SNS COLLEGE OF ENGINEERING
Kurumbapalayam (Po), Coimbatore – 641 107
An Autonomous Institution
Accredited by NBA – AICTE and Accredited by NAAC – UGC with 'A' Grade
Approved by AICTE, New Delhi & Affiliated to Anna University, Chennai

DEPARTMENT OF CSE (IoT & CYBER SECURITY INCLUDING BLOCKCHAIN TECHNOLOGY)

19IT103 – COMPUTATIONAL THINKING AND PYTHON PROGRAMMING

- **A readable, dynamic, pleasant, flexible, fast and powerful language**

UNIT II DATA TYPES, EXPRESSIONS, STATEMENTS

- Python interpreter and interactive mode, debugging; values and types: int, float, boolean, string , and list; variables, expressions, statements, tuple assignment, precedence of operators, comments; Illustrative programs: exchange the values of two variables, circulate the values of n variables, distance between two points.

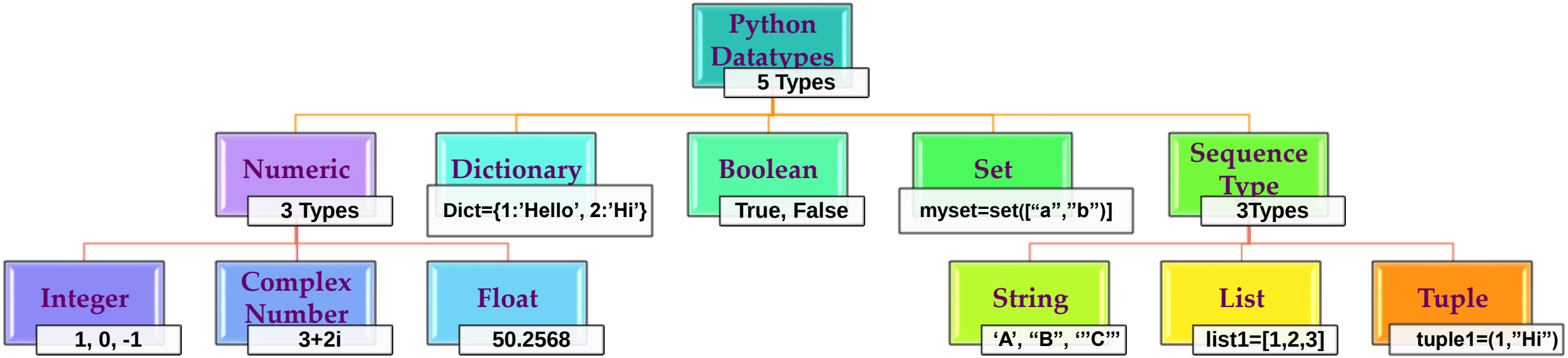
Recap

- **Debugging:** Programming errors are called bugs and the process of tracking them down is called debugging.
  - Synatx Errors
  - Run time Errors
  - Semantic Errors

Values and Types

- A value is one of the basic things a program works with.
- These values belong to different types.
- **Data types** are the classification or categorization of data items.
- It represents the **kind of value** that tells what operations can be performed on a particular data.
- Everything is an **object** in Python programming, **data types** are actually **classes** and **variables** are **instance (object)** of these classes.

