LECTURE HANDOUTS

L01

MCA

I / I

Course Name with Code : **19CAB04 – Problem Solving and Python Programming**

Course Faculty : **Dr.M.Moorthy**

Unit: I- INTRODUCTION TO COMPUTING AND PROBLEM SOLVING

Date of Lecture: 27.01.2021

Topic of Lecture: Computing Devices

Introduction : (Maximum 5 sentences)

Any electronic equipment controlled by a CPU, including desktop and laptop **computers**, smart phones and tablets. It usually refers to a general-purpose **device** that can accept software for many purposes in contrast with a dedicated unit of equipment such as a network switch or router.

Prerequisite knowledge for Complete understanding and learning of Topic: (Max. Four important topics)

- **Knowledge on Computing and Computers**
- **Knowledge on Programming C and C++**

Detailed content of the Lecture:

Any electronic equipment controlled by a CPU, including desktop and laptop **computers**, smart phones and tablets. It usually refers to a general-purpose **device** that can accept software for many purposes in contrast with a dedicated unit of equipment such as a network switch or router.

Computing is any goal-oriented activity requiring, benefiting from, or creating computing machinery. It includes the study and experimentation of algorithmic processes and development of both hardware and software.

It has scientific, engineering, mathematical, technological and social aspects.

Computing devices are the electronic devices which take inputs, process the inputs and then calculate results from the inputs.

Or in simple words, these are basically the mathematical devices which can do the math much in faster way.

One of the most common computing devices is computer.

These devices ranging from the simple Calculator, Printers, Digital Cameras, Mobile phones, Smart phones , PC Desktops, Laptops and many more.

A **computational problem** is a problem that a computer might be able to solve or a question that a computer may be able to answer. For example, the problem of **factoring**

"Given a positive integer *n*, find a nontrivial prime factor of *n*."

is a computational problem.

A computational problem can be viewed as an infinite collection of *instances* together with a, possibly empty, set of *solutions* for every instance.

For example, in the factoring problem, the instances are the integers *n*, and solutions are prime numbers *p* that describes nontrivial prime factors of *n*.

Computational problems are one of the main objects of study in theoretical computer science.

Problem solving means the ability to formulate problems, think creatively about solutions, and express a solution clearly and accurately.

Video Content / Details of website for further learning (if any):

https://www.tutorialspoint.com/python/index.htm

Important Books/Journals for further learning including the page nos.:
Kenneth A. Lambert, “Fundamentals of Python: First Programs”, Cengage Learning, 2012
(**Page No:38**).

Course Faculty

Verified by HOD

LECTURE HANDOUTS

L02

MCA

I / I

Course Name with Code : **19CAB04 – Problem Solving and Python Programming**

Course Faculty : **Dr.M.Moorthy**

Unit: I- INTRODUCTION TO COMPUTING AND PROBLEM SOLVING

Date of Lecture: 29.01.2021

Topic of Lecture: Identification of Computational Problems

Introduction : (Maximum 5 sentences)

A **computational problem** is a problem that a computer might be able to solve or a question that a computer may be able to answer. For example, the problem of **factoring**

"Given a positive integer *n*, find a nontrivial prime factor of *n*."

is a computational problem.

Prerequisite knowledge for Complete understanding and learning of Topic: (Max. Four important topics)

- **Knowledge on Computing and Computers**
- **Knowledge on Programming C and C++**

Detailed content of the Lecture:

Types of computational problems

A **decision problem** is a computational problem where the answer for every instance is either yes or no. An example of a decision problem is *primality testing*:

"Given a positive integer *n*, determine if *n* is prime."

A decision problem is typically represented as the set of all instances for which the answer is *yes*. For example, primality testing can be represented as the infinite set

$L = \{2, 3, 5, 7, 11, ...\}$

In a **search problem**, the answers can be arbitrary strings. For example, factoring is a search problem where the instances are (string representations of) positive integers and the solutions are (string representations of) collections of primes.

A search problem is represented as a relation consisting of all the instance-solution pairs, called a *search relation*. For example, factoring can be represented as the relation

R = {(4, 2), (6, 2), (6, 3), (8, 2), (9, 3), (10, 2), (10, 5)...}

which consist of all pairs of numbers (n, p), where p is a nontrivial prime factor of n.

