1. What is a variable in Python? - Explain what a variable is, how it is declared, and what types of data can be assigned to variables in Python.

Variable is name given to point the data in python.

Variable declaration rules:

1. Variable doesn't start with any number or special character rather than_
2. variable doesn't contain any special characters rather than _
3. varible can be start from _ or any alphabets.
4. optional choose Variable name which representative of data meaning
5. optional variable should not be longer length

In variable you can assign numerical type ,string type,sequence type ,maping data like dictionary type, or set type data

2. What is the purpose of f-strings in Python? - Explain how f-strings work for string formatting in Python and provide an example of how they are used to insert variables into a string.

When you want to display dynamic data along with string in that case you use f"string.

Eg.

Name="lava"

Print(f"Welcome { Name }") o/p –**Welcome lava**

For defining ‘f’ string you should write ‘f’ key and after that using double quotes or single quotes you can write string.

Inside of your f string with help of {} you can specify dynamic data.

3. What are the different data types in Python? - Describe the basic data types in Python such as integers, floats, strings, lists, tuples, dictionaries, and sets. Provide examples for each.

1. **Numerical type:** floats,int,complex

Int: in you can store integers type data

e.g no=23

float: in float you can store decimal type data.

e.g no=4.87

Complex: in this you can store complex type data real and imaginary

e.g 2+3j

2. **Sequential Type:- list,tuple,dictionary,string**

String: string in python is sequential collection of one or more **Unicode characters**

List: list is collection of heterogeneous data, it is ordered and indexed,

It is mutable, list can be defined with help of [ ]

Tuple: tuple is collection of heterogeneous data,it is ordered and indexed,

It is immutable,tuple can be defined with help of ( )

Dictionary: it is maping data type which stores data in the form of key value pairs

Keys in dictionary are unique ,dictionary is defined using {}

Set: set is collection of heterogeneous data ,it is unordered and unindexed, in set you can add or remove elements, empty can be defined using set using set() ,set elements are defined in {}

4. Explain how conditional statements work in Python.
- Describe the use of if, elif, and else statements in Python with an example. Explain their control flow.

when we want perform some operation Based on conditions in that case we can use conditional statements ,
in python we having if ,eilf and else statements for performing conditional operations
if : it is conditional statement used to define conditions and the block code will be executed once the condition is true

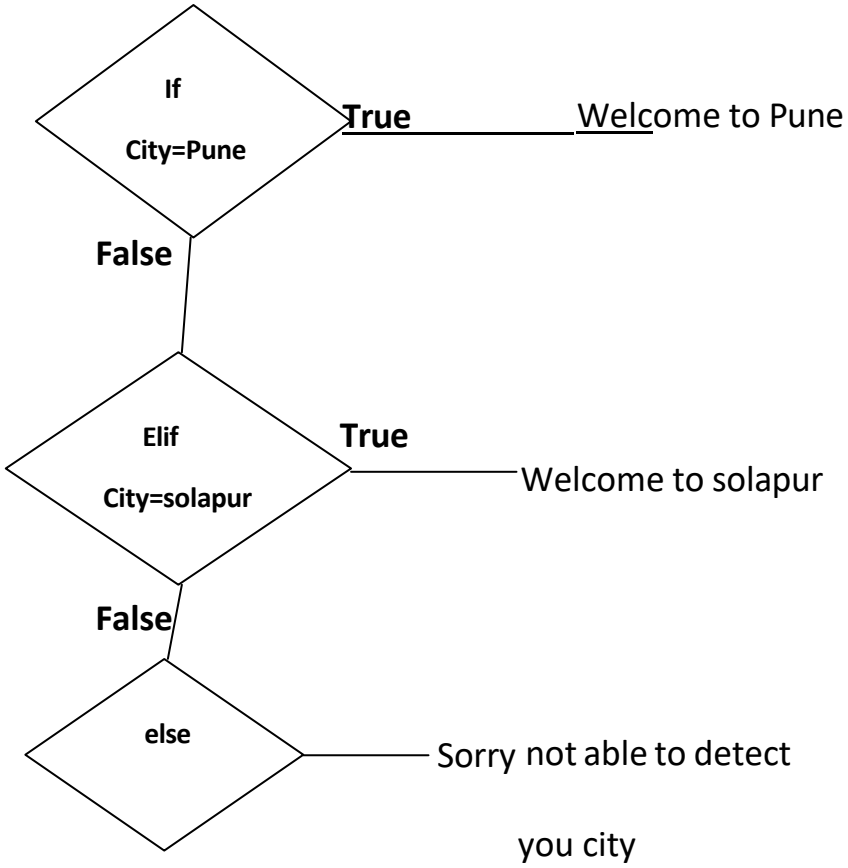
eilf: in your code if you having multiple conditions then you can define those conditions in **elif** , it executes when previous condition is not true.

else: it will excuted when no any conditions are true , you can't use else without if

```
city='Solapur'
if city=='Pune':
  print("Welcome to Pune")
elif city=='Solapur';
 print("Welcome to Solapur')
else:
 print("sorry we not able to detect your city")
```

Output:- Welcome to Solapur

Control Flow:-



5. What is the difference between a list and a tuple in Python?
- Explain the characteristics of both lists and tuples, including mutability, and when to use each.

