Artificial Intelligence

Lab 1

Agenda

Introduction to AI

- What is AI ?
- Applications of AI
- Branches of AI

Introduction to Python 3.X

- Installing Python 3
- Basic Syntax
- Installing the necessary Python packages

Package

The package will include the implementation of the algorithms we explain in the section.

At the end of each lab the required tasks will be announced with their **deadline**.

Each group will upload the code to the **GoogleDrive** shared with the given a valid **Gmail**.

- Groups 4 – 6 members.
- Registration Form (link), Deadline **28/2/2019**.

Deadlines are **not extendable**.

Avoid **cheating** or copying algorithms (even from online resources).

Refer to the template and document for more details. (links will be shared on course-sites).

Attendance & Year-work

- QR code will be used.
- Make sure you have a copy of it
- QRs are found on Course-sites (Enroll)
- Grades will be recorded course-sites.
- **Individual Tasks** only will be submitted through Course-sites.

Artificial Intelligence

"Can a machine think and behave like humans do?"

Artificial Intelligence is a way of making a computer intelligent (software think intelligently), in the similar manner the intelligent humans think.

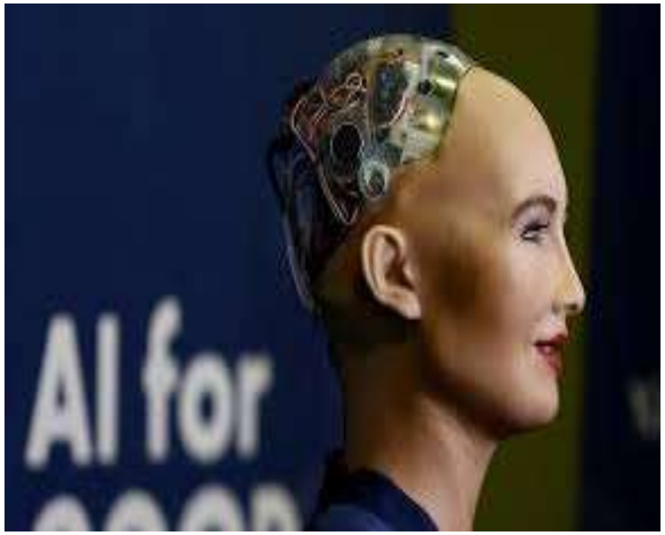
By mimicking the way the human brain learns, thinks, and takes action, we can build a machine that can do the same.

Applications of AI

AI has been used across many industries and it continues to expand rapidly.

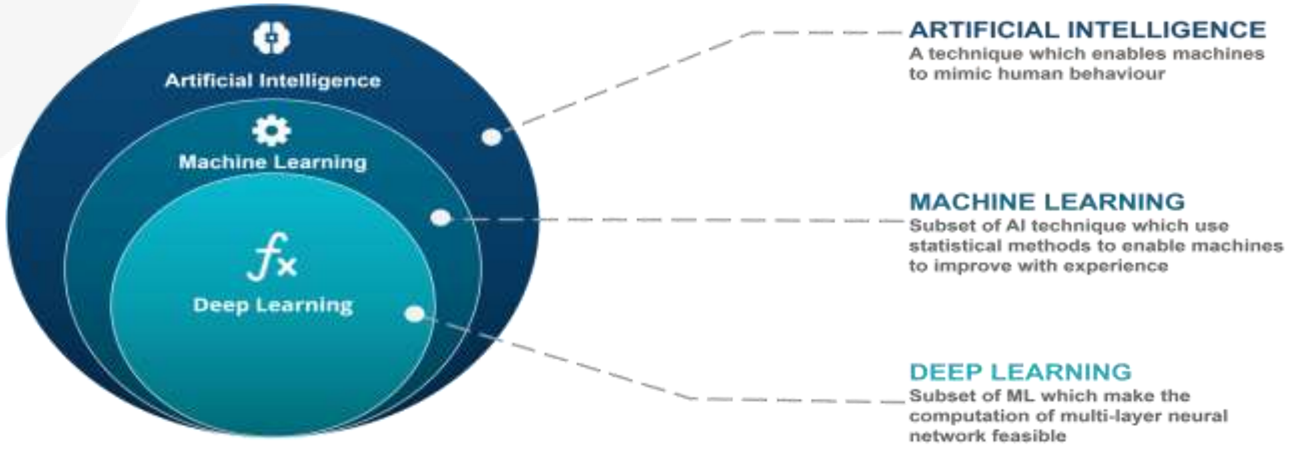
Some of the most popular areas include:

- Computer Vision
- Natural Language Processing
- Speech Recognition
- Expert Systems
- Games
- Robotics



Branches of AI

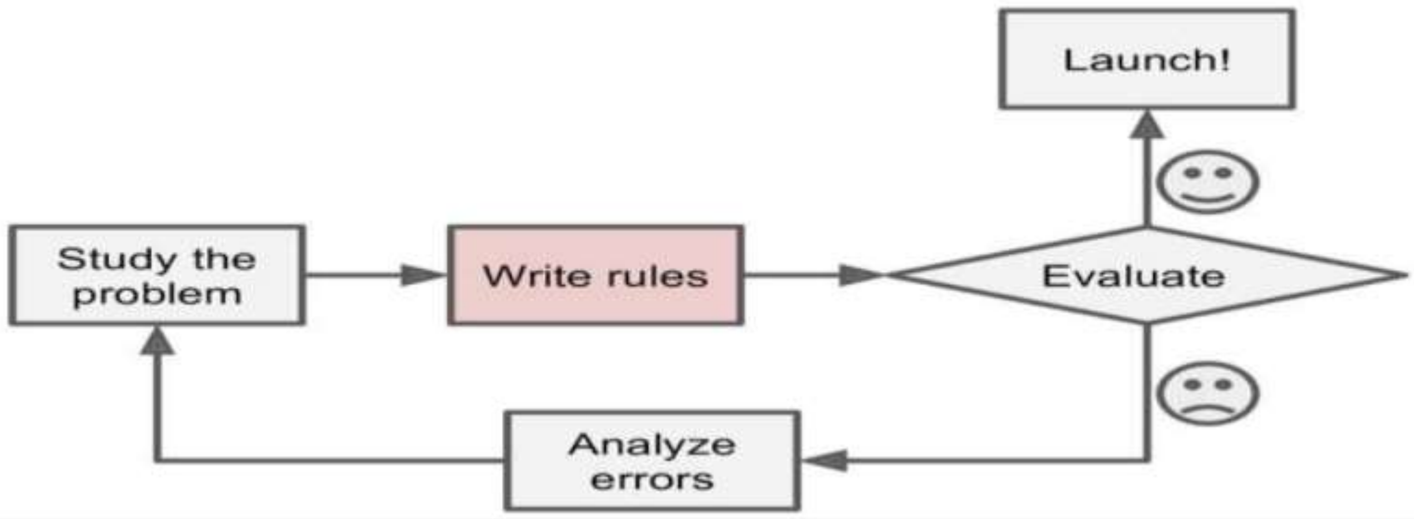
- Machine learning and pattern recognition (The most popular).
  - We design and develop software that can learn/train from data. Based on these learning models, we perform classification/predictions on unknown data.
  - One of the main constraints here is that these programs are limited to the power of the data.
  - If the dataset is small, then the learning models would be limited as well.


Artificial Intelligence
Machine Learning
Deep Learning
ARTIFICIAL INTELLIGENCE
A technique which enables machines to mimic human behaviour
MACHINE LEARNING
Subset of AI technique which use statistical methods to enable machines to improve with experience
DEEP LEARNING
Subset of ML which make the computation of multi-layer neural network feasible

Branches of AI

Logic-based AI

- A program written in logic-based AI is basically a set of statements in logical form that express facts and rules about a particular problem domain.



Branches of AI

Search

- The Search techniques are used extensively in AI programs.
- These programs examine a large number of possibilities and then pick the most optimal path.
- For example, this is used a lot in strategy games such as Chess, networking, resource allocation, scheduling, and so on.

