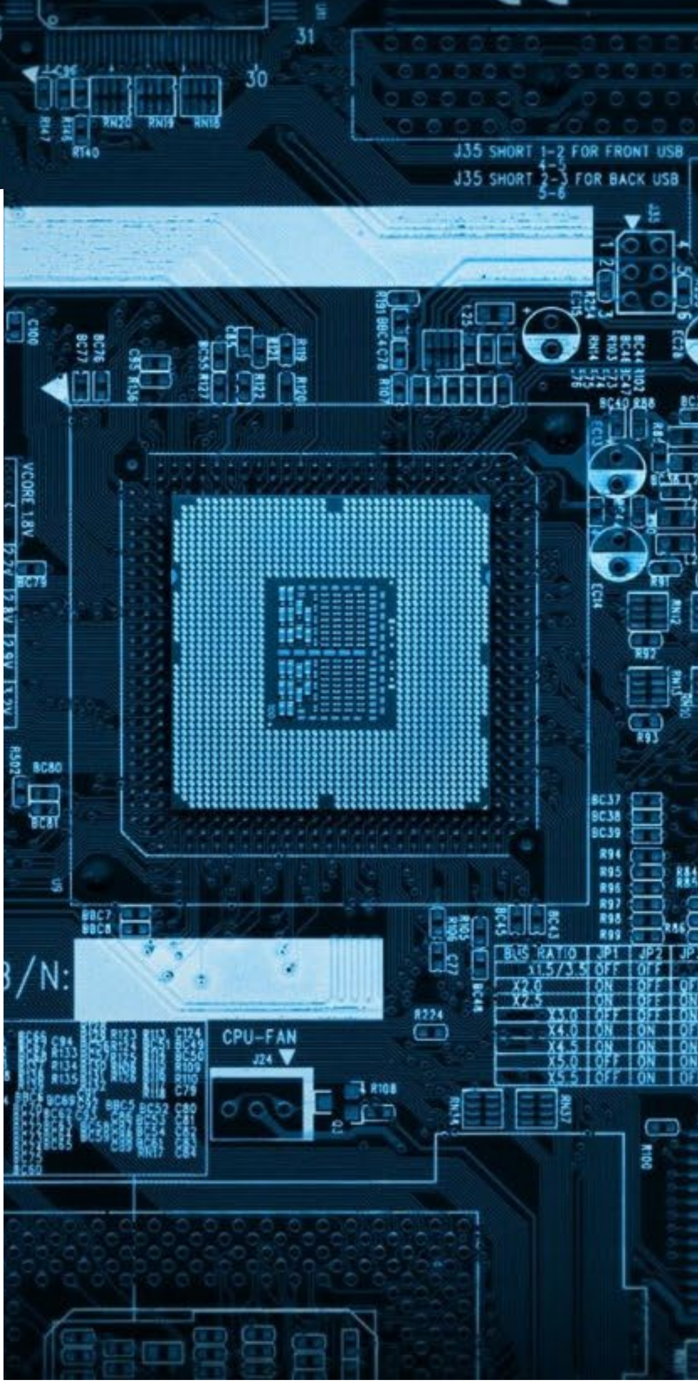
SCMA 249

Computer Programming in Actuarial Science

1/2022

LAB 1

Introduction



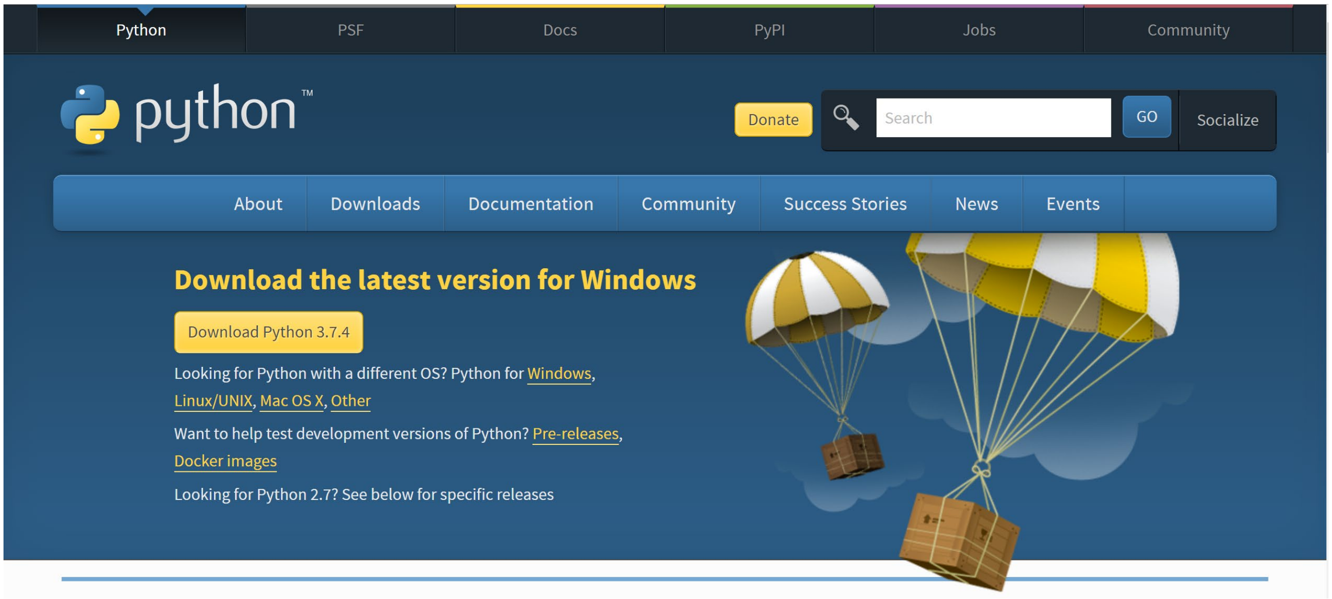
LAB 1
Introduction

1.1 Getting Started

Python is an open source, high-level programming language developed by Guido van Rossum in the late 1980s and presently administered by Python Software Foundation. It came from the ABC language that he helped create early on in his career. Python is a powerful language that you can use to create games, write GUIs, and develop web applications. It is a high-level language. Reading and writing codes in Python is much like reading and writing regular English statements. Because they are not written in machine-readable language, Python programs need to be processed before machines can run them.

