BACHELOR OF COMPUTER APPLICATIONS

SEMESTER 5

DCA3104
PYTHON PROGRAMMING

Unit 3

Python Variables and Datatypes

Table of Contents

1. INTRODUCTION

Now that you have understood how essential Python is for developing any application, especially web-based ones, the next step is to learn different elements of the language. Till now, you have learnt that alphabets are an essential part of learning. In the next unit, you understood why you need a dedicated workspace, or a notebook to fit our analogy. The next

Datatypes and variables make up the alphabets of Python. Python essentially is a programming language and its primary use is to calculate. Python's concept of numbers is different from what we have learnt since childhood. Through datatypes, you can instruct the computer to store the required data in the desired way. Why do every data need to be stored differently? Because each data takes up a different space. A computer needs to organise its memory in such a way that no memory block, or space, goes empty or is wasted.

Take an example. A family of four need to go to a relative's place. Now for the transportation of 4 members, getting an empty bus will be called a waste of space, money, and fuel. Most of the seats on the bus will go empty. Now, you will also not get a bicycle for them. The bicycle does not have enough space for all four members. Similarly, computers categorize different kinds of data that require different memories. This way, each data type has enough memory to store its value.

For example, when one is recording someone's age, it is usually done in whole numbers. Whereas, when recording someone's height, you need a decimal number. Intuitively, one might say that a whole number requires less space for storage in comparison to a decimal number. Thus, both these numbers will be allocated different sizes of memories by the computer.

Through this chapter, you will come to understand the working of Python and its syntax. Once you understand the syntax, you will be able to write basic programs that include various operators, variables, and datatypes. Various real-world examples will also guide you through the varied applications of Python and its straightforward syntax.

1.1 Learning Objectives

After studying the chapter, you will be able to:

- *Explain the concept of variables and how to declare and assign a value to them.*
- *Describe the difference and uses of local and global variables.*
- *Describe how to delete a variable from the memory.*
- *Describe and familiarize yourself with basic categories of data types used by Python.*
- *Understand various Python keywords and their functions in a program.*

2. PYTHON VARIABLES

Different programming languages have different mechanisms for storing or calling any data. You can retrieve data directly or indirectly from the computer storage. Languages such as C and C++ allow the user to access the memory directly. Whereas, there are some programming languages such as Java, Visual Basic, and more that follow such a mechanism that you can only retrieve data through indirect methods.

STUDY NOTE

Python is an object-oriented language, thus all variables, functions, and modules are treated as objects.

All data that is stored through a programming language needs a variable. Variable is a reserved memory location. It is created when the user assigns a value to it.

As mentioned in the previous unit as well, Python is an object-oriented language. That means Python uses objects to store and designate all types of data stored in the memory. What is an object? It can be defined as an entity of the memory that contains any value or data. In Python, data is defined either as an object or through its relation with an object.

An object has three characteristics. These are as explained below.

Identity: Identity can be called the address of the memory where the object is to be stored. Each identity is unique so that all objects are defined differently.

Type: Type refers to the operations that can be performed on the data that is stored in the object. In Python, the built-in function type () is used to define the type of the object.

Value: The data contained in the object is called its value.

In Python, a variable is defined as the object that can store only one value and have a single attribute. The interpreter allocates memory to a variable depending upon its data type.

SELF-ASSESSMENT QUESTIONS - 1

1. In Python, all variables are __________.
2. Objects have three characteristics. (True/False)
3. Identity refers to the data stored in the object. (True/False)
4. C and C++ let the user access the memory directly. (True/False)
5. The data contained in the object is called _______.
   A. Value
   B. Type
   C. Store
   D. Identity

3. IDENTIFIERS RULES

Identifiers are names defined by the user to denote a variable. In the above example, we represented the variables by the names 'a' and 'b'. Thus, these are identifiers. Identifiers are the basic blocks of a Python program. We can use one identifier to represent one single variable only in a particular program.

Rules to Follow While Naming Identifiers

You cannot use any word as an identifier. Because of the presence of keywords in Python, there are certain rules that you will have to follow while naming identifiers. These are explained below.

