15-110 Refresher Session : Week 8(Exam Review)

No Calculators, only Brains !!

1. **Kahoot Time !!**
   Please use your devices to scan the QR code and let's play!

---

2. **Short Code Question**
   In the following questions, solve the problem using a maximum of the number of lines indicated in each part.

   1. (1 line) Return the list L after dropping the first n and last k elements.
      `def dropFirstAndLast(L, n, k):`

   2. (3 lines) Return the elements at the even indices in the reverse order
      `def evenIndicesReverse(L):`

   3. (2 lines) Return the middle digit of the middle element in L - a list of n-digit numbers. You can assume that L has odd number of elements
      `def middleOfMiddleL(L, n):`

   4. (1 line) Return the sum of every 3rd number in the list from the beginning.
      `def sum3rd(L):`

   Note : Please try out all the list functions

---

3. **Act like a Computer**

```
def helper(x , d = 4) :
    return (x + d) * 3

def mystery(L,x,y):
    v = 0
    while v < len(L):
        L = L + [helper(x)]
        L[v] = L[v] + helper(x , 2)
        if v % 2 == 0 :
            L.pop(v)
        v +=1
    maximum = max(L)
    k = L.index(maximum)
    return (k, L[y::-1])
```

What does the function above return when `mystery([2 , 9 , 3 , 4], 10 , 2)` is executed?

---

4. **Act like a Programmer**
   A multiple linear regression can be modelled using the equation $y = w_1x_1 + w_2x_2 + ... + w_nx_n + b$, where y is the dependent variable, $w_1, w_2, w_3, ...w_n$ are the weights and $x_1, x_2, x_3, ...x_n$ are the independent variables, and b is the bias.

   Your task is to write a function `findY(W, X, b)` which takes two lists `W` and `X` with elements $w_0, w_1, w_2, ...w_{n-1}$ and $x_0, x_1, x_2, ...x_{n-1}$ respectively and a number `b`. The function has to calculate and return y. You can assume that the length of both the lists are equal.

```
findY([0.1, 0.2, 0.3], [10, 20, 30] , 0.8) returns 14.8
findY([0.06, 0.34], [1, 2] , 0.1) returns 0.84
findY([0, 0, 0, 1], [1, 2 , 2 , 4] , 0.5) returns 4.5
```

5. **Act like a Computer**

```
def mystery2(L) :
    print(sorted(L))
    print(L)
    if (sorted(L) == L.sort()):
        print("Then why is it not the same")
    print(L[::-1])
    L = [L] + [len(L)]
    L.append([1 , 9])
    print(L[1])
    return L[L[-2]::-2]
```

(a) What would `mystery2([6 ,4 ,5])` print and return ?

(b) What would `mystery2([20, 9])` print and return

6. **Act like a Programmer**
This questions has two parts and you have to implement both parts to get full credit. *You can use other helper functions when necessary*

(a) `isFunnyNumber(x)` : A number is said to be funny if it is positive, has two digits and the sum of the squares of the digits is even. Your task is to write a function `isFunnyNumber(n` which takes a number `n` and returns `True` if the number is funny and `False` otherwise.

```
isFunnyNumber(38) returns False
isFunnyNumber(37) returns True
isFunnyNumber(381) returns False
isFunnyNumber(-38) returns False
```

(b) `nthFunnyNumber(L, n)` : In this part, you have to implement a function `nthFunnyNumber(L)` which takes a list `L` and an integer `n` and returns the nth funny number in the list. If there aren't enough funny numbers in the list, return -1.

```
nthFunnyNumber([1 , 38, 3 , 103, 45 , 44, 17 , 5 , 20] , 4) returns -1
nthFunnyNumber([1 , 38, 3 , 103, 45 , 44, 17 , 5 , 20] , 3) returns 20
```

7. **Act like a Programmer**
Implement a function `sieveOfPrimes(L)` which takes a list of numbers `L`, removes all the numbers which are not prime, and returns the list.

```
sieveOfPrimes([4, 6, 7, 9. 10, 23, 17, 9) returns [7, 23, 17]
sieveOfPrimes([4]) returns []
```

8. **Act like a Programmer**
A football league directory was hacked and several of the match statuses were edited in the database. The league management found that there were certain patterns in the changes made as follows:
   (a) Each win was edited to a number greater than 1
   (b) Each loss was changed to a number less than 0
   (c) Each draw was scaled to a number between 0 and 1 (both inclusive)

   As the programmer of the team, your task is as follows:
   (a) Create a function `reverseHack(L)` which takes a list of lists `L` with each row denoting the performance of a team and each element in the inner list is a value denoting that particular team's win/loss/draw.
   (b) Reinstate the match status to original status. The original status : Each win is 1 point, draw is 0.5 and loss is 0
   (c) Print the scores of the top three teams as shown in the example.

```
reverseHack(
 [[4, 4 , 3],
  [-17, 0, -19],
  [9, -8, 0.3],
  [-0.5, 14, 1 ],
  [7, 0, 2],
  [-1, -10, -2 ]] returns  [3, 2.5, 1.5]
```