1, 10, 3))
[1, 4, 4, 6, 7, 8, 10]
>>> combine(['a', 1, 'z'], [2, 4, 'y'])
[1, 2, 4, 'a', 'y', 'z']
```

Ans -

```
def combine(la, lb):
    lst=[]
    lst.extend(la)
    lst.extend(lb)
    lst.sort()
    return lst
```

Q.no -17.The *transpose* of a matrix M, denoted M^T, is formed by interchanging the rows and columns of M. That is, a mxn matrix is transformed into a nxm matrix. $[M^T]_{ij} = [M]_{ji}$. Write a function that returns the transpose of a matrix.

Examples

```
>>> M = [[1,2,3], [4,5,6]]
>>> transpose(M)
[[1, 4], [2, 5], [3, 6]]
>>> transpose([[1, 2]])
[[1], [2]]
>>> transpose([[3]])
[[3]]
```

Ans - def transpose(matrix):
return map(list, zip(*matrix))

Q.no -18 .**Question:**

[MCQ] What will be the value of c?

```
>>> a = [1,1]
>>> b = [2,2]
>>> c = a + b * 3
```

1) [1, 1, 2, 2, 1, 1, 2, 2, 1, 1, 2, 2]
2) [1, 1, [2, 2, 2, 2, 2, 2]]
3) [1, 1, 2, 2, 2, 2, 2, 2]
4) [1, 1, [2, 2], 3]
5) Error

Ans - 3

Q.no -19. Write a function calCumulativeSum(numbers) that takes in a list of numbers as argument and returns the cumulative sum of the list. That is, the new list where the *i* element is the sum of the first *i* + 1 elements from the original list. For example, the cumulative sum of [1, 2, 3] is [1, 3, 6].

Examples

```
>>> calCumulativeSum([1,2,3])
[1, 3, 6]
>>> calCumulativeSum([2,2,2])
[2, 4, 6]
>>> calCumulativeSum([2,4,6])
[2, 6, 12]
```

Ans -
```
def calCumulativeSum(numbers):
    lst=[]
    value=0
    while(value<=len(numbers)-1):
        lst.append((sum(numbers[j] for j in range(0,value+1))))
        value=value+1
    return lst
```

Q.no -20. Write a function combineList(list1, list2) that takes in two lists as arguments and return a list that combines all the elements in the two list.

Examples

```
>>> combineList([1,2], [3, 4])
[1, 2, 3, 4]
>>> combineList(range(5), range(5))
[0, 1, 2, 3, 4, 0, 1, 2, 3, 4]
>>> combineList(range(5), ['a', 'b', 'c'])
[0, 1, 2, 3, 4, 'a', 'b', 'c']
```

Ans -
```
def combineList(list1, list2):
    lst=[]
    lst.extend(list1)
    lst.extend(list2)
    return lst
```

Q.no -21.Write a function (list1, list2) that takes in two lists as arguments and return a list that is the result of removing elements from list1 that can be found in list2.

Examples

```
ubtractList
>>> subtractList(range(5), range(4))
[4]
>>> subtractList([1,2,3,4,5], [2, 4])
[1, 3, 5]
>>> subtractList (['a', 'b', 'c', 'd'], ['x', 'y', 'z'])
['a', 'b', 'c', 'd']
```

Ans -
```
def subtractList(list1, list2):
    lst=[]
    for num in list1:
        if num not in list2:
            lst.append(num)
    return lst
```

Q.no -22. Write a function countLetters(word) that takes in a word as argument and returns a list of tuples that shows the number of times each letter appears. The letters must be sorted in alphabetical order.

Examples

```
>>> countLetters('google')
[('e', 1), ('g', 2), ('l', 1), ('o', 2)]
>>> countLetters('apple')
[('a', 1), ('e', 1), ('l', 1), ('p', 2)]
>>> countLetters('microsoft')
[('c', 1), ('f', 1), ('i', 1), ('m', 1), ('o', 2), ('r', 1), ('s', 1),
('t', 1)]
```

Ans -
```
def countLetters(word):
    chr = []
    count = []
    for c in sorted(word):
        if c not in chr:
            chr.append(c)
            count.append(1)
        else:
            count[-1] += 1
    return zip(chr, count)
```

Q.no -23. Write a function getNumbers(number) that takes in a number as argument and return a list of numbers as shown in the samples given below.

Examples

```
>>> getNumbers(10)
[100, 64, 36, 16, 4, 0, 4, 16, 36, 64, 100]
>>> getNumbers(9)
[81, 49, 25, 9, 1, 1, 9, 25, 49, 81]
>>> getNumbers(8)
[64, 36, 16, 4, 0, 4, 16, 36, 64]
>>> getNumbers(0)
[0]
```

Ans -
```
def getNumbers(num):
    lst = []
    for i in range(-num,num+1,2):
        lst.append(i*i)
    return lst
```

Q.no -24. Write a function getSumOfFirstDigit(numList) that takes in a list of positive numbers and returns the sum of all the first digit in the list.

Examples

```
>>> getSumOfFirstDigit([12, 23, 34, 45, 56])
15
>>> getSumOfFirstDigit([1, 23, 456, 7890])
14
>>> getSumOfFirstDigit([])
0
```

Ans -
```
def getSumOfFirstDigit(num):
    answer=[]
    for number in num:
        number=str(number)
        number=number[0]
        answer.append(int(number))
    return sum(answer)
```

Q.no -25.**Question:**

[MCQ] What are the final values of 'a' and 'b' in the following operations?

```
a = [ ]
b = [0]
a.append([ ])
b.append([0])
a.extend([ ])
b.extend([0])
```

1) a = [[ ], [ ]]
   b = [0, 0, 0]
2) a = [ ]
   b = [0, [0]]
3) a = [[ ]]
   b = [0, [0], 0]
4) a = [[ ], [ ]]
   b = [0, [0], 0]
5) None of the above.

Ans - 3

Q.no -26. List comprehension offers a concise way to derive a new list from an existing list or sequence. Given a list of numbers, write a function that returns the numbers that are greater than the average.

Examples

```
   >>> nums = [ 3, -1, 4, -2, 0 ]  # returns a list of numbers greater
than 0
    >>> [x for x in nums if x>0 ]
    [3, 4]
    >>> word = 'cultivate'
    >>> [ x for x in word if x in 'aeiou' ] # get vowels
    ['u', 'i', 'a', 'e']
    >>> getAboveAverage([1, 2, 3, 4])
    [3, 4]
```

Ans - def getAboveAverage(nums):
    return [x for x in nums if x > (sum(nums)/float(len(nums)))] # complete the list comprehension