APPENDIX – A
Python and Anaconda

A.1A Quick Survey of Python and its Data Structures

Python 3.8 is a high level, interpreted and object orientedscripting language released on October 14th, 2019. It is a free and open-source. Download and install the latest version of Python by following the steps given in the installation document.

Once you install it, a Python program can be executed either in command line by directly typing the code and pressing enter to get the output or using a text editor. All python files have to be saved with .py extension. A Python program can also be executed in an Integrated Development Environment (IDE). IDLE is Python's IDE which is automatically installed when we install Python. It works on Windows, Unix, and macOS. When an IDLE is opened, an interactive Python Shell is opened.

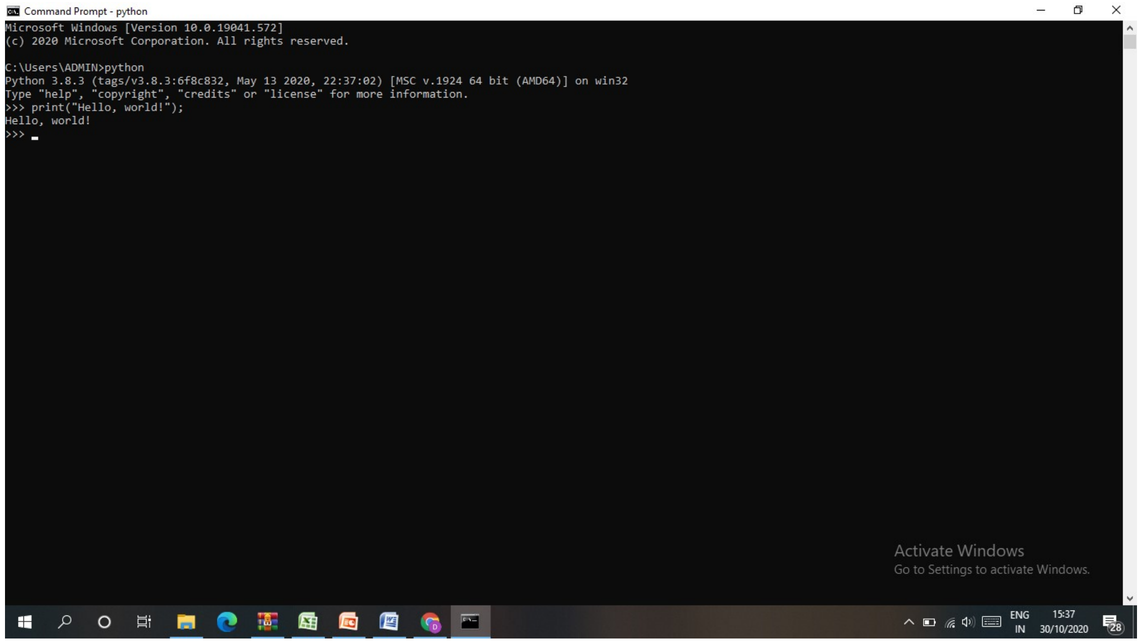
First Python Code:

Method 1:

Open a command prompt and type,

print("Hello, world!")

