Name: Shaik Sameer Mujahid
Batch: Data Science with AI (April 1st)

Python

1. What is python?

Python is a high-level programming language that is known for its code readability and simplicity. It is commonly used in automation, artificial intelligence, data analysis, and online applications development. Python's design emphasizes readability of code, and its ease of use makes it accessible to both beginner and experienced programmers.

The main key features of python are:

Syntax: python's syntax is clear, making it easy to learn and write the code

Libraries: python has a vast number of standard and third-party libraries. These are used to extend the capabilities of the applications

Versatile: python is used in various domains, like in developing web applications using Django, Flask, and in data science using Pandas, NumPy and other libraries, it is also used in machine learning using TensorFlow, Scikit-Learn

2. What are the rules for creating variable name?

Rules for creating variable name

Rule 1: variable name must begin with a letter or an underscore(_)

Example: 'variable', '_variable'

Rule 2: the name can contain digits and underscores but not spaces

Example: 'variable1'

Rule 3: cannot start with digits

Rule 4: variables are case sensitive which means variable, Variable. VARIABLE each are distinct

3. What is the difference between count () and len ()?'

Count()

The count() is used to count the number of occurrences of the specific element in a sequence

Example:

List=[1,4,6,7,7,8]

Count=List.count(7)

Output: 2

Len()

The len() is used to determine the total number of elements in a sequence

Example:

List= List=[1,4,6,7,7,8]

Length= len(List)

Output: 6

4. What is type-casting and type () function in python?

Type-casting is known as type conversion, it is defined as a process when the value of one data type is converted to another datatype

Example:

X = ‘123’

Y = int(X)

Print(type(Y))

Output: <class 'int'>

There are few exceptions in this type casting

1. We cannot convert a non-numeric string into an integer
2. We cannot convert a non-numeric sting into a flo
3. Cannot convert list into an integer

These will raise a TypeError in the console

5. What are strings and string slicing? Explain with examples.

Strings:

A string is defined as a sequence of characters enclosed within single quotes (`' '`), double quotes (`" "`), or triple quotes (`''' '''` or `""" """`). Strings are immutable, meaning once created, their characters cannot be modified.

Example:

string = ‘hello’

string = “hello”

string = “’hello

world’”

triple quotes are used for multi line strings

Slicing:

String slicing is a way to extract or separate a portion of a string using range of indices

Syntax: string[start:stop:step]

start: The starting index of the slice, defaults to be beginning of unmentioned

stop: The ending index of the slice, Defaults to be the end of unmentioned

step: The step size or interval between indices, defaults to 1 if unmentioned.

Example:

String =’hello world’

New_string= string[0:5]

Output: hello

New_string= string[:7]

Output: hello w

New_string= sting[::-1]

Output: dlrow olleh

6. What are the data types in python explain them in details?

Data Types:

1. String(str): it represents a sequence of characters enclosed in single, double or triple quotes

string = ‘hello’

string = “hello”

string = ‘’’hello

world’’’

2. Integer(int): It represents whole numbers both positive and negative without any fractional parts

x = 10

y = -5

z = 0

3. Flot(float): it represents real numbers containing both integer and fractional parts

x = 10.5

y = -3.14

z = 0.0

4. List(list): it is defined as an ordered, mutable collection of items it is enclosed in square brackets([]) lists can contain of different data types

x = [1, 2, 3, 4, 5]

y = ['a', 'b', 'c']

z = [1, 'a', 3.14]

5. Sets(set): it is and unordered collection data type that is iterable, mutable and has no duplicate item, it is enclose in curly braces({})

x = {1, 2, 3, 4, 5}

y = {'a', 'b', 'c'}

6. Tuple(tuple): it is defined as an ordered, immutable collection of items it is enclosed in parentheses(()) lists can contain of different data types

x = (1, 2, 3)

y = ('a', 'b', 'c')

z = (1, 'a', 3.14)

7. Dictionary(dict): it is defined as an ordered, mutable collection of ley-value pairs it is enclosed in curly braces({}). Keys must be unique and immutable

x = {'name': 'Alice', 'age': 25, 'city': 'New York'}

y = {1: 'one', 2: 'two'}

dictionary does not accept mutable as mutable as keys

7. Explain for loop and while loop with examples?

For loop:

The for loop in Python is used to iterate over a sequence (such as a list, tuple, string, or range) or other iterable objects. It allows you to execute a block of code multiple times, once for each item in the sequence.