Installing Python

Go to www.python.org and download the latest version of Python. You would normally opt to download the latest version.

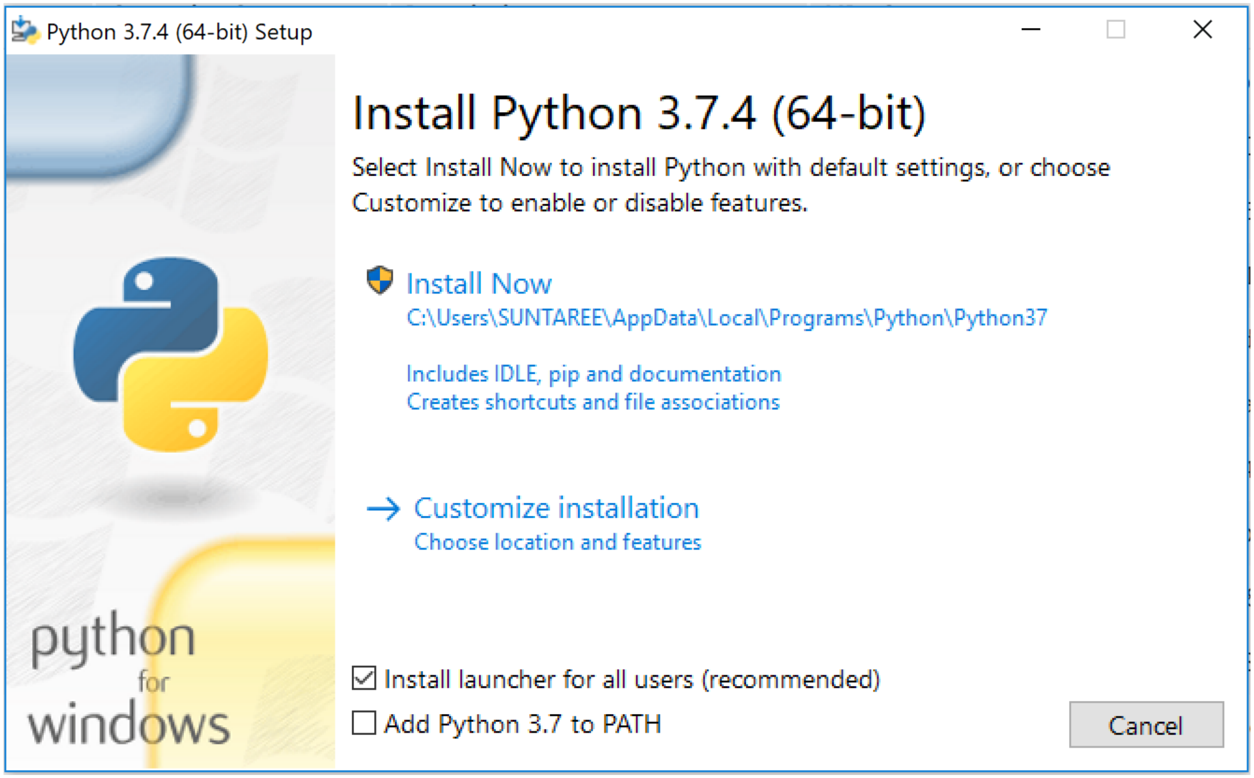


Download the latest version for Windows

Looking for Python with a different OS? Python for Windows, Linux/UNIX, Mac OS X, Other

Want to help test development versions of Python? Pre-releases, Docker images

Looking for Python 2.7? See below for specific releases


Python 3.7.4 (64-bit) Setup

Install Python 3.7.4 (64-bit)

Select Install Now to install Python with default settings, or choose Customize to enable or disable features.

Install Now
C:\Users\SUNTAREE\AppData\Local\Programs\Python\Python37

Includes IDLE, pip and documentation
Creates shortcuts and file associations

→ Customize installation
Choose location and features

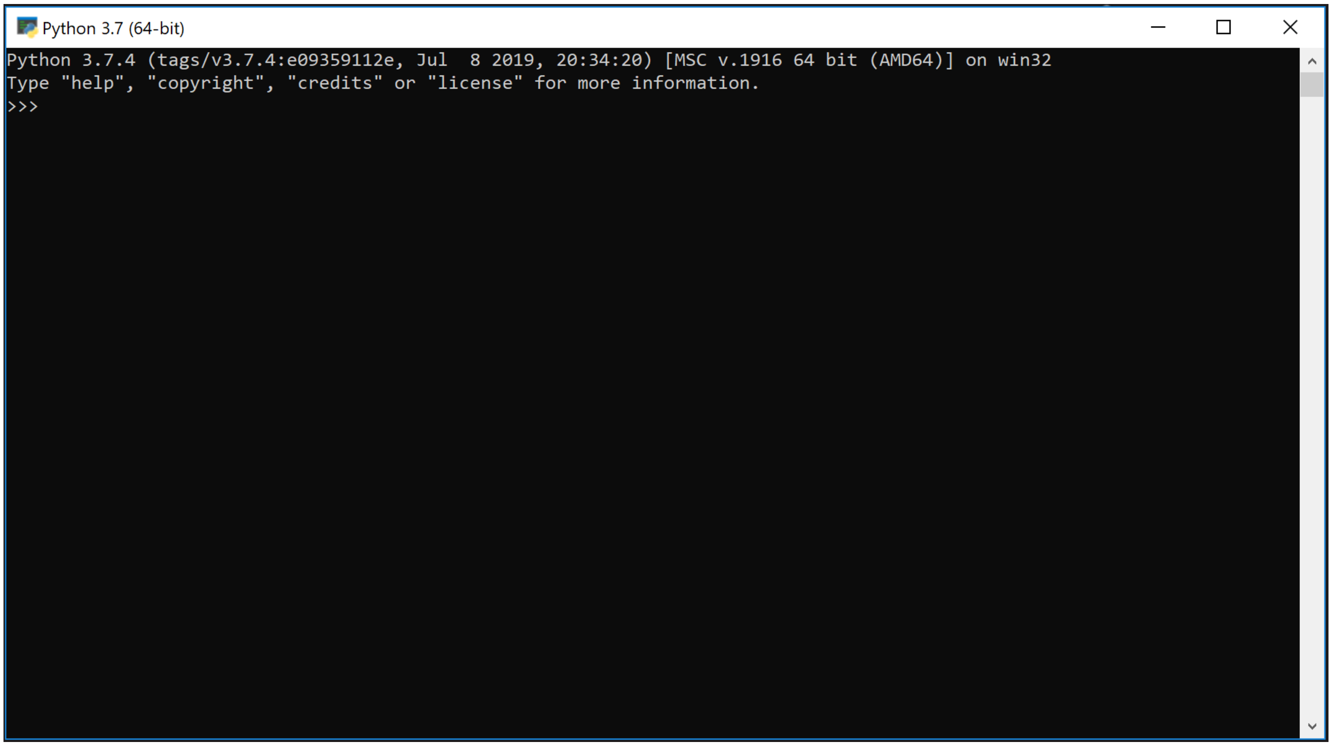
☑ Install launcher for all users (recommended)
□ Add Python 3.7 to PATH