- An identifier can have lowercase letters, uppercase letters, digits, and or an underscore. Thus, identifiers such as Book65_a, variable_1, this_is_a_variable, are acceptable.
- However, identifiers cannot begin with a digit. Thus, you cannot use 1_var or 8variable.
- Special symbols, including !, @, #, $, % cannot be used in an identifier.
- You cannot use a keyword as an identifier. We will study keywords in Python later in the unit.
- There is no particular limit to the length of the identifier. You can make it as long as you want but for sake of ease, keep it comprehensible.

STUDY NOTE

Words like get, print, return, etc., are keywords and cannot be used as identifiers.

Convention to be Followed While Naming Identifiers

These are not rules that you must follow while naming identifiers in Python. However, these conventions are followed so that a program can be understood by other users as well.

- Class names begin with an uppercase letter whereas all identifiers begin with a lowercase letter.
- Private identifiers should begin with an underscore (_).

- Using identifiers longer than one character is advised.
- One should keep in mind that Python is case-sensitive. Thus, variable and Variable will be considered differently.

SELF-ASSESSMENT QUESTIONS - 2

6. Keywords can be used as identifiers. (True or False)
7. Python_P23 is a legal identifier. (True / False)
8. ____________ cannot be used in an identifier.
9. Which of the following is a valid identifier name?
   A. !identify
   B. Identify_1
   C. Identify
   D. Keyword1
10. _________ Identifiers should begin with an underscore (_).

4. DECLARING VARIABLE AND ASSIGNING VALUE

Now that you have understood the rules that need to be followed while naming an identifier, here are the ways that you can declare a variable in Python.

Number Variable: Variables that are used to save number values are numeric variables. Python supports three types of numbers integers, floating-point numbers, and complex numbers. Take the example provided below to understand how these variables are declared.

```
num = 20
float_num = 26.90
a = 3 + 4j
```

String Variable: Strings are a sequence of characters. These are declared under single or double-quotes. Some examples are given below.

```
str_1 = ‘Example1’
str_2 = “Example2”
```

In Python, you do not have to declare the variable before you can assign a value to it. Once you assign the value, automatically the memory is reserved and the data is stored in the location. To assign a value to a variable, the equal sign (=) is used.
For example:

```
a = 10
b = 20
b-a
#Output
10
```

Here, the operand to the left side of the equal sign is called variable. The operand on the right side is the data stored in that variable.

Along with numeric values, such as integers, decimals, etc., you can also assign string values to a variable. The string is a sequence of characters whose sequence cannot be altered. They are always written inside single inverted commas.
For example:

```
a = ‘best’
b = ‘friend’
a + b
#Output
‘bestfriend’
```

4.1 Multiple Assignment

You can assign one value to several variables at the same time. For example:

```
a = b = c = 1
```

You can also assign different values to different variables in the same command. For example,

```
a, b, c = 1, 2, 3
```

In this example, the interpreter will assign the value as per the sequence in which they are defined. Thus, a will be assigned with value 1, b with 2, and c with 3.

You can use the method to swap the values of two variables as well. For example,

```
var_1, var_2 = 3, 8
var_1, var2 = var_2, var_1
```

When you get the output of the program, var_1 will have the value 8 and var_2 will have 3.

4.2 Local Variables

Local variables are defined inside a function. A function is a set of commands that are arranged in a different block than the whole program.

The local variables can only be used inside the function. They have no meaning when called outside of it.

For example,

```
def f ():
    print (a)
    a = 44
a = 33
f ()
print (a)
```

The above program will fail to show output and give an error instead. This is because the variable a is local to the function f and cannot be declared or defined outside of it. Here is an example that will work:

```
def f ():
    a = ‘Python’
    print (a)
f ()
```

4.3 Global Variables

Global variables are defined and declared outside a function. You can use and call them anywhere in the program, even inside a function.

For example,

```
def f ():
     print (a)
a = 64
f ()
```

Here, the output will be 64. As you can see, the variable 'a' is defined and declared outside the function and then called inside it. Thus, a is a global variable.