Heuristics Search

- Heuristic is a way used in some approaches to search to measure how far a node in a search tree seems to be from a goal.
- Heuristic predicates that compare two nodes in a search tree to see if one is better than the other.
- They are used extensively in AI in fields such as robotics, search engines, and so on.

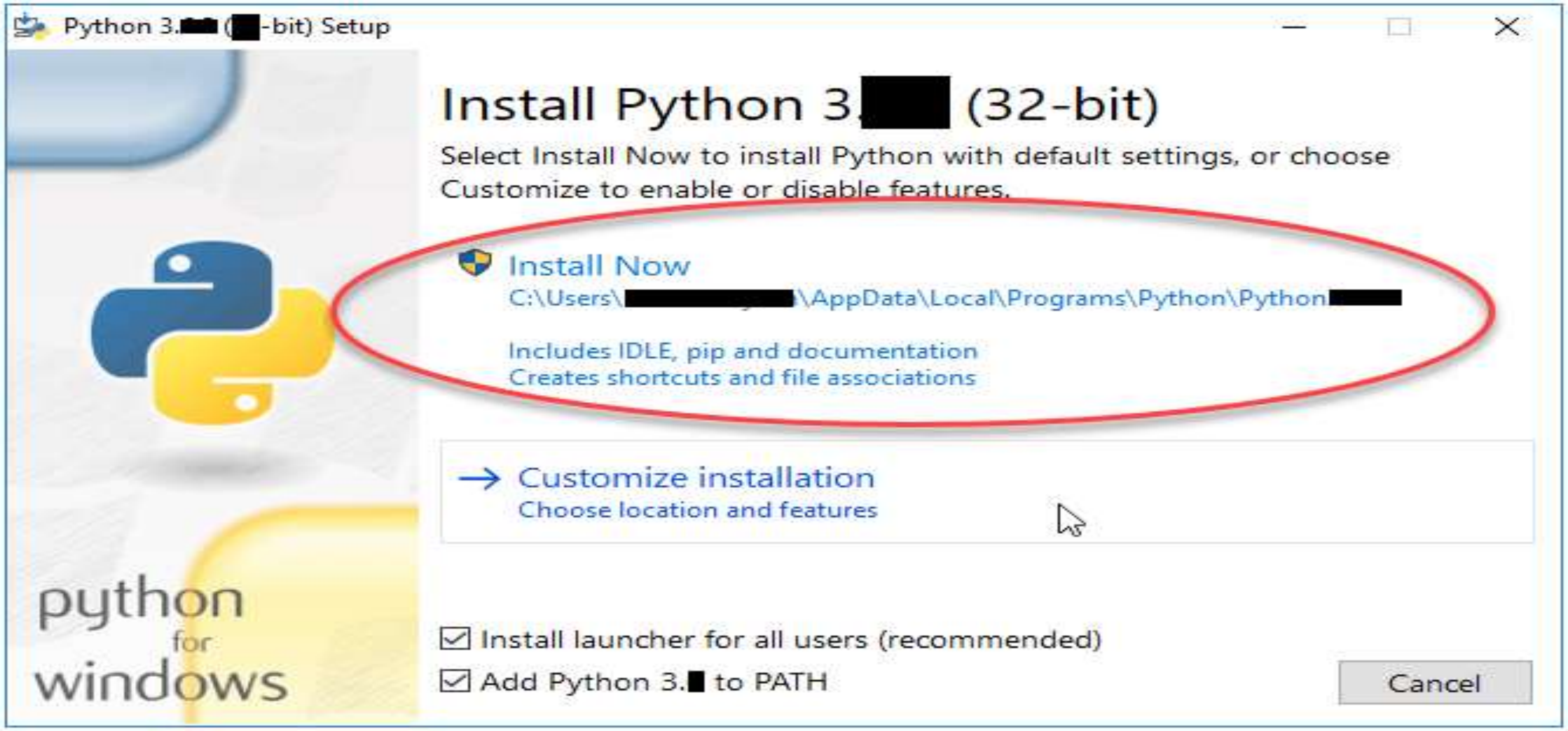
Introduction to Python 3.X

Installing Python 3.X

- Python is an open source scripting language.
- It supports Object Oriented.
- Multi-purpose (Web, GUI, Scripting, etc.)
- Python is a case sensitive.
- You can download Python 3.7.2 from here and Pycharm (IDE) from here.

Install Python

Run the exe for install Python then click on Install Now. When it finishes, you can see a screen that says the Setup was successful. click on "Close".



Python 3.[redacted] ([redacted]-bit) Setup

Install Python 3.[redacted] (32-bit)

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

Install Now
C:\Users\[redacted]\AppData\Local\Programs\Python\Python[redacted]

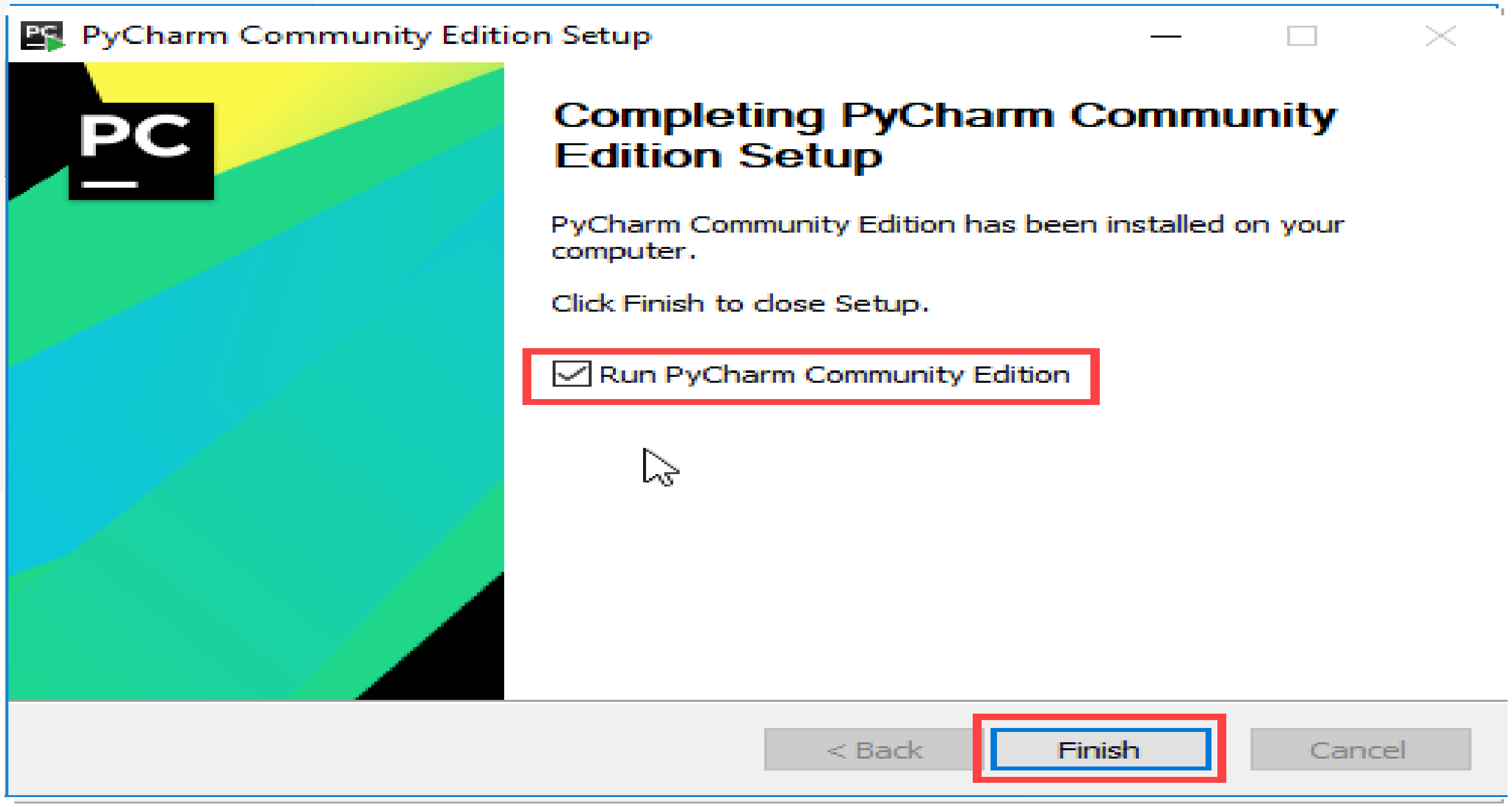
Includes IDLE, pip and documentation
Creates shortcuts and file associations

→ Customize installation
Choose location and features

☑ Install launcher for all users (recommended)
☑ Add Python 3.[redacted] to PATH

Install PyCharm

Run the exe for install PyCharm. The setup wizard should have started. Click “Next”.