Python is a flexible and dynamic language that you can use in different ways. You can use it interactively when you simply want to test a code or a statement on a line-by-line basis or when you're exploring its features. You can use it in script mode when you want to interpret an entire file of statements or application program.

To use Python interactively, you can use either the Command Line window or the IDLE Development Environment.

The Command Line Window

The command line is the most straightforward way to work with Python. You can easily visualize how Python works as it responds to every completed command entered on the **>>> prompt**. It may not be the most preferred interaction with Python, but it is the simplest way to explore how Python works.



```
Python 3.7.4 (tags/v3.7.4:e09359112e, Jul  8 2019, 20:34:20) [MSC v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

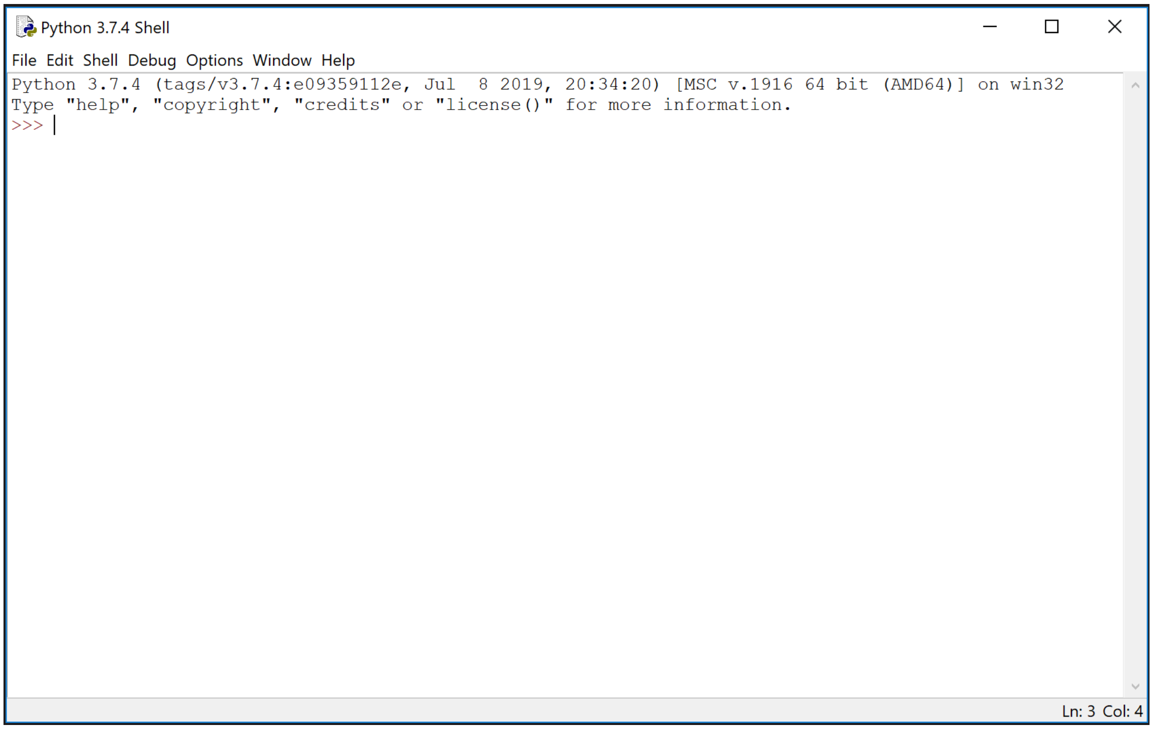
To see how Python works, you can use the print command to print the universal program "Hello, World!" by typing "print("Hello, World!")" at the >>>prompt. Press enter to tell Python that you're done with your command. Very quickly, the command line window will display Hello, World! on the following line.

Python responded correctly because you gave it a command in a format that it requires. To see how it responds when you ask it to print the same string using a wrong syntax for the print command, type and enter the following command on the Python command prompt, Print("Hello, World!"). This is how Python will respond: "Syntax error: invalid syntax" **Be careful when you type the command in Python because it is a case-sensitive language.**

IDLE: Python's Integrated Development Environment (IDE)

The IDLE (Integrated Development and Learning Environment) tool is included in Python's installation package but you can choose to download more sophisticated third party IDEs.

The IDLE tool offers a more efficient platform to write your code and work interactively with Python. You can access IDLE on the same folder where you found the command line icon or on the start menu. As soon as you click on the IDLE icon, it will take you to the Python Shell window (see the following figure).