4.4 Delete A Variable

In a huge program where lots of variables are used, it is good practice to delete those which have served their purpose. This helps in maintaining the memory and free up space in RAM that can be used to perform and handle other tasks. To delete a variable, the keyword del is used in Python. the del keyword is used to delete an object in Python and since variables are objects as well, the keyword can be used in this case as well.

For example:

```
a = “hello”
del a
print (a)
```

In the output will, the interpreter will give the error that 'a' is not defined.

Activity I

Write a program that works as a calculator. That is, the user provides two values to the program and it adds, subtracts, multiplies, and divides them.

SELF-ASSESSMENT QUESTIONS - 3

11. _______ keyword is used to delete a variable.
12. __________ variables can be used only inside the function where they are defined.
13. In Python, declaring the variable before assigning value to it is mandatory. (True/False)
14. The print function is used to get a value on the output screen. (True/False)
15. Which of the following statement shows multiple assignments?
    A. a1, a2, a3 = 8
    B. a1 =8, b1 = 8
    C. 8 = a1, a2, a3
    D. 8 = a1, 8 = a2

5. PYTHON DATATYPES

The data that is stored in the memory can be categorised into various types. Python's interpreter assigns space in the memory as per the requirement for the type. Each such type is called a data type. These data types can perform different operations.

The categorisation of data types is necessary. For example, when defining someone's name, you will need a string of characters whereas, for a student's roll number, you will need a numeric value.

STUDY NOTE

To get the data type of an object, you can use the function type ().

Python has some data types built-in by default. These standard data types are also called primitive data types. The categories of data types in Python are as follows.

Category	**Data Type**
Text Type	str (string)
Numeric Type	int (integers), float (decimal), complex (complex numbers)
Sequence Type	list, tuples, range
Mapping Type	dict (dictionary)
Set Types	set, frozenset
Boolean Type	bool
Binary Type	bytes, bytearray, memoryview

Strings

Strings are text type data types. These are a series of characters that are saved in the memory in the sequence that they are provided. They are enclosed in either single or double-quotes. However, while using strings, you should keep in mind that the quote that you have used to enclose the string cannot be used inside the string. For example, "won't" is valid. However, 'won't' will give you a syntax error. Here are some examples of string.

```
str = “Hello”
str_1 = ‘World’
str_2 = “Goodbye!”
```

In Python, strings are immutable. Hence, once declared and assigned a value, they cannot be changed. When you assign a new value to it, another object is created.

When you need to save long sentences or paragraphs using string and need to divide the lines, you will have to use ‘\n’ wherever you need a line break.

For example, str = “I will learn Python. \n I want to learn to code”

When you print the string, it will look like this:
I will learn Python.
I want to learn to code.

Another thing to keep in mind is the use of triple quotes, such as “ “ “ or ‘ ‘ ‘. When using triple quotes, the new line will not escape the string and will be considered a part of it.

For example:

```
str = “ “ “I will learn to code with Python.
It is an easy programming language.
It is used to develop various web-based applications. “ “ “
```

The output of this string will be as follows:

```
#Output

I will learn to code with Python.
It is an easy programming language.
It is used to develop various web-based applications.
```

To find the length of the string, that is the number of characters used in the string, you can use the function len (). For example:

```
str = “play with me”
len (str)
```

The output will be 11.

Another thing to note is that Python, unlike other programming languages, does not support character type. There is no way to store a single character other than using string. A single character can be extracted from a string using slicing. Slicing is a method through which the index of a character in a string is used to extract it from the complete string. Here is an example to make you understand better.

STUDY NOTE

In all programming languages, the indexing begin from 0 (zero) onwards.

```
str = ‘playing’
print (str [0])
```

The output will be: p

Similarly, you can slice a substring as well. For this, you will have to use a colon in a way that is shown in the example.

```
print (str [0:3])
```

The output, in this case, will be: pla

If you do not specify the end of the index, then the range will be up to the end of the string.

For example:

```
print (str [3:])
```

This will give an output of ying

If you want to slice the string from the end, then you will have to use negative indexing. Note that the negative indexing, that is, from right to left, starts from -1.