PyCharm Community Edition Setup

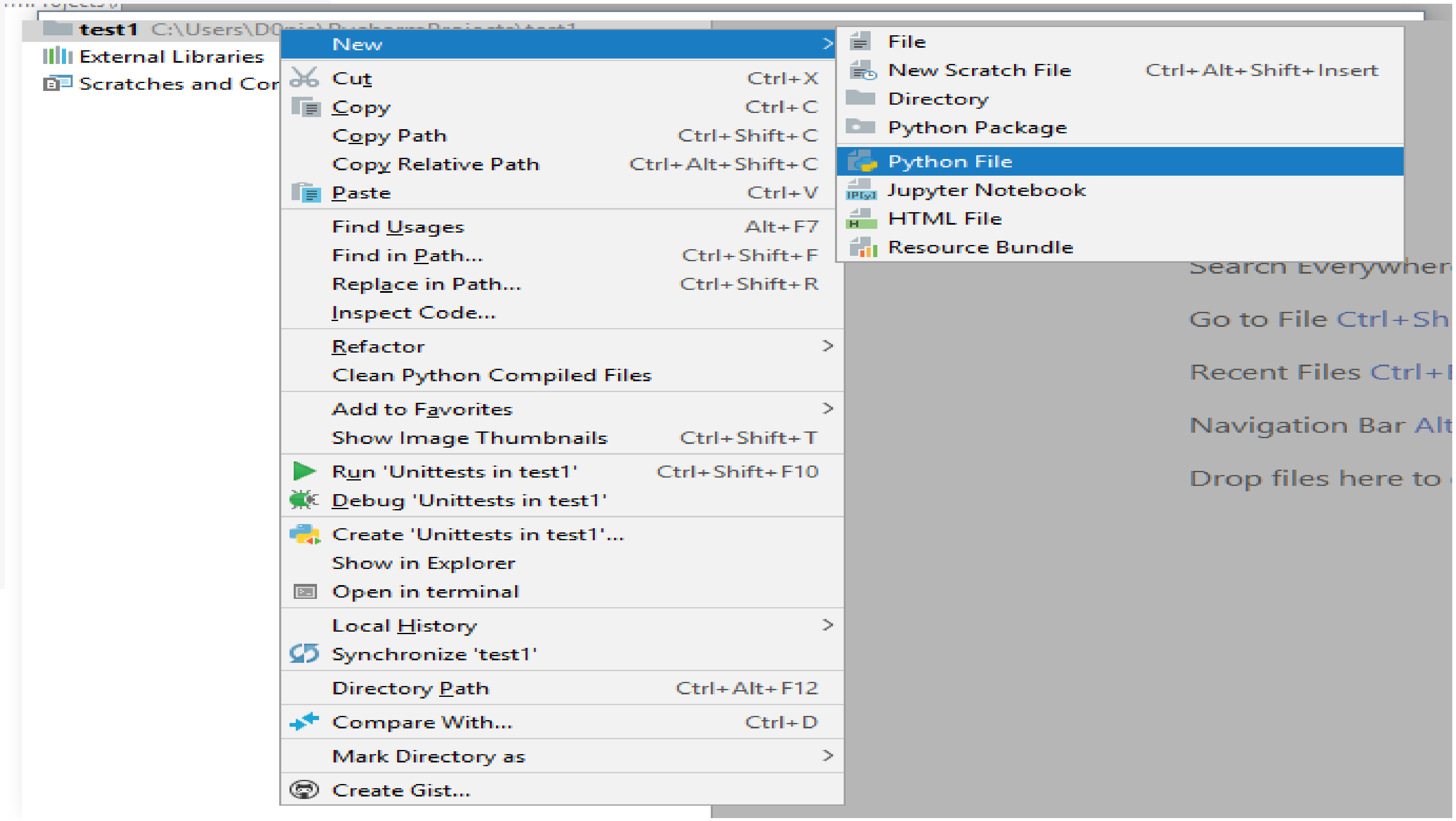
Completing PyCharm Community Edition Setup

PyCharm Community Edition has been installed on your computer.

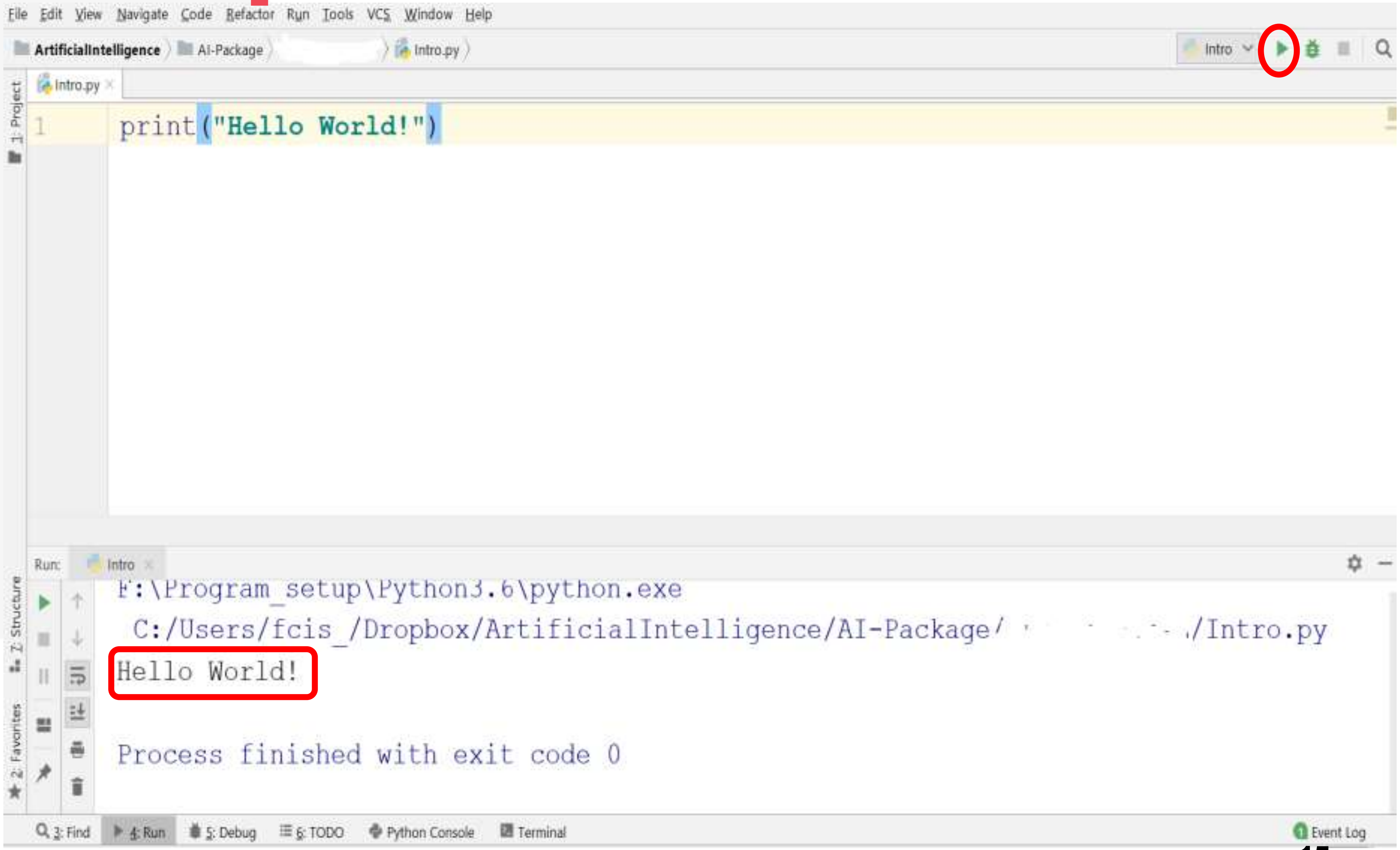
Click Finish to close Setup.