```
Python 3.7.4 (tags/v3.7.4:e09359112e, Jul  8 2019, 20:34:20) [MSC v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> |
```

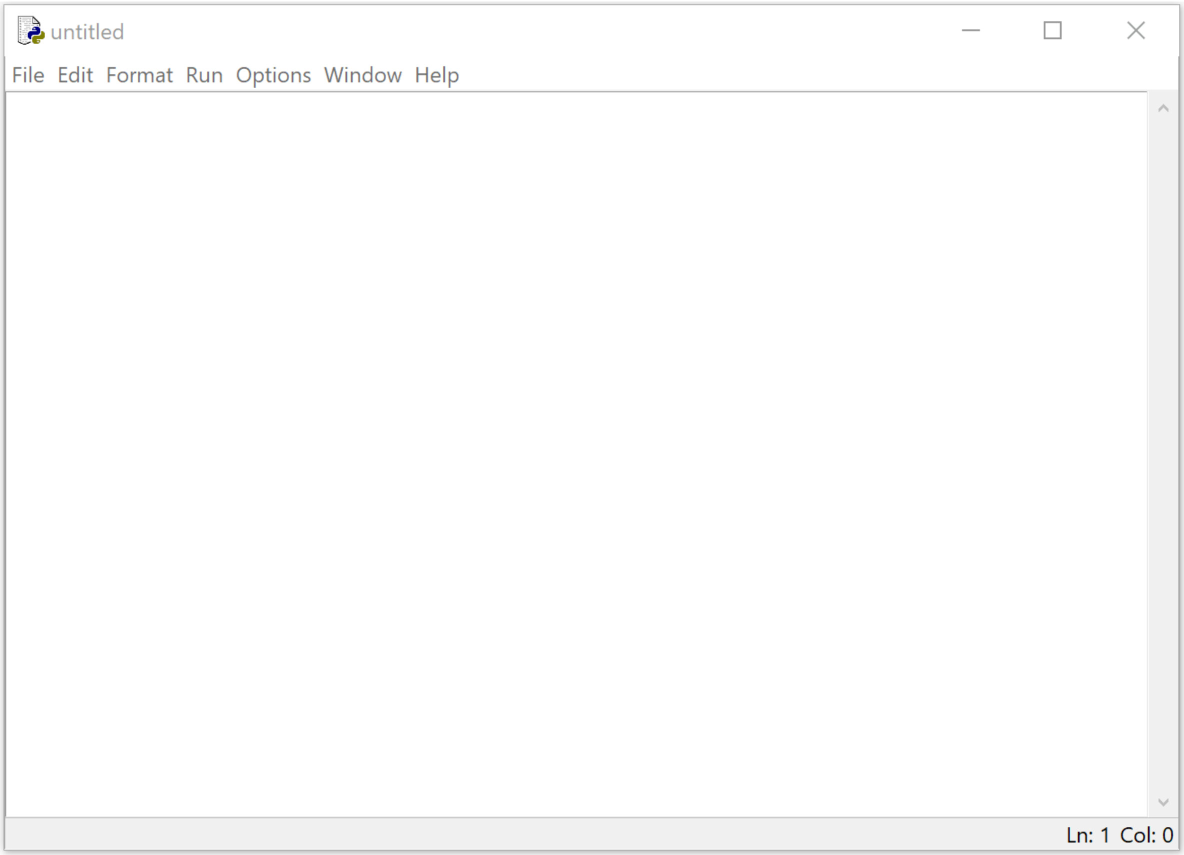
The **Python Shell Window** has dropdown menus and a >>>prompt that you have seen earlier in the command line window. Here you can type and enter statements or expressions for evaluation in the same way that you used the command line earlier. This time however, IDLE's editing menu allows you to scroll back to your previous commands, cut, copy, and paste previous statements and make modifications. IDLE is quite a leap from the command line interaction.

The Python Shell window has the following menu items: File, Edit, Shell, Debug, Options, Windows, and Help. The *Shell* and *Debug* menus provide capabilities you would find useful when creating larger programs.

The File Window

The items on the File menu allows you to create a new file, open an old file, open a module, and/or save your session. When you click on the 'New File' option, you will be taken to a new window, a simple and standard text editor where you can type or edit your code. Initially, this file window is named 'untitled' but its name will soon change as you save your code.

The File window's menu bar varies only slightly with the Shell Window. It doesn't have the 'Shell' and 'Debug' menu found in the Shell Window but it introduces two new menus: the Run and the Format menu. When you choose to Run your code on the file window, you can see the output on the Shell Window.



The Script Mode

When working in script mode, you won't automatically see results the way you would in interactive mood. To see an output from a script, you'll have to run the script and/or invoke the **print()** function within your code.

1.2 Python Syntax

Python syntax refers to the set of rules that defines how human users and the system should write and interpret a Python program. If you want to write and run your program in Python, you must familiarize yourself with its syntax.

Python keywords are reserved words in Python that should not be used as variable, constant, function name, or identifier in your code.

A Python Identifier is a name given to a function, class, variable, module, or other objects that you'll be using in your Python program. Any entity you'll be using in Python should be appropriately named or identified as they will form part of your program. Here are Python naming conventions that you should be aware of:

- An identifier can be a combination of uppercase letters, lowercase letters, underscores, and digits (0-9). Hence, the following are valid identifiers: myClass, my_variable, var_1, and print_hello_world.
- Special characters such as %, @, and $ are not allowed within identifiers.

- An identifier should not begin with a number. Hence, 2variable is not valid, but variable2 is acceptable.
- Python is a case-sensitive language and this behavior extends to identifiers. Thus, Labor and labor are two distinct identifiers in Python.
- You cannot use Python keywords as identifiers.
- Class identifiers begin with an uppercase letter, but the rest of the identifiers begin in lowercase.
- You can use underscores to separate multiple words in your identifier.

You should always choose identifiers that will make sense to you even after a long gap. Hence, while it is easy to set your variable to c = 2, you might find it more helpful for future reference if you use a longer but more relevant variable name such as count = 2.

Remarks

Multi-line statements

A statement may span over several lines. To break a long statement over multiple lines, you can wrap the expression inside parentheses, braces, and brackets. This is the preferred style for handling multi-line expressions. Another way to wrap multiple lines is by using a backslash (\) at the end of every line to indicate line continuation.

Comments

When writing a program, you'll find it helpful to put some notes within your code to describe what it does. A comment is very handy when you have to review or revisit your program. It will also help another programmer who might need to go over the source code. You can write comments within your program by starting the line with a hash (#) symbol. A hash symbol tells the Python interpreter to ignore the comment when running your code. For multi-line comments, you can use a hash symbol at the beginning of each line. Alternatively, you can also wrap multi-line comment with triple quotes.

1.3 Variables and Data Types

Variables

A variable is like a container that stores values that you can access or change. It is a way of pointing to a memory location used by a program. You can use variables to instruct the computer to save or retrieve data to and from this memory location.

Python differs significantly from languages such as Java, C, or C++ when it comes to dealing with variables. Other languages declare and bind a variable to a specific data type. This means that it can only store a unique data type. Hence, if a variable is of integer type, you can only save integers in that variable when running your program.

Python is a lot more flexible when it comes to handling variables. If you need a variable, you'll just think of a name and declare it by assigning a value. If you need to, you can change the value and data type that the variable stores during program execution.

There are some conditions to name the variables which mentioned in the previous part, Python identifiers.

Data Types

Python handles several data types to facilitate the needs of programmers and application developers for workable data. These include strings, numbers, Booleans, lists, date, and time. The followings will provide some explanation of few basic data types.

Strings

Strings are sequences of character data. The string type in Python is called **str**. String literals may be delimited using either single or double quotes. All the characters between the opening delimiter and matching closing delimiter are part of the string:

```
>>> print("I love SCMA 249.")
I love SCMA 249.

>>> print(' I love SCMA 249.')
I love SCMA 249.
```

A string in Python can contain as many characters as you wish. The only limit is your machine's memory resources. A string can also be empty:

```
>>> ''
''
```

What if you want to include a quote character as part of the string itself? Your first impulse might be to try something like this:

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

As you can see, that doesn't work so well. The string in this example opens with a single quote, so Python assumes the next single quote, the one in parentheses which was intended to be part of the string, is the closing delimiter. The final single quote is then a stray and causes the syntax error shown.

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. If a string is to contain a single quote, delimit it with double quotes and vice versa:

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

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

Integers

There is effectively no limit to how long an integer value can be. Of course, it is constrained by the amount of memory your system has, as are all things, but beyond that an integer can be as long as you need it to be:

```
>>> print(123123123123123123123123123123123123123123123123 + 1)
123123123123123123123123123123123123123123123124
```

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
`0o` (zero + lowercase letter `'o'`) `0O` (zero + uppercase letter `'O'`)	Octal	8
`0x` (zero + lowercase letter `'x'`) `0X` (zero + uppercase letter `'X'`)	Hexadecimal	16

```
>>> print(0o10)
8

>>> print(0x10)
16

>>> print(0b10)
2
```

Floating-Point Numbers

The **float** type in Python designates a floating-point number. **float** values are specified with a decimal point. Optionally, the character e or E followed by a positive or negative integer may be appended to specify scientific notation:

```
>>> 4.2
4.2
>>> type(4.2)
<class 'float'>
>>> 4.
4.0
>>> .2
0.2

>>> .4e7
4000000.0
>>> type(.4e7)
<class 'float'>
>>> 4.2e-4
0.00042
```

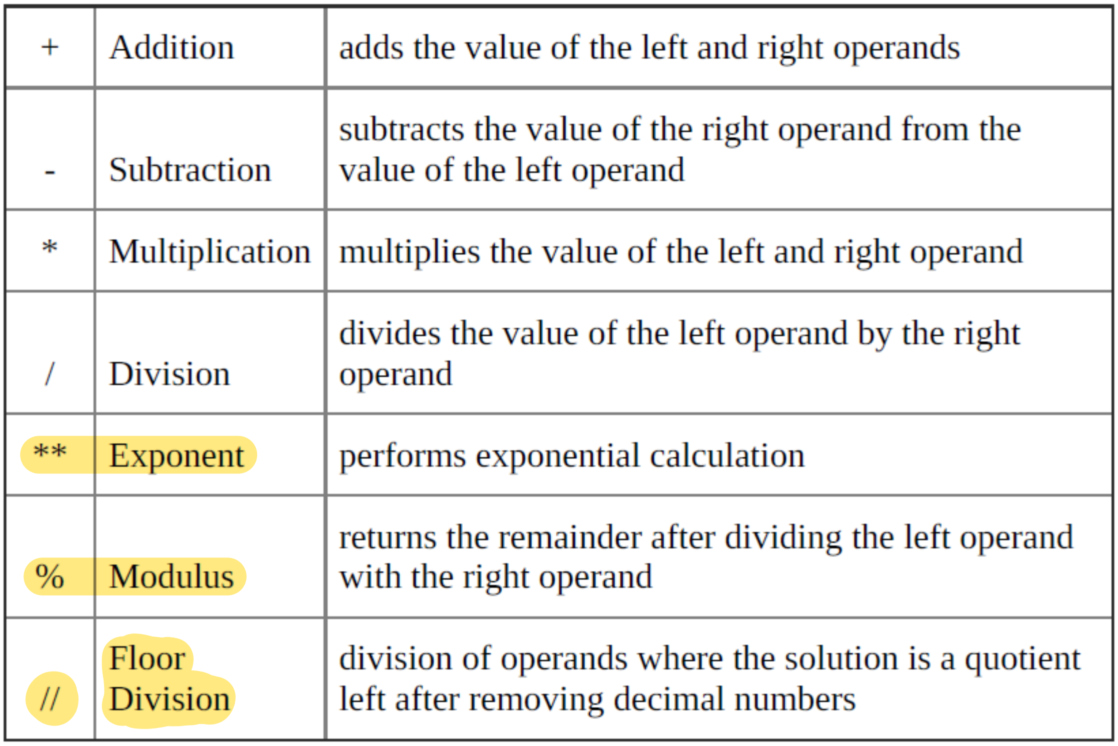
Assigning Value to Variables

If more than one variable have the same value, you can assign the value to those variables simultaneously.