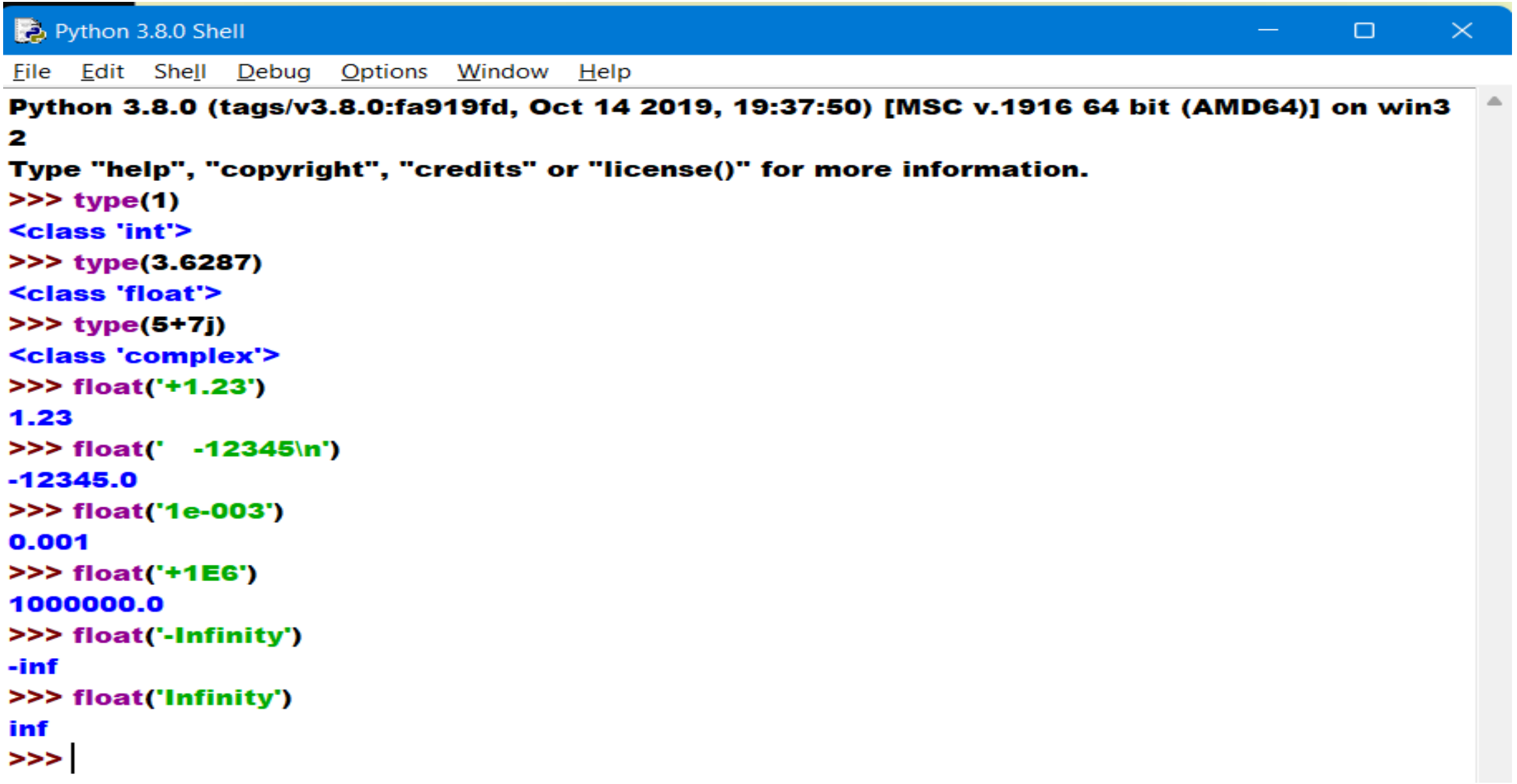
Python Datatypes



Numeric Data Type

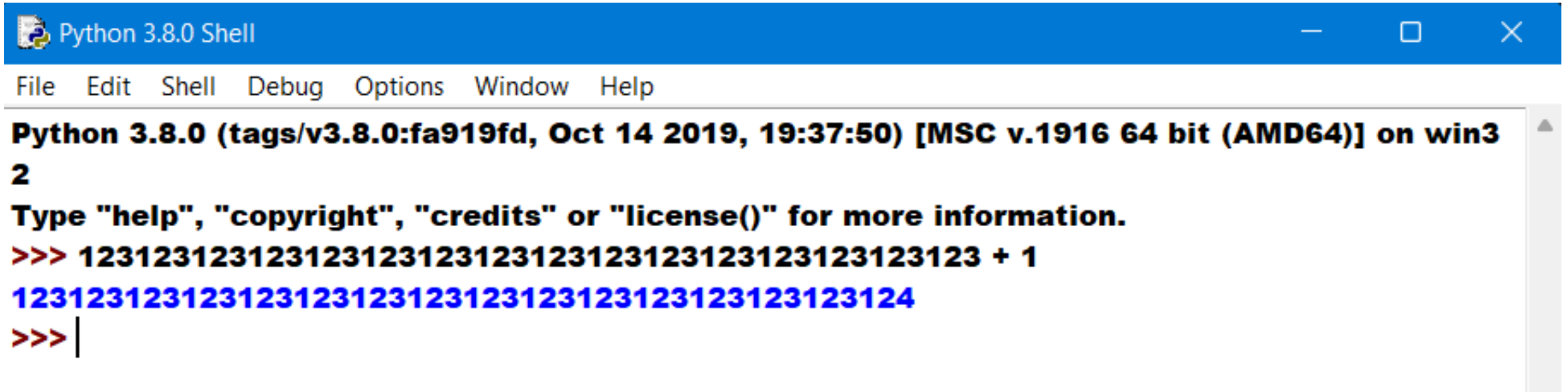
- Numeric data type represent the data which has numeric value.
- Numeric value can be integer, floating number or even complex numbers.
- **Integers** – This value is represented by **int class**. It contains **positive or negative whole numbers** (without fraction or decimal).
- **Float** – This value is represented by **float class**. It is a **real number with floating point** representation. It is specified by a decimal point.
- **Complex Numbers** – Complex number is represented by **complex class**. It is specified as **(real part) + (imaginary part)j**. For example – 2+3j

Numeric Data Type



Numeric Data Type - Integer

- In Python 3, there is effectively no limit to how long an integer value can be.
- It is constrained by the amount of memory of your system.