List	Tuple
List can be defined by []	Tuple is defined by ()
List is slower due to dynamic size and Mutability	Tuple is faster due to immutability
It is suitable for collection of items may Change	It is suitable for collection of items that may not change
List having more inbuilt functions	Tuple having less inbult functions
List consumme more memory due to Mutability	Tuple is consume less memory due to immutablity

6. What is type casting in Python?
- Explain how type casting works in Python and how to convert one data type into another, providing examples.

Typecasting- it defines converting one data type into another datatype

In python we having 2 types of type casting

1) Implicit typecasting
2) Explicit typecasting

Implicit typecasting : when you executing code python will dynamically type caste the data
e.g
print('1'+2,type('1'+2))
output: '12' <class str>

here 1 is string type and 2 is numeric type
python dynamically converted 2 as string type and returned concated result

Explicit tycasting: in this you have to tell python to convert the given data to specified type by using methods
e.g:
print(1+'2')
output:- Type Error was throughn
in this scenario you have to explicitly convert 2 to int
print(1+int('2'))
output:- 3

7. How does the is operator differ from the == operator in Python?
- Explain the difference between identity comparison (is) and equality comparison (==) in Python.

1) **==** is equal to operator it only compares' the values.
e.g;

```
x = [1, 2, 3]
y = [1, 2, 3]
z = x
if x == y:
    print("True")
    print(id(x))
    print(id(y))
else:
    print("False")
```

output:-
True
1570673221952
1570673369920

Here address of x and y are different still it returns true and executed if block

2)IS – it is identity operator it compare values and address of both variables. If they are true,Then it will return true.

```
x = [1, 2, 3]
y = [1, 2, 3]
z = x


# Case 1: Identity comparison (is)
if x is y:
    print("True")
else:
    print("False")
    print(id(x))
    print(id(y))

# Case 2: Comparing references (is)
if x is z:
    print("True")
    print(id(x))
    print(id(z))
else:
   print("False")
```

output:-
False
1570673221952
1570673369920
True
1188639039808
1188639039808

Here in case-1 x & y are having different address and same values hence it returns **false**
In case-2 x & z are having same address and same value **hence it returns true**

8. What is a dictionary in Python?
Describe the concept of a dictionary in Python, including how to create one, add, and access elements using keys.

Dictionary: it is mapping data type which stores data. in the form of key value pairs
Keys in dictionary are unique ,dictionary is defined using {} and dict() method

D=dict()
this method will create the empty dictionary
d={}
this is also an empty dictionary
d={'item':'t-shirt','size':34}
you can also defined dictionary with elements

To add element In dictionary with help of key

```
d={}
d['name']="lavanya"
```

output:-{'name':'Lavanya'}

update :- with help of update method you can add more than one elements in dictionary

```
d={}
d.update({"Age":24,"Gender":"F"})
print(d)
```

output:- {'Age': 24, 'Gender': 'F'}

9. What are the advantages of using f-strings over other string formatting methods?

- Discuss the benefits of using f-strings in Python compared to older formatting techniques like % formatting or str.format().

1.Evaluating Expressions

f-strings work well when handling expressions and calculations directly within your strings. Instead of calculating values beforehand, you can perform operations right inside the curly braces.

2.Conditional expressions

One of the most powerful features of f-strings is their ability to include conditional logic. You can use the ternary operator to make your string formatting respond to conditions.

3. **performance wise f string is speed than other**

4. F-strings can also help with debugging by using the '=' specifier (introduced in Python 3.8) to display both variable names and their values,
as shown below:

```
s=10
print(f"{s=}")
output:
s=10
```

10. What is the difference between mutable and immutable data types in Python?

- Explain the difference with examples of mutable and immutable data types in Python, such as lists (mutable) and strings (immutable).

