Class 1
Python Programming

Getting Started — Installation

aCubeIT
Institute of Artificial Intelligence

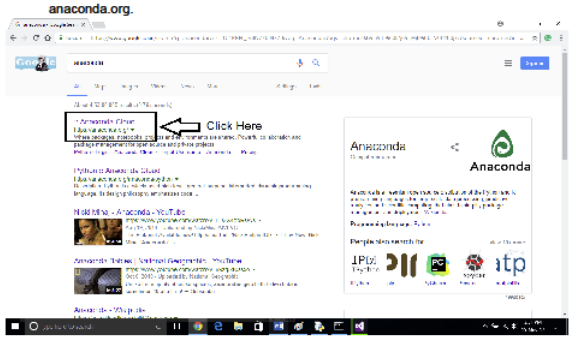
- Installing Anaconda (mandatory for continuing with the course)
- Anaconda can be downloaded from anaconda.org and can be installed like any other normal software.
- Make sure you select Python **3.x** while downloading Anaconda**.**
- Instructions for installing Anaconda

Getting Started — Installation

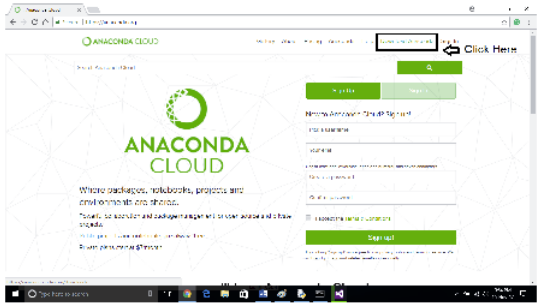
aCubeIT
Institute of Artificial Intelligence

Installing Python with Anaconda(Recommended)

- Search Anaconda on your favorite search engine and open link with address anaconda.org.



- Hover over Download Anaconda on the Anaconda homepage.



Why Python ?

```c
#include <stdio.h>
int main()
{
    // printf() displays the string inside quotation
    printf("Hello, World!");
    return 0;
}
```

C Code

```java
class A {
public static void main(String args[]){
    System.out.println("Hello World");
}
}
```

JAVA Code

```python
print('Hello World')
```

PYTHON Code

Python Programs

- In computing, a program is a specific set of ordered operations for a computer to perform.

- A program is a **sequence of definitions and commands:**
  - Definitions **evaluated**
  - Commands **executed**

  by Python interpreter in a shell.

Python Objects

aCubeIT
Institute of Artificial Intelligence

- Objects are
  - Scalar (cannot be subdivided)

    Example: int, float

  - Non-scalar (have internal structure that can be accessed)

    Example: Employee, Student

- Programs manipulate data objects

Python Objects: Scalar

- **int** - represent integers, Ex. 5
- **float** - represent real numbers, Ex. 3.27
- **bool** - represent Boolean values True and False
- **NoneType** – special and has one value, None
- Can use **type()** to see the type of an object

Scalar Objects: Examples

```
print('Hello World')
```

```
Hello World
```

```
type(5)
```

```
int
```

```
type('Start')
```

```
str
```

```
type(3.0)
```

```
float
```

Objects: Type Conversions

aCubeIT
Institute of Artificial Intelligence

- Can convert object of one type to another
- Example: float object to int and visa versa

```
float(3)
```

```
3.0
```

```
int(7.333)
```

```
7
```

```
str(3)
```

```
'3'
```

Python: Printing to Console

aCubeIT
Institute of Artificial Intelligence

- To show output from code to a user, use print command

```
In [8]: int(3+2)

Out[8]: 5
```

  - **Out[8]** tells you its an interaction within the shell only

```
In [9]: print(3+2)
5
```

  - No **Out[8]** means it is actually shown to a user, apparent when you edit / run files)

Python: Printing to Console

aCubeIT
Institute of Artificial Intelligence

- What happens here?

```
In [10]: 3+5
         print(3+2)

5
```

Python: Expressions

aCubeIT
Institute of Artificial Intelligence

- Combine **objects** and **operators** to form expressions
- An expression has a **value**, which has a **type**