For example:

```
print (str [-4:-2])
```

The output will be yi

5.1 Numbers

Number data types are used to store numeric values. These are categorised as an immutable data type. An immutable data type is one in which changing the value of the data type results in a newly allocated object. You can create a number data type simply by assigning a value to it. For example:

```
var = 8
```

You can change the value stored in var by reassigning another value. For example,

```
var = 8
var_1 = 6
var = var_1
```

The output of the program, when you print var will give the value 6.

In Python, the number data type is further divided into four types. These are explained below.

Integers

These are the most commonly used data types. As defined in mathematics, integers are numbers ranging from negative to positive, including zero. However, due to constrictions of memory, in Python, the integers have a range from -2,147,483,648 (-2^31) through 2,147,483,647 (2^31 - 1).

Integers are represented in decimal format, that is, with base 10. However, one can also define them using the octal and hexadecimal format. If you are using integers in octal or hexadecimal format, then ensure that you use the prefix 0 and 0x respectively. To specify that the data type that you are using is an integer, you can use the constructor function provided in the example below.

For example:

```
a = int (60)
```

Long Integers

Long integers are a subtype of integers. These are when you need a number that is greater than 231 or lesser than -231. Long integers in Python do not have a range and are limited as per the virtual memory of your computer system. The memory should be large enough to accommodate and save a long integer. While defining a long integer, you should use the suffix ‘l’ or ‘L’. These can also be defined in the form of decimal, octal, and/or hexadecimal.

For example, 100, -856, 0X35 are examples of integers.

Whereas 826353738L, 0X52462354L and -6253902628L are the examples of long integers.

Study Note: To represent long integers, you can use both ‘l’ and ‘L’ but the uppercase ‘L’ is preferred to avoid confusion of lowercase ‘l’ with ‘1’ (number 1).

Floating Point Number

Floating point numbers or as commonly known as, float, are numbers with a decimal point. They have two parts; one is the decimal point part and the other is the exponent part. The latter is optional and it is mandatory to define it while using a float. They are assigned 8 bytes of memory. The 52 bits are taken up by mantissa, 11 by the exponent, and the one remixing bit is for the negative or positive sign.

The exponent part is denoted by an uppercase or lowercase ‘e’ or ‘E’. The sign mentioned just before ‘e’ is the sign of the exponent. The absence of mention of sign, in any case, will mean that it is positive.

Some examples of floating-point real numbers are 9.0, 55.8977, -658.99, 52.66-E37, -88.223e6, etc.

To explicitly define that a variable stores a floating data type, you can use the function float ().

Complex Numbers

Complex numbers in Python are represented by two floating numbers taken in an order of a + bj. Here, a represents the real part and b is the imaginary part of the complex number. The imaginary part is always followed by a lowercase ‘j’ or uppercase ‘J’. (Usually, ‘j’ in lowercase is used). Some examples of complex numbers are 7 + 4j, -8.9 +5j, -876j, 2e4j, etc.

In some cases, if you want to print or extract only the imaginary or real part of the complex number, then you can do so by using its data attributes. This has been shown through an example provided below. A variable can be specified to store a complex number with the help of the function complex (ij)

```
complex = 44.9 + 89e4j
print (complex.real)
print (complex.imag)
#The output will be like this:
44.9
89e4
```

You can also get the conjugate of the complex number. If a complex number is defined as a + ib, then its conjugate will be a – ib. Using the attribute complex.conjugate will print the conjugate of the complex number in Python.

5.2 Sequence Type

The sequence data type is another essential data type that is commonly used in Python to store a series of data together. These are further divided into the following types.

Lists:

Lists work similarly as arrays in C. The one difference is that in lists, you can store values of different data types together. The items in a list are enclosed inside a square bracket []. Each value is separated with a comma. The values saved in a list are indexed and ordered. Thus, when you print a list, the order of the values stored in it does not change.

A list is alterable. You can add items to the list, remove them, delete the list, etc., once it has been created. You can also store the same values in a list. As they have a different index, the user will be able to differ between the same data.

For example,

```
list_1 = [23, 56, 766, “passed”]
```

As mentioned, the indexing in the list starts from 0 as well. Thus, in the above example, element 23 has an index of 0, 56 has 1, and so on.

