Black Origin

Conditional Statements

if-elif-else (SYNTAX)

```
if(condition) :
        Statement1
elif(condition):
        Statement2
else:
         StatementN
```

Black Origin

Let’s Practice

Question 1: Movie Ticket Pricing Based on Age Write a program that takes the user's age and prints the ticket price:

Age below 5 → "Free entry!"
Age between 5 and 12 → "Child Ticket: 100 Taka"
Age between 13 and 59 → "Regular Ticket: 250 Taka"
Age 60 or above → "Senior Ticket: 150 Taka"

Black Origin

Let’s Practice

Question 2: Water Level Indicator Write a program that asks the user to enter the current water level in liters.
Respond with:
≥ 100 liters → "Overflow Warning!"
70–99 liters → "Tank is full."
30–69 liters → "Tank is half-filled."
< 30 liters → "Refill the water tank."

Black Origin

Let’s Practice

Question 3: Determine the Type of Triangle Write a program that takes three sides of a triangle and determines its type:

If all sides are equal → "Equilateral Triangle"
If any two sides are equal → "Isosceles Triangle"
If all sides are different → "Scalene Triangle"

Black Origin

Nesting

One statement under another statement

Example:

```
if(condition) :
        if(condition) :
                Statement1
```

Black Origin

Let's Practice

Print output for:

A =5 & G = M

A=2 & G =F

```
A = int(input("A :"))
G = input("M/F :")

if ((A ==1 or A==2) and G == "M"):
    print("fee is 100")
elif (A ==3 or A==4 or G == "F"):
    print("fee is 200")
elif (A ==5 and G == "M"):
    print("fee is 300")
else:
    print("no fee")
```

Black Origin

Conditional Statements

Single line if / Ternary Operator

<var>= <val1> if <condition> else <val2>

```
food = input ("food : ")
eat = "Yes" if food == "cake" else "no"
print(eat)
```

<stt1> if <condition> else <stt2>

```
food = input ("food : ")
print("sweet") if food == "cake" else print("not sweet")
```

Black Origin

Conditional Statements

Clever if / Ternary Operator

<var>=(false_val, true_val) [<condition>]

```
age = int(input("age :"))
vote=("no", "yes") [age>=18]
print(vote)
```

```
salary=float(input("salary :"))
tax=salary*(0.1, 0.2) [salary>=500000]
print(tax)
```

Best Practices

- Simple Instructions
- One instruction per task
- Short & meaningful variable names
- Use appropriate comments
- Avoid complex expressions

Best Practices

Calculate Simple Interest

```
a = float(input("a : "))
b = float(input("b : "))
c = float(input("c : "))
print(a*b*c/100)
```

```
principal = float(input("principalAmount : "))
rate = float(input("rate : "))
time = float(input("Year : "))
simpleInterest = (principal*rate*time)/100
print(simpleInterest)
```

Let’s Practice

Write a program that asks the user to enter a number and checks whether it is divisible by 2. Print "Divisible by 2" if it is, otherwise print "Not Divisible by 2".

Let’s Practice

Write a program that takes three numbers as input and prints the largest among them. If two or more numbers are equal and the largest, print "There is a tie".

Let's Practice

Write a program that takes four numbers as input and prints the maximum value among them using a built-in function.

Let’s Practice

Write a program that asks the user to enter a number and checks if it is a multiple of 5. Print "Multiple of 5" if it is, otherwise print "Not a multiple of 5".

Chapter 3

Lists, Tuples, Dictionary & Set

Lists in Python

- A built-in data type that stores set of values

```
marks = [87, 64, 33, 95, 76]          #marks[0], marks[1]...
```

Lists in Python

- It can store elements of different types (integer, float, string, etc.)

```
student = [”Karan”, 85,“Delhi”]      #student[0], student[1]..

student[0] =“Arjun”                  #allowed in python

len(student)                         #returns length
```

List Slicing

Similar to String Slicing

```
list_name[ starting_idx : ending_idx ] #ending idx is not included

marks = [87, 64, 33, 95, 76]

marks[ 1 : 4 ] is [64, 33, 95]

marks[ : 4 ] is same as marks[ 0 : 4]

marks[ 1 : ] is same as marks[ 1 : len(marks) ]

marks[ -3 : -1 ] is [33, 95]
```

List Methods

```
list = [2, 1, 3]

list.append(4)                  #adds one element at the end [2, 1, 3, 4]

list.sort( )                    #sorts in ascending order [1, 2, 3]

list.sort( reverse=True )       #sorts in descending order [3, 2, 1]

list.reverse( )                 #reverses list [3, 1, 2]

list.insert( idx, el )          #insert element at index
```

List Methods

```
list = [2, 1, 3, 1]

list.remove(1)    #removes first occurrence of element [2, 3, 1]

list.pop( idx )    #removes element at idx
```