Python_Assignment Part-1

Tuesday, November 8, 2022 12:31 AM

Assignment Part-1

Q1. Why do we call Python as a general purpose and high-level programming language?

Ans: Python is a general-purpose language that is broadly applicable across application domains. It is designed to be used in a range of applications like Full Stack development, Web development, Data Science, Automation, Scripting, Gaming etc..
Python is a high-level programming language, because its syntax is very clear and more readable like English language.

Q2. Why is Python called a dynamically typed language?

Ans: Python is a dynamically typed language. In python don't require data type declaration like other programming langaages ex: C, C++, JAVA etc.

Ex: name = 'JoyJoy', age = 19
In able example, not declarning any data type of specified variable of name and age. Python creates reference to variables and its values. Python determines during run-time its data types.

Q3. List some pros and cons of Python programming language?

Ans: Pros: Its free & Open Source, Easy to learn and read, Large Community, Extensive libraries, Highly Scalable,
Portable, Interpreted language.
Cons: Slow compare to other langugage due to line-by-line exection of a code, doesn't support for andriod development, consumes a lot of memory space.

Q4. In what all domains can we use Python?

Ans: Full stack development, Web Development, Automation, Scripting, Game Development, Machine Learning and Artificial Intelligence, Data Science and Data Visualization, Desktop GUI, Web Scraping Applications, Business Applications etc…

Q5. What are variable and how can we declare them?

Ans: Python Variable is like place holder/reserved memory location to store values.
Ex: name = 'JoyJoy, age = 19 # name & age are variables and its store the values.

Q6. How can we take an input from the user in Python?

Ans: using 'input' function, we can prompt key-in/type the required values.
Ex: name = input("Type your name to login : ")

```
>>> name = input("Enter your name to login: ")
Enter your name to login: JoyJoy
>>> name
'JoyJoy'
>>>
```

Q7. What is the default datatype of the value that has been taken as an input using input() function?

Ans: In Python input() function by default it returns string object.

Q8. What is type casting?

Ans: Type casting is the process of converting of one data type to another data type.
Ex:

```
>>> age = '19'
>>> print(type(age))
<class 'str'>
>>> print(type(int(age)))
<class 'int'>
>>>
```

Q9. Can we take more than one input from the user using single input() function? If yes, how? If no, why?

Ans: Using single input() function, we can take multiple input from the user by defining variable

with seperator and using methods and list compreshension.

- **Using split() method:** split() method helps in getting multiple inputs from users. It breaks the given input by the specified seperators.

```
>>> name, age = input("Enter your name and age : ").split()
Enter your name and age : JoyJoy 19
>>> print(name, age)
JoyJoy 19
>>>
```

- **Using List compreshension:** List comprehension offers a shorter syntax for creating a list from any existing iternable object.

```
>>> x, y = [int(x) for x in input("Enter two values: ").split()]
Enter two values: 1 2
>>> print(x, y)
1 2
>>>
```

Q10. What are keywords?

Ans: Python keywords are reserved words and cannot be used as variable names, function name, or for any other identifiers. Below are the list of reserved keywords.
['False', 'None', 'True', 'and', 'as', 'assert', 'async', 'await', 'break', 'class', 'continue', 'def', 'del', 'elif', 'else', 'except', 'finally', 'for', 'from', 'global', 'if', 'import', 'in', 'is', 'lambda', 'nonlocal', 'not', 'or', 'pass', 'raise', 'return', 'try', 'while', 'with', 'yield']

Q11. Can we use keywords as a variable? Support your answer with reason.

Ans: In Python, keywords are the reserved names that are built-in to Python, so a keyword cannot be used as an identifier - they have a special meaning and we cannot use them as identifier names.

Q12. What is indentation? What's the use of indentaion in Python?

Ans: Python indentation is a way of telling the Python interpreter that a series of statements belong to a single block of code. Block of code is represented using colon(:).

```
>>> name = "JoyJoy"
>>> if name == "JoyJoy":
...     print("Welcome to Python JoyJoy")
...
Welcome to Python JoyJoy
>>>
```

Q13. How can we throw some output in Python?

Ans: using print() function, we display the results

Q14. What are operators in Python?

Ans: Below are the list of operators in Python.