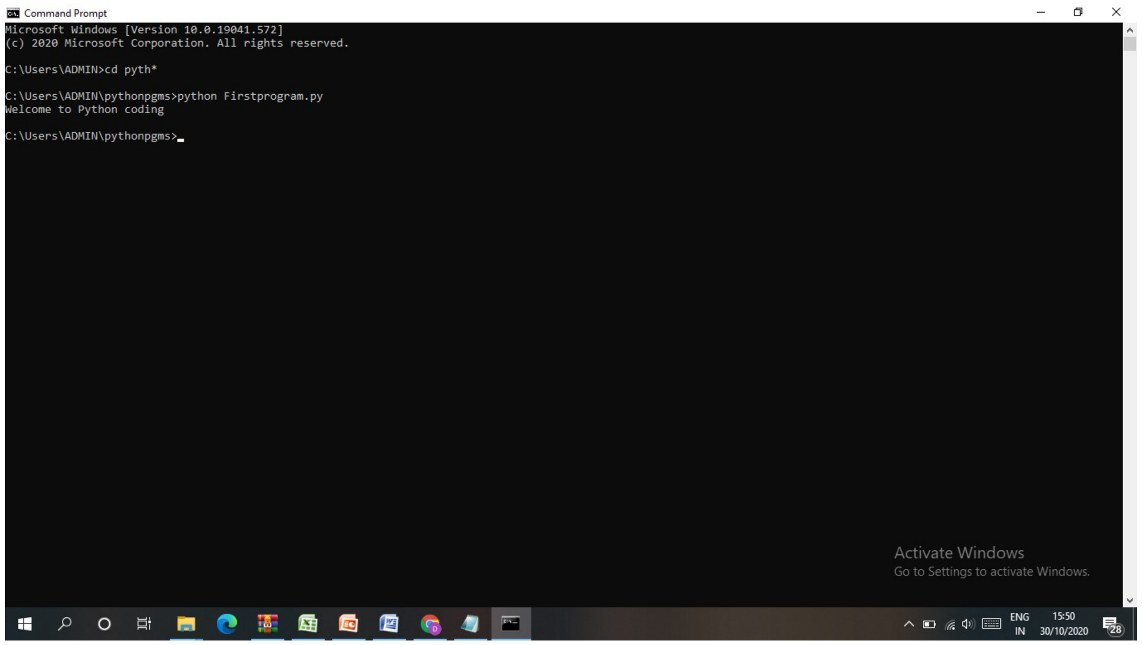
Then press Enter. You will get the output.



```
Microsoft Windows [Version 10.0.19041.572]
(c) 2020 Microsoft Corporation. All rights reserved.

C:\Users\ADMIN>python
Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:37:02) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> print("Hello, world!");
Hello, world!
>>>
```

Method 2:

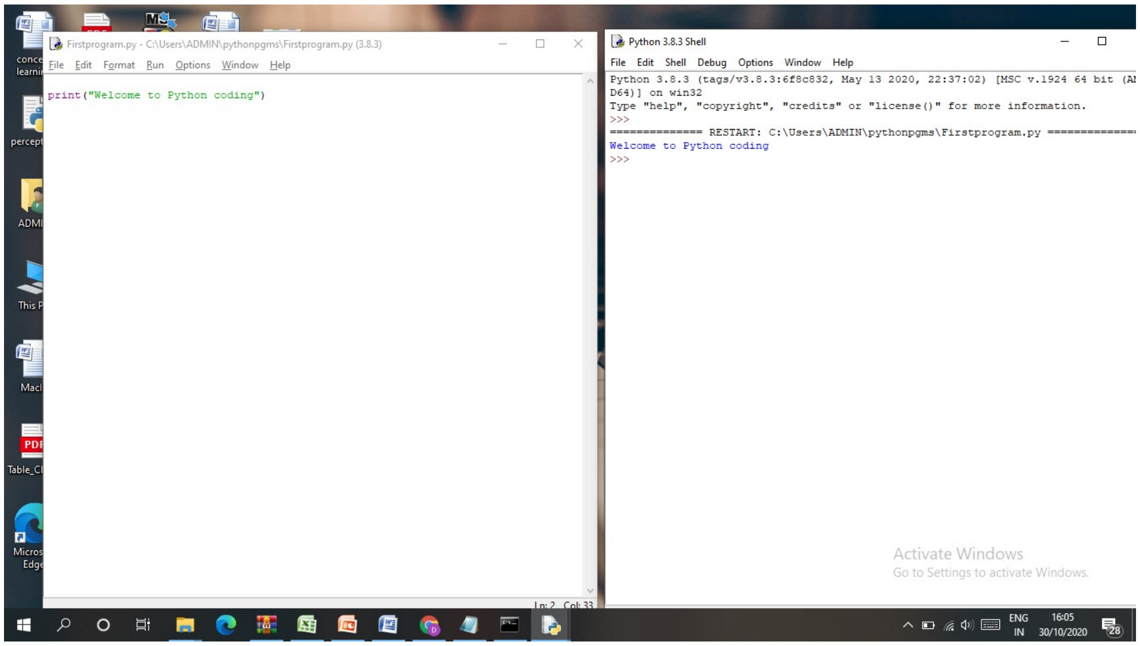
The second way of executing a Python program is to open a text editor like Notepad and type the code. Save the file with .py extension. Now, in the command prompt, type python program_name.py. You will get the output.



```
Microsoft Windows [Version 10.0.19041.572]
(c) 2020 Microsoft Corporation. All rights reserved.

C:\Users\ADMIN>cd pyth*

C:\Users\ADMIN\pythonpgms>python Firstprogram.py
Welcome to Python coding

C:\Users\ADMIN\pythonpgms>_
```