A **counting problem** asks for the number of solutions to a given search problem. For example, a counting problem associated with factoring is

"Given a positive integer n, count the number of nontrivial prime factors of n."

A counting problem can be represented by a function f from $\{0, 1\}^*$ to the nonnegative integers. For a search relation R, the counting problem associated to R is the function

$f_R(\mathrm{x}) = |\{y: R(x, y)\}|$.

An **optimization problem** asks for finding a "best possible" solution among the set of all possible solutions to a search problem. One example is the *maximum independent set* problem:

"Given a graph G, find an independent set of G of maximum size."

Optimization problems can be represented by their search relations.

In a **function problem** a single output (of a total function) is expected for every input, but the output is more complex than that of a decision problem, that is, it isn't just "yes" or "no". One of the most famous examples is the *traveling salesman* problem:

"Given a list of cities and the distances between each pair of cities, find the shortest possible route that visits each city exactly once and returns to the origin city."

Video Content / Details of website for further learning (if any):

https://www.tutorialspoint.com/python/index.htm

Important Books/Journals for further learning including the page nos.:

https://www.cs.utah.edu/~zachary/computing/lessons/uces-1/uces-1/node1.html

Course Faculty

Verified by HOD

LECTURE HANDOUTS

L03

MCA

I / I

Course Name with Code : **19CAB04 – Problem Solving and Python Programming**

Course Faculty : **Dr.M.Moorthy**

Unit: I- INTRODUCTION TO COMPUTING AND PROBLEM SOLVING

Date of Lecture: 30.01.2021

Topic of Lecture: Pseudocodes and Flowcharts

Introduction : (Maximum 5 sentences)

Pseudocode: It's simply an *implementation of an algorithm* in the form of *annotations* and *informative text* written in *plain English*. It has no syntax like any of the programming language and thus **can't be compiled or interpreted by the computer**.

A **flowchart** is a type of diagram that represents a workflow or process. A flowchart can also be defined as a diagrammatic representation of an algorithm, a step-by-step approach to solving a task.

Prerequisite knowledge for Complete understanding and learning of Topic: (Max. Four important topics)

- **Knowledge on Computing and Computers**
- **Knowledge on Programming C and C++**

Detailed content of the Lecture:

Pseudocode: It's simply an *implementation of an algorithm* in the form of *annotations* and *informative text* written in *plain English*. It has no syntax like any of the programming language and thus **can't be compiled or interpreted by the computer**.

Advantages of Pseudocode

- **Improves** the readability of any approach. It's one of the best approaches to start implementation of an algorithm.
- **Acts as a bridge** between the program and the algorithm or flowchart. Also works as a rough documentation, so the program of one developer can be understood easily when a pseudo code is written out. In industries, the approach of documentation is essential. And that's where a pseudo-code proves vital.
- The **main goal** of a pseudo code is to **explain what exactly each line of a program should do**, hence making the code construction phase easier for the programmer.

The flowchart shows the steps as boxes of various kinds, and their order by connecting the boxes with arrows. This diagrammatic representation illustrates a solution model to a given problem. Flowcharts are used in analyzing, designing, documenting or managing a process or program in various fields.

There are four general types:

- *Document flowcharts*, showing controls over a document-flow through a system
- *Data flowcharts*, showing controls over a data-flow in a system
- *System flowcharts*, showing controls at a physical or resource level
- *Program flowchart*, showing the controls in a program within a system

Video Content / Details of website for further learning (if any):

https://www.tutorialspoint.com/python/index.htm

Important Books/Journals for further learning including the page nos.:

https://ceng.eskisehir.edu.tr/emrekacmaz/bil158/flowchart-pseudocode-examples.pdf

