

Chapter 1

Introduction 1.1

What is Python? Python is a high-level scripting language which can be used for a wide variety of text processing, system administration and internet-related tasks. Unlike many similar languages, it's core language is very small and easy to master, while allowing the addition of modules to perform a virtually limitless variety of tasks. Python is a true object-oriented language, and is available on a wide variety of platforms. There's even a python interpreter written entirely in Java, further enhancing python's position as an excellent solution for internet-based problems. Python was developed in the early 1990's by Guido van Rossum, then at CWI in Amsterdam, and currently at CNRI in Virginia. In some ways, python grew out of a project to design a computer language which would be easy for beginners to learn, yet would be powerful enough for even advanced users. This heritage is reflected in python's small, clean syntax and the thoroughness of the implementation of ideas like object-oriented programming, without eliminating the ability to program in a more traditional style. So python is an excellent choice as a first programming language without sacrificing the power and advanced capabilities that users will eventually need. Although pictures of snakes often appear on python books and websites, the name is derived from Guido van Rossum's favorite TV show, "Monty Python's Flying Circus". For this reason, lots of online and print documentation for the language has a light and humorous touch. Interestingly, many experienced programmers report that python has brought back a lot of the

1.2 The very Basics of Python

There are a few features of python which are different than other programming languages, and which should be mentioned early on so that subsequent examples don't seem confusing. Further information on all of these features will be provided later, when the topics are covered in depth. Python statements do not need to end with a special character – the python interpreter knows that you are done with an individual statement by the presence of a newline, which will be generated when you press the "Return" key of your keyboard. If a statement spans more than one line, the safest course of action is to use a backslash (\) at the end of the line to let python know that you are going to continue the statement on the next line; you can continue using backslashes on additional continuation lines. (There are situations where the backslashes are not needed which will be discussed later.) Python provides you with a certain level of freedom when composing a program, but there are some rules which must always be obeyed. One of these rules, which some people find very surprising, is that python uses indentation (that is, the amount of white space before the statement itself) to indicate the presence of loops, instead of using delimiters like curly braces ({}) or keywords (like "begin" and "end") as in many other languages. The amount of indentation you use is not important, but it must be consistent within a given depth of a loop, and statements which are not indented must begin in the first column. Most python programmers prefer to use an editor like emacs, which automatically provides consistent indentation; you will probably find it easier to maintain your programs if you use consistent indentation in every loop, at all depths, and an intelligent editor is very useful in achieving this.

1.4 Basic Principles of Python

Python has many features that usually are found only in languages which are much more complex to learn and use. These features were designed into python from its very first beginnings, rather than being accumulated into an end result, as is the case with many other scripting languages. If you're new to programming, even the basic descriptions which follow may seem intimidating. But don't worry – all of these ideas will be made clearer in the chapters which follow. The idea of presenting these concepts now is to make you aware of how python works, and the general philosophy behind python programming. If some of the concepts that are introduced here seem abstract or overly complex, just try to get a general feel for the idea, and the details will be fleshed out later

1.4.1 Basic Core Language

Python is designed so that there really isn't that much to learn in the basic language. For example, there is only one basic structure for conditional programming (if/else/elif), two looping commands (while and for), and a consistent method of handling errors (try/except) which apply to all python programs. This doesn't mean that the language is not flexible and powerful, however. It simply means that you're not confronted with an overwhelming choice of options at every turn, which can make programming a much simpler task.

1.4.2 Modules

Python relies on modules, that is, self-contained programs which define a variety of functions and data types, that you can call in order to do tasks beyond the scope of the basic core language by using the import command. For example, the core distribution of python contains modules for processing files, accessing your computer's operating system and the internet, writing CGI scripts (which handle communicating with pages displayed in web browsers), string handling and many other tasks. Optional modules, available on the Python web site (<http://www.python.org>), can be used to create graphical user interfaces, communicate with data bases, process image files, and so on. This structure makes it easy to get started with python, learning specific skills only as you need them, as well as making python run more efficiently by not always including every capability in every program.

1.4.3 Object Oriented Programming

Python is a true object-oriented language. The term "object oriented" has become quite a popular buzzword; such high profile languages as C++ and Java are both object oriented by design. Many other languages add some object-oriented capabilities, but were not designed to be object oriented from the ground up as python was. Why is this feature important? Object oriented program allows you to focus on the data you're interested in, whether it's employee information, the results of a scientific experiment or survey, setlists for your favorite band, the contents of your CD collection, information entered by an internet user into a search form or shopping cart, and to develop methods to deal efficiently with your data. A basic concept of object oriented programming is

encapsulation, the ability to define an object that contains your data and all the information a program needs to operate on that data. In this way, when you call a function (known as a method in object-oriented lingo), you don't need to specify a lot of details about your data, because your data object "knows" all about itself. In addition, objects can inherit from other objects, so if you or someone else has designed an object that's very close to one you're interested in, you only have to construct those methods which differ from the existing object, allowing you to save a lot of work. Another nice feature of object oriented programs is operator overloading. What this means is that the same operator can have different meanings