**SYSTEM REQUIREMENTS**

**HARDWARE REQUIREMENTS:**

HARDWARE REQUIREMENTS:

* Processor : Intel Pentium.
* RAM : 8 GB

**SOFTWARE REQUIREMENTS:**

SOFTWARE REQUIREMENTS:

* OS : Windows 7
* Software : COLAB

**SOFTWARE DESCRIPTION:**

**Python**

Python is one of those rare languages which can claim to be both *simple* and powerful. You will find yourself pleasantly surprised to see how easy it is to concentrate on the solution to the problem rather than the syntax and structure of the language you are programming in. The official introduction to Python is Python is an easy to learn, powerful programming language. It has efficient high-level data structures and a simple but effective approach to object-oriented programming. Python's elegant syntax and dynamic typing, together with its interpreted nature, make it an ideal language for scripting and rapid application development in many areas on most platforms. I will discuss most of these features in more detail in the next section.

## **Features of Python**

### **Simple**

Python is a simple and minimalistic language. Reading a good Python program feels almost like reading English, although very strict English! This pseudo-code nature of Python is one of its greatest strengths. It allows you to concentrate on the solution to the problem rather than the language itself.

### **Easy to Learn**

As you will see, Python is extremely easy to get started with. Python has an extraordinarily simple syntax, as already mentioned.

### **Free and Open Source**

Python is an example of a FLOSS (Free/Libré and Open Source Software). In simple terms, you can freely distribute copies of this software, read its source code, make changes to it, and use pieces of it in new free programs. FLOSS is based on the concept of a community which shares knowledge. This is one of the reasons why Python is so good - it has been created and is constantly improved by a community who just want to see a better Python.

### **High-level Language**

When you write programs in Python, you never need to bother about the low-level details such as managing the memory used by your program, etc.

### **Portable**

Due to its open-source nature, Python has been ported to (i.e. changed to make it work on) many platforms. All your Python programs can work on any of these platforms without requiring any changes at all if you are careful enough to avoid any system-dependent features.

You can use Python on GNU/Linux, Windows, FreeBSD, Macintosh, Solaris, OS/2, Amiga, AROS, AS/400, BeOS, OS/390, z/OS, Palm OS, QNX, VMS, Psion, Acorn RISC OS, VxWorks, PlayStation, Sharp Zaurus, Windows CE and PocketPC!

You can even use a platform like [Kivy](http://kivy.org) to create games for your computer and for iPhone, iPad, and Android.

### **Interpreted**

This requires a bit of explanation.

A program written in a compiled language like C or C++ is converted from the source language i.e. C or C++ into a language that is spoken by your computer (binary code i.e. 0s and 1s) using a compiler with various flags and options. When you run the program, the linker/loader software copies the program from hard disk to memory and starts running it.

Python, on the other hand, does not need compilation to binary. You just run the program directly from the source code. Internally, Python converts the source code into an intermediate form called bytecodes and then translates this into the native language of your computer and then runs it. All this, actually, makes using Python much easier since you don't have to worry about compiling the program, making sure that the proper libraries are linked and loaded, etc. This also makes your Python programs much more portable, since you can just copy your Python program onto another computer and it just works!

### **Object Oriented**

Python supports procedure-oriented programming as well as object-oriented programming. In procedure-oriented languages, the program is built around procedures or functions which are nothing but reusable pieces of programs. In object-oriented languages, the program is built around objects which combine data and functionality. Python has a very powerful but simplistic way of doing OOP, especially when compared to big languages like C++ or Java.

### **Extensible**

If you need a critical piece of code to run very fast or want to have some piece of algorithm not to be open, you can code that part of your program in C or C++ and then use it from your Python program.

### **Embeddable**

You can embed Python within your C/C++ programs to give scripting capabilities for your program's users.

### **Extensive Libraries**

The Python Standard Library is huge indeed. It can help you do various things involving regular expressions, documentation generation, unit testing, threading, databases, web browsers, CGI, FTP, email, XML, XML-RPC, HTML, WAV files, cryptography, GUI (graphical user interfaces), and other system-dependent stuff. Remember, all this is always available wherever Python is installed. This is called the Batteries Included philosophy of Python.