Method 3:

The third way of executing a Python program is to use an IDE. IDLE is the Python supported IDE.Open the IDLE and we can see the Python shell opened. Use the File tab to open the program. Then use the Run tab to execute. The program output can be seen in the console.


Firstprogram.py - C:\Users\ADMIN\pythonpgms\Firstprogram.py (3.8.3)
File Edit Format Run Options Window Help
Python 3.8.3 Shell
File Edit Shell Debug Options Window Help
Ln: 2 Col: 33

```python
print("Welcome to Python coding")
```

```
Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:37:02) [MSC v.1924 64 bit (AM
D64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
============== RESTART: C:\Users\ADMIN\pythonpgms\Firstprogram.py ==============
Welcome to Python coding
>>>
```

Comments

A single line can be commented using #. Multiple lines can be commented using triple quotes ‘’’ ‘’’ or “”” “””.

Indentation

Python follows strict indentation to define a block of code. This indentation level determines the grouping of statements. Normally, indentation errors occur if tabs are inconsistent.

Tokens

Identifiers, keywords, literals, operators,and delimiters are called as tokens. Tokens are separated by whitespaces.

Identifiers

Identifier is a name for a variable, function or a class defined by the user.An identifier can be named with alphabets (A-z), numbers(0–9) and an underscore (_). It cannot start with a digit. It cannot have special symbols like, @, #, $, %, etc.

Reserved words (keywords)

Keywords are predefined and cannot be used as an identifier. Python also has a list of keywords:

else import pass None break except in raise class finally is
return and continue for lambda try as def from nonlocal while
assert del global not with async elif if or yield

Reserved classes of identifiers

Besides keywords thereare some special reserved identifiers. They are represented with leading and trailing underscore characters.

For example,

The specially defined methods such as __new__() and __init__() that are used in the construction of objects.

Variables

In Python, when values are assigned to a variable, the variable holds the reference of the object or value. Python is also a type-inferred language, so we don't need to explicitly define the variable type. The type is automatically inferred by the value.

Example:

```
num1 = 15
num2 = 15.15
ch = ‘a’
str = “string”
print (num1)
print (num2)
print (str)
print(ch)
```

We can assign a single value to multiple variables.

```
a = b = 20
```

We can also assign multiple values to multiple variables, for example:

```
x, y, z = 10, 15.2, "string"
print (x)
print (y)
print (z)
```

Literals

Literals are constant values of built-in data types.

i. **String literals**

String literals can be enclosed within single, double, or triple quotes.

```
str = "Python is very easy"
```

ii. Character literals

Character literal is a single character enclosed by single or double quotes.

```
ch1 = "C"
ch2 = ‘a’
```

iii. Numeric literals

There are three types of numeric literals such as integers, floating point numbers, and imaginary/complex numbers. Integer literals can be binary literals, decimal literals, octal literals and hexadecimal literals.

Binary Literal starts with ‘0b’.

```
bnum = 0b101011
```

Decimal Literal

```
dnum = 125
```

Octal Literal starts with ‘0o’

```
onum = 0o147
```

Hexadecimal Literal starts with ‘0x’

```
hnum = 0x5E
```

Float Literal

```
fnum1= 15.5
fnum2 = 1.5e3
```

Complex Literal has two parts, real and imaginary

```
x = 3.14j
print(x, x.real, x.imag)
3.14j   3.14    0.0
```

iv. Boolean Literal

A Boolean literal can take any of the two values: True or False. It can be used in expressions also.

```
a = True
x = True + 10
y = False + 10
print("x:", x)
print("y:", y)
```

v. Special literals

Python has one special literal called None.

x = None

Literal Collections

There are four types of literal collections called List literals, Tuple literals, Dictionary literals, and Set literals.

