# Python workbook

## Complete Python for beginners

## Index

**Module 1- Software setup and First Python Script**

**Module 2- Data types**

**Module 3- Sequence Types**

- Set Type
- Frozen Set
- range type
- bytes and bytearray
- Dictionary
- Immutability
- Map of Lists Usecase

**Module 4- Special Types**

- None type
- Escape chars
- Constants
- del keyword
- Data Types Summary

**Module 5- Operators and Operands**

- Arithmetic Operators
- Assignment Operators
- Comparison Operators
- Logical Operators

**Module 6- Input and Output functions**

- Introduction
- print
- print and string formatting
- input
- reading multiple inputs

**Module 7- More Programs**

- Read and Display student details
- Average of three numbers
- Area of a circle
- Using the math module

**Module 8- Flow control statements**

- Introduction
- If Else Syntax
- Find even or odd number
- IF Else Ladder
- Handle Zero
- If-Else Ladder
- While Syntax
- Display numbers from 1 to 20
- Odd numbers between given numbers
- For Syntax
- using for loop
- product of numbers in a list
- multiplication table of a given number
- break
- continue
- assert
- More Programs - Remove Duplicates in List
- More Programs - Count Vowels in a word
- More Programs - Handle employee Details

**Module 9- More Programs**

- Reverse a String
- Reverse using split and join
- Reverse the words in a string
- Reverse the characters in the words
- Remove Duplicate Chars
- Count the characters
- Print Right Angled Triangle
- Print Pyramid Pattern
- Find substrings in a given string

**Module 10- Command line arguments**

- Introduction
- command line arguments
- command line arguments in PyCharm

- product of command line arguments

**Module 11- Functions**

- Introduction
- first function
- returning a result
- return multiple values
- local and global variables
- accessing global variable with the same name
- assign function to a variable
- function inside another
- function as parameter to an other
- returning functions
- pass any type
- Recursion
- recursion
- keyword arguments
- default arguments
- BMI Usecase
- *args and **kwargs
- passing optionals params to other functions

**Module 12- Lambdas**

- Introduction
- Lambda - Cube of a given number
- Lambda - Even or Odd
- Lambda - Sum of two numbers
- Using a filter
- Using the map function
- Using reduce function
- Decorators
- Decorator that doubles the result of a function
- Using @ Decorator
- Decorating Strings
- Decorator Chaining
- Generators

- Create a generator
- Keywords in python

**Module 13- Modules**

- Introduction
- Create a module and use it
- Different ways to import
- Use Math Module
- Use Random Module

**Module 14- List Comprehensions**

- List Comprehensions
- Cube of numbers in a list
- Even numbers in a list
- Product of numbers in a list
- Common elements in a list

**Module 15- Object Oriented Programming**

- Introduction
- Create first class
- Use Parameterized Constructor
- Define a instance method
- Create Getter and Setter methods
- Define instance methods
- Methods vs Constructors
- Define static field
- Count the number of Objects
- Create a Inner Class
- Garbage Collection
- GC Methods
- Use Destructor
- Patient Clinicals Usecase

**Module 16- Encapsulation**

- Introduction
- Private Fields and Name Mangling

- Implementing Encapsulation

**Module 17-Inheritance**

- Inheritance
- Inheritance in action
- Inheriting Functionality
- Overriding
- Using Super()

**Module 18- Polymorphism**

- Introduction
- DuckTyping
- DuckTyping for Dependency Injection
- Operator Overloading
- Runtime Polymorphism

**Module 19- Polymorphism**

- Abstract Classes and Interfaces
- Create an Abstract Class
- Create an Interface

**Module 20- Exception handling assertions and logging**

- Introduction
- Exception Class Hierarchy
- Handling Exceptions
- Using Finally
- Else
- Create and Raise Custom Exceptions
- More Programs - Custom Exceptions
- Logging in action
- Logging Configuration
- Log Exceptions
- Using Assertions

### Module 21- Files

- Introduction
- Write string to a file
- Read from a file
- Write multiple strings
- Check if the file exists
- Pickle
- UnPickle

### Module 22- Regular Expressions

- Introduction
- Sequence Characters
- search()
- findall() and match()
- split()
- substitute()
- Quantifiers
- using quantifiers
- Matching dates
- Special Characters

