Elisha Vernee Hart

Revise The Previous Topics

The fundamental question pertaining to computer science is, “What can be computed and/or calculated?”

Machine language is a binary language which consists of 0 and 1.

A, Ascii Value is 65, and the binary number is 1000001.

Syntax is a set of rules that defines which sequence of characters and symbols constitute a well-formed string.

Semantics is a well-formed string that have a meaning and saying what exactly is the meaning?

There are ways things can be syntactically correct and semantically correct.
There is an example of using, “noun, verb, noun” within a correct sentence.
An example is: Car is vehicle. Car is the noun, the word “is” is the helping verb, and vehicle is a noun. This would be syntactically correct because a car is a vehicle in actuality.
Another example is: Car is hotel. The car is the noun, the word “is” is the helping verb, and hotel is a noun. This would be semantically correct because a car is a noun and a hotel is a noun. The statement does not make any sense due to a car not being a hotel, therefore this would be considered semantically correct.

Python can be worked in an Interactive Mode and Scripting Mode.

A token is a single or smallest element of a programming language. It is also known as lexical unit. The types of tokens are keywords, identifiers, literals, operators, and punctuators.

Keywords are reserved words that have special meanings.

Python 3.6 or lower have 33 keywords. Python 3.7, 3.8, and 3.10 have 35 keywords. Python 3.9 have 36 keywords.

Keywords includes words such as “True” and “False.” The letter “T” in True, must be capitalized and the letter “F” in False must be capitalized. There are keywords called None, if, else, elif while, for, break, continue, with, class, pass, and, or, not, is, in, try, except, finally, raise, assert, from, as, import, def, return, yield, global, nonlocal, lambda, del, async, await.

Identifier is a name that is given to a function, variable, object, class or module. The rules of identifiers is that it should be a combination of a letter, and those letters can be a-z and A-Z. Digits which are digits between 0-9, and an underscore or "_". Identifiers should not start with a digit.

The topic of Identifiers was continued. An identifier should not be a keyword. It can be any length, and it is case sensitive.

Literal is a constant. This includes Boolean Literal which is True and/or False. Numeric Literal which includes int, float, and complex.

Integers represents "int", this includes examples such as -34, 23, 10 and so on.

Floating point numbers and the word "float" is used in the compiler. This includes numbers such as -343434.34343, 3434.44545, and so forth.

There is Special Literal which is None.

There is String Literal which would be anything within quotation marks or quotes known as a "string." Like 'Hello', "Hi", and so on.

Literal collections is things such as list, tuple, sets, dictionary, and so on.

The Punctuators include symbols such as , : ; ( ) @ .

Datatype is the classification that identifies which type of value a variable has.

There are different built in datatypes used in Python. These are categorized into several classes such as Boolean, Numeric which includes int, float, and complex. Sequences which includes str, list, tuple, bytes, and bytearray. The sets includes set and frozenset. The Mappings includes dict. For example, in the compiler this was typed,

```
a=(2,3,4, 'Hi', [234234],)
print(type(a))
```

Output showed <class tuple'>

Another example is:

```
a={2,3,4, 'Hi',}
print(type(a))
```

Output showed <class set'>

The fundamental or basic datatype includes int, float, bool, str, and complex.

Operators are symbols that carryout computation. These operators include arithmetic operators such as +, -, /, %, //, *, **

The operator symbol "+" will perform addition if both or one of the operand is Numeric or Boolean. For an example:
4+True
Output: 5

Due to True being 1. Therefore 4+1=5.

The operator symbol "+" will perform if both operands are sequenced or are the same type. For example, 'hello' + [23] will show an error.

The operator symbol "*" will perform multiplication if both or one of the operand is Numeric or Boolean. An example is 2*3 and the output shows 6.

The operator symbol "*" will perform repetition if one of the operand is sequenced and the other is "int." For example, 2 * 'Python' the output will show 'PythonPython'

The operator symbol "/" performs division. It always returns the float result. For example, 12/34 and the output will show 0.3529411764705886.

Another example is 12/3 and the output showing 4.0.

The operator symbol "//" is called floor division. If both the operands are int the ans or answer will be int. If one of the operand is a float the output will be float. For example, 34//35 the output will show 0.

The power of operator is "**" An example would be 2**3 and the output would be 8 since 2 to the 3rd power is equal to 8.

Relational Operator or Conditional Operator which is >, >=, <, <=, ==, !=. For example, 3 >= 3 and the output will show True. The output will always be a Boolean value.

Logical Operators include and, or, not. When using the word “and”, if one of the operand is False, the output will be False. For example, 1 and False will be False.

If both the operands are False values, like 0, 0.0, False, then the output will be the first operand.

A range is a function which generates a range of integers.
The range(End), will generate integers for 0 to End-1.
An example in the compiler,
range(10)
Output: range(0,10)

Another example of a range is:

```
for i in range (1):
  print (i)
```

Output: 0

Another example:

```
for i in range (15):
  print i
```

Output:0
1
2
3
4
5
6
7
8
9
10
11
12
13
14

An example to print range going horizontally

```
for i in range (15):
  print(i, end = ‘ ‘)
```

Output: 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14

For an range, End should not be 0 or less than 0, otherwise nothing will show.
Another example is:

```
for i in range (-10, 1):
  print (i, end = ‘ ‘)
```

Output: -10, -9. -8. -7, -6, -5, -4, -3, -2, -1, 0

Range that has Begin, End, and Step. If there is -10, 11, 5, the number 5 represents to skip every 5 numbers. The end is 11 show it will show the number 10 in the compiler. It is showing 0-10, which is 11.
An example is:

```
for i in range (-10, 11, 5):
   print(i, end = ‘ ‘)
```

Output: -10 -5 0 5 10

Another example is:

```
for i in range (-10, 11, 2):
   print (i, end = ‘ ‘)
```

Output: -10 -8 -6 -4 -2 0 2 4 6 8 10

Example:

```
for i in range (10, 0, -10):
  print (i, end = ‘ ‘)
```

Output: 10

What is indexing? It is the process of accessing elements. Types of index:

1.) Positive Index is being used to access elements from left to right. Starts from 0, and ends at n-1, n is the total number of elements.
2.) Negative Index is being used to access elements right to left. Starts from -1, end at -n, n is total number of elements.

Indexing is applicable only on sequence.

A slicing operator returns sub-sequence. The types are seq[Begin: End] seq [Begin:End:Step]