Here are the methods that you can implement in the list. These are built-in and can be used using the functions that are listed in the table below.

TABLE 1: METHODS FOR LISTS

Method	Use
insert ()	To add an element to the list at a specified position or index
extend ()	To add the elements of a list to another list
count ()	To count the number of elements, present in the list.
append ()	To add an element at the end of the list
clear ()	To remove all the elements of the list.
pop ()	To remove an element from the list at a specific index.
reverse ()	To reverse the order of the list
sort ()	To sort the list

Tuples

Tuple functions are similar to lists but with one difference. They are immutable. Values stored in a list can be changed and altered even after the list has been created. Such a feature may become an issue in some programs. Thus, in such cases, tuples can be used. Tuples are enclosed in parentheses with each element separated by a comma. They are ordered as well and allow repeating a single value.

For example,

```
tuple1 = (3, 6, 9)
```

To join two tuples together, you can use the + operator. It is shown in the example below.

```
tuple1 = (3, 6, 9)
tuple2 = (“a”, “b”, “c”)
tuple3 = tuple1 + tuple2
```

To add items to the end of the tuple, you can use append () similarly as in a list. Other functions and methods such as remove (), del, etc., work similarly in both tuples and lists.

5.3 Boolean

Boolean type is used to store one of the two values: true or false. Bool is used to test whether the result of an expression is true or false. This can be used when you are comparing two values. If you run such a conditional statement as shown in the example, Python will give True or False as a result.

For example, print (7 ==8)

This statement will result in an output showing False.

The bool () function is used to verify whether a value is true or false. Here, being a true value means that it is acceptable by Python.

For example,

```
print (bool (89))
```

This will return the value as True in the output window.

5.4 Set

Just like lists and tuples, sets are used to store a collection of data. Through set, you can store multiple items in a single variable. The collection is unordered and unindexed. That is, the data stored in a set is not saved in a sequence. Hence, you cannot surely say in which order the sets will appear or get printed. They are immutable as well. That means, once a collection of data is stored in a set, it cannot be changed. Also, no two sets can have the same value.

You cannot completely change the values stored in a set but you can add new data. And, example of a set is provided below.

```
set_1 = {“pencil”, “eraser”, “ruler”}
```

Here, note that all the data in the set is enclosed in curly braces {}.

To add new items to a set, you can use the function add (). The method is shown in the example below.

```
set_1 = {“pencil”, “eraser”, “ruler”}
set_1.add (“pen”)
```

When you print the set, you will find four values in it, namely, pencil, eraser, ruler, and pen arranged in random order.

You can add the values of one set to another using the function update (). For example,

```
set_1 = {“pencil”, “eraser”, “ruler”}
set_2 = {“protractor”, “divider”}
set_1.update (set_2)

#The output will be: pencil, eraser, ruler, protractor, and divider.
```

The update () function can be used to add any iterable object such as lists, tuples, etc.

The function remove () or discard () can be used to remove a value from the set. For example,

```
set_1 = {“pencil”, “eraser”, “ruler”}
set_1.remove (“ruler”)
```

This way, you can remove the value “ruler” from the set.

To empty the complete set, you can use the method clear ().

For example, set_1.clear () will remove all the values from the set.

The del keyword will delete the set completely.

For example, del set_1

STUDY NOTE

Dictionaries were not ordered in versions Python 3.6 and earlier. This was updated from Python 3.7.

5.5 Dictionary

Imagine you want to look up a number in the logs of your mobile phone. You do not remember the number, and surely you would not remember the index at which it is saved. How will you search for it then? Using the name of the person whose number you are searching for. The name of the person is called key whereas the phone number will be the value.

In Python, you can save the pair of key: value together in the form of dictionary data type. In a dictionary, one key is associated with only one value. It is enclosed in curly braces {} and each pair of key: value is separated by a comma.

For example:

```
dict_1 = {
“name”: “Aakash”,
“birthyear”: 1998,
“rollnumber”: 891
}
```

The values in a dictionary are stored in order. No duplication of the value of a key is allowed in a dictionary.

Here are the functions and methods that you can use to create, modify, and add to a dictionary.