[✓] Run PyCharm Community Edition

How to create a new project?



Sample Run



```
print("Hello World!")
```

```
F:\Program_setup\Python3.6\python.exe
 C:/Users/fcis_/Dropbox/ArtificialIntelligence/AI-Package/[illegible]/Intro.py
Hello World!

Process finished with exit code 0
```

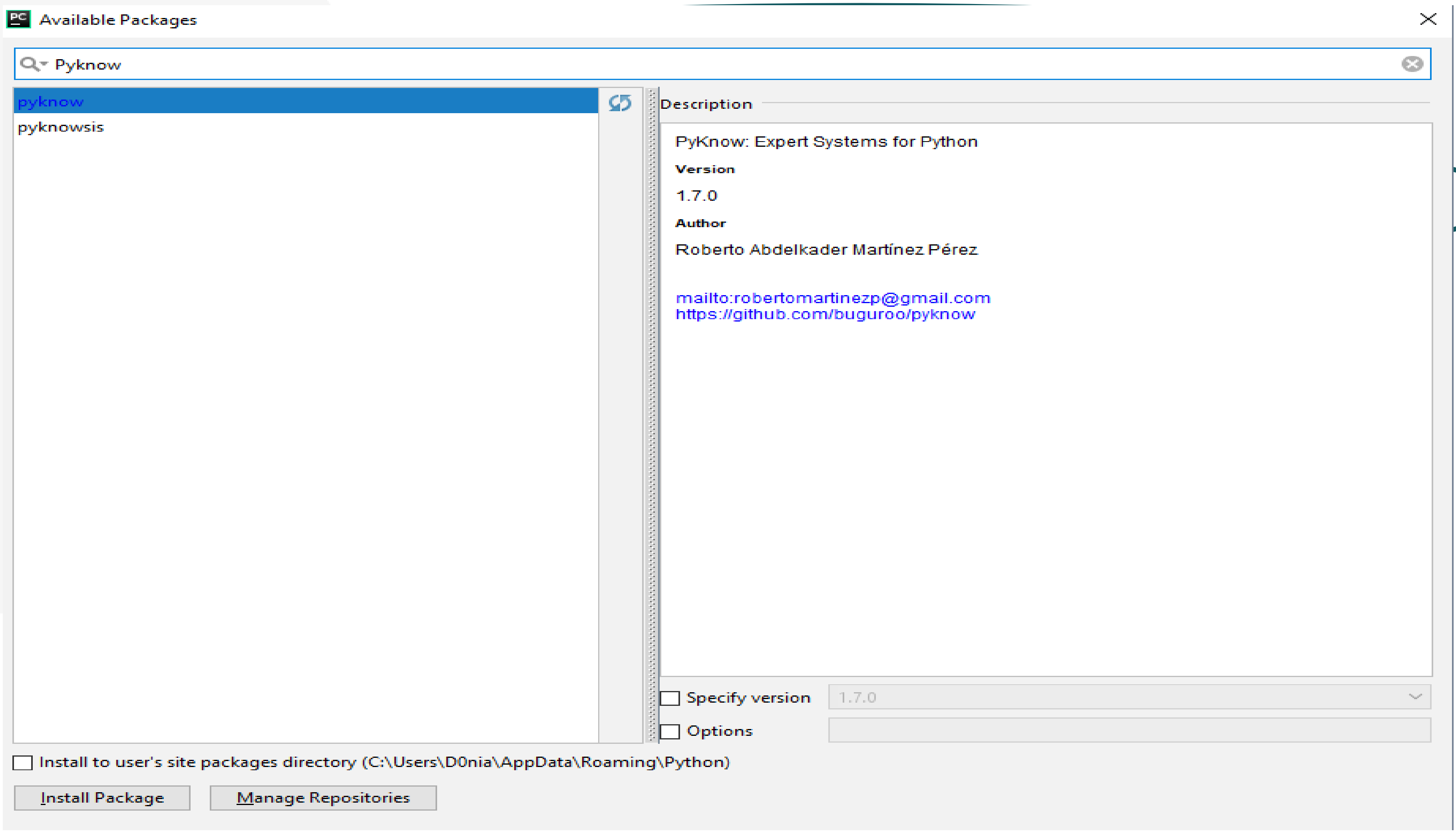
How to install a Package

Packages are imported to use third party code.
(same as library usage)

You can install python package through:

- Pycharm Wizard
- Command Prompt

How to install a package using Pycharm Wizard



PyKnow: Expert Systems for Python

Version

1.7.0

Author

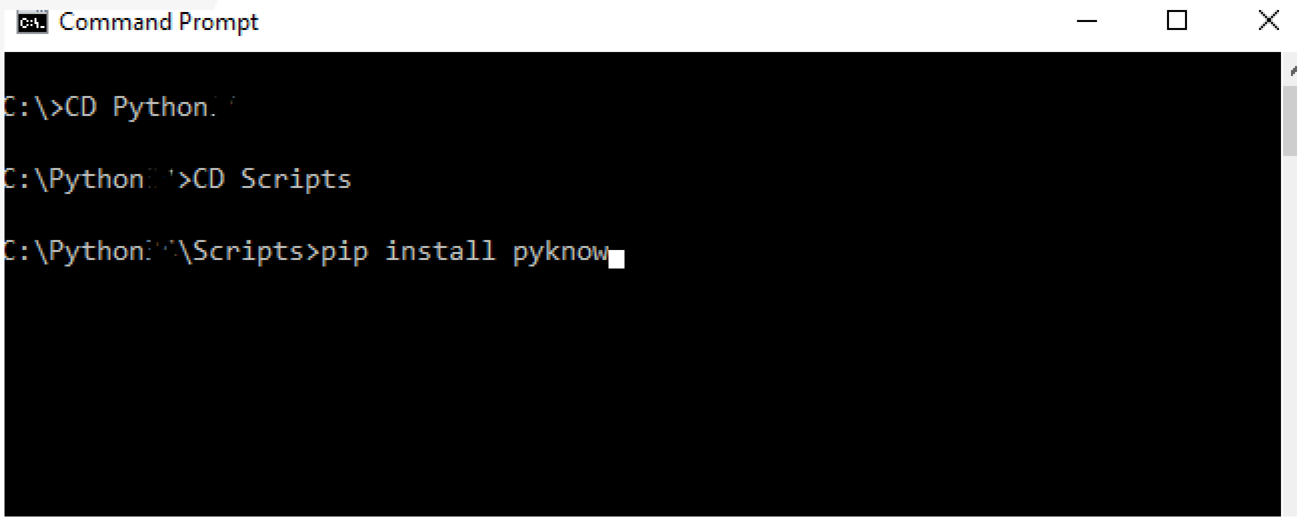
Roberto Abdelkader Martínez Pérez

mailto:robertomartinezp@gmail.com
https://github.com/buguroo/pyknow

Specify version
1.7.0
Options
Install to user's site packages directory (C:\Users\D0nia\AppData\Roaming\Python)
Install Package
Manage Repositories