Course Faculty

Verified by HOD

MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu

DESIGNING YOUR FUTURE
Estd. 2000

IQAC

LECTURE HANDOUTS

L04

MCA

I / I

Course Name with Code : **19CAB04 – Problem Solving and Python Programming**

Course Faculty : **Dr.M.Moorthy**

Unit: I- INTRODUCTION TO COMPUTING AND PROBLEM SOLVING

Date of Lecture: 02.02.2021

Topic of Lecture: Algorithms

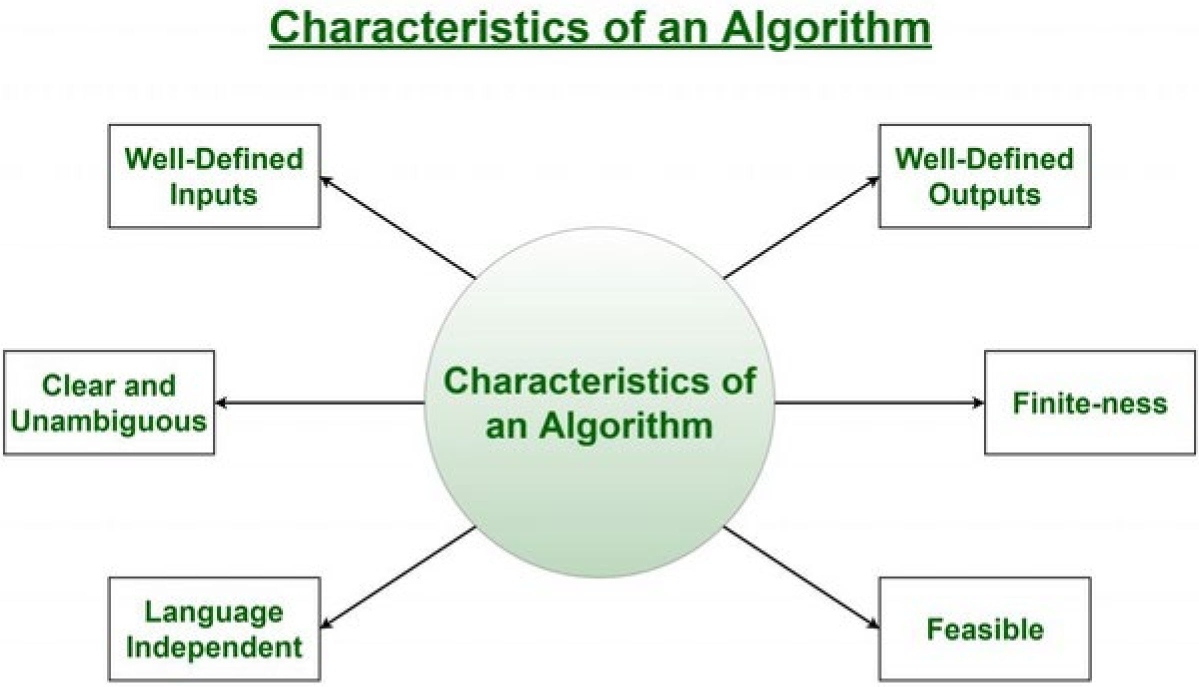
Introduction : (Maximum 5 sentences)

The word **Algorithm** means "a process or set of rules to be followed in calculations or other problem-solving operations". Therefore Algorithm refers to a set of rules/instructions that step-by-step define how a work is to be executed upon in order to get the expected results.

Prerequisite knowledge for Complete understanding and learning of Topic:
(Max. Four important topics)

- **Knowledge on Computing and Computers**
- **Knowledge on Programming C and C++**

Detailed content of the Lecture:



Advantages of Algorithms:

- It is easy to understand.
- Algorithm is a step-wise representation of a solution to a given problem.
- In Algorithm the problem is broken down into smaller pieces or steps hence, it is easier for the programmer to convert it into an actual program.

Disadvantages of Algorithms:

- Writing an algorithm takes a long time so it is time-consuming.
- Branching and Looping statements are difficult to show in Algorithms.

How to Design an Algorithm?

In order to write an algorithm, following things are needed as a pre-requisite:

1. The **problem** that is to be solved by this algorithm.
2. The **constraints** of the problem that must be considered while solving the problem.
3. The **input** to be taken to solve the problem.
4. The **output** to be expected when the problem is solved.
5. The **solution** to this problem, in the given constraints.