```
>>> a = b = c = 1.0
```

Basic Python Operators

Python does a good job of processing mathematical expressions with its basic arithmetic operators. You can easily make programs to automate tasks such as computing tax, tips, discounts, or rent.



+	Addition	adds the value of the left and right operands
-	Subtraction	subtracts the value of the right operand from the value of the left operand
*	Multiplication	multiplies the value of the left and right operand
/	Division	divides the value of the left operand by the right operand
**	Exponent	performs exponential calculation
%	Modulus	returns the remainder after dividing the left operand with the right operand
//	Floor Division	division of operands where the solution is a quotient left after removing decimal numbers

Concept

Flowchart

What is a flowchart?

- A flowchart is a picture (graphical representation) of the problem solving process.
- A flowchart gives a step-by-step procedure for solution of a problem.

Elements of a flowchart:

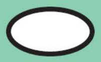
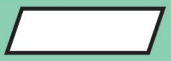
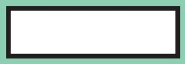
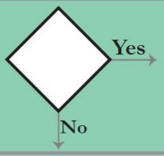
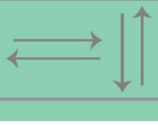
- Various geometrical shaped boxes represent the steps of the solution.
- The boxes are connected by directional arrows to show the flow of the solution.

Uses of a flowchart:

- To specify the method of solving a problem.
- To plan the sequence of a computer program.
- Communicate ideas, solutions.

Info

Flowchart symbols and their purpose

Flowchart symbols	Geometric shape	Purpose
Ellipse		Ellipse is used to indicate the start and end of a flowchart. Start written in the ellipse indicates the beginning of a flowchart. End or Stop or Exit written in the ellipse indicates the end of the flowchart.
