Machine Learning Laboratory

(410302)

BE Sem I Honors in AI/ML

Academic Year: 2021-22

Lab Assignment No.1 Part 2

Name: Aboli Marathe

Roll Number: 41301

Branch: Department of Computer Engineering

1. Write a function find_max that accepts three numbers as arguments and returns the largest number among three. Write another function main, in main () function accept three numbers from user and call find_max.

```
def find_max(x, y, z):
    if x > y and x > z:
        return x
    elif y > z:
        return y
    else:
        return z

def main():
    num1 = int(input('Enter the first number:'))
    num2 = int(input('Enter the second number:'))
    num3 = int(input('Enter the third number:'))
    maximum = find_max(num1,num2,num3)
    print('The largest number is', maximum)

main()
```

```
Enter first number 5
Enter second number 3
Enter third number 2
The largest number is 5
```

2. Write a function, is_vowel that returns the value true if a given character is a vowel, and otherwise returns false. Write another function main, in main () function accept a string from user and count number of vowels in that string.

```
def is_vowel(l):
    if l in ['a','e','i','o','u','A','E','I','O','U']:
        return True
    else:
        return False

def main():
    counter = 0
    text = input('Enter a string: ')
    for ch in text:
        if(is_vowel(ch)):
            counter += 1

    print('Number of vowels are', counter)

main()
```

```
 Enter a text: Aboli
 Number of vowels are 3
```

3. Write a function named is_prime, which takes an integer as an argument and returns true if the argument is a prime number, or false otherwise. Also, write the main function that displays prime numbers between 1 to 500.

```
def is_prime(num):
    for i in range(2,num):
        if num%i == 0:
            return False
    return True

def main():
    for val in range(2,501):
        if is_prime(val):
            print(val)

main()
```

```
 157
 163
 167
 173
 179
 181
 191
 193
 197
 199
 211
 223
 227
 229
 233
 239
 241
```

```
251
257
263
269
271
277
281
283
293
307
311
313
317
331
337
347
349
353
359
367
373
379
383
389
397
401
409
419
421
431
433
439
443
449
457
461
463
467
479
487
491
499
```

4. Write a function in python to find the sum of the cube of elements in a list. The list is received as an argument to the function, in turn, the function must return the sum. Write the main function which invokes the above function.

```
def cubesum(vals):
    cubetotal = 0
    for val in vals:
        cubetotal += val**3
    return cubetotal

def main():
    nums = [1,2,3,4]
    sum = cubesum(nums)
```

```
        print(sum)

    main()
```

```
     100
```

5. Write the definition of a function zero_ending(scores) to add all those values in the list of scores, which are ending with zero and display the sum. For example: If the scores contain [200, 456, 300, 100, 234, 678] The sum should be displayed as 600.

```
    def zero_ending(scores):
        sum = 0
        for num in scores:
            if num%10 == 0:
                sum += num
        return sum

    scores =  [200, 456, 300, 100, 234, 678]
    total = zero_ending(scores)
    print(total)
```

```
     600
```

6. Write a definition of a method count_now(places) to find and display those place names, in which there are more than 5 characters. For example: If the list places contain ["DELHI","LONDON","PARIS","NEW YORK","DUBAI"] The following should get displayed: LONDON NEW YORK

```
    def count_now(places):
        for place in places:
            if len(place) > 5:
                print(place)

    places = ["DELHI","LONDON","PARIS","NEW YORK","DUBAI"]
    count_now(places)
```

```
     LONDON
     NEW YORK
```

7. Write a method in python to display the elements of list thrice if it is a number and display the element terminated with '#' if it is not a number. For example, if the content of list is as follows: ThisList= ['41', 'DROND', 'GIRIRAJ', '13', 'ZARA'] The output should be 414141 DROND# GIRIRAJ# 131313 ZARA#

```
    def parselist(l):
        for cur in l:
            if cur.isdigit():
```

```
            print(cur*3)
        else:
            print(cur+'#')

list1 = ['41','DROND','GIRIRAJ', '13','ZARA']
parselist(list1)
```

```
414141
DROND#
GIRIRAJ#
131313
ZARA#
```

8. For a given list of values in descending order, write a method in python to search for a value with the help of Binary Search method. The method should return position of the value and should return -1 if the value not present in the list.

```
def binary_search(items, item):
    first = 0
    last = len(items)-1
    while(first<=last):
        mid = (first+last)//2
        if items[mid]==item:
            return mid
        elif items[mid]>item:
            first=mid+1
        else:
            last=mid-1
    return -1


l = [64,10,53,12,6,34,99]
index = binary_search(l,2)
if index == -1:
    print('Element not found')
else:
    print('Element found at position',index+1)
```

```
Element not found
```

```
index = binary_search(l,64)
if index == -1:
    print('Element not found')
else:
    print('Element found at position',index+1)
```

```
Element found at position 1
```

9. Write a function half_and_half that takes in a list and change the list such that the elements of the second half are now in the first half. For example, if the size of list is even

and content of list is as follows : my_liist = [10,20,30,40,50,60] The output should be [40,50,60,10,20,30] if the size of list is odd and content of list is as follows : my_liist = [10,20,30,40,50,60,70] The output should be [50,60,70,40,10,20,30]

```
def half_and_half(templist):
    if len(templist)%2 == 0:
        start = 0
    else:
        start = 1

    L = len(templist)//2

    for i in range(L):
        temp = templist[i]
        templist[i] = templist[i+L+start]
        templist[i+L+start] = temp

l = [10,20,30,40,50,60,70]
half_and_half(l)
print(l)
```

```
 [50, 60, 70, 40, 10, 20, 30]
```

10. Write a function that accepts a dictionary as an argument. If the dictionary contains replicate values, return an empty dictionary, otherwise, return a new dictionary whose values are now the keys and whose keys are the values.

```
def swapping(d):
    L = list(d.values())
    for value in L:
        if L.count(value) > 1:
            return dict()

    new_dict = {}
    for k, v in d.items():
        new_dict[v] = k
    return new_dict


d = {'a':1,'a2':2,'a3':3}
print(d)
n = swapping(d)
print(n)

d = {'a':1,'a2':2,'a3':3}
print(d)
n = swapping(d)
print(n)
```

```
 {'a': 1, 'a2': 2, 'a3': 3}
 {1: 'a', 2: 'a2', 3: 'a3'}
```

```
{'a': 1, 'a2': 2, 'a3': 3}
{1: 'a', 2: 'a2', 3: 'a3'}
```

✓ 0s completed at 12:08 AM