1.Introduction to python and its Features(simple, highlevel, interpreted language).

Python is an object oriented, high-level-general purpose, interpreted programming language, created by Guido van rossum in 1991. Python has clear syntax and large libraries support. Python uses in web dev, Machine learning, ai , automation etc.

FEATURES →
Simple : python is easy to read and write because it have a simple clear and syntax
High-leven : python code is same as english which makes it easy to understand
Interpreted : python uses interpreter which executes python code line by line on runtime and display output

2.History and evolution of python.
Python was created by Guido van rossum in 1991, intended to create a language which is easy to read and write and provide more easy syntax.
Evolution :
1919 : python 0.9.0 released publically first time.
1994 : python 1.0 released.
2000 : python 2.0 released, introduced **PEP8** , garbage collection and automatic memory management
2008 : python 3.0 released,introduced a new standard library, new data types.

3.Advantages of using python python over other programming languages.

- Cross platform support (windows, linux, macos)
- Simple and minimalist syntax
- Huge libraries support
- Large community : for learning resource and open source contributions
- Versatile application
-

4.installing python and setting up the development environment.
For windown go to python official website and download latest python installer and install it, check for path environment variables, download a code editor like vscode install python extention then create a file with .py extention and write hellow world program then run it.

5.write and execute your first python program.
Install latest python open a code editor create a file with.py extension write print(“hello world”) in it open terminal locate to directory where your file saved, type python filename.py and hit enter it will execute the fiel and display output.

6.Understanding pep 8 guideline.
pep8 is official style guide for python that provides the best practice to write python programs.
Pep 8 includes →
Code layout, use of indentation, max-line length, naming-conventions, use of white space, and case syntax

7.indentation, comments and naming-conventions in python.
Indentation : according to pep 8 use 4 space for indentation to indicate each level of depth in block
Comments : # symbo for single line comment, wrap text in “””” “””” for multiline comment
Naming-conventions : for naming variables and functions use snake_case, for classes use Pascal Case,

8.Write rateable and maintainable code.
According to pep 8

- follow the naming conventions for variables function and classes,
- meaning full names for variable and functions
- use comments for sections of code,
- use complex logics and repeated code as functions,
- use object oriented concepts for better code reusability.

9.Understanding data types: integer, float, string, list, tuples, set, dictionaries.
Int: integer, contain only whole numbers without fractional portion, can be negative or positive

Float: floating point, contain decimal portion, can be positive or negative

Str: strings are sequences of characters enclosed in double or single quotes**.** strings are immutables

List : list is a collection of similer and dissimiler items which is orderable, mutable and indexed and allow duplicate value. List represented by [ ] brakets.

Tuple: tuple is a collection of similer and dissimiler items which is ordered,indexed and immutable and allow duplicate value. Tuple represented by ( )

Dictionaries: a collection of unique key : value pair which is ordered mutable, allow multiple data types value

set : set is a collection of unique items which multiple data type, set is unordered and mutable represented by { }

10.Python variables and memory allocation.
Python is dynamically typed,and we don't need to specify the type before creating a variable.
Python has a auto memory management and a garbage collection that allocates delicate memory on runtime.

11.Python operators: arithmetic, comparison, logical, bitwise

Arithmetic:

+ : addition
- : subtraction
* : multiplication
/ : division
// : floor division
% : modulus

Comparison operators

== : Equal
!= : NotEqual
< : less than
> : greater than
<= : less than Equal
>= : greater than Equal

logical operators

and : logical and operator
or : logical or operator
not : logical not operator

bitwise

& : bitwise and
| : bitwise or
~ : bitwise not
^ : bitwise xor
>> : right shift
<< : left shift

12.Introduction to conditional statements if else , elif

conditional statements are used to execute a block of code when a certain condition is true
If : execute a block of code when a condition is true
else : when if statements condition is false than statements written in else block are executes
elif : to check another condition when if condition is false and

13.Nested if else condition.

When we have to check for an another condition inside a if else blick we nest another if else block inside it it called nested if else conditon

14.Introduction to for and while loop.
For loop is a sequence control loop it iterates over a sequence or an iterable and executes a block of code for each item in sequence.

```
for i in range(1,11):
        print(i)
```

While loop is entry control loop, it executes a block of code while a certain condition is true.

```
X = 0
while x < 10:
    print(x)
    X += 1
```

15.How loop works in python
Loop used for execute a block of code repetedly while a certain conditon is true.
Every loop has an initilizer a condtion and a update statemetn and block that executed on condtion true while conditon is treu initilizer value got updated and block executed

```
  x = 0
  while x < 10 :
     print(x)
     x += 1
```

16.Using loop with collections (list, tuple, etc).

```
l1 = [1,3,4,5,7]
  t = (2,3,5,7,8)

mydict = {
     "name" : "kp",
     "age"  : 23
  }
```

loop with list

```
   for item in l1:
      print(item)
```

loop with tuple

```
   for item in t:
      print(item)
```

loop with dictionary

```
   for k in mydict.keys():
      print(mydict[k])
```

20.Defining and calling functions in Python.
Functions are blocks of code that are executes on calling and used for reusing the complex logic and repeated code .
To define a function use def keyword , write function name followed by () parenthesis
To call a function type function name followed by parenthesis()

21.Function arguments (position, keyword and default).
positional arguments : arguments are assigned to parameters into the order they are passed

keyword : when passed argument and parameter has same name, the order of passing not matter as long as name is same

default : when no arguments are passed to function, then default value are applied to function is it defined a default parameter

22.Scope of variables in python
Scope refers to the area or section of program where a variable can be accessible and modified

Scopes in python:
Global : when variable defined on top of program or outside any block or function it has a global scope and can be accessed anywhere throughout program

local : when variable defined inside a block of code it has a local scope and it only accessible inside that block

enclosed : when have a function or a block inside of another function or block than variables defined inside outer most block are accessible inside nested block

23) Built-in methods for strings, lists, etc
strings methods →
len()
strip()
count()
upper()
lower()
title()
capitalize()
replace()
find()
split()
join()
isalpha()

isalnum()
isdigit()
startswith()
endswith()

list methods
len()
append()
insert()
extend()
pop()
remove()
index()
reverse( )
sort()
clear()
count()

24) Understanding the role of break, continue, and pass in Python loops
Break: break terminates the loop when a specific condition is true regardless of loop's condition
Continue: continue skip the current iteration of a loop when a certain condition is true and continue over next iteration
Pass: pass uses as a placeholder and do nothing, in a loop when a statement is required we use pass

25) Understanding how to access and manipulate strings.
strings written in double quotations and are immutable in python,
to access we store a string into variable : string = "some text"

to access string character use indexing =
string[0] : s
string[1] : o
or a range--
string[0:4] : some

string manipulation using methods :
use len(string)
upper()
lower()
title()
capitalize()
replace()
split()
f-string
etc..

26) Basic operations: concatenation, repetition, string methods (upper(), lower(), etc.).

concatenation --> concat a string with another using + operator
string = "kapil"
string = "hello " + string
string += " how are you"

using f"" string format (f-string)
string = f"hello {string} how are you"

repetition string : repeating a string using * operator
repeated = string * 3

string method :
upper()
lower()
title()
capitalize()
replace()
split()
zfill()

27) String slicing.

string slicing is a method to create a substring out of original string
string slicing done by specifing a range in [ ]
substring = string [start: end : step]

28) Using map(), reduce(), and filter() functions for processing data.

map(function, iterable)
map apply a function on all the items of a list and return a new list

filter(function, iterable)
filter apply a function on all the items of a list and and filter the results based on condition inside function