Parallelogram		A parallelogram is used to read data (input) or to print data (output).
Rectangle		A rectangle is used to show the processing that takes place in the flowchart.
Diamond		A diamond with two branches is used to show the decision making step in a flowchart. A question is specified in the diamond. The next step in the sequence is based on the answer to the question which is “Yes” or “No”.
Arrows		Arrows are used to connect the steps in a flowchart, to show the flow or sequence of the problem solving process

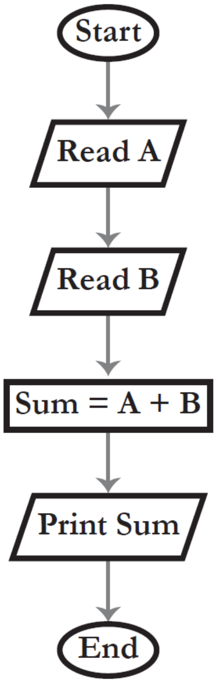
Guidelines

Drawing a flowchart

- **Identify** input and output.
- **Apply** reasoning skills to solve the problem.
- **Draw** the flowchart using the appropriate symbols and arrows to show the sequence of steps in solving the problem.

Example 1

Flowchart - How to find sum of two numbers

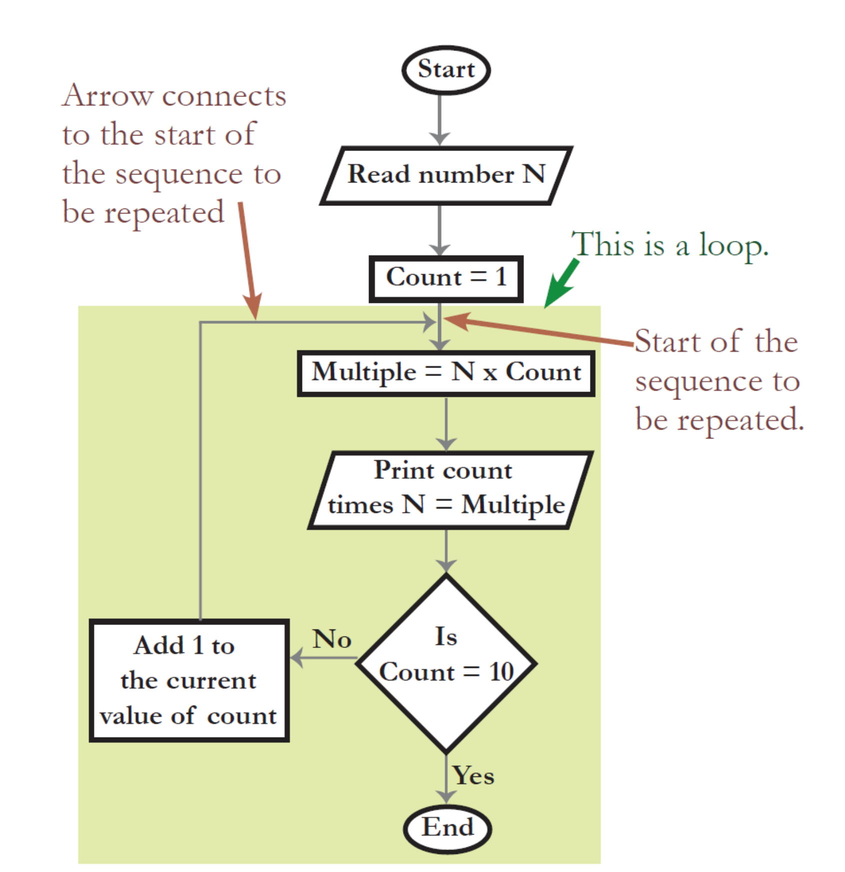


Finding sum of 845 and 247

Start
|
A= 845
|
B= 247
|
Sum= 845+ 247
|
Sum= 1092
|
End

Example 2

Flowchart of multiplication table for a given number N



Multiplication table of 12

Start

N = 12

Count =1

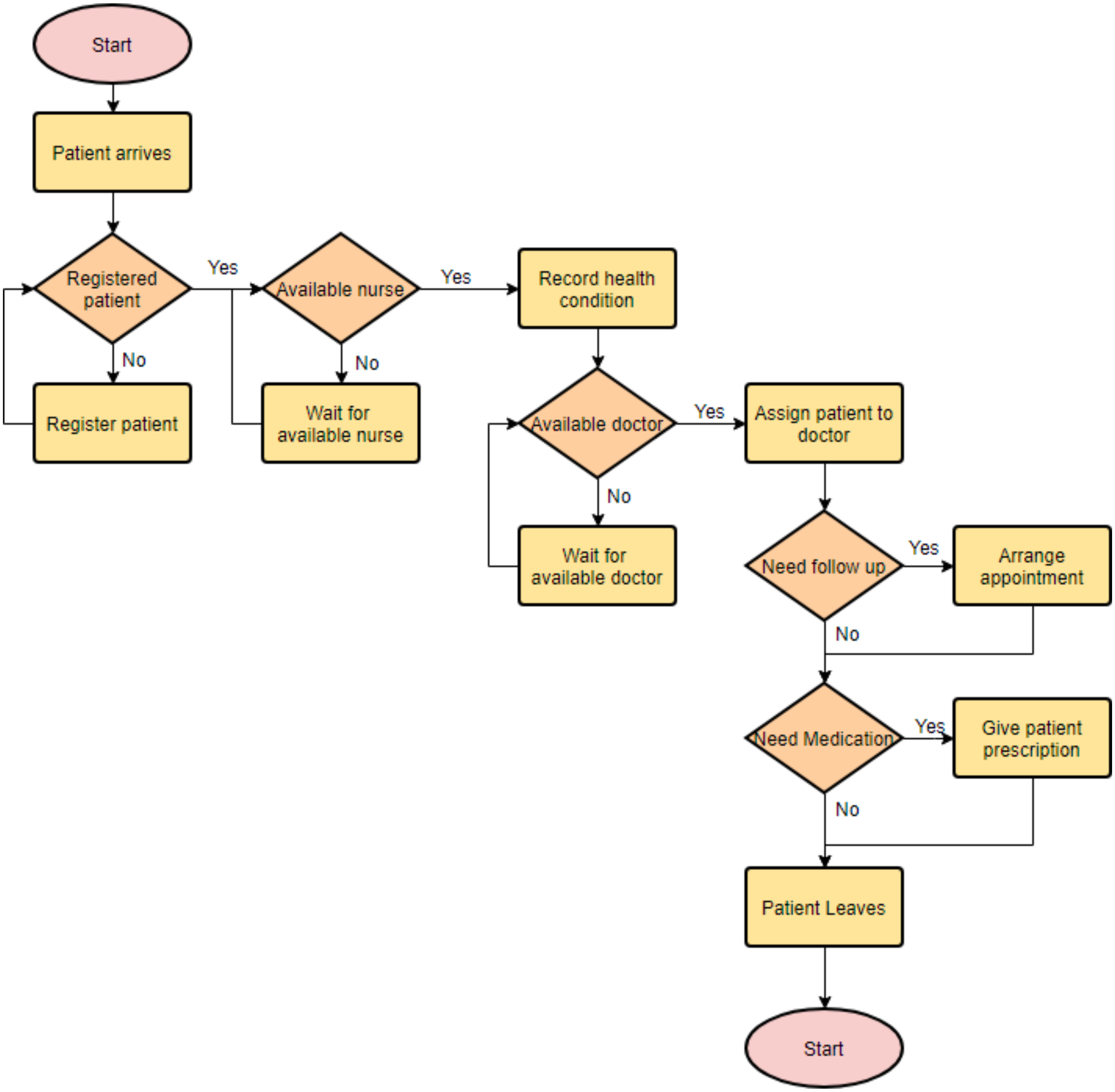
12 * 1 = 12
Count = 1+1 = 2
12 * 2 = 24
Count = 2+1 = 3
......
Count = 9+1 = 10
12 * 10 = 120

Count =10

End

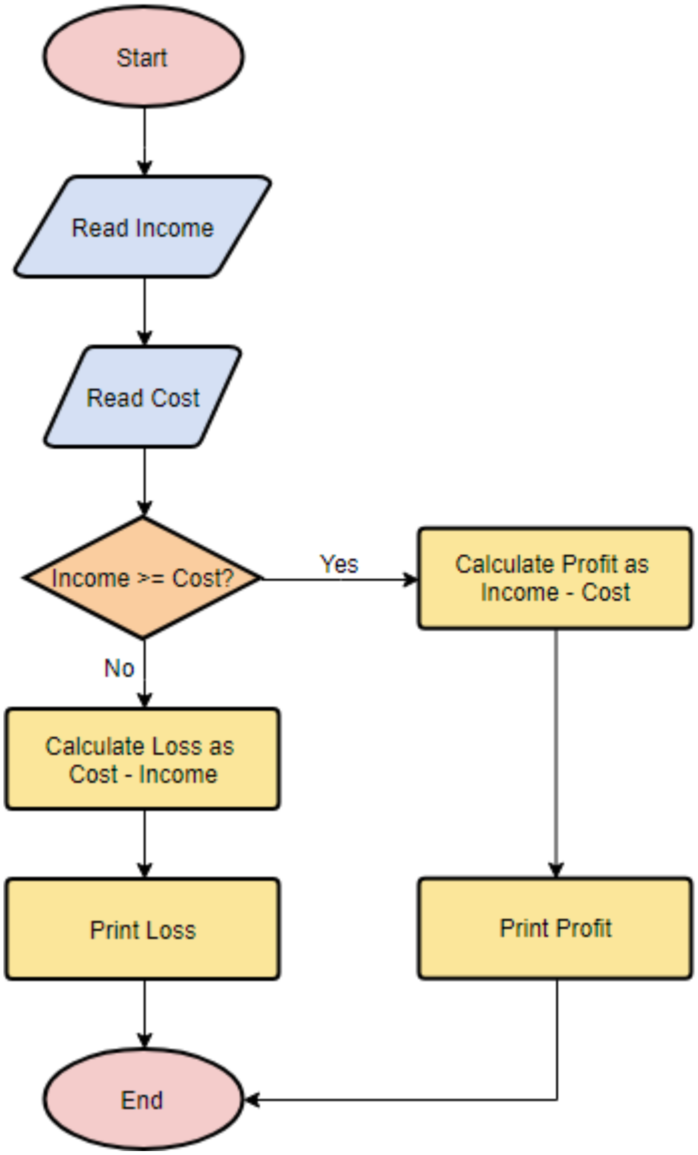
Example 3- Medical Service

This is a hospital flowchart example that shows how clinical cases shall be processed. This flowchart uses decision shapes intensively in representing alternative flows.

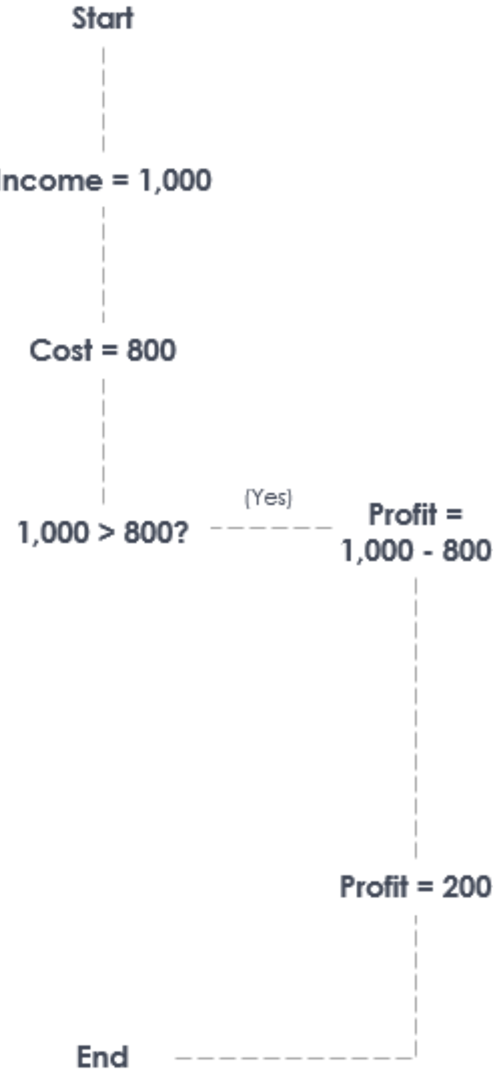


Example 4- Calculate Profit and Loss

The flowchart example below shows how profit and loss can be calculated.



Find the profit/loss when income = 1,000, cost = 800



