Python - Day 4 (Beginner)

Topic :- Randomisation, List, List Operations, for loops, Range function,

(1) Randomisation

Randomization lets your program produce unpredictable results, important for games, simulations, etc.

(1.1) Importing Random Module

```
import random
```

(1.2) Random integer

```
random.randint (1, 10)
```

Returns any integer between 1 and 10 (inclusive)

(1.3) Random float

```
random.random ()
```

Returns a float between 0.0 → 1.0
Multiply it to get a custom range:

```
random.random () * 10
```

# 0.5 → 5.0 (if the output of random.random is 0.5 multiplying it with 10 can give 5.0 so we can have custom Output)

(2) Lists in python

What is a list?

A list is a collection of items stored in one variable

It can contain:

- Strings
- numbers
- floats
- Even other lists

Example:

fruits = ["apple", "banana", "mango"]

Key Notes to Remember About lists

- lists are ordered → items keep their position
- lists are indexed starting from 0
- lists are mutable → you can change them
- lists allow duplicates

Accessing lists items:

```
fruits[0]  # apple
fruits[1]  # banana
fruits[-1] # last item: mango
```

Updating an item

```
fruits[1] = "grapes"
```

Adding items:

```
fruits.append("Orange") # adds one item
fruits.Extend(["Kiwi", "pear"]) # adds multiple
                                  items.
```

Nested list

A nested list means a list inside another list

Think of it like:

- A shelf with multiple boxes
- Each box contains items
- the main list = shelf
- sub-lists = boxs inside the shelf

(1) what is a Nested list?

```
fruits = ["apple", "banana", "mango"]
vegetables = ["carrot", "broccoli", "spinach"]

food = [fruits, vegetables]
```

Here

```
food = [
        ["apple", "banana", "mango"],
        ["carrot", "broccoli", "spinach"]
        ]
```

A nested list is simply a list where Each Element is another list.

(2) why do we use Nested lists?

Because Something Sometimes data has leve

Example

- Row & colums (like a grid)
- chessboard
- 2D coordinates

Nested lists help us organize data in structured form

(3) Accessing items in Nested lists
To access items inside nested lists, we use two indices

format
Outer list [inner list index][item index]

Example:

```
food = [
        ["apple", "banana", "mango"],
        ["carrot", "broccoli", "Spinach"]
    ]
print(food[0][1])   # banana
print(food[1][2])   # Spinach
```

Explanation
- food[0] = first Sublist = ["apple", "banana", "mango"]
- food[0][1] = Second item in Sublist = "banana"

(4) changing value in Nested lists
you can update items inside nested lists the Same way.

```
food[1][0] = "tomato"
```

this changes "Carrot" → "tomato"

(5) Visual Understanding (Very Important)
Think of a nested lists as row

Row 0 → ["apple", "banana", "mango"]
Row 1 → ["carrot", broccoli", "spinach"]

So:

food [row][column]

Examples:

food [0][0] → apple
food [0][2] → mango
food [1][1] → broccoli

(3) Splitting strings into lists

Useful for taking multiple inputs at Once

```
names = "mohit, aman, Sakshi"
names list = names. Split(", ")
```

(4) for loops

what is a for loop?

A for-loop is used when you want to repeat an action for Each item in a list, range, or Sequence

Example:

```
for fruit in fruits:
        print (fruit)
```

This prints Each fruit One by One.

Imptant Notes About for loops

- you do not need to know the index
- The loop variable (eg., fruits) takes each value from the list
- loop body must be indented

Summing using loops

```
total = 0
for num in [1, 2, 3]:
    total += num
```

(5) Range function

Used to generate a sequence of numbers

```
for i in range (1, 6):
    print(i)
```

prints 1 to 5

With step:

```
for i in range (0, 10, 2):
    print(i)
```

(6) Important Concept Notes for Day 4

Lists :-

- Keep items of the Same type when possible (clean)
- index start at 0
- Use .append() for adding one item.
- Use .extend() for adding multiple
- Use .Split() to convert a string → list

Random

- random.choice (list) is best for games
- random.randint (a, b) includes both a & b
- random.random () always gives float 0-1
- random.shufle(l) shuffles the list randomly
  list name.

for Range

- range (n) goes from 0 to n-1
- range (a,b) goes a - b-1
- range (a,b, Stop) skips numbers

for loops

- Use when you know how many times you want to repeat
- loop variable name should be meaningful (not always i)
- Great for lists, summation, searching, automation.

(*)(*) Mini project - Rock, paper, Scissors (*)(*)

Important Notes About this project

- ALWAYS convert input using .lower()
- validate user input → avoid crashes
- Use random.choice() for clean code
- Winning Conditions must be Explicitly written
- Understanding the logic not just memorise.

Key takeaways of Day 4

- lists help store multiple values.
- for loops help process those values.
- Randomisation makes programs unpredictable and fun