Name : Sufyan Ahmad

ID : 13062

Subject: Data Sciences

Time: 9 Am to 12:00 Am

BS (CS,SE)

Instructor: M.Ayub Khan

Q1. a. What are variables in python explain with help of Python coded examples?

Ans

A Python variable is a reserved memory location to store values. In other words, a variable in a python program gives data to the computer for processing.

Every value in Python has a datatype. Different data types in Python are Numbers, List, Tuple, Strings, Dictionary, etc. Variables can be declared by any name or even alphabets like a, aa, abc, etc.

Example

```
# Declare a variable and initialize it

f = 0

print f

# re-declaring the variable works

f = 'guru99'

print f
```

example

```
# Declare a variable and initialize it

f = 0

print(f)

# re-declaring the variable works

f = 'guru99'

print(f)
```

b. What are the rules to define a variable in python?

Ans

A variable can have a short name (like x and y) or a more descriptive name (age, carname, total_volume). Rules for Python variables:

A variable name must start with a letter or the underscore character

A variable name cannot start with a number

A variable name can only contain alpha-numeric characters and underscores (A-z, 0-9, and _)

Variable names are case-sensitive (age, Age and AGE are three different variables)

Q2. a. What are data types, how many data types are used in python explain with

the help of Python coded examples ?

answer

Data types are the classification or categorization of data items. Data types represent a kind of value which determines what operations can be performed on that data. Numeric, non-numeric and Boolean (true/false) data are the most used data types.

There are 5 data types in python

Numbers

Number stores numeric values. The integer, float, and complex values belong to a Python Numbers data-type.

Python creates Number objects when a number is assigned to a variable. For example

```
a = 5

print("The type of a", type(a))


b = 40.5

print("The type of b", type(b))


c = 1+3j

print("The type of c", type(c))

print(" c is a complex number", isinstance(1+3j,complex))
```

Sequence Type

The string can be defined as the sequence of characters represented in the quotation marks. In Python, we can use single, double, or triple quotes to define a string

Example - 1

```
str = "string using double quotes"
print(str)
s = '''A multiline
string'''
print(s)
```

Output:

```
string using double quotes
A multiline
string
```

Boolean

Boolean type provides two built-in values, True and False. These values are used to determine the given statement true or false. It denotes by the class bool.

example.

```python
# Python program to check the boolean type
print(type(True))
print(type(False))
print(false)
```

Output:

```
<class 'bool'>
<class 'bool'>
NameError: name 'false' is not defined
```

Set

Python Set is the unordered collection of the data type. It is iterable, mutable(can modify after creation), and has unique elements. In set, the order of the elements is undefined; it may return the changed sequence of the element.

example.

```python
# Creating Empty set
set1 = set()

set2 = {'James', 2, 3,'Python'}

#Printing Set value
print(set2)

# Adding element to the set

set2.add(10)
```

```python
print(set2)

#Removing element from the set
set2.remove(2)
print(set2)
```

Output:

```
{3, 'Python', 'James', 2}
{'Python', 'James', 3, 2, 10}
{'Python', 'James', 3, 10}
```

Dictionary

Dictionary is an unordered set of a key-value pair of items. It is like an associative array or a hash table where each key stores a specific value. Key can hold any primitive data type, whereas value is an arbitrary Python object.

example.

```python
d = {1:'Jimmy', 2:'Alex', 3:'john', 4:'mike'}

# Printing dictionary
print (d)

# Accesing value using keys
print("1st name is "+d[1])
print("2nd name is "+ d[4])

print (d.keys())
print (d.values())
```

Output:

```
1st name is Jimmy
2nd name is mike
{1: 'Jimmy', 2: 'Alex', 3: 'john', 4: 'mike'}
dict_keys([1, 2, 3, 4])

dict_values(['Jimmy', 'Alex', 'john', 'mike'])
```

b. Write a program in python in which integer value is changed in to string data type as well as explain in detail.

Answer

In Python an integer can be converted into a string using the built-in str() function. The str() function takes in any python data type and converts it into a string. But use of the str() is not the only way to do so. This type of conversion can also be done using the "%s" keyword, the .format function or using f-string function.