- Syntax for a simple expression

  <object> <operator> <object>

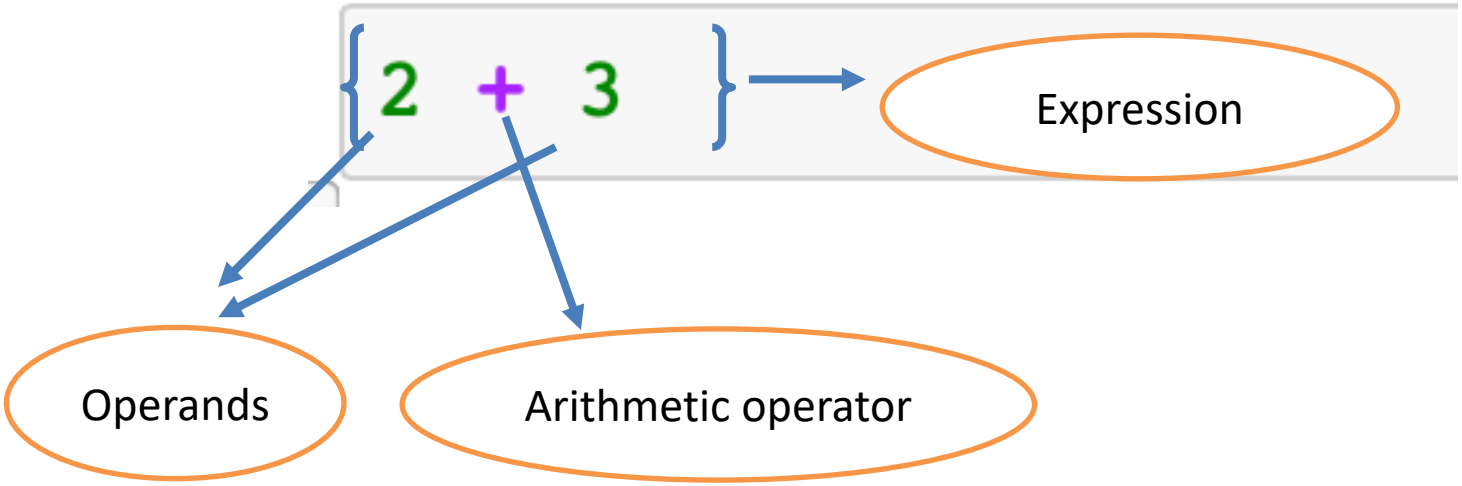
- **What are operators?**

aCubeIT
Institute of Artificial Intelligence

OPERATORS

Operators:

- **Operators** are special **symbols** in Python that **carry out arithmetic or logical computation**.
- The **value** that the operator operates on is called the **operand**.



Python: Basic Operators

- Python language supports the following types of operators.
  - **Arithmetic Operators**
  - **Comparison (Relational) Operators**
  - **Assignment Operators**
  - **Logical Operators**
  - Bitwise Operators
  - Membership Operators
  - Identity Operators
- Let us have a look into highlighted operators one by one.

Python: Arithmetic Operators

aCubeIT
Institute of Artificial Intelligence

Operator	Description	Example
+ Addition	Adds values on either side of the operator.	a + b = 30
- Subtraction	Subtracts right hand operand from left hand operand.	a – b = -10
* Multiplication	Multiplies values on either side of the operator	a * b = 200
/ Division	Divides left hand operand by right hand operand	b / a = 2

Python: Arithmetic Operators

Operator	Description	Example
% Modulus	Divides left hand operand by right hand operand and returns remainder	b % a = 0
** Exponent	Performs exponential (power) calculation on operators	a**b =10 to the power 20
//	Floor Division - The division of operands where the result is the quotient in which the digits after the decimal point are removed. But if one of the operands is negative, the result is rounded away from zero (towards negative infinity) −	9//2 = 4 and 9.0//2.0 = 4.0, -11//3 = -4, -11.0//3 = -4.0

