TOPIC 3: PYTHON 3 BASICS

Introduction.

Python was designed for readability, and has some similarities to the English language with influence from mathematics. Python uses new lines to complete a command, as opposed to other programming languages which often use semicolons or parentheses.

Python relies on indentation, using whitespace, to define scope; such as the scope of loops, functions and classes. Other programming languages often use curly-brackets for this purpose.

Objectives

Objectives by the end of this topic you should be able to:

- Explain statements and indentation
- Describe different types of comments
- Explain various python reserved keywords

Learning activities

Learning Activity 3.1: Reading

Read further on reserved keywords.

Learning Activity 3.2: Journal

Summarize various python reserved keywords from python 3 documentation

Learning Activity 3.3: Discussion

Discuss how PEP 8 coding convention, a set of recommendation, can help you write your Python code more readable

Assessment

Topic resources

1. The Python Tutorial¶. (n.d.). Retrieved from https://docs.python.org/3/tutorial/index.html
2. Mueller, J. P. (n.d.). *Beginning Programming with Python For Dummies*. S.l.: For Dummies.
3. (n.d.). Python 3.7.4 documentation. Retrieved from https://docs.python.org/3
4. (n.d.). Git Handbook. Retrieved from https://guides.github.com/introduction/git-handbook/
5. Shaw, Z. (2017). *Learn Python 3 the hard way: a very simple introduction to the terrifyingly beautiful world of computers and code*. Boston: Addison-Wesley.
6. Bader, D. (2018). *Python tricks: the book*. Vancouver, BC: Dan Bader.
7. Downey, A. B. (2015). *Think Python*. Sebastopol: OReilly.
8. Ramalho, L. (2016). *Fluent Python:*Beijing: OReilly.

URL Links

https://www.geeksforgeeks.org/python-set-4-dictionary-keywords-python/?ref=rp
https://www.geeksforgeeks.org/keywords-python-set-2/?ref=rp
https://www.geeksforgeeks.org/keywords-python-set-1/

TOPIC: PYTHON 3 BASICS (Statement, Indentation and Comment in Python)

Statements

Instructions written in the source code for execution are called statements. There are different types of statements in the Python programming language like Assignment statement, Conditional statement, Looping statements etc. These all help the user to get the required output. For example, n = 50 is an assignment statement.

Multi-Line Statements: Statements in Python can be extended to one or more lines using parentheses (), braces {}, square brackets [], semi-colon (;), continuation character slash (\). When the programmer needs to do long calculations and cannot fit his statements into one line, one can make use of these characters.

Example :

```
Declared using Continuation Character (\):
s = 1 + 2 + 3 + \
    4 + 5 + 6 + \
    7 + 8 + 9


Declared using parentheses () :
n = (1 * 2 * 3 + 7 + 8 + 9)


Declared using square brackets [] :
footballer = ['MESSI',
        'NEYMAR',
        'SUAREZ']


Declared using braces {} :
```

```
x = {1 + 2 + 3 + 4 + 5 + 6 +
     7 + 8 + 9}
```

Declared using semicolons(;) :

```
flag = 2; ropes = 3; pole = 4
```

Indentation

A block is a combination of all these statements. Block can be regarded as the grouping of statements for a specific purpose. Most of the programming languages like C, C++, Java use braces { } to define a block of code. One of the distinctive features of Python is its use of indentation to highlight the blocks of code. Whitespace is used for indentation in Python. All statements with the same distance to the right belong to the same block of code. If a block has to be more deeply nested, it is simply indented further to the right. You can understand it better by looking at the following lines of code:

```
# Python program showing
# indentation

site = 'gfg'

if site == 'gfg':
    print('Logging on to geeksforgeeks...')
else:
    print('retype the URL.')
print('All set !')
```

Output:

```
Logging on to geeksforgeeks...

All set !
```

The lines print(‘Logging on to geeksforgeeks…’) and print(‘retype the URL.’) are two separate code blocks. The two blocks of code in our example if-statement are both indented four spaces. The final print(‘All set!’) is not indented, and so it does not belong to the else-block.

```
j = 1
while(j<= 5):
    print(j)
    j = j + 1
```

Output:

```
1

2

3

4

5
```

To indicate a block of code in Python, you must indent each line of the block by the same whitespace. The two lines of code in the while loop are both indented four spaces. It is required for indicating what block of code a statement belongs to. For example, j=1 and while(j<=5): is not indented, and so it is not within while block. So, Python code structures by indentation.