Syntax: *str(integer_value)*

Example

```
num = 10

# check  and print type of num variable

print(type(num))

# convert the num into string

converted_num = str(num)

# check  and print type converted_num variable

print(type(converted_num))
```

Syntax: *“%s” % integer*

```
num = 10

# check  and print type of num variable

print(type(num))

# convert the num into string and print

"% s" % num
```

Q3. Why print() and type functions are used in python explain with the help of python coded examples for each function and explain in detail as well ?

answer

Print()

The print() function prints the specified message to the screen, or other standard output device.

The message can be a string, or any other object, the object will be converted into a string before written to the screen.

Example

Print more than one object:

```
print("Hello", "how are you?")
```

Example

Print a tuple:

```
x = ("apple", "banana", "cherry")
print(x)
```

Function

In Python, a function is a group of related statements that performs a specific task.

Functions help break our program into smaller and modular chunks. As our program grows larger and larger, functions make it more organized and manageable.

Example of a function

```
def greet(name):
    """
    This function greets to
    the person passed in as
    a parameter
    """
    print("Hello, " + name + ". Good morning!")
```

Q4. How addition operator is used to update the values of variables explain with the help of Python coded example as well as explain the program?
Answer :

AnswAns4:

Addition operator is used to update the values of variables

```
 x = 6 # initialize x

print(x)

 x = x + 1 # update x

print(x)
```

If you try to update a variable that doesn't exist, you get an error because Python evaluates the expression on the right side of the assignment operator before it assigns the resulting value to the name on the left. Before you can update a variable, you have to initialize it, usually with a simple assignment. In the above example, x was initialized to 6 Or

 a = a + 1

Since these operations are so common, Python has �shortcut� operators that make typing and understanding easier

 +=

These 2 expressions produce the same result:er

a = a + 5

a += 5

Other operators that work the same way include -=, *=, /=

Precedence

Most modern programming languages interpret compound math

operations in the same way

They generally follow accepted mathematical practice in formal

expressions

In the following expression:

```
>>> 3 + 4 * 5
```

```
23
```

The rules of precedence require that a multiplication in an

expression be performed before an addition

We can change precedence by using parentheses:

```
>>> (3 + 4) * 5
35
```

The specific Python rules of precedence are here

The numpy module

The Anaconda distribution includes numpy, a general purpose

�scientific computing� collection of useful functions

numpy contains many good analytical functions as well as simpler

trig and utility functions

To use numpy, your module must import it (I�m assuming

we�re working in a text editor here):

```
import numpy
sinOneHalf = numpy.sin(.5)
myAngleInDegrees = 180
myAngleInRadians = numpy.deg2rad(myAngleInDegrees)
# and so on�
```

Here is some code that exercises trig functions and constants

in numpy

In the file I frequently use the \ character to indicate that a

python statement extends onto the next line in the text

I also use the str() command to convert numerical

information to a string

You need to do this if you mix strings and numerical

data in a single print statement

Q5. What type of errors do occur in Python, write the a program with different types of errors as well as write separate correction code in python as well as explain the errors?

Syntax errors

Python will find these kinds of errors when it tries to parse your program, and exit with an error message without running anything. Syntax errors are mistakes in the use of the Python language, and are analogous to spelling or grammar mistakes in a language like English: for example, the sentence *Would you some tea?* does not make sense – it is missing a verb

```
myfunction(x, y):
    return x + y

else:
    print("Hello!")

if mark >= 50
    print("You passed!")

if arriving:
    print("Hi!")
esle:
    print("Bye!")

if flag:
print("Flag is set!")
```

Runtime errors

If a program is syntactically correct – that is, free of syntax errors – it will be run by the Python interpreter. However, the program may exit unexpectedly during execution if it encounters a *runtime error* – a problem which was not detected when the program was parsed, but is only revealed when a particular line is executed. When a program comes to a halt because of a runtime error, we say that it has crashed.

```
for x in range(a, b):
    print("(%f, %f, %f)" % my_list[x])
```

Logical errors

Logical errors are the most difficult to fix. They occur when the program runs without crashing, but produces an incorrect result. The error is caused by a mistake in the program's logic. You won't get an error message, because no syntax or runtime error has occurred

```
product = 0
for i in range(10):
    product *= i

sum_squares = 0
for i in range(10):
    i_sq = i**2
sum_squares += i_sq

nums = 0
for num in range(10):
    num += num
```