List literalsareenclosed within square brackets.

List_Items = ["I1", "I2", "I3"]

Tuple literalsareenclosed within brackets.

Tuple_Items = (“I1”, “I2”, “I3”)

Dictionary literals are enclosed within curly braces.

gender = {'m':'male', 'f':'female'}

Set literals are enclosed within curly braces.

animals = {“cat, “dog”, “rabbit”}

print(List_Items)

print(Tuple_Items)

print(gender)

print(animals)

Operators

Python supports arithmetic operators, assignment operators, comparison operators, logical operators, bitwise operators and special operators.

Arithmetic Operators:

Operator	**Description**
+	Binary addition: Adds two operands
-	Binary Subtraction: Subtracts two operands
*	Binary Multiplication: Multiplies two operands
/	Binary division: Divides two operands
%	Modulus: Returns the remainder
**	Exponent: a**b is equal to a^b
//	Floor Division: Returns the quotient and rounds the decimal value

Assignment Operators:

Operator	**Description**
=	Assigns the RHS operand to LHS operand
+=	Adds and then assign the RHS operand to LHS operand
-=	Subtracts and then assign the RHS operand to LHS operand
*=	Multiplies and then assign the RHS operand to LHS operand
/=	Divides and then assign the RHS operand to LHS operand
%=	Modulus and then assign the RHS operand to LHS operand
**=	Exponent and then assign the RHS operand to LHS operand
//=	Floor Division and then assigns the RHS operand to LHS operand

Comparison Operators:

Operator	**Description**
==	Returns true if LHS operand is equal to RHS operand
!=	Returns true if LHS operand is not equal to RHS operand
>	Returns true if LHS operand is greater than the RHS operand
<	Returns true if LHS operand is less than the RHS operand
>=	Returns true if LHS operand is greater than and equal to RHS operand
<=	Returns true if LHS operand is less than and equal to RHS operand

Logical Operators:

Operator	**Description**
and	Returns true if LHS expression and RHS expression are true
or	Returns true if LHS expression or RHS expression are true
not	Takes single expression and returns the complement of it.

Bit Wise Operators:

Operator	**Description**
&	Bitwise AND
|	Bitwise OR
^	Bitwise XOR
~	Bitwise NOT

Shift Operators:

Operator	Description
<<	Left shift
>>	Right shift

Special Operators:

Operator	Description
is	Identity operators
is not	
in	Membership operators
not in	

Data types

Variables can hold values of any data type and depending on the type of value; the memory will be allocated by the interpreter.

- Numerical data type can be int, float or complex
- String data type
- Boolean data type takes only two values “True” and “False”.

type() keyword

type() is used to find the data type of a variable.

Structure of a program

A Python program is constituted of code blocksthat are executed as a unit. These code blocks can be a module, a function body, or a class definition. The main module for a Python script is called __main__.

To get input from user, use the function,

input()

It always gets the input as string. To convert the string to integer, use the

int()

name = int(input(what is your name?))

To print output to Console, use the function

print()

print("Your age is",age)

Control Statements

1. **if statement**

```
x =10
y =20
if x>y:
print("if stmt executed")
```

2. **if…else statement**

```
x =10
y =20
if x>y:
print("if stmt executed")
else:
print("else stmt executed")
```

3. **if…elif…else statement**

```
x = 40
y = 20
z = 50
if x>y and x>z:
print("x is greater")
elif y>x and y>z:
print("y is greater")
else:
print("z is greater")
```

Nested if statement is also supported.

Looping statements

1. **while:**

```
x = 10
while x>5:
 print(x)
 x = x-1
print("while loop executed")
```

2. **While loop with else:**

```
x = 10
while x>5:
print(x)
 x = x-1
print("while loop executed")
else:
print("Inside else")
```

Loop statements may have an else clause which is executed when the loop terminates or when the while condition becomes false, but not when the loop is terminated by a break statement.