How to install Package using CMD

1) Run CMD and redirect to python path
2) Type CD Scripts
3) Type pip install PackageName

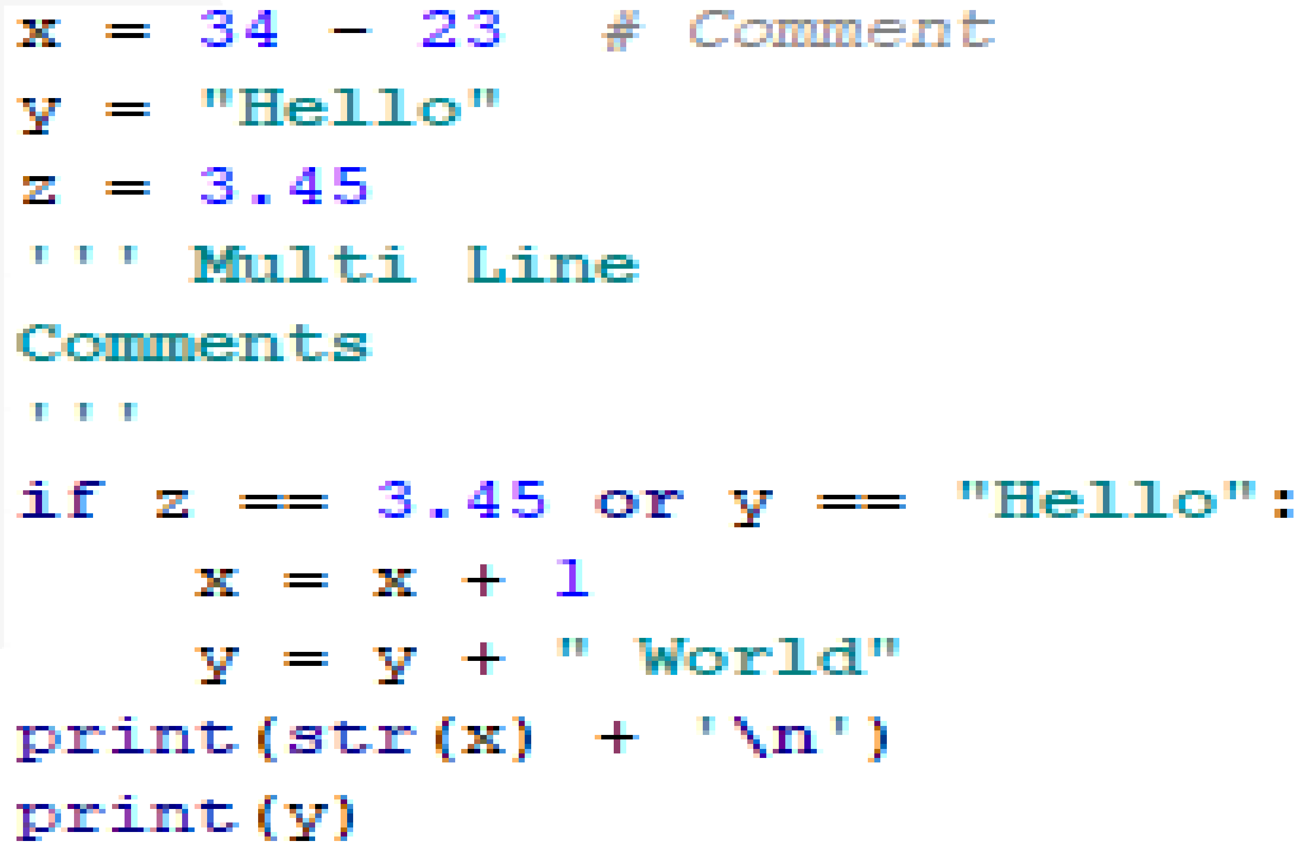


```
C:\>CD Python

C:\Python >CD Scripts

C:\Python \Scripts>pip install pyknow
```

A Code Sample



```
x = 34 - 23  # Comment
y = "Hello"
z = 3.45
''' Multi Line
Comments
'''
if z == 3.45 or y == "Hello":
    x = x + 1
    y = y + " World"
print(str(x) + '\n')
print(y)
```

Enough to understand the code

- Assignment uses **=** and comparison uses **==**
- For numbers **+ - / %** are as expected.
  - Special use of **+** for string concatenation.
  - Special use of **%** for string formatting (as with printf in C).
- Logical operators are words (**and**, **or**, **not**).
- The basic printing command is **`print`**(parameter).
- The first assignment to a variable creates it.
  - Variable types don't need to be declared.
  - Python figures out the variables types on its own.

Basic Data-Types

- Integers

  Z = 5

- Floats

  X = 3.456

- Strings
  - Can use “” or ‘’ to specify.

    “abc” or ‘abc’ (Same thing).

Whitespace - Indentation

- Whitespace is meaningful in Python: especially indentation and placement of newlines.

- No braces { } to mark blocks of code in Python.

- Use consistent indentation instead.
  - The first line with less indentation in outside of block.
  - The first line with more indentation starts a nested block.

Whitespace - Indentation

```
#Works Fine
if 5 > 2:
   print("Five is greater than two!")

#Python will give you an error if you skip the indentation:
if 5 > 2:
print("Five is greater than two!")
```

Comments

Start comment with # the rest of the line will be ignored.

For Multi-Line comment use treble quotes

`""" """.`

```
def myFunction():
    """ this is my function
    Function does ."""
    pass
```

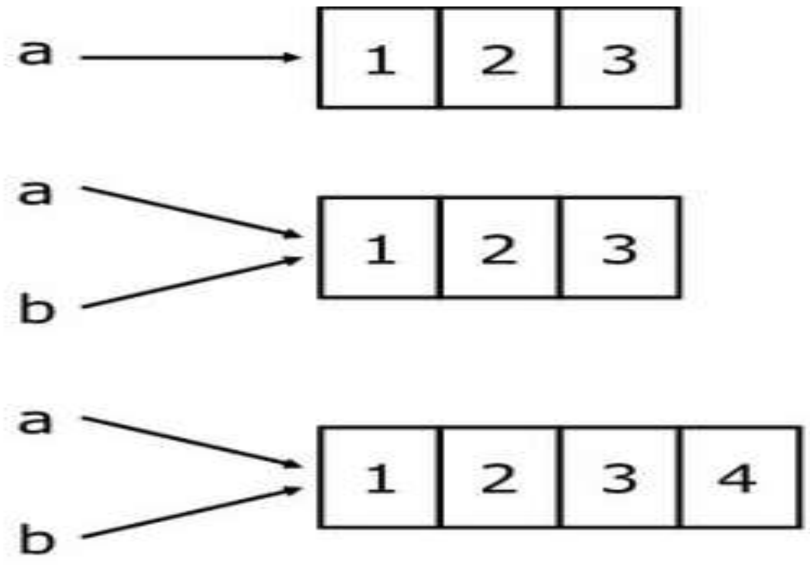
Assignment Operator

- An assignment operator assigns a value to its left operand based on the value of its right operand.

- Collections are assigned by reference.

Example

```
a = [1,2,3] # a now references to the list
[1,2,3]
b = a # b now references to what a references.
a.append(4)
print(b) # prints [1,2,3,4]
```



Example

```
x = 3
y = x
y = 4
print (x) # prints 3, as X is not
affected
```

Casting

- int()
  - constructs an integer number from an integer literal, a float literal (by rounding down to the previous whole number), or a string literal (providing the string represents a whole number)

- Example:

```
x = int(1)   # x will be 1
y = int(2.8) # y will be 2
z = int("3") # z will be 3
```

Casting

- float()
  - constructs a float number from an integer literal, a float literal or a string literal (providing the string represents a float or an integer)

- Example:

```
x = float(1)     # x will be 1.0
y = float(2.8)   # y will be 2.8
z = float("3")   # z will be 3.0
w = float("4.2") # w will be 4.2
```

Casting

- str()
  - constructs a string from a wide variety of data types, including strings, integer literals and float literals

- Example:

```
x = str("s1") # x will be 's1'
y = str(2)    # y will be '2'
z = str(3.0)  # z will be '3.0'
```

String

- String literals in python are surrounded by either single quotation marks, or double quotation marks 'hello' is the same as "hello".