Video Content / Details of website for further learning (if any):

https://www.tutorialspoint.com/python/index.htm

Important Books/Journals for further learning including the page nos.:

Kenneth A. Lambert, “Fundamentals of Python: First Programs”, Cengage Learning, 2012. **(Page No:24)**

Course Faculty

Verified by HOD

DESIGNING YOUR FUTURE
Estd. 2000

IQAC

LECTURE HANDOUTS

L05

MCA

I / I

Course Name with Code : **19CAB04 – Problem Solving and Python Programming**

Course Faculty : **Dr.M.Moorthy**

Unit: I- INTRODUCTION TO COMPUTING AND PROBLEM SOLVING

Date of Lecture: 03.02.2021

Topic of Lecture: Building blocks of Algorithms

Introduction : (Maximum 5 sentences)

BUILDING BLOCKS OF ALGORITHMS (statements, state, control flow, functions)

Algorithms can be constructed from basic building blocks namely, sequence, selection and iteration.

Prerequisite knowledge for Complete understanding and learning of Topic: (Max. Four important topics)

- **Knowledge on Computing and Computers**
- **Knowledge on Programming C and C++**

Detailed content of the Lecture:

Statements:

Statement is a single action in a computer.

In a computer statements might include some of the following actions

Ø input data-information given to the program
Ø process data-perform operation on a given input
Ø output data-processed result

State:

Transition from one process to another process under specified condition with in a time is called state.

Control flow:

The process of executing the individual statements in a given order is called control flow.

The control can be executed in three ways

1. sequence
2. selection
3. iteration

Sequence:

All t Function is a sub program which consists of block of code(set of instructions) that performs a particular task.

For complex problems, the problem is been divided into smaller and simpler tasks during algorithm design.

Benefits of Using Functions

- Reduction in line of code
- Code reuse
- Better readability
- Information hiding
- Easy to debug and test
- Improved maintainability

The instructions are executed one after another is called sequence execution.

Video Content / Details of website for further learning (if any):
https://www.tutorialspoint.com/python/index.htm

Important Books/Journals for further learning including the page nos.:
http://www.brainkart.com/article/Building-blocks-of-algorithms-(statements,-state,-control-flow,-functions)_35892/

Course Faculty

Verified by HOD

LECTURE HANDOUTS

L06

MCA

I / I

Course Name with Code : **19CAB04 – Problem Solving and Python Programming**

Course Faculty : **Dr.M.Moorthy**

Unit: I- INTRODUCTION TO COMPUTING AND PROBLEM SOLVING

Date of Lecture: 05.02.2021

Topic of Lecture: Python interpreter and Interactive Mode

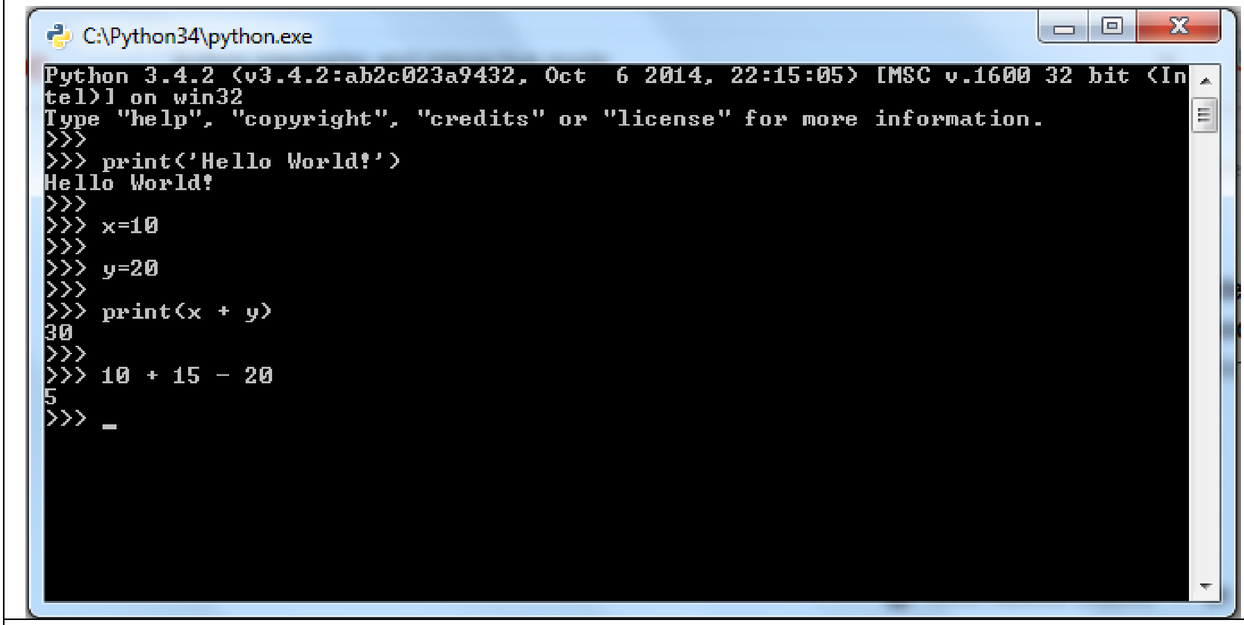
Introduction : (Maximum 5 sentences)

