Assignment Day 2 Homework Solution¶
Data Science 101 Course
Name-chaithra k
Mail Id – chaithravalanja@gmail.com

Questions 1:
Create an empty list. Accept 10 numbers from the user and append to it the list if it is an even number.
In [11]:

```
a=[]
for i in range(10):
    b=int(input("Enter elements:"))
    a.append(b)
even=[]
odd=[]
for j in a:
    if(j%2==0):
        even.append(j)
    else:
        odd.append(j)
print("The even list",even)
```

```
Enter elements:1
Enter   elements:2
Enter   elements:3
Enter   elements:4
Enter   elements:5
Enter   elements:6
Enter   elements:7
Enter   elements:8
Enter   elements:9
Enter elements:10
The even list [2, 4, 6, 8, 10]
```

Question 2
Create a notebook on LIST COMPREHENSION. This exercise is to put you in a Self learning mode
List Comprehension vs For Loop in Python
Suppose, we want to separate the letters of the word human and add the letters as items of a list. The first thing that comes in mind would be using for loop.
For loop
We will see the difference between for loop and list comprehension.

Iterating through a string Using for Loop

In [12]:
```
h_letters = []

for letter in 'human':
    h_letters.append(letter)

print(h_letters)
```
```
['h', 'u', 'm', 'a', 'n']
```
List Comprehension.

List comprehension is an elegant way to define and create lists based on existing lists.

Let’s see how the above program can be written using list comprehensions.

In [13]:
```
h_letters = [ letter for letter in 'human' ]
print( h_letters)
```
```
['h', 'u', 'm', 'a', 'n']
```
Conditionals in List Comprehension

List comprehensions can utilize conditional statement to modify existing list (or other tuples). We will create list that uses mathematical operators, integers, and range()

In [14]:
```
# Using if with List Comprehension
number_list = [ x for x in range(20) if x % 2 == 0]
print(number_list)
```
```
[0, 2, 4, 6, 8, 10, 12, 14, 16, 18]
```
In [15]:
```
# Nested IF with List Comprehension
num_list = [y for y in range(100) if y % 2 == 0 if y % 5 == 0]
print(num_list)
```
```
[0, 10, 20, 30, 40, 50, 60, 70, 80, 90]
```
In [16]:
```
# if...else With List Comprehension
obj = ["Even" if i%2==0 else "Odd" for i in range(10)]
print(obj)
```
```
['Even', 'Odd', 'Even', 'Odd', 'Even', 'Odd', 'Even', 'Odd', 'Even', 'Odd']
```
Key Points to Remember

List comprehension is an elegant way to define and create lists based on existing lists.

List comprehension is generally more compact and faster than normal functions and loops for creating list.

However, we should avoid writing very long list comprehensions in one line to ensure that code is user-friendly.

Remember, every list comprehension can be rewritten in for loop, but every for loop can’t be rewritten in the form of list comprehension.

Questions 3:
You have seen in the videos how powerful dictionary data structure is.
In this assignment, given a number n, you have to write a program that generates a dictionary d which contains (i, i*i), where i is from 1 to n (both included).
Then you have to just print this dictionary d.
Example: Input: 4 will give output as {1: 1, 2: 4, 3: 9, 4: 16}
Input Format: Take the number n in a single line.
Output Format: Print the dictionary d in a single line.
In [17]:

```
n = int(input(" "))
d = {i:i*i for i in range(1,n+1)}
print(d)
```

```
 10
{1: 1, 2: 4, 3: 9, 4: 16, 5: 25, 6: 36, 7: 49, 8: 64, 9: 81, 10: 100}
```

Questions 4:
In [18]:

```
pos=0
moves={"UP":1j,
       "DOWN":-1j,
       "LEFT":-1,
       "RIGHT":1}

#Set inputs
data=["UP 5",
    "DOWN 3",
    "LEFT 3",
    "RIGHT 2"]

#Move robot on valid moves
for inp in data:
    parts=inp.split()
    mv=parts[0]
    val=parts[1]
    print (mv, val)
    pos += moves[mv]*int(val)

#get distance
print('Rounded distance from zero position:', round(abs(pos)))
```

```
UP 5
DOWN 3
```

```
LEFT 3
RIGHT 2
Rounded distance from zero position: 2
```

```
In [ ]:
```

```
In [3]:
```