3. **for:**

```
x = 5
for i in range(x):
    print(x)
print("for loop executed")
```

Built-in range() function is used in the for statement to iterate over a sequence of numbers. It defines the start, stop and step size as range(start, stop,step_size). step_size defaults to 1 if not provided.

Like while statement, for statement can also have an optional else clause.

break and continue Statements

The break keyword is used to stop a loop. In such cases, the else part is ignored.

```
list = [0, 1, 2, 3, 4]
for i in list:
    print(i)
    if i ==3:
      break
else:
print("No items in the list.")
```

pass Statement

The pass statement can be used when a statement is required syntactically but the program requires no action.

Import statement

We can import Python modules . For example, Import a module numpy as
import numpy

Arrays

Arrays are used to store multiple values of homogeneous type in one single variable. We can access an array element by referring to the index number.
names = [“Sam”, “Ayden”, “John”, “Paul”]

Defining Functions

The keyword def is used to define a function definition.The keyword is followed by the function name and the parenthesized list of formal parameters.The statements within the body of the function start at the next line, and must be indented.The return statement returns a value from a function.

```
def add():
a=5
b=7
print(a+b)
```

Default Argument Values

We can also specify a default value for one or more arguments.

Classes

Classes can be defined to bundle data and functionality together. The keyword “class” is used to define a class followed by the classname.

```
class classname:
        block of statements
```

Example:

```
class student:
   def __init__(self, regno, name):
self.regno = regno
      self.name = name
   def display(self):
      print(self.regno)
      print(self.name)
s1 = student(227616, "Sam" )
print(s1.regno)
print(s1.name)
s1.display();
```

An __init__() method is used for initiating a newly-created class instance. The first argument to this method is self.

Class object is created using function notation.

obj = Myclass()

Attributes are referenced using dot operator.

obj.attribute1

Data Structures

Python provides built-in data structures such as: lists, tuples, dictionaries, strings, sets and frozensets.

Lists, strings and tuples are ordered sequences of objects. Strings can contain only characters whereas lists and tuples can contain any type of objects.Lists and tuples are like arrays.

Tuples are immutable whereas Lists are mutables. Sets are mutable unordered sequence of unique elements whereas frozensets are immutable sets.

List is an ordered sequence of items. They are enclosed in brackets.

list1 = [1, 227616, "sam"]

Tuple is an ordered sequence of items same as a list. They are enclosed in parentheses.

tup = (1, 227616, "ayden")

Dictionary is an unordered collection of key-value pairs. They are enclosed with curly brackets.

dic = {"a":1, "b":2, "c":2}

Set is an unordered collection of unique items. They are enclosed in curly braces.

animals= { “cat”, “dog”, “tiger”}

A.2 Anaconda Version

Since, this book concentrates on machine learning instead of basic programming of Python, one needs more packages like NumPy, Pandas, SciPy, MatplotLib and Scikit-Learn. Installing all these can be done but there is an easy way out. That is installing an open source python version called – Anaconda.

Anaconda is a Python distribution that is extremely popular for implementing machine learning algorithms. It is developed by Continuum Analytics Inc (https://www.anaconda.com). It has all the necessary packages required to implement machine learning algorithms. It by default has many packages that are necessary for implementing machine learning algorithms such as NumPy, SciPy, MatplotLib, Pandas, IPython and CPython. It comes with Spyder, an IDE for developing Python programs. It also has a platform independent package manager called conda environment for installing Python packages. One can down Anaconda and install it like any other Windows package. Then, Spyder environment can be used to develop Python programs.

Conda is a special package manager. One can openthe Anaconda command prompt and can use either pip command or conda install for installing additional packages as:

```
>> conda install ‘package name’
```

There are many packages available in special channels. They can be installed as:

```
>> conda install -c channel name package
```

One can also use pip install as:

```
>> pip install package
```

One can create a virtual environment and install packages. This helps to create a separate environment for say deep learning programs or packages of certain versions that may be necessary. The commands that can be used are given below:

```
>> conda create –name sample numpy=1.7
```

Then this environment can be activated as:

```
>> activate sample
```

Finally, the virtual environment can be deactivated as:

```
>> deactivate
```