Day 04.

Randomisation and Python Lists.

Sometime or usually we want to have a different outputs every time or a random output from given set of avaiable item.

This is basically used in games or where the program interact with users.

There, we use Randomisation.

The random moudle is a built in module to generate the pseudo-random variable.

It can be used to perfor some action randomly such as to get random number, selecting a random elements from a list, shuffle element randomly and much more.

Random Number.

1. Choice() — Choice() is an inbuilt function in the python programming language that returns a random item from a list, tuple or String.

eg. import random ← It's imp to import the module we want to use.

list1 = [1, 2, 3, 4, 5, 6] → Tuple.

print (random.choice(list1))

random → Accesing the Module

choice(list1) → In module calling choice function with list1 as parameter.

2. randrange(beg, end, step)

The random ~~mould~~ module offers a function that can generate random numbers from a specified range and also allowing rooms for steps to be included called — randrang().

```
import random
print("random.randrange(20, 50, 3))
```

3. random()

This method is used to generate a float random number less than 1 and greater or equal to 0.

4. Seed()

Seed function is used to save the state of a random function so that it can generate some random numbers on multiple Executions of the code on the same machine (for a specific seed value). The seed value is the previous value number generated by generator. For the first time where there is no previous value, it uses current system time.

5. Shuffle () —

It is used to shuffle a sequence (list).

Shuffling means changing the position of the elements of the sequence. Here, the shuffling operation is in place.

```
import random
#declaring a list
Sample_list = ['A', 'B', 'C', 'D']
print ("original list: ")
print (sample_list)
#first shuffle.
random.shuffle (sample_list)
print ( After the first shuffle: ")
print (sample_list)
```

** random.shuffle (sample_list) originally shuffle the real variable.

6. Uniform (a, b)

This function is used to generate a floating point random number between the numbers mentioned in its arguments. It takes two arguments, lower limit (included in generation) and upper limit (not included in generation).

$\longrightarrow [a, b)$

Coding Exercise 4.1

Heads or Tails.

```
import random
side=random.randint(0,1)
if randomside==1:
    print("Head")
else:
    print("Tails")
```

Python Lists.

Lists is a data structure in python.

datastructure is a way data are stored

previously we used

a = 10

This is used to store single data.

Sometime we have to store many small data to a single variable which have connection.

fruits = [item1, item2]

List declaration.

List can store datatype, it can have mixed datatype.

fruits = ["Cherry", "Apple", "Pear"]

Lists also support indexing and -

eg.

```
print(fruits[1])    #Starts from 0.
```

→ Apple

Also, we have -negative index.

-1 → returns the last item in the list

-2 → return the second last in the list.

and so on.

Basically negative means that its going to start shifting the cursor from end.

- We can change any particular item in list by just assigning new item at that same index

like,

```
fruits[0] = "Mango"
print(fruits)
```

→ ['Mango', 'Apple', 'Pear']

item changed #

- Also we can add new items to the List

```
fruits.append("Grapes")
```

→ append() add new item to Grapes.

Refers to DOCS for more ways and different ways of adding items.

Coding Exercise 4.2

str.split() ⟶ It's a function that directly ~~convert~~ convert str in list by separating out the commas using

str.split(',').

```
str = "Hello, I am, Abhishek Kushwaha"
op = str.split(",")
print(op)
```

⟶ ['Hello', 'I am', 'Abhishek Kushwaha']

Solution

```
names = input("Give me everybody's names :")
names = names.split(",")
person_who_pay = ~~random.randint(len(names))~~
                 random.~~randrange~~ randint(0, len(names)-1)
person_who_pay = names[person_who_pay]
print(f"{person_who_pay} is going to by the meal today!")
```

Or use choice()

```
person_who_pay = random.choice(names)
```

Single line! cool!!

Nested List.

Nested list are nothing but a list comprehension within another list comprehension within another lists.

Fruits = [["Apple", "Mango", "Grapes"],
["Pear", "fruits1", "fruits2"]]

Simply, A nested list is a lits of lists, or any list that has another list as an element (a sublist).

- They can be helpful if you want to create a matrix or need to store a sublist along with other data types.

Lets take a Example:-

matrix = [[0, 1, 2],
[3, 4, 5],
[6, 7, 8],
[9, 10, 11]]

we can access each item from the Nested list. like if we want to return 11, then first we have to note whats the ~~no~~ column no and row.no.

= matrix[column-1][row-1]

For, 11,
column = 3, row = 4.

11 = matrix[2][3] ←

Index Errors.

While working with list, if you get an error most offer that's Index Error.

Let's understand what Index Error is:

When we created the list, it has the items and they are assigned with some rows and column no for accessing.

When accessing or checking, if we pass the row & column no, which is not present or unaccessible we get Index Error.

eg.

```
Lists = [['1', '2', '3'],
         ['4', '5', '6']
         ['7', '8', '9']]
```

In this list we have items starting from

row – 0
Column – 0 } to → row – 2
Column – 2

If we want to access an item at [3, 3] we will land with Index Error stating index value out of range.

Means, there is no item at Lists[3][3].

Project:- Rock, Paper, Scissors.

Rock =

Paper =

Scissors =

```
user_choise = int(input(" 0 for Rock, 1 for paper, 2 for scissors"))
computer_choice = random.randint(0, 2)
game_images = [rock, paper, scissors]
if user_choice == computer_choice:
    print("Its a draw!")
elif user_choice == 0 and computer choice == 1:
    print("You win!")
elif computer_choice == 0 and user_choice == 2:
    print("You lose")
elif computer_choice > user_choice:
    print("You loose")
else:
    print("You typed an invalid number, You lose!")
```

Coding Exercise 4.3. (Refer to Exercise & directory).

```
hoziontal = int(position[0]-1)
Vertical = int(position[1]-1)
```

where you want to put the treasure.

```
map[hoziontal][vertical] = "Treasure emoji"
print(map)
```