Comments

Python developers often make use of the comment system as, without the use of it, things can get real confusing, real fast. Comments are the useful information that the developers provide to make the reader understand the source code. It explains the logic or a part of it used in the code. Comments are usually helpful to someone maintaining or enhancing your code when you are no longer around to answer questions about it. These are often cited as a useful programming convention that does not take part in the output of the program but improves the readability of the whole program. There are two types of comment in Python:

Single line comments : Python single line comment starts with hashtag symbol with no white spaces (#) and lasts till the end of the line. If the comment exceeds one line then put a hashtag on the next line and continue the comment. Python's single line comments are proved useful for supplying short explanations for variables, function declarations, and expressions. See the following code snippet demonstrating single line comment:

```
1 # This is a comment
2 # Print "Python Programming !" to console
3 print("Python Programming")
4
```

```
1 a, b = 1, 3 # Declaring two integers
2 sum = a + b # adding two integers
3 print(sum) # displaying the output
4
```

Multi-line string as comment : Python multi-line comment is a piece of text enclosed in a delimiter ("""") on each end of the comment. Again there should be no white space between delimiter (""""). They are useful when the comment text does not fit into one line; therefore needs to span across lines. Multi-line comments or paragraphs serve as documentation for others reading your code. See the following code snippet demonstrating multi-line comment.

```
"""
This would be a multiline comment in Python that
spans several lines and describes geeksforgeeks.
A Computer Science portal for geeks. It contains
well written, well thought
and well-explained computer science
and programming articles,
quizzes and more.
…
"""
print("Python Multiple Line Commnents")

```

```
'''This topic on Python Multiple Line Commnents gives you a
perfect example of
multi-line comments'''

print("Python Multiple Line Commnents")

```

Keywords in Python

True : This keyword is used to represent a boolean true. If a statement is true, "True" is printed.

False : This keyword is used to represent a boolean false. If a statement is false, "False" is printed.
True and False in python are same as 1 and 0.

None : This is a special constant used to **denote a null value or a void**. **Its important to remember, 0, any empty container(e.g empty list) do not compute to None**.
It is an object of its own datatype – NoneType. It is not possible to create multiple None objects and can assign it to variables.

and : This a logical operator in python. "and" Return the first false value .if not found return last**.**
The truth table for "and" is depicted below.

Truth Table for and

A	B	A and B
True	True	True
True	False	False
False	True	False
False	False	False

3 and 0 **returns 0**
3 and 10 **returns 10**
10 or 20 or 30 or 10 or 70 returns **70**

The above statements might be a bit confusing to a programmer coming from a language like **C** where the logical operators always return boolean values(0 or 1). Following lines are straight from the python docs explaining this:

The expression x and y first evaluates x; if x is false, its value is returned; otherwise, y is evaluated and the resulting value is returned.

The expression x or y first evaluates x; if x is true, its value is returned; otherwise, y is evaluated and the resulting value is returned.

Note that neither and nor or restrict the value and type they return to False and True, but rather return the last evaluated argument. This is sometimes useful, e.g., if s is a string that should be replaced by a default value if it is empty, the expression s or 'foo' yields the desired value. Because not has to create a new value, it returns a boolean value regardless of the type of its argument (for example, not 'foo' produces False rather than ".)

or : This a logical operator in python. "or**"** Return the first True value if not found return last**.** The truth table for "or" is depicted below.

Truth Table for or

A	B	A or B
True	True	True
True	False	True
False	True	True
False	False	False

3 or 0 **returns 3**
3 or 10 **returns 3**
0 or 0 or 3 or 10 or 0 returns **3**

not : This logical operator **inverts the truth value**. The truth table for "not" is depicted below.

Truth table for not

A	not A
True	False
False	True

```
# Python code to demonstrate
# True, False, None, and, or , not

# showing that None is not equal to 0
# prints False as its false.
print (None == 0)

# showing objective of None
# two None value equated to None
# here x and y both are null
# hence true
x = None
y = None
print (x == y)

# showing logical operation
# or (returns True)
print (True or False)

# showing logical operation
# and (returns False)
print (False and True)

# showing logical operation
# not (returns False)
print (not True)
```

Output:

```
False

True

True

False

False
```

assert : This function is used for **debugging purposes**. Usually used to check the correctness of code. If a statement evaluated to true, nothing happens, but when it is false, "**AssertionError**" is raised . One can also **print a message with the error, separated by a comma**.

break : "break" is used to control the flow of loop. The statement is used to **break out of loop and passes the control to the statement following immediately after loop.**

continue : "continue" is also used to control the flow of code. The keyword **skips the current iteration of the loop**, but **does not end the loop**.
Illustrations of break and continue keywords can be seen in the link below.

Loops and Control Statements (continue, break and pass) in Python

class : This keyword is used to **declare user defined classes**. For more info. click here.

def : This keyword is used to **declare user defined functions**. For more info. click here.

if : It is a control statement for decision making. **Truth expression forces control to go in "if" statement block.**

else : It is a control statement for decision making. **False expression forces control to go in "else" statement block.**

elif : It is a control statement for decision making. It is short for "**else if**"
if, else and elif conditional statements are explained in detail here .

del : del is used to **delete a reference to an object**. Any variable or list value can be deleted using del

Revision questions

1. What are reserved keywords in python?
2. Describe two ways of writing comments in python
3. Why is indentation important in python
4. Define a statement