When commands are read from a keyboard, the interpreter is said to be in *interactive mode*. In this mode it prompts for the next command with the *primary prompt*, usually three greater-than signs (>>>); for continuation lines it prompts with the *secondary prompt*, by default three dots (...). The interpreter prints a welcome message stating its version number and a copyright notice before printing the first prompt:

Prerequisite knowledge for Complete understanding and learning of Topic: (Max. Four important topics)

- **Knowledge on Computing and Computers**
- **Knowledge on Programming C and C++**

Detailed content of the Lecture:



```
Python 3.4.2 (v3.4.2:ab2c023a9432, Oct  6 2014, 22:15:05) [MSC v.1600 32 bit (In
tel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>>
>>> print('Hello World!')
Hello World!
>>>
>>> x=10
>>>
>>> y=20
>>>
>>> print(x + y)
30
>>>
>>> 10 + 15 - 20
5
>>> _
```

Video Content / Details of website for further learning (if any):

https://www.tutorialspoint.com/python/index.htm

Important Books/Journals for further learning including the page nos.:

Kenneth A. Lambert, “Fundamentals of Python: First Programs”, Cengage Learning, 2012. (**Page No:45**)

Course Faculty

Verified by HOD

MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu

DESIGNING YOUR FUTURE
Estd. 2000

LECTURE HANDOUTS

L07

MCA

I / I

Course Name with Code : **19CAB04 – Problem Solving and Python Programming**

Course Faculty : **Dr.M.Moorthy**

Unit: I- INTRODUCTION TO COMPUTING AND PROBLEM SOLVING

Date of Lecture: 06.02.2021

Topic of Lecture: Variables and Identifiers

Introduction : (Maximum 5 sentences)

Variables are nothing but reserved memory locations to store values. This means that when you create a variable you reserve some space in memory.

Based on the data type of a variable, the interpreter allocates memory and decides what can be stored in the reserved memory. Therefore, by assigning different data types to variables, you can store integers, decimals or characters in these variables.

Prerequisite knowledge for Complete understanding and learning of Topic: (Max. Four important topics)

- **Knowledge on Computing and Computers**
- **Knowledge on Programming C and C++**

Detailed content of the Lecture:

Python variables do not need explicit declaration to reserve memory space. The declaration happens automatically when you assign a value to a variable. The equal sign (=) is used to assign values to variables.

The operand to the left of the = operator is the name of the variable and the operand to the right of the = operator is the value stored in the variable. For example −

```
counter = 100          # An integer assignment
miles   = 1000.0       # A floating point
name    = "John"       # A string

print counter
print miles
print name
```

Here, 100, 1000.0 and "John" are the values assigned to *counter*, *miles*, and *name* variables, respectively. This produces the following result −

```
100
1000.0
John
```

Multiple Assignment

Python allows you to assign a single value to several variables simultaneously. For example −

```
a = b = c = 1
```

Here, an integer object is created with the value 1, and all three variables are assigned to the same memory location. You can also assign multiple objects to multiple variables. For example −

```
a,b,c = 1,2,"john"
```

Here, two integer objects with values 1 and 2 are assigned to variables a and b respectively, and one string object with the value "john" is assigned to the variable c.