Python: Arithmetic Operators

- **Data types of outputs** when arithmetic operations are performed on int and float

  - i + j —> the sum (if both are **int**, result is **int**)
  - i - j —> the difference (if **either or both** are **floats**, result is **float**)
  - i * j —> the product (if **either or both are floats**, result is **float**)
  - i / j —> division (result is **float**)

Examples: Arithmetic Operators

```
a = 21
b = 10

c = a + b
print("Addition - Value of c is ", c)

c = a - b
print("Subtraction - Value of c is ", c)

c = a * b
print("Multiplication - Value of c is ", c)
```

Examples: Arithmetic Operators

```
c = a / b
print("Division - Value of c is ", c)

c = a % b
print("Remainder - Value of c is ", c)

a = 2
b = 3
c = a**b
print("Power- Value of c is ", c)
```

Examples: Arithmetic Operators

```
a = 10

b = 5

c = a//b

print("Quotient - Value of c is ", c)
```

Result:

Addition - Value of c is 31

Subtraction - Value of c is 11

Multiplication - Value of c is 210

Division - Value of c is 2.1

Remainder - Value of c is 1

Power- Value of c is 8

Quotient - Value of c is 2

Python: Relational Operators

aCubeIT
Institute of Artificial Intelligence

Operator	Description	Example
==	If the values of two operands are equal, then the condition becomes true.	(a == b) is not true.
!=	If values of two operands are not equal, then condition becomes true.	(a != b) is true.
>	If the value of left operand is greater than the value of right operand, then condition becomes true.	(a > b) is not true.

Python: Relational Operators

aCubeIT
Institute of Artificial Intelligence

Operator	Description	Example
<	If the value of left operand is less than the value of right operand, then condition becomes true.	(a < b) is true.
>=	If the value of left operand is greater than or equal to the value of right operand, then condition becomes true.	(a >= b) is not true.
<=	If the value of left operand is less than or equal to the value of right operand, then condition becomes true.	(a <= b) is true.

Examples: Relational Operators

- Let us take

  a = 5

  b = 6

- Find the result for: (Comparison evaluates to a **Boolean**)
  - a == b
  - a != b
  - a > b
  - a == b **(*Result*: False)**
  - a != b **(*Result*: True)**
  - a > b **(*Result*: False)**

Python: Assignment Operators

aCubeIT
Institute of Artificial Intelligence

Operator	Description	Example
=	Assigns values from right side operands to left side operand	c = a + b assigns value of a + b into c
+= Add AND	It adds right operand to the left operand and assign the result to left operand	c += a is equivalent to c = c + a
-= Subtract AND	It subtracts right operand from the left operand and assign the result to left operand	c -= a is equivalent to c = c - a
*= Multiply AND	It multiplies right operand with the left operand and assign the result to left operand	c *= a is equivalent to c = c * a
/= Divide AND	It divides left operand with the right operand and assign the result to left operand	c /= a is equivalent to c = c / ac /= a is equivalent to c = c / a

Python: Assignment Operators

aCubeIT
Institute of Artificial Intelligence

Operator	Description	Example
%= Modulus AND	It takes modulus using two operands and assign the result to left operand	c %= a is equivalent to c = c % a
**= Exponent AND	Performs exponential (power) calculation on operators and assign value to the left operand	c **= a is equivalent to c = c ** a
//= Floor Division	It performs floor division on operators and assign value to the left operand	c //= a is equivalent to c = c // a

Examples: Assignment Operators


```
a = 21
b = 10
c = 0

c = a + b
print("Addition       - Value of c is ", c)

c += a
print("Addition       - Value of c is ", c)

c *= a
print("Multiplication - Value of c is ", c)
```