### Module 23- Date and Time

- Date and Time
- Time since epoch
- Finding the current date and time
- Combining Date and Time
- Sorting dates
- sleep()
- Knowing the execution time of a program
- Validate Credit Card Usecase
- Project Management Usecase

### Module 24- Threads

- Introduction
- Main Thread
- Thread using a function

- Printing Thread Names
- Thread extending the Thread Class
- Thread using a class
- Multithreading in action
- using sleep()
- The TicketBooking usecase
- Thread Synchronization
- Add more logic
- Synchronization using lock
- Synchronization using semaphore
- Thread Communication
- Using a boolean flag
- Run and summarize
- Thread Communication using wait and notify
- Use wait and notify
- Queues and Thread Communication
- Producer Consumer Pattern
- Three Types of queues
- Types of queues

**Module 25- Networking**

- Introduction
- Downloading a HTML
- Downloading a Image
- Socket Programming
- Create a server
- Create a client
- File Server
- File Client
- Sending Emails
- Email Client
- Run Mail Client

**Module 26- Database Operator**

- Introduction
- Install MySql

- Install MySql workbench
- Install mysql connector
- Setup the database
- Connect to the DB from Python
- READ
- READ - fetchall
- CREATE
- DELETE
- Database Operations

**Module 27- Additional Content – Postgre SQL**

- Introduction
- Setup the database
- Install psycopg2
- Connect and Insert

**Module 28- Work with mongo**

- Mongo Introduction
- Install Mongo
- Use Mongo Console
- Install pymongo
- API walkthrough
- insert one
- insert many
- read
- update
- delete

**Module 29- Debugging Fundamental**

- Usecase
- Debugging in PyDev
- Debugging in PyCharm

**Module 30- Additional Content – Virtual Environment**

- Using Virtual Environment

**Module 31- Additional Content – Unit Testing**

- Introduction
- Create and run a test
- Assert for exceptions
- setUp and tearDown

**Module 32- Additional Content – Numpy**

- Introduction
- Numpy in action
- linespace logspace and more
- Using math functions
- Array Comparison
- More comparison Functions
- Copying arrays
- Slicing
- Multidimensional arrays
- Few more attributes
- Few more functions

## About the Author

Jagdish Rathod, CCNP (RS/SP/SEC), AWS Solution Architect Professional, Palo Alto, Certified is an ambitious and highly experienced senior technical instructor. He has been instructing networking classes for more than 5 years, covering subjects including Cloud, Python, Automation, SDWAN, Routing and Switching, Service Provider, and Security (CCNA to CCNP). Additionally, he has been creating and revising the course material. He has helped other engineers obtain certifications and pass the lab exams.

In addition to developing courses and providing technical workforce training in a variety of networking environments, Jagdish is an expert at designing, planning, coordinating, maintaining, troubleshooting, and implementing changes to various aspects of multi-scaled, multi-platform, multi-protocol complex networks. He has worked on tasks such as running and maintaining PCs and peripherals and running network control programmes for diverse data communication networks in LAN, MAN, and WAN environments.

In addition to providing instructor-led training domestically in numerous Indian states, Jagdish has also done so internationally in the UK, USA, Canada, Australia, Singapore, Malaysia, and UAE. Additionally, he has served as a freelance Cisco Certified Instructor for numerous international corporate clients.

## Acknowledgment

First and foremost, I want to express my gratitude to God for continuing to bless me and for being my support system. You have given me the strength and assurance I need to have faith in myself and go after my goals.Without the trust I have in you, I could never have accomplished this

Second, I want to thank my family for being understanding of my late-night computer work. Without their help and encouragement, I would not have been able to spend as much time creating this workbook.

I also want to thank my students for participating in my trainings and using my workbooks. I think that my workbooks have assisted students in improving their knowledge of the topic and relevant technologies, and I will keep creating workbooks for the latest technological versions.