Besides the standard library, there are various other high-quality libraries which you can find at the Python Package Index.

**TESTING PRODUCTS:**

System testing is the stage of implementation, which aimed at ensuring that system works accurately and efficiently before the live operation commence. Testing is the process of executing a program with the intent of finding an error. A good test case is one that has a high probability of finding an error. A successful test is one that answers a yet undiscovered error.

Testing is vital to the success of the system. System testing makes a logical assumption that if all parts of the system are correct, the goal will be successfully achieved. . A series of tests are performed before the system is ready for the user acceptance testing. Any engineered product can be tested in one of the following ways. Knowing the specified function that a product has been designed to from, test can be conducted to demonstrate each function is fully operational. Knowing the internal working of a product, tests can be conducted to ensure that “al gears mesh”, that is the internal operation of the product performs according to the specification and all internal components have been adequately exercised.

**PYTHON and images**

* The help in PYTHON is very good, use it!
* An image in PYTHON is treated as a matrix
* Every pixel is a matrix element
* All the operators in PYTHON defined on Matrices can be used on images: +, -, \*, /, ^, sqrt, sin, cos etc.
* **PYTHON can import/export several image formats**
  + BMP (Microsoft Windows Bitmap)
  + GIF (Graphics Interchange Files)
  + HDF (Hierarchical Data Format)
  + JPEG (Joint Photographic Experts Group)
  + PCX (Paintbrush)
  + PNG (Portable Network Graphics)
  + TIFF (Tagged Image File Format)
  + XWD (X Window Dump)
  + PYTHON can also load raw-data or other types of image data
* **Data types in PYTHON**
  + Double (64-bit double-precision floating point)
  + Single (32-bit single-precision floating point)
  + Int32 (32-bit signed integer)
  + Int16 (16-bit signed integer)
  + Int8 (8-bit signed integer)
  + Uint32 (32-bit unsigned integer)
  + Uint16 (16-bit unsigned integer)
  + Uint8 (8-bit unsigned integer)

**Images in PYTHON**

Binary images: {0, 1}

• Intensity images: [0, 1] or uint8, double etc.

• RGB images: m-by-n-by-3

• Indexed images: m-by-3 color map

• Multidimensional images m-by-n-by-p (p is the number of layers)

**IMAGE TYPES IN PYTHON**

Outside PYTHON images may be of three types i.e. black & white, grey scale and colored. In PYTHON, however, there are four types of images. Black & White images are called binary images, containing 1 for white and 0 for black. Grey scale images are called intensity images, containing numbers in the range of 0 to 255 or 0 to 1. Colored images may be represented as RGB Image or Indexed Image.

In RGB Images there exist three indexed images. First image contains all the red portion of the image, second green and third contains the blue portion. So for a 640 x 480 sized image the matrix will be 640 x 480 x 3. An alternate method of colored image representation is Indexed Image. It actually exist of two matrices namely image matrix and map matrix. Each color in the image is given an index number and in image matrix each color is represented as an index number. Map matrix contains the database of which index number belongs to which color.

**IMAGE TYPE CONVERSION**

* RGB Image to Intensity Image (rgb2gray)
* RGB Image to Indexed Image (rgb2ind)
* RGB Image to Binary Image (im2bw)
* Indexed Image to RGB Image (ind2rgb)
* Indexed Image to Intensity Image (ind2gray)
* Indexed Image to Binary Image (im2bw)
* Intensity Image to Indexed Image (gray2ind)
* Intensity Image to Binary Image (im2bw)
* Intensity Image to RGB Image (gray2ind, ind2rgb)

**Key Features**

* High-level language for technical computing
* Development environment for managing code, files, and data
* Interactive tools for iterative exploration, design, and problem solving
* Mathematical functions for linear algebra, statistics, Fourier analysis, filtering, optimization, and numerical integration
* 2-D and 3-D graphics functions for visualizing data
* Tools for building custom graphical user interfaces

Functions for integrating PYTHON based algorithms with external applications and languages, such as C, C++, FORTRAN, Java, COM, and Microsoft Excel.