- Strings can be output to screen using the print function.

- For example:

```
print("hello").
```

String functions

```
b = "Hello, World!"
print(b[2:5]) # Prints llo
print(len(b)) # Prints 13
print(b.replace("H", "J")) # Prints Jello World
print("Enter your name:")
x = input() # Gets input from user
print("Hello, " + x)
```

Operators

Python divides the operators in the following groups:

- Arithmetic operators
- Assignment operators
- Comparison operators
- Logical operators
- Identity operators
- Membership operators
- Bitwise operators

Arithmetic operators

Operator	Name	Example
+	Addition	x + y
-	Subtraction	x - y
*	Multiplication	x * y
/	Division	x / y
%	lus	x % y
**	Exponentiation	x ** y
//	Floor division	x // y

x to the power y

5 / 2 will return 2.5 and 5 // 2 will return 2

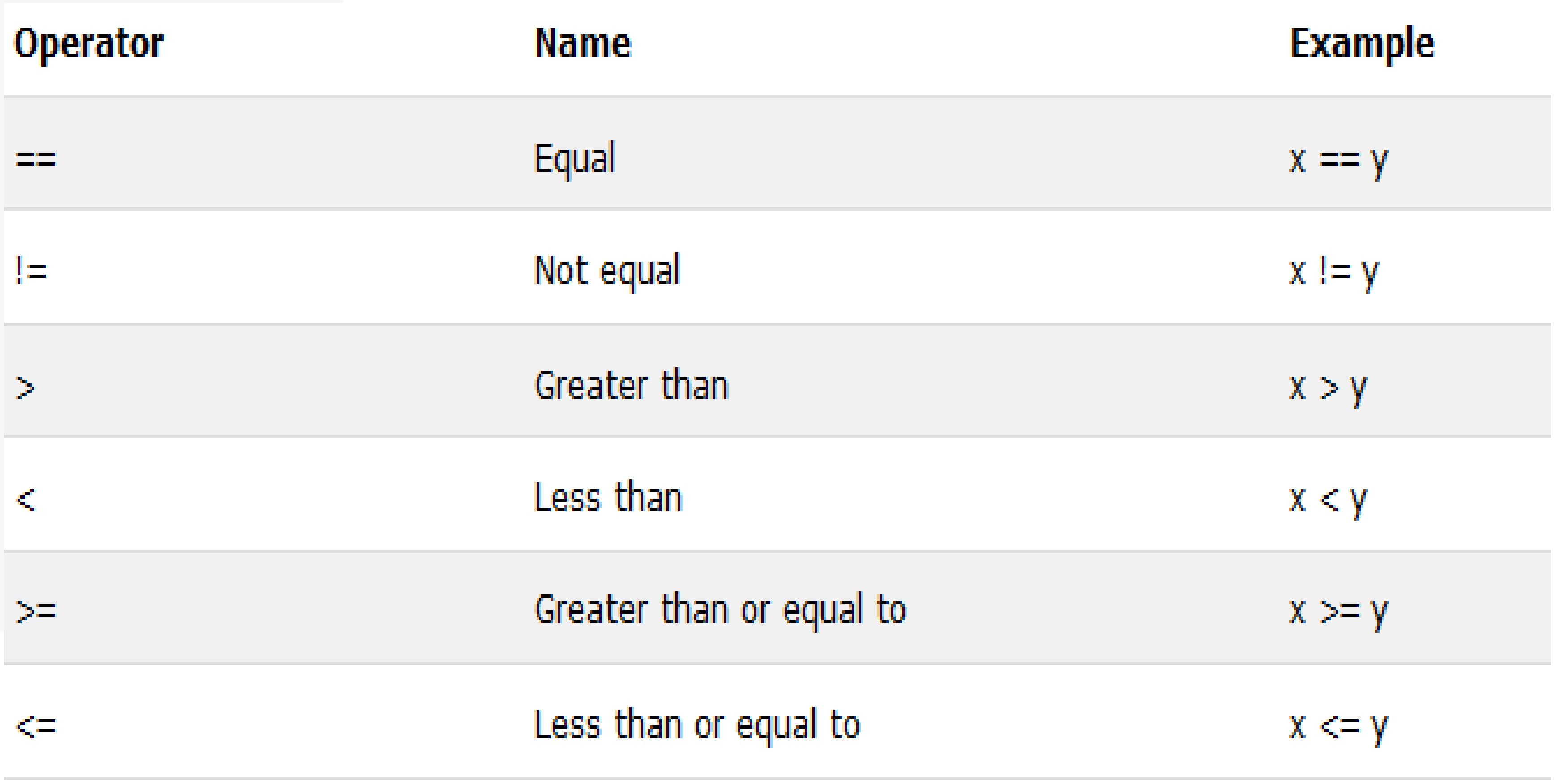
Assignment operators

Operator	Example	Same As
=	x = 5	x = 5
+=	x += 3	x = x + 3
-=	x -= 3	x = x - 3
*=	x *= 3	x = x * 3
/=	x /= 3	x = x / 3
%=	x %= 3	x = x % 3
//=	x //= 3	x = x // 3
**=	x **= 3	x = x ** 3
&=	x &= 3	x = x & 3
|=	x |= 3	x = x | 3
^=	x ^= 3	x = x ^ 3
>>=	x >>= 3	x = x >> 3
<<=	x <<= 3	x = x << 3