**Jagdish Rathod**

**M.Tech (CE- Gold Medalist) CCNA/CCNP (RS/SP/SEC), AWS, Paloalto**

## Feedback

Please send feedback if there are any issues with respect to the content of this workbook. I would also appreciate suggestions from you which can improve this workbook further. Kindly send your feedback and suggestions at info@hazynetwork.com

## Who this course is for:

- Any student who wants to learn python in depth
- Developers who are getting started with Python
- Networking professionals who want to learn pythons quickly and in depth
- Experienced developers who want to fill in any gaps in their knowledge

## What you'll learn

- Master the Features of Python Language
- Install Python Virtual Machine and the Eclipse IDE(PyDev)
- Execute your first python program
- Learn various simple types as well as collection types
- Define logic using conditional statements ,looping constructs
- Use the different types of operators
- See the input and output functions in action
- Create and use functions , Lambdas Decorators and Generators
- Learn what Object Oriented Programming is the four OOPs principles
- Implement inheritance, abstraction, polymorphism and encapsulation
- Understand interfaces, their importance, and their uses
- Use abstract classes and interfaces to implement abstraction
- Handle Exceptions
- Read and Write files using the Files API
- Do pattern matching using Regular expressions
- Deal with data and time
- All in simple steps

## Requirements

- Nothing is prerequisite about any programming language
- Internet Connection
- Windows / Linux / Mac OS X Machine knowledge

## Beginners welcome: no need to know anything about Python!

## Exam

Many people are planning to give their exams and rest assured, we have **entire exam preparation section + lots of quizzes** in order for you to practice and be prepared.

If anyone is planning to give exams immediately, please get back along at info@hazynetwork.com and I'll be happy to guide at personal level :)

## PYTHON INTRODUCTION

- Python is an object-oriented programming language created by Guido Rossum in 1989. It is ideally designed for rapid prototyping of complex applications. It has interfaces to many OS system calls and libraries and is extensible to C or C++. Many large companies use the Python programming language include NASA, Google, YouTube, BitTorrent, etc.
- Python programming is widely used in Artificial Intelligence, Natural Language Generation, Neural Networks and other advanced fields of Computer Science. Python had deep focus on code readability & this class will teach you python from basics.
- Python is a general purpose, dynamic, high level and interpreted programming language. It supports Object Oriented programming approach to develop applications. It is simple and easy to learn and provides lots of high-level data structures.
- Python is easy to learn yet powerful and versatile scripting language which makes it attractive for Application Development.
- Python's syntax and dynamic typing with its interpreted nature, makes it an ideal language for scripting and rapid application development

## PYTHON FEATURES

Python provides lots of features that are listed below.

**1) Easy to Learn and Use**

Python is easy to learn and use. It is developer-friendly and high level programming language.

**2) Expressive Language**

Python language is more expressive means that it is more understandable and readable.

**3) Interpreted Language**

Python is an interpreted language i.e. interpreter executes the code line by line at a time. This makes debugging easy and thus suitable for beginners.

**4) Cross-platform Language**

Python can run equally on different platforms such as Windows, Linux, Unix and Macintosh etc. So, we can say that Python is a portable language.

**5) Free and Open Source**

Python language is freely available at offical web address.The source-code is also available. Therefore it is open source.

**6) Object-Oriented Language**

Python supports object oriented language and concepts of classes and objects come into existence.

**7) Extensible**

It implies that other languages such as C/C++ can be used to compile the code and thus it can be used further in our python code.

**8) Large Standard Library**

Python has a large and broad library and provides rich set of module and functions for rapid application development.

**9) GUI Programming Support**

Graphical user interfaces can be developed using Python.

**10) Integrated**

It can be easily integrated with languages like C, C++, JAVA etc.

## PYTHON HISTORY

- Python laid its foundation in the late 1980s.
- The implementation of Python was started in the December 1989 by Guido Van Rossum at CWI(Centrum Wiskunde&Informatica) in Netherland.
- In February 1991, van Rossum published the code (labeled version 0.9.0) to alt.sources.
- In 1994, Python 1.0 was released with new features like: lambda, map, filter, and reduce.

- Python 2.0 added new features like: list comprehensions, garbage collection system.
- On December 3, 2008, Python 3.0 (also called "Py3K") was released. It was designed to rectify fundamental flaw of the language.
- ABC programming language is said to be the predecessor of Python language which was capable of Exception Handling and interfacing with Amoeba Operating System.

## PYTHON APPLICATIONS AREA

Python is known for its general purpose nature that makes it applicable in almost each domain of software development. Python as a whole can be used in anysphere of development.

Here, we are specifying applications areas where python can be applied.

**1) Web Applications**

We can use Python to develop web applications. It provides libraries to handle internet protocols such as HTML and XML, JSON, Email processing, request, beautifulSoup, Feedparser etc. It also provides Frameworks such as Django, Pyramid, Flask etc to design and develop web based applications. Some important developments are: PythonWikiEngines, Pocoo, Python BlogSoftware etc.