Video Content / Details of website for further learning (if any):

https://www.tutorialspoint.com/python/index.htm

Important Books/Journals for further learning including the page nos.:

John V Guttag, “Introduction to Computation and Programming Using Python”, Revised and Expanded Edition, MIT Press , 2013. (**Page No: 11**)

Course Faculty

Verified by HOD

MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu

DESIGNING YOUR FUTURE
Estd. 2000

IQAC

LECTURE HANDOUTS

L08

MCA

I / I

Course Name with Code : **19CAB04 – Problem Solving and Python Programming**

Course Faculty : **Dr.M.Moorthy**

Unit: I- INTRODUCTION TO COMPUTING AND PROBLEM SOLVING

Date of Lecture: 09.02.2021

Topic of Lecture: Arithmetic Operators

Introduction : (Maximum 5 sentences)

Arithmetic operators are used to perform mathematical operations like addition, subtraction, multiplication and division.

There are 7 arithmetic operators in Python :

1. Addition
2. Subtraction
3. Multiplication
4. Division
5. Modulus
6. Exponentiation
7. Floor division

Prerequisite knowledge for Complete understanding and learning of Topic: (Max. Four important topics)

- **Knowledge on Computing and Computers**
- **Knowledge on Programming C and C++**

Detailed content of the Lecture:

Operator	Description	Syntax
+	Addition: adds two operands	x + y
–	Subtraction: subtracts two operands	x – y
*	Multiplication: multiplies two operands	x * y
/	Division (float): divides the first operand by the second	x / y
//	Division (floor): divides the first operand by the second	x // y

%	Modulus: returns the remainder when first operand is divided by the second	x % y
**	Power : Returns first raised to power second	x ** y

Video Content / Details of website for further learning (if any):

https://www.tutorialspoint.com/python/index.htm

Important Books/Journals for further learning including the page nos.:

Allen B. Downey, "Think Python: How to Think Like a Computer Scientist", Second Edition, Shroff/O'Reilly Publishers, 2016. (**Page No:28**)

Course Faculty

Verified by HOD

MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu

DESIGNING YOUR FUTURE
Estd. 2000

IQAC

LECTURE HANDOUTS

L09

MCA

I / I

Course Name with Code : **19CAB04 – Problem Solving and Python Programming**

Course Faculty : **Dr.M.Moorthy**

Unit: I- INTRODUCTION TO COMPUTING AND PROBLEM SOLVING

Date of Lecture: 10.02.2021

Topic of Lecture: Statements

Introduction : (Maximum 5 sentences)

Statements. Instructions written in the source code for execution are called statements. There are different types of statements in the Python programming language like Assignment statement, Conditional statement, Looping statements etc. These all help the user to get the required output.

Prerequisite knowledge for Complete understanding and learning of Topic: (Max. Four important topics)

- **Knowledge on Computing and Computers**
- **Knowledge on Programming C and C++**

Detailed content of the Lecture:

1 if statements

An **if statement** consists of a boolean expression followed by one or more statements.

2 if...else statements

An **if statement** can be followed by an optional **else statement**, which executes when the boolean expression is FALSE.

3 nested if statements

You can use one **if** or **else if** statement inside another **if** or **else if** statement(s).

4 while loop

Repeats a statement or group of statements while a given condition is TRUE. It tests the condition before executing the loop body.

5 for loop

Executes a sequence of statements multiple times and abbreviates the code that manages the loop variable.

6 nested loops

You can use one or more loop inside any another while, for or do..while loop.

break statement

Terminates the loop statement and transfers execution to the statement immediately following the loop.

continue statement

Causes the loop to skip the remainder of its body and immediately retest its condition prior to reiterating.

pass statement

The pass statement in Python is used when a statement is required syntactically but you do not want any command or code to execute.

Video Content / Details of website for further learning (if any):

https://www.tutorialspoint.com/python/index.htm

Important Books/Journals for further learning including the page nos.:

Kenneth A. Lambert, "Fundamentals of Python: First Programs", Cengage Learning, 2012. (**Page No:119**)

Course Faculty

Verified by HOD

LECTURE HANDOUTS

L10

MCA

I / I

Course Name with Code : **19CAB04 – Problem Solving and Python Programming**

Course Faculty : **Dr.M.Moorthy**

Unit: II - CONDITIONALS AND FUNCTIONS **Date of Lecture:** 12.02.2021

Topic of Lecture: Boolean Values

Introduction : (Maximum 5 sentences)
In Python, the two Boolean values are True and False (the capitalization must be exactly as shown), and the Python type is bool.