Signed Shift Right

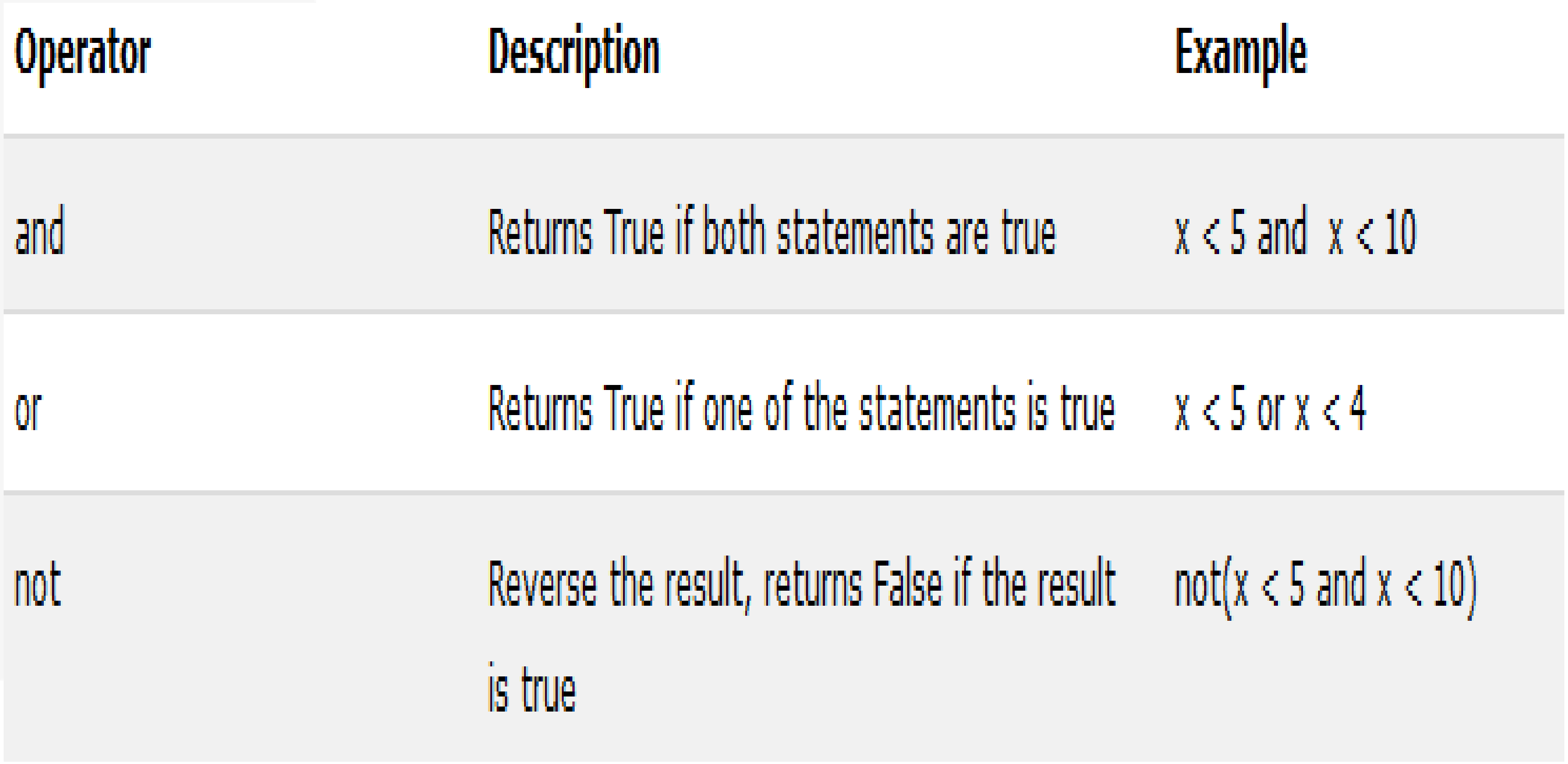
Shift Left

Comparison operators



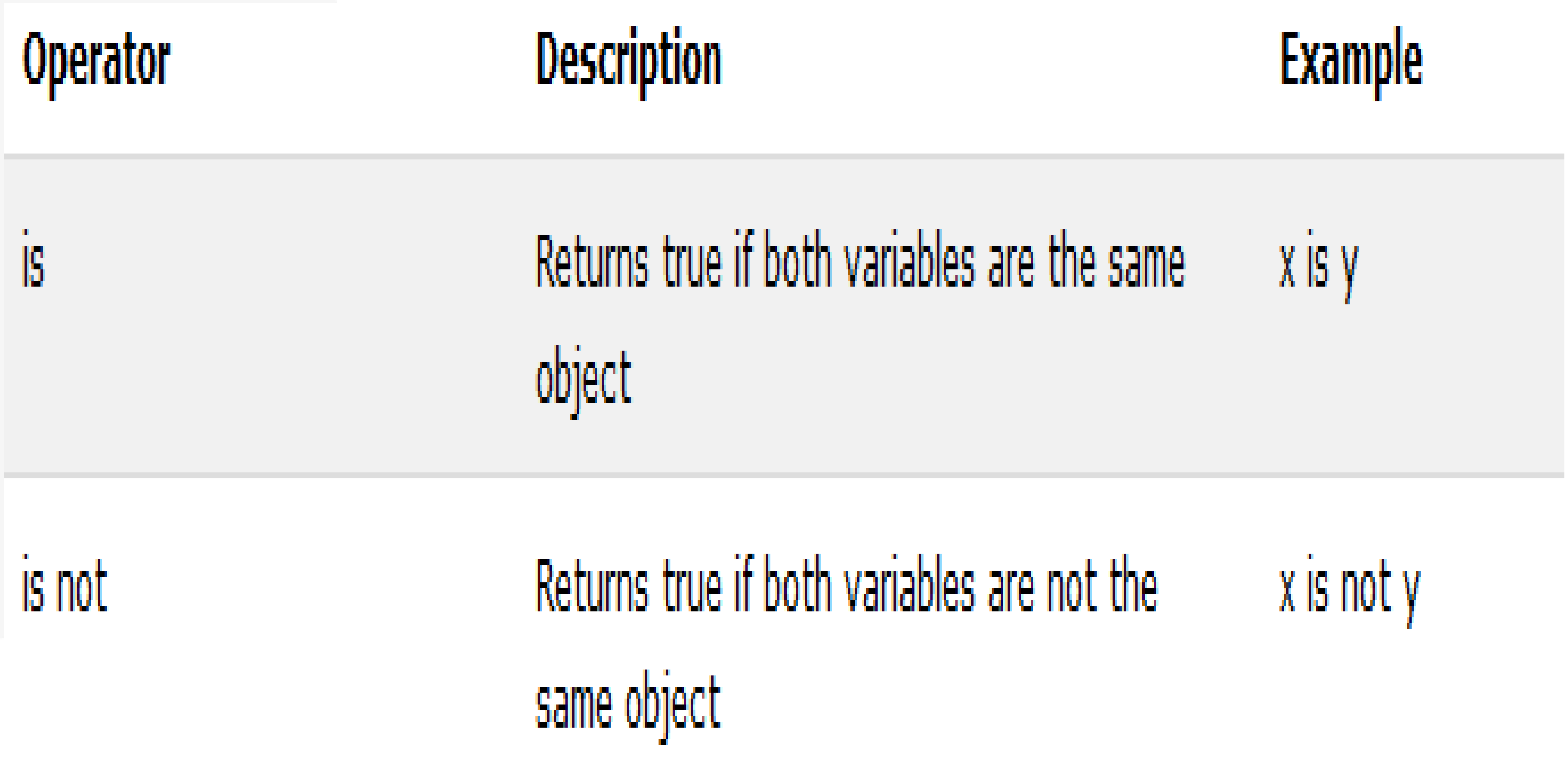
Operator	Name	Example
==	Equal	x == y
!=	Not equal	x != y
>	Greater than	x > y
<	Less than	x < y
>=	Greater than or equal to	x >= y
<=	Less than or equal to	x <= y

Logical Operators



Operator	Description	Example
and	Returns True if both statements are true	x < 5 and x < 10
or	Returns True if one of the statements is true	x < 5 or x < 4
not	Reverse the result, returns False if the result is true	not(x < 5 and x < 10)