**2) Desktop GUI Applications**

Python provides Tk GUI library to develop user interface in python based application. Some other useful toolkits wxWidgets, Kivy, pyqt that are useable on several platforms. The Kivy is popular for writing multitouch applications.

**3) Software Development**

Python is helpful for software development process. It works as a support language and can be used for build control and management, testing etc.

**4) Scientific and Numeric**

Python is popular and widely used in scientific and numeric computing. Some useful library and package are SciPy, Pandas, IPython etc. SciPy is group of packages of engineering, science and mathematics.

**5) Business Applications**

Python is used to build Bussiness applications like ERP and e-commerce systems.

Tryton is a high level application platform.

**6) Console Based Application**

We can use Python to develop console based applications. For example: IPython.

**7) Audio or Video based Applications**

Python is awesome to perform multiple tasks and can be used to develop multimedia applications. Some of real applications are: TimPlayer, cplay etc.

**8) 3D CAD Applications**

To create CAD application Fandango is a real application which provides full features ofCAD.

**9) Enterprise Applications**

Python can be used to create applications which can be used within an Enterprise or an Organization. Some real time applications are: OpenErp, Tryton, Picalo etc.

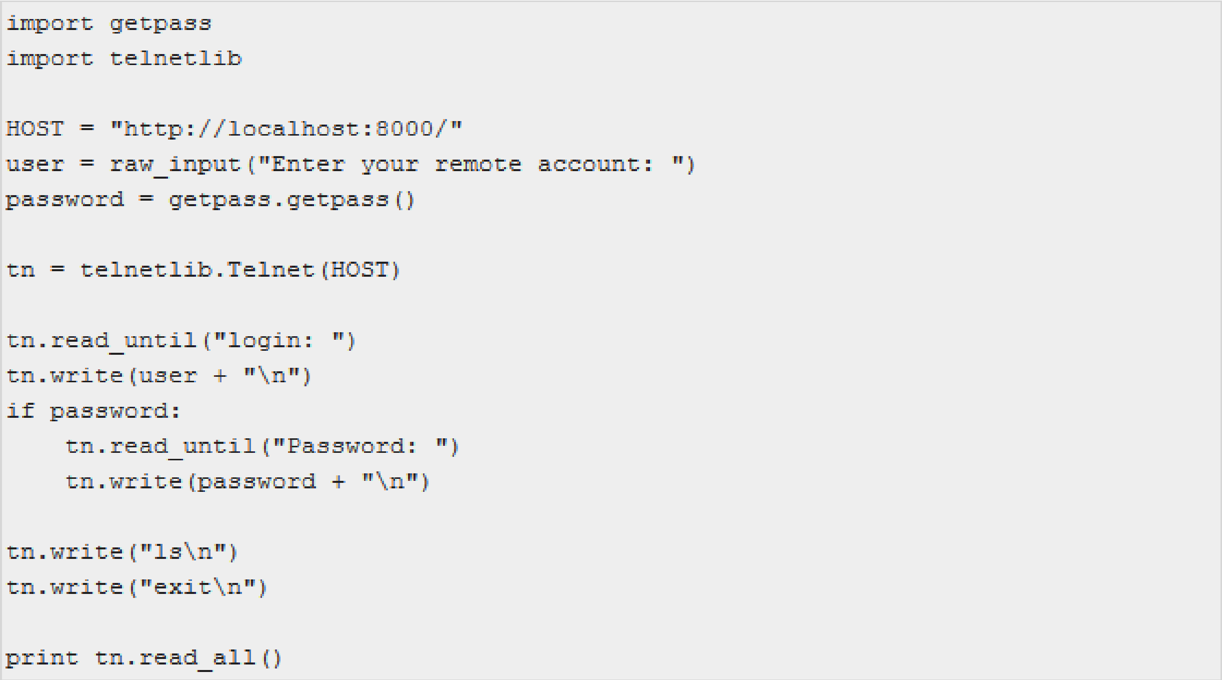
**10) Applications for Images**

Using Python several application can be developed for image. Applications developed are: VPython, Gogh, imgSeek etc.