Syntax:

```
for i in range:
   #code
```

Example:

```
for I in range (1,5)
        print(i)
```

1

2

3

4

5

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

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

Output:

apple

banana

cherry

While loop:

The while loop in Python repeatedly executes a block of code as long as a given condition is True. It is used when the number of iterations is not known beforehand and depends on a certain condition being met.

Syntax:

```
While condition:
    #code
```

Example:

```
count = 0
while count < 5:
    print(count)
    count = count+1
```

Output:

1

2

3

4

5

Using a Brak statement:

```
count = 0
while True:
    print(count)
    count += 1
    if count >= 5:
        break
```

output:

0

1

2

3

4

8. What are the control statements and where do we use them? Explain with examples.

Control Statements:

Control statements are used to control the flow of execution in a program. They enable you to make decisions, repeat tasks, and break out of loops

Conditional statements:

If:

Syntax:

If condition

Else

Syntax:

If condition:

    #code

Else:

    #code

Elif

Syntax:

If condition:

    #code

Elif another_condition:

    #code

Else: # code

Example:

x = 10

if x > 0:

```
        print("x is positive")
elif x < 0:
        print("x is negative")
else:
        print("x is zero")
```

Output: x is positive

Break: it is used to exit a loop when a certain condition is satisfied.

Example:

```
for i in range(10):
        if i == 5:
                break
        print(i)
```

Output:

0

1

2

3

4

9. What is the difference between list and tuples? Explain some list methods with examples.

Lists and Tuples are both used to store collections of items in Python, but they have some differences:

1. Mutability:

Lists: Mutable, meaning their elements can be changed after creation. You can add, remove, or modify items in a list.

Tuples: Immutable, meaning once a tuple is created, you cannot change its elements

Syntax:

Lists are enclosed in square brackets [ ].

Tuples are enclosed in parentheses ( ).

List Methods:

Append(): adds an element to the end of the list

fruits = ['apple', 'banana', 'cherry']

fruits.append('orange')

print(fruits)

Output: ['apple', 'banana', 'cherry', 'orange']

Extend(): adds multiple elements to the list

fruits = ['apple', 'banana', 'cherry']

fruits2 = ['orange', 'pear']

fruits.extend(fruits2)

print(fruits)

Output: ['apple', 'banana', 'cherry', 'orange', 'pear']

Insert(): Inserts an element at a specified position in the list

fruits = ['apple', 'banana', 'cherry']

fruits.insert(1, 'orange')

print(fruits)

Output: ['apple', 'orange', 'banana', 'cherry']

Remove(): Removes the first occurrence of a specified element from the list

fruits = ['apple', 'banana', 'cherry']

fruits.remove('banana')

print(fruits)

Output: ['apple', 'cherry']

Pop(): Removes and returns the element at the specified index

fruits = ['apple', 'banana', 'cherry']

popped_fruit = fruits.pop(1)

print(fruits)

print(popped_fruit)

Output: ['apple', 'cherry']

'banana'

Index(): Returns the index of the first occurrence of a specified value.

```
fruits = ['apple', 'banana', 'cherry']

index = fruits.index('banana')

print(index)
```

Output: 1

Count(): Returns the number of occurrences of a specified element in the list

```
fruits = ['apple', 'banana', 'cherry', 'banana']

count = fruits.count('banana')

print(count)
```

Output: 2

Sort(): Sorts the elements of the list in ascending order

```
numbers = [3, 1, 4, 1, 5, 9, 2, 6]

numbers.sort()

print(numbers)
```

Output: [1, 1, 2, 3, 4, 5, 6, 9]

10. What is class and object?

Class:

A class is a blueprint or template for creating objects. It defines the data and function that will be used to modify the data in it.