Examples: Assignment Operators


```
c /= a
print("Division       - Value of c is ", c)

c  = 2
c %= a
print("Remainder      - Value of c is ", c)

c **= a
print("Power          - Value of c is ", c)

c //= a
print("Quotient       - Value of c is ", c)
```

Examples: Assignment Operators

- **Result**

  Addition - Value of c is 31

  Addition - Value of c is 52

  Multiplication - Value of c is 1092

  Division - Value of c is 52.0

  Remainder - Value of c is 2

  Power - Value of c is 2097152

  Quotient - Value of c is 99864

Python Logical Operators

aCubeIT
Institute of Artificial Intelligence

Operator	Description	Example
and Logical AND	If both the operands are true then condition becomes true.	(a and b) is true.
or Logical OR	If any of the two operands are non-zero then condition becomes true.	(a or b) is true.
not Logical NOT	Used to reverse the logical state of its operand.	Not(a and b) is false.

BINDING VALUES AND VARIABLES

Binding variables and values

- Equal sign is an assignment of a value to a variable name.
  - pi = 3.14159
  - pi_approx = 22/7
- **Value stored in computer memory**
- An assignment **binds name to value**
- Retrieve value associated with name or variable by invoking the name. **Ex: by typing pi**.

Why give names to expressions

- To reuse names instead of values.
- With meaningful names it is easier to understand and change code later.

Good Programming Practice:

Always use meaningful names for variables to convey the intent / purpose of the variable.

Changing bindings

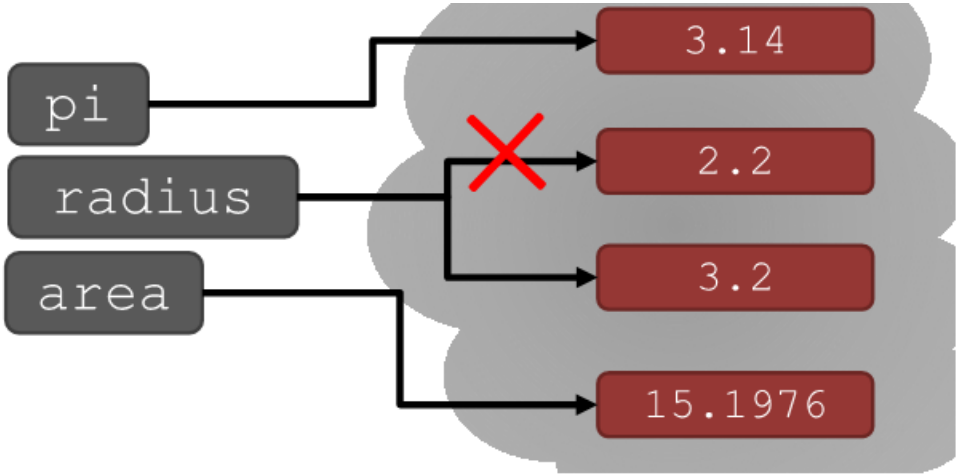
- In the following assignment example

```
pi = 3.14159 radius = 2.2
# area of circle
area = pi * (radius ** 2)
radius = radius+1
```

- What happens when **radius** value is changed?

Changing bindings

- Can re-bind variable names using new assignment statements.
- Previous value may still stored in memory but lost the handle for it.



- What happens to the **value of area?**
  - **Value of area doesn't change**

Exercise: Changing bindings

- What will this print?

```
x = 3
y = x + 2
y = 2*y
x = y - x
print(x, y)
```

- **Result: 7,10**

Input/Output : print()

- Used to output stuff to console
- Keyword is **print**

Input/Output :

```
input('Enter an Integer')
```

```
Enter an Integer
```

- Prints whatever is in the quotes
- User types in something and hits enter
- Binds that value to a variable

Examples: input/output statements

- Print statement: You have already seen the use of print statement in the examples of operators.
  - **x = 1**
  - **print(x)**
- Input statement takes input from console and binds the value to the assigned variable
  - **text = input("Type anything:")**
  - **print(text)**