There are several such applications which can be developed using Python

**11) Now Automation**

## Python - Network Programming Introduction



```
import getpass
import telnetlib

HOST = "http://localhost:8000/"
user = raw_input("Enter your remote account: ")
password = getpass.getpass()

tn = telnetlib.Telnet(HOST)

tn.read_until("login: ")
tn.write(user + "\n")
if password:
    tn.read_until("Password: ")
    tn.write(password + "\n")

tn.write("ls\n")
tn.write("exit\n")

print tn.read_all()
```

## PYTHON VARIABLES

- Variable is a name which is used to refer memory location. Variable also known as identifier and used to hold value.
- In Python, we don't need to specify the type of variable because Python is a type infer language and smart enough to get variable type.
- Variable names can be a group of both letters and digits, but they have to begin with a letter or an underscore.
- It is recommended to use lowercase letters for variable name. Rahul and rahul both are two different variables
- **Note - Variable name should not be a keyword.**
- Declaring Variable and Assigning Values
- Python does not bound us to declare variable before using in the application. It allows us to create variable at required time.
- We don't need to declare explicitly variable in Python. When we assign any value to the variable that variable is declared automatically.

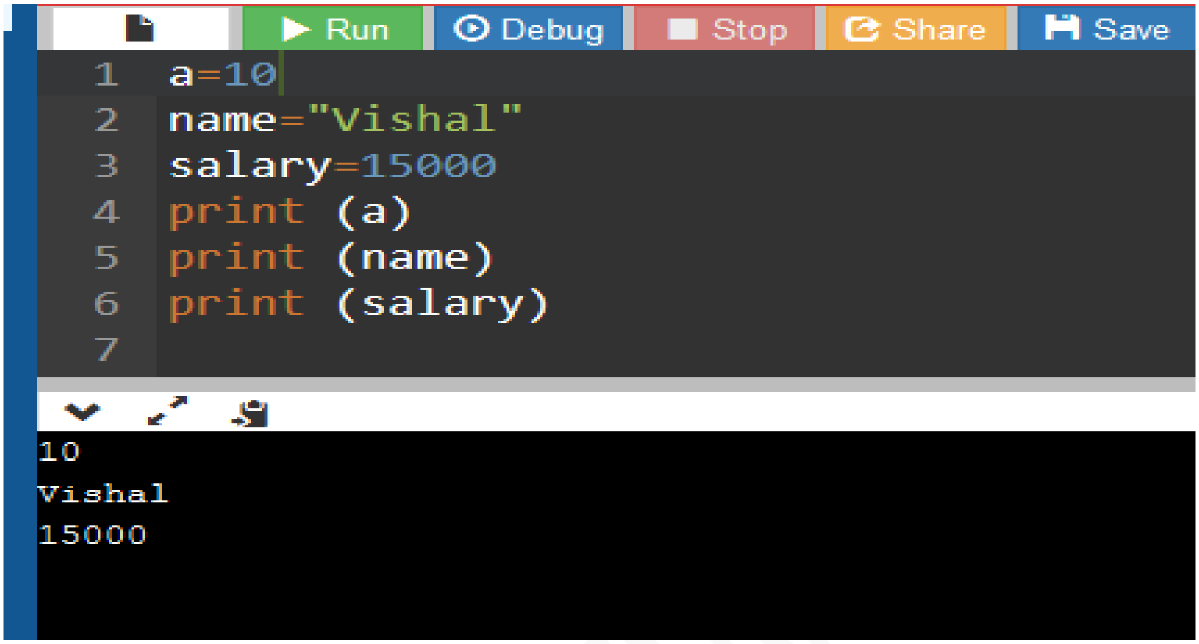
The equal (=) operator is used to assign value to a variable.

Eg:

```
a=10

name=”Vishal”

salary=15000

print (a)

print (name)

print (salary)
```

The equal (=) operator is used to assign value to a variable.

Exmaple:



```
a=10
name="Vishal"
salary=15000
print (a)
print (name)
print (salary)
```

```
10
Vishal
15000
```

**Multiple Assignment**

- Python allows us to assign a value to multiple variables in a single statement which is also known as multiple assignment.
- We can apply multiple assignments in two ways either by assigning a single value to multiple variables or assigning multiple values to multiple variables. Lets see given examples.