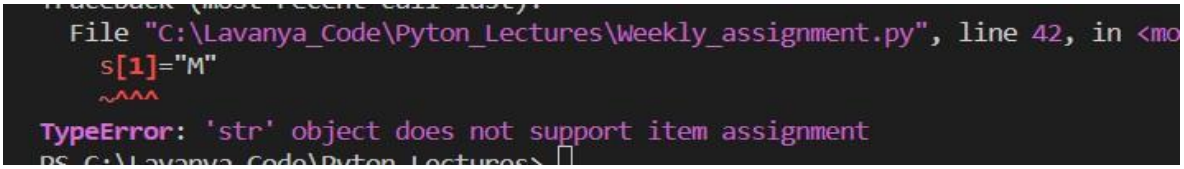
- **Mutable data type:** this data types are not allowed to make any

changes in original data

Eg. String,Tuple

```
s='la'
s[1]="M"
```

Output:-



- Immutable data type: **this data types are allowed to make any changes in original data**

  Eg. List,dictionary

```
s=[1,2]
s[1]="M"
print(s)
```

Output:-
[1,"M']

Section B: Correct the Code

1. **Code:**

```
x = 10
y = 5
if x > y
    print("x is greater than y")
```

Task: Fix the syntax error in the code.

Ans:

```
x = 10
y = 5
if x > y:
    print("x is greater than y")
```

2. **Code:**

```
name = "Alice"
age = 30
print("My name is {} and I am {} years old.".format(name, age))
```

Task: Rewrite the code using an f-string to format the string.

Ans:

```
name = "Alice"
age = 30
print(f"My name is {name} and I am {age} years old.".format(name, age))
```

3. **Code:**

```
fruits = ["apple", "banana", "cherry"]
print(fruits[3])
```

Task: Correct the code to print the last element of the list.

ANS:-

```
fruits = ["apple", "banana", "cherry"]
print(fruits[-1])
```

4. **Code:**

```
number = 12
if number > 10
   print("Greater than 10")
elif number == 10
   print("Equal to 10")
else:
   print("Less than 10")
```

Task: Fix the syntax error in the conditional statement.

ANS:

```
number = 12
if number > 10:
   print("Greater than 10")
elif number == 10:
   print("Equal to 10")
else:
   print("Less than 10")
```

5. **Code:**

```
a = 5
b = 10
print(f"Sum of a and b is: {a + b}")
```

Task: Explain why this code is correct and what the f-string does here.

ANS:

Boz expression of a+b is given in {}

6. **Code:**

```
age = 18
if age >= 18
   print("You are an adult")
else:
   print("You are a minor")
```

Task: Fix the indentation issue in the code.

ANS:-

```
age = 18
if age >= 18:
   print("You are an adult")
else:
   print("You are a minor")
```

7. **Code:**

```
x = "10"
y = 5
print(x + y)
```

Task: Correct the type error so that the program adds the two values as numbers.

ANS:-

```
x = "10"
y = 5
print(int(x) + y)
```

8. **Code:**

```
num = 0
if num > 0:
    print("Positive number")
elif num < 0:
    print("Negative number")
else:
    print("Zero")
```

Task: Correct the code to print whether the number is positive, negative, or zero correctly.

No correction found

9. **Code:**

```
colors = ("red", "green", "blue")
colors[1] = "yellow"
print(colors)
```

Task: Correct the code to show how tuples work with immutability.

ANS:-

```
colors =list( ("red", "green", "blue")) colors[1] = "yellow"
colors=tuple(colors)
print(colors)
```

10. **Code:**

```
fruits = ["apple", "banana", "cherry"] print(fruits["banana"])
```

Task: Fix the issue and explain how to access list elements by index.

ANS:- list element can accessible by index of element.

```
fruits = ["apple", "banana", "cherry"] print(fruits[1])
```

Section C: Write a Program

1. Write a Python program that declares two variables, x and y, and swaps their values using a third variable.
Show how to perform variable swapping and explain the steps in the program.

```
a=10
b=20
print("befor swaping a and b",a,b)
temp=a
a=b
b=a
print("after swaping a and b",a,b)
```

here storing the a value in temporary variable so that data will not overwritten after assigning to b and then assigning a to b and b to temp

output:-

```
PS C:\Lavanya_Code\Pyton_Lectures> python Weekly_assignment.py
befor swaping a and b 10 20
after swaping a and b 20 20
PS C:\Lavanya_Code\Pyton_Lectures>
```

2. Write a program that asks for a user's name and age, and then prints out "Hello, my name is <name> and I am <age> years old" using f-strings.
Ensure to take input from the user and format the string using an f-string.

```
name=input("enter Name: ")
age=int(input("Enter age: "))
print(f"Hello, my name is {name} and I am {age} years old")
```

output:-

```
PS C:\Lavanya_Code\Pyton_Lectures> python Weekly_assignment.py
enter Name: lava
Enter age: 24
Hello, my name is lava and I am 24 years old
```

3. Write a Python program that checks if a number is positive, negative, or zero using if-elif-else statements.

The program should ask for a number from the user and print an appropriate message based on the number.

```
14  num=int(input("Enter Number: "))
15  if num > 0:
16      print("Positive number")
17  elif num < 0:
18      print("Negative number")
19  else:
20      print("Zero")
```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Lavanya_Code\Pyton_Lectures> python Weekly_assignment.py
Enter Number: 20
Positive number
PS C:\Lavanya_Code\Pyton_Lectures> python Weekly_assignment.py
Enter Number: -20
Negative number
PS C:\Lavanya_Code\Pyton_Lectures> python Weekly_assignment.py
Enter Number: 0
Zero
PS C:\Lavanya_Code\Pyton_Lectures>
```

4. Write a Python program that calculates the square of a number entered by the user using f-strings.

Take a number as input, calculate its square, and print the result using f-string formatting.