TABLE 2: METHODS FOR DICTIONARY

Method	Use
clear ()	To remove all the elements of a dictionary
copy ()	To return a copy of the dictionary
get ()	To return the value of a specific key
keys ()	To get a list of the keys included in a dictionary
update ()	To update the dictionary by adding key:value pairs
values ()	To get a list of values present in the dictionary
pop ()	To remove the element of the specified key

SELF-ASSESSMENT QUESTIONS – 4

16. Tuples are ________ data types and thus, cannot be changed after they are created.
17. ______________ is the data type in which value is stored with a key instead of an index.
18. ____________ data types store two values, true and false.
19. To signify that a number is a long integer, it is suffixed by __________.
20. You can remove a value from a set with method _____________

6. PYTHON KEYWORDS

Every programming language has a definite syntax and rules that need to be followed while witting a program. Python reserves some of the names that cannot be used as programmer-defined identifiers. These predefined identifiers are called keywords. Here is a list of keywords used in Python.

- and
- assert
- break
- while
- continue
- class
- del
- elif
- else
- def
- exec
- except
- for
- from
- global
- finally
- if
- is
- lambda
- not
- in
- pass
- or
- raise
- return

- try
- print
- yield

Till now, we have learnt to use global (to define global variables) and print (to print a variable or value on the output screen).

Activity II

Write a program that combines three or more sequence data types. Try making different combinations such as lists with tuples, tuples with dictionaries, etc. Note if you get any syntax error.

SELF-ASSESSMENT QUESTIONS – 5

21. Keywords are predefined identifiers. (True/False)

22. Which is a valid identifier:
    A. you_and
    B. return
    C. @gogo
    D. print(3)

23. ____________ keyword is used to define a class.

24. _____________is a keyword used for conditional statement.

25. ___________ to remove all the elements of a dictionary

7. SUMMARY

- An entity that contains data is called an object. Python uses objects to define any data. An object has three components: an identity, a type, and a value.
- Objects whose value can be changed after creation are called mutable objects. Those whose values cannot be altered once they are created are called immutable.
- Names given to variables in a programming language are called identifiers.
- Variables reserve memory locations to save data.
- = sign is used to assign value to a variable. In Python, a variable can be assigned a value without declaring it before.
- Python has various data types divided among categories: number type, sequence type, text type, mapping type, Boolean type, set type, and binary type.
- The numeric type has four kinds of data types: integers, long integers, floating type real numbers, and complex numbers.
- Sequence type has two data types, lists and tuples. These are ordered collections of data. They have indices. Lists are mutable data types whereas tuples are immutable.
- A string is a collection of characters. It is defined by enclosing it in single or double-quotes.
- Sets are an unordered collection of data that does not have any index.
- Dictionary is used to store data in the pair of key: value.
- The words that cannot be used as programmer-defined identifiers are called keywords.

8. GLOSSARY

Identifier: It is the name used to identify a variable, function, or class.

Variables: The names that are used to locate the memory of a value.

Global Variables: A variable that is declared outside of a function and can be accessed anywhere in the program.

Data types: The classification of values stored to perform any task in a Python program.

Float: Numbers that are represented with a decimal point or have a fraction part.

Lists: It is an ordered sequence of elements that is mutable.

Tuples: They are sequence data type to store a collection of data in order and are immutable.

Boolean: Boolean is a data type that is used to store two values: True or False.

Set: Data type that is used to store multiple values in a single variable.

Dictionary: It is an unordered collection of data arranged in the form of key: value pair.

Keywords: Words reserved in Python that are used to define the syntax and structure of Python.