Numeric Data Type - Integer

- The following strings can be prepended to an integer value to indicate a base other than 10:

Prefix	Interpretation	Base
0b (zero + lowercase letter 'b') 0B (zero + uppercase letter 'B')	Binary	2 (1, 0)
0o (zero + lowercase letter 'o') 0O (zero + uppercase letter 'O')	Octal	8 (0, 1, 2, 3, 4, 5, 6, 7)
0x (zero + lowercase letter 'x') 0X (zero + uppercase letter 'X')	Hexadecimal	16 (0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F)

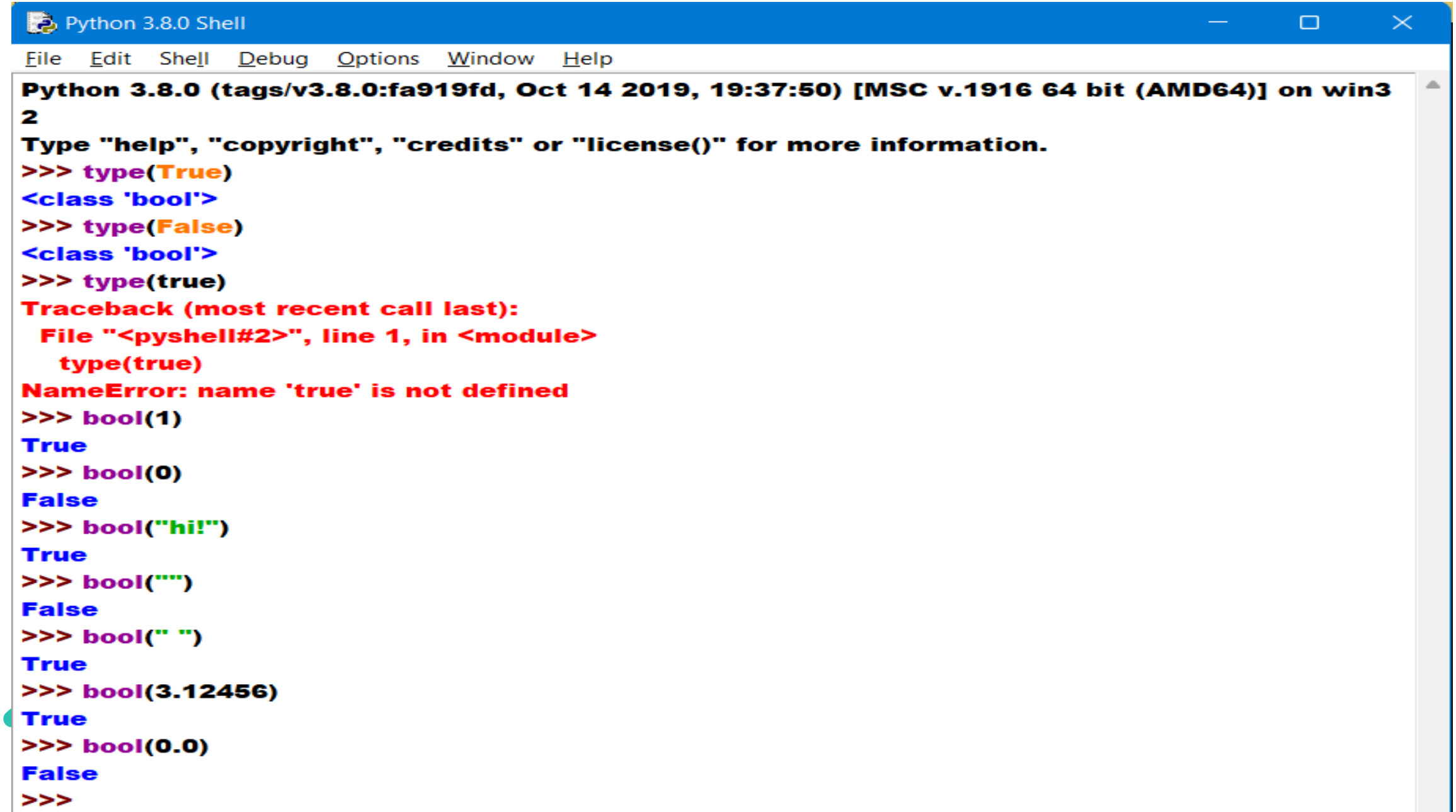
Numeric Data Type - Integer

```
>>> print(0o10)
8
>>> print(0x10)
16
>>> print(0b10)
2
>>> type(0o10)
<class 'int'>
>>> type(0o10)
<class 'int'>
>>> type(0x10)
<class 'int'>
>>> 0b10
2
>>>
```