Syntax:

```
class ClassName:
        # varables
        # functions
```

Example:

```
class Person:
   def __init__(self, name, age):
      self.name = name
      self.age = age
```

```
    def greet(self):
        print(f"hello, I am {self.name}")


person1 = Person("sam", 19)
person2 = Person("ram", 20)
person1.greet()
person2.greet()
```

hello, I am sam

hello, I am ram

Objects:

objects are defined as entities that represent instances of classes

Example:

```
class Person:
    def __init__(self, name, age):
        self.name = name
        self.age = age
    def greet(self):
        print("hello, I am {self.name} of age(self.age}”)
```

Output: person1 = Person("sam", 19)

hello, I am sam of age 19

11. What is method overloading and method overriding?

Method overloading refers to defining multiple methods in a class with the same name but with different parameters The purpose of method overloading is to provide multiple implementations of a method based on the type of arguments passed to it.

```
class math:
    def add(self, a, b):
        return a + b
    def add(self, a, b, c):
        return a + b + c
math_ops = math()
```

print(math_ops.add(2, 3))

Output: This will raise TypeError

print(math_ops.add(2, 3, 4))

Output: 9

Method Overriding:

Method overriding occurs when a subclass provides a specific implementation of a method that is already defined in its superclass. The purpose of method overriding is to change the functionality of an existing method

Example:

```
class Greeting:
    def greet(self):
        print("Greetings")
class GoodMorning(Greeting):
    def greet(self):
        print("Good morning")
class GoodAfternoon(Greeting):
    def greet(self):
        print("Good afternoon")
morning = GoodMorning()
afternoon = GoodAfternoon()
morning.greet()
afternoon.greet()
# Output: Good morning
# Output: Good afternoon
```

12. Write a program to remove duplicates in a string and explain the code. Example: string = ‘google’. output: [‘g’, ‘o’, ‘l’, ‘e’]

```
def remove (string):
    unique = set(string)
     result = list(unique)
    return result
```

```
string = 'google'

unique = remove(string)

print(unique)
```

Output: ['e', 'g', 'l', 'o']

1. Created a function remove where input is string and the output is result containing the string
2. Converting string to set using (set(string))
3. Converting ser to list for result
4. Returning the result

13. Explain arguments and parameters in python.

Arguments:

Arguments are defined as the actual values or variables passed to a function when calling it. These values are assigned to the corresponding parameters defined in the function

Syntax:

```
function_name(argument1, argument2, ...)
```

Parameters:

Parameters are defined as placeholders or variables defined in the function signature that specify what kind of arguments a function can accept

Syntax:

```
def function_name(parameter1, parameter2, ...):
        #code
```

Example:

```
def greet(name, message):
   print(f"Hello, {name}")
   print(message)
greet("sam", "How are you")
```

14. Write a program that calculates the highest score from a List of scores. student_scores = [78,65,89,86,55,91,64,89]. Output: The highest score in the class is: 91. Note: you are not allowed to use max() function.

```
def calculate_score(scores):
   max = float('-inf')
   for score in scores:
```

```
        if score > max:
            max = score
    return max
student_scores = [78, 65, 89, 86, 55, 91, 64, 89]
highest_score = calculate_score (student_scores)
print(f"The highest score in the class is: {highest_score}")
```

Output: The highest score in the class is: 91

15. Write a function for weather forecasting by sky colors. Blue, lite gray and dark gray indicates 'There is no rain'; 'Its about to rain' and 'Its about to heavy rain'. When the color name matches with any of the above mentioned, it should print, as color indicates accordingly. If not matches, it should print 'Its raining'.

```
def weather (color):
    if color == 'Blue':
        print("There is no rain")
    elif color == 'lite gray':
        print("It's about to rain")
    elif color == 'dark gray':
        print("It's about to heavy rain")
    else:
        print("It's raining")
weather ('Blue')
weather ('lite gray')
weather ('dark gray')
weather ('Red')
```

Output: There is no rain

It's about to rain

It's about to heavy rain

It's raining

Name: Shaik Sameer Mujahid
Batch: Data Science with AI (April 1st)

16. Create random list of values. Write a program to find the missing letters/elements in a list. Example: list1 = [1,3,5], output: missing_elements = [2,4]

```
def find _elements(lst):
    start = 1
    end = max(lst)
    reference_list = list(range(start, end + 1))
    missing_elements = [elem for elem in reference_list if elem not in lst]
    return missing_elements
import random
list1 = random.sample(range(1, 5), 7)
list1.sort()
missing_elements = find _elements (list1)
print(f"Original List: {list1}")
print(f"Missing Elements: {missing_elements}")
```

Output:

Original List: [3, 4, 6, 7, 8, 9, 10]

Missing Elements: [1, 2, 5]