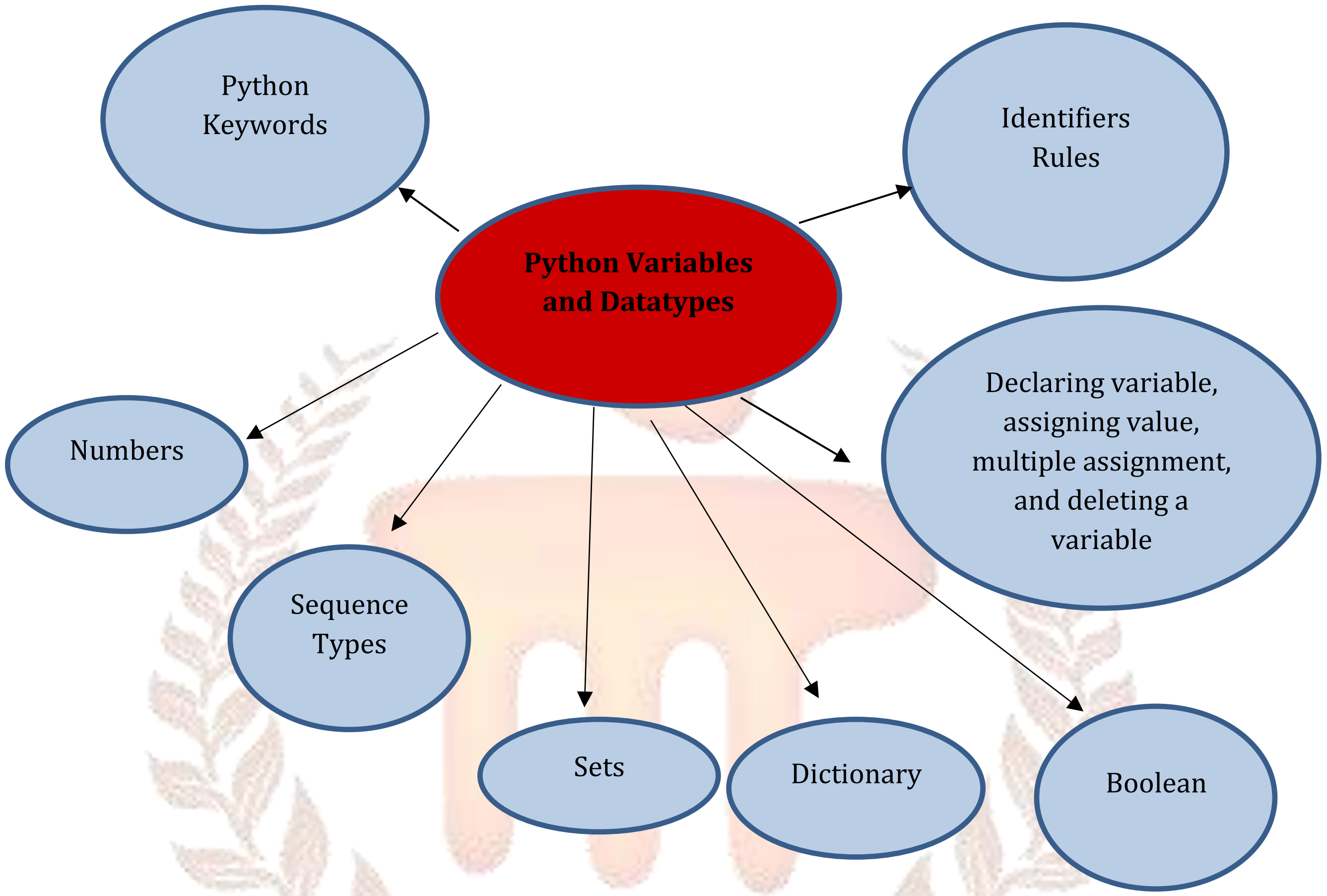
Python Keywords
Identifiers Rules
Python Variables and Datatypes
Numbers
Declaring variable, assigning value, multiple assignment, and deleting a variable
Sequence Types
Sets
Dictionary
Boolean

Fig 1: Conceptual Map

9. CASE STUDY

College database creation using python

There is a college admission in an engineering college and thousands of new students have joined in each course as per the mark obtained in the 12th standard.

These students belonged to different schools with different subject category and have joined different courses as per the percentage scored. The college has planned to create a database using Python to segregate the students as per their name, enrolment number, school, city, and course.

With the help of the python code, they have created a database with all the student information's from their personal address to academic details.

So, they're using information forms of lists and tuples for his or her simple management and understanding. Once the info is made, the school can unleash identity cards for these a hundred students.