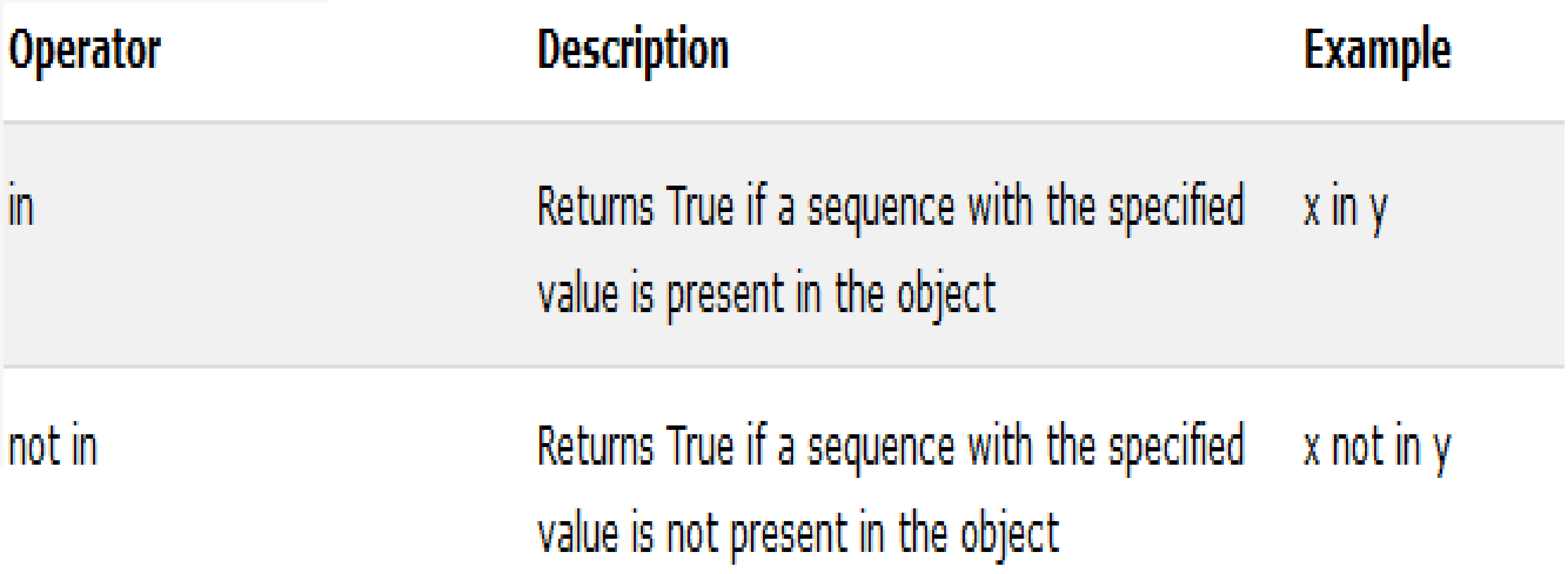
Identity Operators



Operator	Description	Example
is	Returns true if both variables are the same object	x is y
is not	Returns true if both variables are not the same object	x is not y

Membership Operators

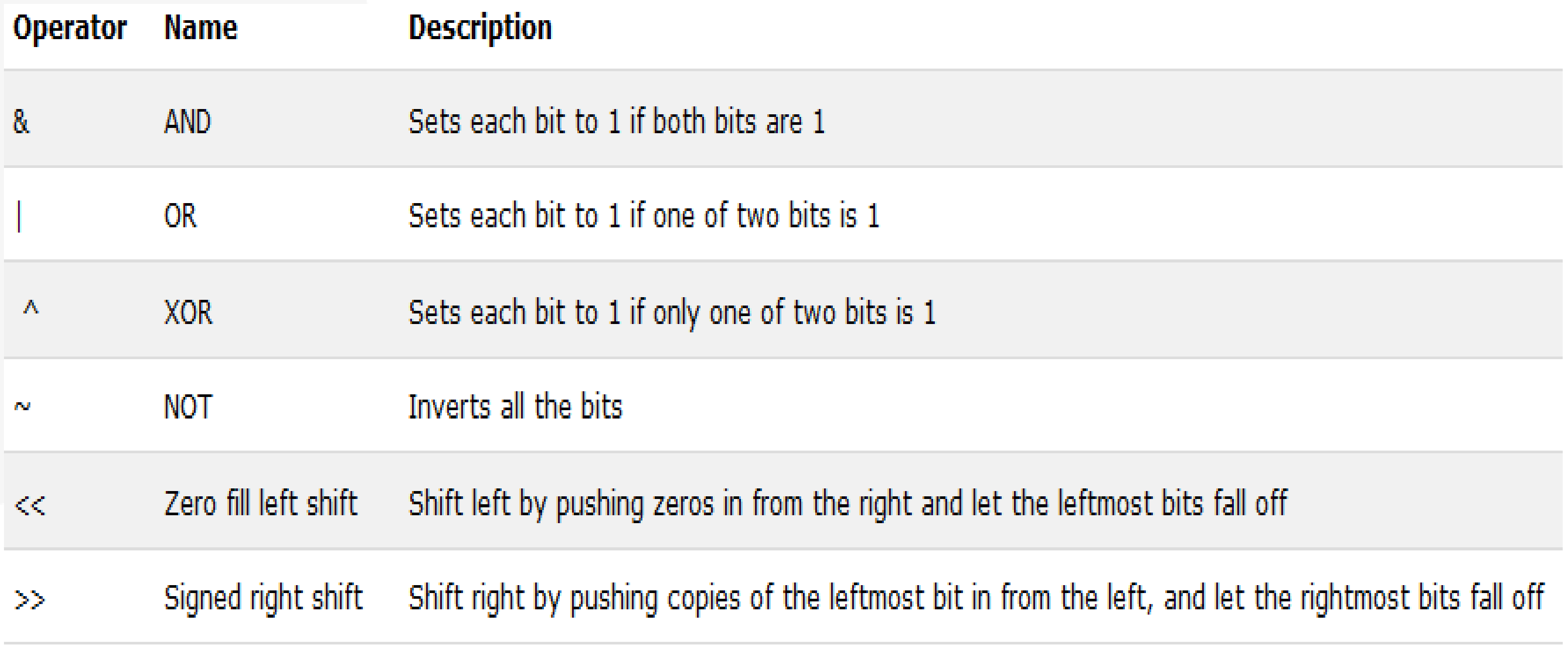
Membership operators are used to test if a sequence is presented in an object:



Operator	Description	Example
in	Returns True if a sequence with the specified value is present in the object	x in y
not in	Returns True if a sequence with the specified value is not present in the object	x not in y

Bitwise Operators

Bitwise operators are used to compare (binary) numbers:



Operator	Name	Description
&	AND	Sets each bit to 1 if both bits are 1
|	OR	Sets each bit to 1 if one of two bits is 1
^	XOR	Sets each bit to 1 if only one of two bits is 1
~	NOT	Inverts all the bits
<<	Zero fill left shift	Shift left by pushing zeros in from the right and let the leftmost bits fall off
>>	Signed right shift	Shift right by pushing copies of the leftmost bit in from the left, and let the rightmost bits fall off

If ... Else

```
a = 200
b = 33
if b > a or b >= a:
  print("b is greater than a")
elif a == b:
  print("a and b are equal")
else:
  print("a is greater than b")

print("A") if a > b else print("B")
```

While Loops

```
i = 1
while i < 6:
 print(i)
 i += 1
 if(i == 3):
       break
 else:
       continue
```

For Loops

```
fruits = ["apple", "banana", "cherry"] #list
for x in fruits:
  if x == "banana":
    continue
  print(x)

#The range() function returns a sequence of numbers,
starting from 0 by default, and increments by 1 (by
default), and ends at a specified number.
for x in range(6): #From 0 to 5
 print(x)
```

Hands On - Check Password

A website requires the users to input username and password to register. Write a program to check the validity of password input by users.

Following are the criteria for checking the password:

- 1. At least 1 letter between [a-z]
- 2. At least 1 number between [0-9]
- 3. At least 1 letter between [A-Z]
- 4. At least 1 character from [$#@]
- 5. Minimum length of password: 6 and Maximum length of password: 12

Hint: use ‘import re’ for regex checking:
example: re.search("[^A-Za-z0-9$#@]", p)

Example:

If the following password is given as input to the program:

ABd1234@1

Then, the output of the program should be: Accepted Password

If the following password is given as input to the program:

2We3345*#

Then, the output of the program should be: Rejected Password

Solution

Hands On

Write a program that accepts three numbers as input and sorts them in descending order.

Input numbers are separated by a space.

```
Input three integers:
30 40 35
After sorting the said integers:
40 35 30
```

Solution

Hands On

Write a program to check a triangle is equilateral, isosceles or scalene. Go to the editor

Note :

An equilateral triangle is a triangle in which all three sides are equal.

A scalene triangle is a triangle that has three unequal sides.

An isosceles triangle is a triangle with (at least) two equal sides.

```
Input lengths of the triangle sides:
x: 6
y: 8
z: 12
Scalene triangle
```

Solution

Staff E-Mails:

- Donia Gamal: donia.gamaleldin@cis.asu.edu.eg
- Zeina Rayan: zeinarayan@hotmail.com
- Salma Elgayar: salma.elgayar@cis.asu.edu.eg
- Sarah Osama: sarah.osama@cis.asu.edu.eg
- Asmaa Bahy: abk84.ak@gmail.com
- Marwa Salah: Marwa_salim@cis.asu.edu.eg
- Doaa Ezzat: doaa.ezzat@gmail.com

Questions?