1. **Readable:** Python is a very readable language.

2. **Easy to Learn:** Learning python is easy as this is a expressive and high level programming language, which means it is easy to understand the language and thus easy to learn.

3. **Cross platform:** Python is available and can run on various operating systems such as Mac, Windows, Linux, Unix etc. This makes it a cross platform and portable language.

4. **Open Source:** Python is a open source programming language.

5. **Large standard library:** Python comes with a large standard library that has some handy codes and functions which we can use while writing code in Python.

6. **Free:** Python is free to download and use. This means you can download it for free and use it in your application. See: [Open Source Python License](https://docs.python.org/3/license.html). Python is an example of a FLOSS (Free/Libre Open Source Software), which means you can freely distribute copies of this software, read its source code and modify it.

7. **Supports exception handling:** If you are new, you may wonder what is an exception? An exception is an event that can occur during program exception and can disrupt the normal flow of program. Python supports exception handling which means we can write less error prone code and can test various scenarios that can cause an exception later on.

8. **Advanced features:** Supports generators and list comprehensions. We will cover these features later.

9. **Automatic memory management:** Python supports automatic memory management which means the memory is cleared and freed automatically. You do not have to bother clearing the memory.

## What Can You Do with Python?

You may be wondering what all are the applications of Python. There are so many applications of Python, here are some of the them.  
1. Web development – Web framework like Django and Flask are based on Python. They help you write server-side code which helps you manage database, write backend programming logic, mapping urls etc.

2. Machine learning – There are many machine learning applications written in Python. Machine learning is a way to write a logic so that a machine can learn and solve a particular problem on its own. For example, products recommendation in websites like Amazon, Flipkart, eBay etc. is a machine learning algorithm that recognises user’s interest. Face recognition and Voice recognition in your phone is another example of machine learning.

3. Data Analysis – Data analysis and data visualisation in form of charts can also be developed using Python.

4. Scripting – Scripting is writing small programs to automate simple tasks such as sending automated response emails etc. Such type of applications can also be written in Python programming language.

5. Game development – You can develop games using Python.

6. You can develop Embedded applications in Python.

7. Desktop applications – You can develop desktop application in Python using library like TKinter or QT.

Python syntax

print("Hello, World!")

python comments

#This is a comment

print("Hello, World!")

print("Hello, World!") #This is a comment

Multiline comments

#This is a comment  
#written in  
#more than just one line  
print("Hello, World!")

"""  
This is a comment  
written in  
more than just one line  
"""  
print("Hello, World!")

Creating Variables

x = 5  
y = "John"  
print(x)  
print(y)

Variables do not need to be declared with any particular type and can even change type after they have been set.

x = 4 # x is of type int  
x = "Sally" # x is now of type str  
print(x)

String variables can be declared either by using single or double quotes:

x = "John"  
# is the same as  
x = 'John'

Python allows you to assign values to multiple variables in one line:

x, y, z = "Orange", "Banana", "Cherry"  
print(x)  
print(y)  
print(z)

And you can assign the same value to multiple variables in one line:

x = y = z = "Orange"  
print(x)  
print(y)  
print(z)

Output Variables

The Python print statement is often used to output variables.

To combine both text and a variable, Python uses the + character:

x = "awesome"  
print("Python is " + x)

You can also use the + character to add a variable to another variable:

x = "Python is "  
y = "awesome"  
z =  x + y  
print(z)

For numbers, the + character works as a mathematical operator:

x = 5  
y = 10  
print(x + y)

If you try to combine a string and a number, Python will give you an error:

x = 5  
y = "John"  
print(x + y)

Create a variable outside of a function, and use it inside the function

x = "awesome"  
  
def myfunc():  
  print("Python is " + x)  
  
myfunc()

x = "awesome"  
  
def myfunc():  
  x = "fantastic"  
  print("Python is " + x)  
  
myfunc()  
  
print("Python is " + x)

Built-in Data Types

In programming, data type is an important concept.

Variables can store data of different types, and different types can do different things.

Python has the following data types built-in by default, in these categories:

|  |  |
| --- | --- |
| Text Type: | str |
| Numeric Types: | int, float, complex |
| Sequence Types: | list, tuple, range |
| Mapping Type: | dict |
| Set Types: | set, frozenset |
| Boolean Type: | bool |
| Binary Types: | bytes, bytearray, memoryview |

Print the data type of the variable x:

x = 5  
print(type(x))

Setting the Data Type

In Python, the data type is set when you assign a value to a variable:

|  |  |
| --- | --- |
| **Example** | **Data Type** |
| x = "Hello World" | str |
| x = 20 | int |
| x = 20.5 | float |
| x = 1j | complex |
| x = ["apple", "banana", "cherry"] | list |
| x = ("apple", "banana", "cherry") | tuple |
| x = range(6) | range |
| x = {"name" : "John", "age" : 36} | dict |
| x = {"apple", "banana", "cherry"} | set |
| x = frozenset({"apple", "banana", "cherry"}) | frozenset |
| x = True | bool |
| x = b"Hello" | bytes |
| x = bytearray(5) | bytearray |
| x = memoryview(bytes(5)) | memoryview |

## Strings are Arrays

Get the character at position 1 (remember that the first character has the position 0):

a = "Hello, World!"  
print(a[1])

## Slicing

You can return a range of characters by using the slice syntax.

Specify the start index and the end index, separated by a colon, to return a part of the string.

Get the characters from position 2 to position 5 (not included):

b = "Hello, World!"  
print(b[2:5])

Negative Indexing

Use negative indexes to start the slice from the end of the string:

Get the characters from position 5 to position 1 (not included), starting the count from the end of the string:

b = "Hello, World!"  
print(b[-5:-2])

## String Length

To get the length of a string, use the len() function.

a = "Hello, World!"  
print(len(a))

## String Methods

The strip() method removes any whitespace from the beginning or the end:

a = " Hello, World! "  
print(a.strip()) # returns "Hello, World!"

The lower() method returns the string in lower case:

a = "Hello, World!"  
print(a.lower())

The upper() method returns the string in upper case:

a = "Hello, World!"  
print(a.upper())

The replace() method replaces a string with another string:

a = "Hello, World!"  
print(a.replace("H", "J"))

The split() method splits the string into substrings if it finds instances of the separator:

a = "Hello, World!"  
print(a.split(",")) # returns ['Hello', ' World!']

## Check String

txt = "The rain in Spain stays mainly in the plain"  
x = "ain" in txt  
print(x)

## String Concatenation

a = "Hello"  
b = "World"  
c = a + b  
print(c)

To add a space between them, add a " ":

a = "Hello"  
b = "World"  
c = a + " " + b  
print(c)

## String Format

age = 36  
txt = "My name is John, I am " + age  
print(txt) – error

The format() method takes the passed arguments, formats them, and places them in the string where the placeholders {} are:

age = 36  
txt = "My name is John, and I am {}"  
print(txt.format(age))

The format() method takes unlimited number of arguments, and are placed into the respective placeholders:

quantity = 3  
itemno = 567  
price = 49.95  
myorder = "I want {} pieces of item {} for {} dollars."  
print(myorder.format(quantity, itemno, price))

You can use index numbers {0} to be sure the arguments are placed in the correct placeholders:

quantity = 3  
itemno = 567  
price = 49.95  
myorder = "I want to pay {2} dollars for {0} pieces of item {1}."  
print(myorder.format(quantity, itemno, price))

## Escape Character

txt = "We are the so-called "Vikings" from the north." – error

txt = "We are the so-called \"Vikings\" from the north."