Prerequisite knowledge for Complete understanding and learning of Topic: (Max. Four important topics)

- **Knowledge on Computing and Computers**
- **Knowledge on Programming C and C++**

Detailed content of the Lecture:

Boolean Values

In programming you often need to know if an expression is `True` or `False`.

You can evaluate any expression in Python, and get one of two answers, `True` or `False`.

When you compare two values, the expression is evaluated and Python returns the Boolean answer:

Example

```
print(10 > 9)
print(10 == 9)
print(10 < 9)
```

Example

Print a message based on whether the condition is `True` or `False`:

```
a = 200
b = 33

if b > a:
  print("b is greater than a")
else:
  print("b is not greater than a")
```

Evaluate a string and a number:

```
print(bool("Hello"))
print(bool(15))
```

Video Content / Details of website for further learning (if any):
https://www.tutorialspoint.com/python/index.htm

Important Books/Journals for further learning including the page nos.:

John V Guttag, “Introduction to Computation and Programming Using Python”, Revised and Expanded Edition, MIT Press (**Page No:31**)

Course Faculty

Verified by HOD

MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu

Estd. 2000

LECTURE HANDOUTS

L11

MCA

I / I

Course Name with Code : **19CAB04 – Problem Solving and Python Programming**

Course Faculty : **Dr.M.Moorthy**

Unit: II - CONDITIONALS AND FUNCTIONS **Date of Lecture:** 13.02.2021

Topic of Lecture: Operators Precedence

Introduction : (Maximum 5 sentences)

Operator precedence determines the grouping of terms in an expression and decides how an expression is evaluated.

Certain **operators** have higher **precedence** than others; for example, the multiplication **operator** has a higher **precedence** than the addition **operator**.

Prerequisite knowledge for Complete understanding and learning of Topic:
(Max. Four important topics)

- **Knowledge on Computing and Computers**
- **Knowledge on Programming C and C++**

Detailed content of the Lecture:

Operators	Meaning
()	Parentheses
**	Exponent
+x, -x, ~x	Unary plus, Unary minus, Bitwise NOT
*, /, //, %	Multiplication, Division, Floor division, Modulus
+, -	Addition, Subtraction
<<, >>	Bitwise shift operators
&	Bitwise AND
^	Bitwise XOR
|	Bitwise OR

==, !=, >, >=, <, <=, is, **is not, in, not in**	**Comparisons, Identity, Membership operators**
not	**Logical NOT**
and	**Logical AND**
or	**Logical OR**

Video Content / Details of website for further learning (if any):

https://www.tutorialspoint.com/python/index.htm

Important Books/Journals for further learning including the page nos.:

Allen B. Downey, “Think Python: How to Think Like a Computer Scientist”, Second Edition, Shroff/O‘Reilly Publishers, 2016 (**Page No:43**)

Course Faculty

Verified by HOD

LECTURE HANDOUTS

L12

MCA

I / I

Course Name with Code : **19CAB04 – Problem Solving and Python Programming**

Course Faculty : **Dr.M.Moorthy**

Unit: II - CONDITIONALS AND FUNCTIONS **Date of Lecture:** 16.02.2021

Topic of Lecture: Conditionals

Introduction : (Maximum 5 sentences)

Decision making is anticipation of conditions occurring while execution of the program and specifying actions taken according to the conditions.

Decision structures evaluate multiple expressions which produce TRUE or FALSE as outcome. You need to determine which action to take and which statements to execute if outcome is TRUE or FALSE otherwise.

Prerequisite knowledge for Complete understanding and learning of Topic: (Max. Four important topics)

- **Knowledge on Computing and Computers**
- **Knowledge on Programming C and C++**

Detailed content of the Lecture:

Python programming language assumes any **non-zero** and **non-null** values as TRUE, and if it is either **zero** or **null**, then it is assumed as FALSE value.

Python programming language provides following types of decision making statements.

1 if statements

An **if statement** consists of a boolean expression followed by one or more statements.

2 if...else statements

An **if statement** can be followed by an optional **else statement**, which executes when the boolean expression is FALSE.