Boolean Data Type

- Data type with one of the **two built-in values, True or False**.
- Non-Boolean objects can be evaluated in Boolean context as well and determined to be true or false. It is denoted by the **class bool**.
- **True and False** with capital ‘T’ and ‘F’ are **valid boolean** otherwise python will throw an error.

Boolean Data Type



Sequence Data Type - String

- Strings are **sequences of character data**.
- The string type in python comes under the **class str**.
- String literals may be **delimited using either single or double or triple quotes**.
- All the characters between the **opening delimiter and matching closing delimiter** are part of the string

String Data Type

```
>>> type('Python')
<class 'str'>
>>> type("String")
<class 'str'>
>>> type("'Apple'")
<class 'str'>
>>> type("""KGiSL""")
<class 'str'>
>>>
>>> print("""This is a
string that spans
across several lines""")
This is a
string that spans
across several lines
>>>
```

String Data Type

- What if you want to include a quote character as part of the string itself?

```
>>> print('This string contains a single quote (') character.')
SyntaxError: invalid syntax
>>>
```

- The string in this example opens with a single quote.
- Python assumes the next single quote, the one in parentheses which was intended to be part of the string, is the closing delimiter.

String Data Type

- If you want to include either type of quote character within the string, the simplest way is to **delimit the string with the other type**.

```
>>> print("This string contains a single quote (') character.")
This string contains a single quote (') character.
>>> print('This string contains a double quote (") character.')
```

Escape Sequences in Strings

- Sometimes, you want Python to **interpret a character or sequence of characters within a string differently.** This may occur in one of two ways:
  - **Suppress the special interpretation** that certain characters are usually given within a string.
  - **Apply special interpretation to characters** in a string which would normally be taken literally.

Suppressing Special Character Meaning

Escape Sequence	Usual Interpretation of Character(s) After Backslash	"Escaped" Interpretation
\'	Terminates string with single quote opening delimiter	Literal single quote (') character
\"	Terminates string with double quote opening delimiter	Literal double quote (") character
\<newline>	Terminates input line	Newline is ignored
\\	Introduces escape sequence	Literal backslash (\) character

Suppressing Special Character Meaning

```
>>> print('This string contains a single quote (\') character.')
This string contains a single quote (') character.
>>> print('This string contains a single quote (\") character.')
This string contains a single quote (") character.
>>> print('a

SyntaxError: EOL while scanning string literal
>>> print('a\
b\
c')
abc
>>> print('foo\\bar')
foo\bar
>>>
```

Suppressing Special Character Meaning

- The computer can distinguish between a tab character and a sequence of space characters, but you can't.
  - To a human reading the code, tab and space characters are visually indistinguishable.
- Some text editors are configured to automatically eliminate tab characters by expanding them to the appropriate number of spaces.
- Some Python REPL environments will not insert tabs into code.

Suppressing Special Character Meaning

Escape Sequence	"Escaped" Interpretation
\a	ASCII Bell (BEL) character
\b	ASCII Backspace (BS) character
\f	ASCII Formfeed (FF) character
\n	ASCII Linefeed (LF) character
\N{<name>}	Character from Unicode database with given <name>
\r	ASCII Carriage Return (CR) character
\t	ASCII Horizontal Tab (TAB) character
\uxxxx	Unicode character with 16-bit hex value xxxx

Suppressing Special Character Meaning

Escape Sequence	"Escaped" Interpretation
\Uxxxxxxxx	Unicode character with 32-bit hex value xxxxxxxx
\v	ASCII Vertical Tab (VT) character
\ooo	Character with octal value ooo
\xhh	Character with hex value hh

Suppressing Special Character Meaning

```
>>> print("a\tb")
a       b
>>> print("a\141\x61")
aaa
>>> print("a\nb")
a
b
>>> print('\u2192 \N{rightwards arrow}')
→ →
>>>
```

Raw Strings

- A raw string literal is **preceded by r or R**, which specifies that **escape sequences** in the associated string are **not translated**.

```
>>> print('foo\nbar')
foo
bar
>>> print(r'foo\nbar')
foo\nbar
>>> print('foo\\bar')
foo\bar
>>> print(R'foo\\bar')
foo\\bar
>>>
```