```
25
26  num=int(input("Enter Number: "))
27  print(f"Squre of {num}: {num ** 2}")
```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Lavanya_Code\Pyton_Lectures> python Weekly_assignment.py
Enter Number: 11
Squre of 11: 121
PS C:\Lavanya_Code\Pyton_Lectures>
```

5. Write a Python program that converts a given string into uppercase or lowercase based on user input.

Use conditional statements to convert the string and print it accordingly.

```
24
25  s=input("Enter String")
26  if s.isupper():
27      print(s.lower())
28  else:
29      print(s.upper())
30
```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Lavanya_Code\Pyton_Lectures> python Weekly_assignment.py
Enter StringLAVANYA
lavanya
PS C:\Lavanya_Code\Pyton_Lectures> python Weekly_assignment.py
Enter Stringlavanya
LAVANYA
PS C:\Lavanya_Code\Pyton_Lectures>
```

6. Write a Python program that finds the type of a given variable and prints it.
The program should take a variable as input and print its datatype using the type() function.

```
34  var=input("Enter date")
35  print(type(var))
```

```
PROBLEMS 1  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Lavanya_Code\Pyton_Lectures> python Weekly_assignment.py
Enter datelavanya
<class 'str'>
PS C:\Lavanya_Code\Pyton_Lectures> python Weekly_assignment.py
Enter date1020
<class 'str'>
PS C:\Lavanya_Code\Pyton_Lectures>
```

7. Write a Python program that calculates the total price of items in a shopping cart.
The program should take item prices as input and print the total cost using a list or dictionary.

```
32
33  list=[100,200,300,400,2000]
34  print("Total Cost:",sum(list))
```

```
PROBLEMS 1  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Lavanya_Code\Pyton_Lectures> python Weekly_assignment.py
Total Cost: 3000
PS C:\Lavanya_Code\Pyton_Lectures>
```

8. Write a Python program that checks if a given number is a prime number or not.

The program should use conditionals and a loop to check if the number is prime.

```
32  #
33  n=int(input("Enter Number: "))
34  flag=0
35  for i in range(2,n):
36      if n%i==0:
37          flag=1
38  if(flag==1):
39      print("Number is Not Prime")
40  else:
41      print("Numbet is  Prime")
42
```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Lavanya_Code\Pyton_Lectures> python Weekly_assignment.py
Enter Number: 10
Number is Not Prime
PS C:\Lavanya_Code\Pyton_Lectures> python Weekly_assignment.py
Enter Number: 7
Numbet is  Prime
PS C:\Lavanya_Code\Pyton_Lectures> python Weekly_assignment.py
Enter Number: 2
Numbet is  Prime
PS C:\Lavanya_Code\Pyton_Lectures> python Datatype_python_assignment.py
PS C:\Lavanya_Code\Pyton_Lectures> 59
59
PS C:\Lavanya_Code\Pyton_Lectures> python Datatype_python_assignment.py
PS C:\Lavanya_Code\Pyton_Lectures> python Weekly_assignment.py
Enter Number: 60
Number is Not Prime
PS C:\Lavanya_Code\Pyton_Lectures>
```

9. Write a Python program to add two lists element-wise using a for loop and print the result.

Take two lists of numbers and add corresponding elements using a loop.

```
33
34  l1=[10,20,30,40]
35  l2=[1,2,3,4]
36  l3=[]
37  for i in range(len(l1)):
38      l3.append(l1[i]+l2[i])
39  print(l3)
40
41
42
43
```

```
PROBLEMS 1  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Lavanya_Code\Pyton_Lectures> python Weekly_assignment.py
[11, 22, 33, 44]
PS C:\Lavanya_Code\Pyton_Lectures>
```

10. Write a Python program to sort a list of strings alphabetically and print the sorted list.

The program should take a list of strings and sort them using Python's built-in sorting function.

```
34  l1=['l','a','v','c','n','y','a']
35  l1.sort()
36  print(l1)
37
```

```
PROBLEMS 1  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Lavanya_Code\Pyton_Lectures> python Weekly_assignment.py
['a', 'a', 'c', 'l', 'n', 'v', 'y']
PS C:\Lavanya_Code\Pyton_Lectures>
```