EXAMPLE 5 Auditor Forum allows its customers to access their blood test reports through Internet. At the time of payment, the system creates a unique user ID and password which is printed on the payment receipt. The customer is allowed to log on to Auditor Forum's website on or after the specified date, to access the report. After logging on to the website the customer is required to input his email address after which the system automatically sends the test report to that email address.

After sending the email, the system automatically signs off the customer and displays the message: "Your report has been sent to your email address."

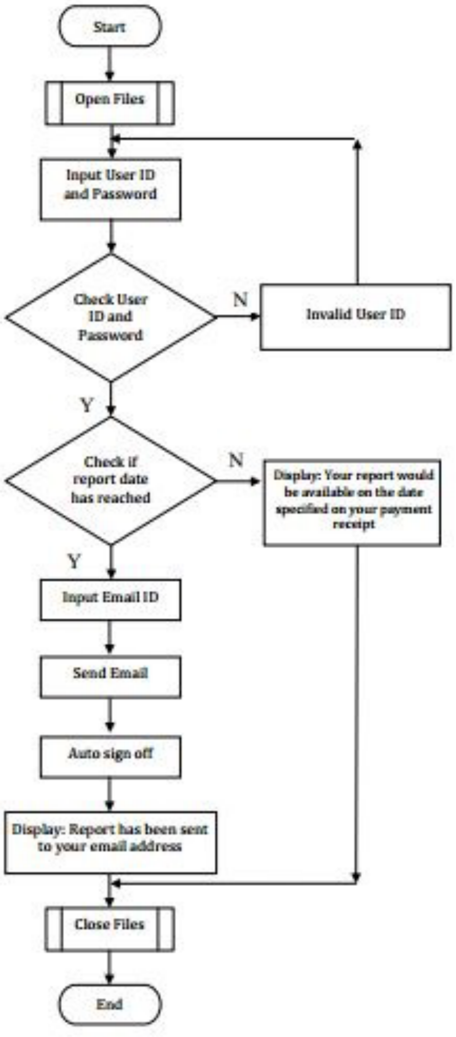
If the customer tries to log on before the report receiving date, the system displays the message:

"Your report would be available on the date specified on your payment receipt."

The user ID and password is valid only for fifteen days after the date on which the report becomes available.

Required:

Draw a program flowchart to depict the above process from customer sign in to automatic sign off.


Start
Open Files
Input User ID and Password
Check User ID and Password
N
Invalid User ID
Y
Check if report date has reached
N
Display: Your report would be available on the date specified on your payment receipt
Y
Input Email ID
Send Email
Auto sign off
Display: Report has been sent to your email address
Close Files
End

References

- https://realpython.com/python-data-types/#integers
- Kittipon P, Kittipob P, Somchai P, Sukree S. Python 101. V1.0.2., Chulalongkorn University Printing House; 2018.
- Andrew J. Python: The Ultimate Beginners Guide!., CreateSpace Independent Publishing Platform; 2016.
- Brian H. A Practical Introduction to Python Programming., Crative Commons Attribution-Noncommercial-Share Alike 3.0; 2015.