With the help of python they are using data types of lists for their ease of management and understanding. Once the database is created and finalised, the college will issue the identity cards for these newly joined students.

***Source**: bestpy.com*

Questions

1) Discuss what changes could the college make in the database to decrease the memory space required to save the data of each child.
2) Can you recommend using data types other than lists and tuples that can be used by the college? Give reasons for your answer.

10. TERMINAL QUESTIONS

SHORT ANSWER QUESTIONS

Q1. What is a variable?

Q2. Give any two examples of valid variable names.

Q3. Write a program that gives the sum of two floats.

Q4. How will you represent a long integer in Python?

Q5. Explain strings with example.

LONG ANSWER QUESTIONS

Q1. What are the four types of numeric data types in Python?

Q2. Write a Python program to calculate the length of a string.

Q3. Write a Python program, that returns the sum of all items in a list.

Q4. Write a Python program to get the largest number from a list.

11. ANSWERS

SELF ASSESSMENT QUESTIONS

1. Objects
2. True
3. False
4. True
5. A. Value
6. False
7. True
8. Special characters
9. B, C, D. identify_1
10. Private
11. del
12. Local
13. False
14. True
15. A. a1, a2, a3 = 8
16. Immutable
17. Dictionary
18. Boolean
19. L
20. Remove or discard
21. True
22. A. you_and
23. class
24. if
25. clear ()

TERMINAL QUESTIONS

SHORT ANSWER QUESTIONS

Answer 1: A variable is the name that the user gives so that they can reserve the space in the computer memory to save a value. Variable name can include any lowercase, uppercase letters along with digits and underscore.

Answer 2: Variables names should not be keywords and should not include special characters. Thus, var_1, variable2, etc., can be valid variable names.

Answer 3:

```
float1 = 8.9
float_2 = 6.9
float_3 = float1 + float_2
print (float_3)
```

Answer 4: To represent a long integer in Python, the number is followed by 'l' or 'L'. Thus, 882722L is a long integer.

Answer 5: Strings are text type data types. These are a series of characters that are saved in the memory in the sequence that they are provided. They are enclosed in either single or double-quotes. However, while using strings, you should keep in mind that the quote that you have used to enclose the string cannot be used inside the string. For example, "won't" is valid. However, 'won't' will give you a syntax error. Here are some examples of string.

```
str = "Hello"
str_1 = 'World'
str_2 = "Goodbye!"
```

LONG ANSWER QUESTIONS

Answer 1: In Python, the numeric data types are of the following four types:

Integers: Negative to positive numbers including 0.

Long Integers: These are same as integers but do not have a range and depend upon the size of the computer memory.

Floating point: Numbers with decimal points.

Complex Numbers: Numbers that have a real and an imaginary part.

Answer 2:

```
def string_length(str1):
   count = 0
   for char in str1:
      count += 1
   return count
print(string_length('AString'))
```

Answer 3:

```
def sum_list(items):
   total = 0
   for num in items:
      total += num
   return total
numbers = [1, 2, 3, 4, 5]
print(sum_list(numbers))
```

Answer 4:

```
def max_num_in_list( list ):
   max = list[ 0 ]
   for a in list:
      if a > max:
         max = a
   return max
print(max_num_in_list([1, 2, -8, 0]))
```

12. SUGGESTED BOOKS AND E-REFERENCES

BOOKS:

- Eric Matthes (2016), Python Crash Course: A Hands-On, Project-Based Introduction to Programming.
- John M. Zelle (2009), Python Programming: An Introduction to Computer Science (Preliminary Second Edition).
- Mark Lutz (2011), Python Programming: A Powerful Object-Oriented Programming (Fourth Edition).
- Sebastian Raschka (2017), Python Machine Learning - Machine Learning and Deep Learning with Python (Edition 2).

E-REFERENCES:

- Python Programming Certification Training Course, last viewed on March 23, 2021, <https://www.edureka.co/python-programming-certification-training>
- Python Tutorials and Sample Programs, last viewed on March 23, 2021, <https://www.w3schools.com/python/>
- History of Python, last viewed on March 23, 2021, <https://en.wikipedia.org/wiki/History_of_Python>