15-110 Refresher Session : Week 7

No Calculators, only Brains !!

1. **Act like a Computer**
   Given the following python function,

```
def ref_1(L ,m , n) :
    x = len(L[m:n])
    y = len(L)
    z = L[x:y:2]
    sum1 = 0
    sum2 = 0
    sum3 = 0
    for i in z :
        sum1 = sum1 + i
    for j in range(len(z)):
        sum2 = sum2 + j
    for k in L[::-1]:
        if k < y :
            sum3 = sum3 + k
    return (sum1, sum2, sum3)
```

   What would the function print for each of the calls below ?

   1. `ref_1([49, 32, 5, 4, 12, 43, 23, 5, 2] , 2 , 7)`
   2. `ref_1([3 ,5, 7, 8 ,9, 10] , 1 , 4)`
   3. *(Think)*: What is the type of data returned by this function ? How is it different from lists?

2. **Act like a programmer** Implement a python function `modifyList(L)` which takes a list `L` as a parameter and returns a modified list with the following properties : elements at even indices are replaced by half its value and elements at odd indices are replaced by double of its value. The returned list should only have integers. For example,

```
modifyList([3 ,5, 7, 8 ,9, 10]) returns [1, 10, 3, 16, 4, 20]
modifyList([49, 32, 5, 4, 12, 43, 23, 5, 2]) returns [24, 64, 2, 8, 6, 86, 11, 10, 1]
```

3. **Act like a computer:**
   Given the function below

```
def ref_2(a, b):
    for c in a:
        for i in range(len(b)):
            if (b[i] not in a):
                print("A", end="")
                b[i] = b[i] + c
            if (c % 2 == b[i] % 2):
                print("B", end="")
                b[i] = b[i] * 2
            elif (b[-1] != c):
                print("C", end="")
                b[i] = c
    return (a , b)
```

   What would `ref_2([4],[2,3])` print and return ?

4. **Act like a programmer:**
   Implement a function `replacePairs(L1, L2)` which takes two lists `L1` and `L2` as parameters and does the following:

   1. If an element of L2 appears in list L1, that element becomes 0 in L2.
   2. If two elements in L1 adjacent to each other are equal, they are replaced by 0s in L1. *For the ease of implementation, you can assume that an element can only be repeated twice in L1 and does not contain 0s initially.*
   3. A new list is created with the following elements in the order : length of L1, L1 with only odd-indexed elements, length of L2 , L2 With only even-indexed elements
   4. The new list is returned

```
replacePairs([3, 3], []) returns [2, [0], 0, []]
replacePairs([2, 4 ,4, 5], [4]) returns [4, [0, 5], 1, [0]]
replacePairs([3, 5, 7, 8, 8], [1, 6, 8]) returns [5, [5, 0], 3, [1, 0]]
```

5. **Additional Practice: Grocery Price Calculator**
   Assume you have two lists :

   1. `quantities` denotes the quantity of each item that needs to be purchased
   2. `prices` denotes the price of each item

   Your task is to implement a function `groceryPrice(quantities, prices)` taking two lists as parameters such that the quantity and price of the first item are the first elements in the list, quantity and price of the second item are the second items in the list, etc. and calculate the total amount you have to pay at checkout. For example,

```
groceryPrice([1 , 2,  3], [1.5 , 2.5 , 3.5]) returns 17.0
groceryPrice([1 , 5,  1], [1.5 , 1 , 3.5]) returns 10.0
```