- Arithmetic operators
- Assignment operators
- Comparison operators
- Logical operators
- Identity operators
- Membership operators

Q15. What is difference between / and // operators?

Ans: / : Division gives decimal point result.
// : Floor Divions gives interger result.
Ex:

```
>>> 9/2
4.5
>>> 9//2
4
>>>
```

Q16. Write a code that gives following as an output.

```
iNeuroniNeuroniNeuroniNeuron
```

```
>>> org = '\"\"\" \n iNeuroniNeuroniNeuroniNeuron \n\"\"\"'
>>> print(org)
"""
 iNeuroniNeuroniNeuroniNeuron
"""
```

Q17. Write a code to take a number as an input from the user and check if the number is odd or even.

```
num = int(input("Enter the number: "))
if num % 2 == 0:
    print("Even Number")
elif num % 2 != 0:
    print("Odd Number")
```

```
~/Python3$ python3 main.py
Enter the number: 2
Even Number
~/Python3$ python3 main.py
Enter the number: 3
Odd Number
~/Python3$
```

Q18. What are boolean operator?

Ans: An expression that evaluates the results to either True = 1 or False = 0

Q19. What will the output of the following?

```
1 or 0 = Ans: 1
0 and 0 = Ans : 0
True and False and True = Ans : False
1 or 0 or 0 = 0 Ans: 1
```

```
>>> 1 or 0
1
>>> 0 and 0
0
>>> True and False and True
False
>>> 1 or 0 or 0
1
>>>
```

Q20. What are conditional statements in Python?

Ans: Conditional statement is used to controls the flow of execution depending on some condition.

Q21. What is use of 'if', 'elif' and 'else' keywords?

Ans: if-elif-else keywords/statement is used in Python for decision-making i,e the program will evaludate test expression and will execute the remaining statements only if the given test expression turns out to be true. This allows validation for multiple expressions.

Q22. Write a code to take the age of person as an input and if age >= 18 display "I can vote". If age is < 18 display "I can't vote".

```python
# Age validation for voting

person_age = int(input("Enter your age: "))

if person_age >= 18:
    print("I can vote.")
elif person_age < 18:
    print("I can't vote.")

```

```
~/Python3$ python3 age.py
Enter your age: 19
I can vote.
~/Python3$ python3 age.py
Enter your age: 18
I can vote.
~/Python3$ python3 age.py
Enter your age: 17
I can't vote.
~/Python3$
```

Q23. Write a code that displays the sum of all the even numbers from the given list.

```
numbers = [12, 75, 150, 180, 145, 525, 50]
```

```python
numbers = [12, 75, 150, 180, 145, 525, 50]

even = []

for i in numbers:
    if i % 2 == 0:
        even.append(i)

print(f"The total sum of even numbers is: {sum(even)}")
```

```
~/Python3$ python3 even.py
The total sum of even numbers is: 392
~/Python3$
```

Q24. Write a code to take 3 numbers as an input from the user and display the greatest no as output.

```python
#Write a code to take 3 numbers as an input from the user and display the greatest no as output.

user1, user2, user3 = input("Enter 3 numbers: ").split()

if (user1 >= user2) and (user1 >= user3):
    greatest = user1
elif (user2 >= user1) and (user2 >= user2):
    greatest = user2
else:
    greatest = user3

print("The greatest number is: ", greatest)

```

```
~/Python3$ python3 greatestno.py
Enter 3 numbers: 11 99 33
The greatest number is:  99
~/Python3$
```

Q25. Write a program to display only those numbers from a list that satisfy the following conditions

- The number must be divisible by five

- If the number is greater than 150, then skip it and move to the next number

- If the number is greater than 500, then stop the loop

```
numbers = [12, 75, 150, 180, 145, 525, 50]
```

```python
# divisible by 5, > 150 skip , > 500 stop Lookup

numbers = [12, 75, 150, 180, 145, 525, 50]

divisible_no = []

# Finding list of divisible no by 5
for i in numbers:
    if i > 150:
        if i > 500:
            break
        continue
    if i % 5 == 0:
        divisible_no.append(i)

print("Numbers divisible by 5 are: ", divisible_no)

```

```
~/Python3$ python3 division.py
Numbers divisible by 5 are:  [75, 150, 145]
~/Python3$
```