**1. Assigning single value to multiple variables**

Eg:

x=y=z=50

print (x)

print (y)

print (z)

**Output:**

>>>

50

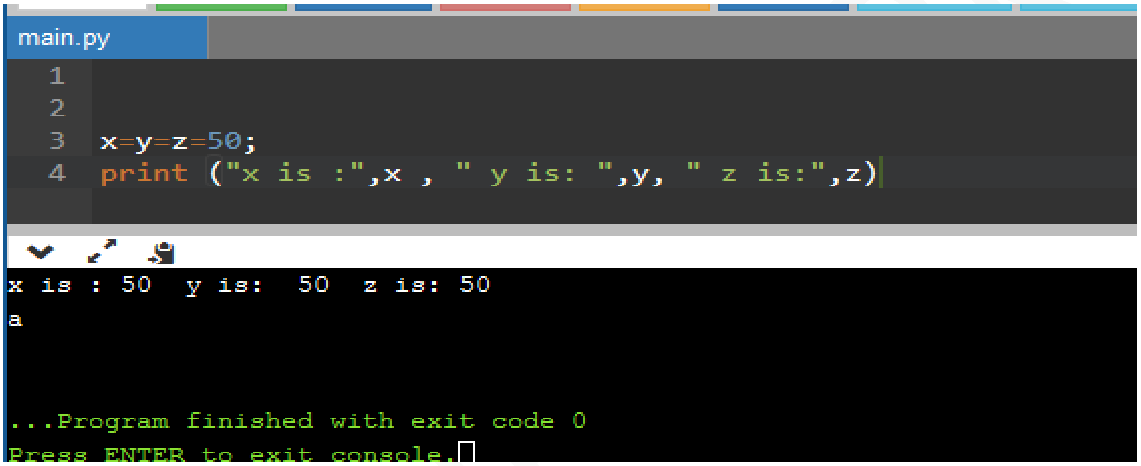
50

50

>>>

### 1. Assigning single value to multiple variables

Eg:



```


x=y=z=50;
print ("x is :",x , " y is: ",y, " z is:",z)
```

```
x is : 50  y is:  50  z is: 50
a


...Program finished with exit code 0
Press ENTER to exit console.
```

### 2.Assigning multiple values to multiple variables:

Eg:

a,b,c=10,20,30

print (a)

print (b)

print (c )

**Output:**

>>>

10

20

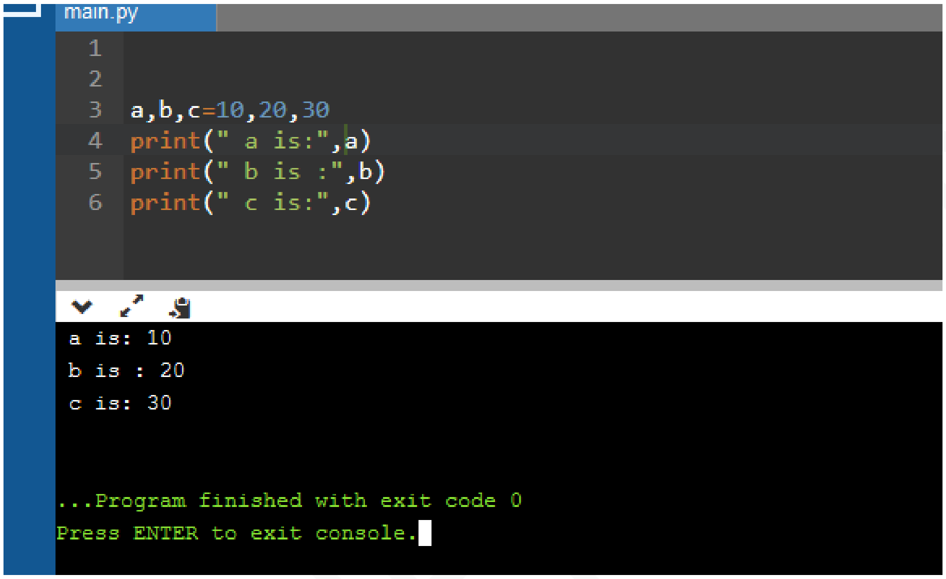
30

>>>

The values will be assigned in the order in which variables appears.

**2.Assigning multiple values to multiple variables:**

Eg:



```
a,b,c=10,20,30
print(" a is:",a)
print(" b is :",b)
print(" c is:",c)
```

```
a is: 10
b is : 20
c is: 30


...Program finished with exit code 0
Press ENTER to exit console.
```

## End of Chapter 1