3 nested if statements

You can use one **if** or **else if** statement inside another **if** or **else if** statement(s).

```
# program using if else

s = input ("Input your name: ")

if s == "Nathiya":

 print( "Hello ", s)

elif s == "Arun":

 print( "I'm so glad to see you ", s)

elif s == "Venkatramana":

 print( "I didn't expect you ",s)

else:

 print("Hello unknown")
```

Video Content / Details of website for further learning (if any):

https://www.tutorialspoint.com/python/index.htm

Important Books/Journals for further learning including the page nos.:

Allen B. Downey, “Think Python: How to Think Like a Computer Scientist”, Second Edition, Shroff/O‘Reilly Publishers, 2016 (**Page No:92**)

Course Faculty

Verified by HOD

MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu

DESIGNING YOUR FUTURE
Estd. 2000

IQAC

LECTURE HANDOUTS

L13

MCA

I / I

Course Name with Code : **19CAB04 – Problem Solving and Python Programming**

Course Faculty : **Dr.M.Moorthy**

Unit: II - CONDITIONALS AND FUNCTIONS **Date of Lecture:** 17.02.2021

Topic of Lecture: Loop Structures

Introduction : (Maximum 5 sentences)

A loop statement allows us to execute a statement or group of statements multiple times. It may be for , while, for each, etc.

Prerequisite knowledge for Complete understanding and learning of Topic: (Max. Four important topics)

- **Knowledge on Computing and Computers**
- **Knowledge on Programming C and C++**

Detailed content of the Lecture:

Python programming language provides following types of loops to handle looping requirements.

Sr.No. Loop Type & Description

1 while loop

Repeats a statement or group of statements while a given condition is TRUE. It tests the condition before executing the loop body.

2 for loop

Executes a sequence of statements multiple times and abbreviates the code that manages the loop variable.

3 nested loops

You can use one or more loop inside any another while, for or do..while loop.

Let us go through the loop control statements briefly

Sr.No. Control Statement & Description

1 break statement

Terminates the loop statement and transfers execution to the statement immediately following the loop.

2 continue statement

Causes the loop to skip the remainder of its body and immediately retest its condition prior to reiterating.

3 pass statement

The pass statement in Python is used when a statement is required syntactically but you do not want any command or code to execute.

Video Content / Details of website for further learning (if any):

https://www.tutorialspoint.com/python/index.htm

Important Books/Journals for further learning including the page nos.:

John V Guttag, "Introduction to Computation and Programming Using Python", Revised and Expanded Edition, MIT Press , 2013. (**Page No:35**)

Course Faculty

Verified by HOD

MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu

Estd. 2000

LECTURE HANDOUTS

L14

MCA

I / I

Course Name with Code : **19CAB04 – Problem Solving and Python Programming**

Course Faculty : **Dr.M.Moorthy**

Unit: II - CONDITIONALS AND FUNCTIONS **Date of Lecture:** 22.08.2020

Topic of Lecture: While and For Loop

Introduction : (Maximum 5 sentences)

Python has two primitive loop commands:

- while loops
- for loops

With the while loop we can execute a set of statements as long as a condition is true.

A for loop is used for iterating over a sequence (that is a list, a tuple, a dictionary, a set, or a string).

Prerequisite knowledge for Complete understanding and learning of Topic:
(Max. Four important topics)

- **Knowledge on Computing and Computers**
- **Knowledge on Programming C and C++**

Detailed content of the Lecture:

Example - while

Print i as long as i is less than 6:

```
i = 1
while i < 6:
  print(i)
  i += 1
```

Example - for

Print each fruit in a fruit list:

```
fruits = ["apple", "banana", "cherry"]
for x in fruits:
  print(x)
```

Exit the loop when x is "banana":

```
fruits = ["apple", "banana", "cherry"]
for x in fruits:
  print(x)
  if x == "banana":
    break
```

Video Content / Details of website for further learning (if any):

https://www.tutorialspoint.com/python/index.htm

Important Books/Journals for further learning including the page nos.:
Allen B. Downey, “Think Python: How to Think Like a Computer Scientist”, Second Edition, Shroff/O‘Reilly Publishers, 2016. (**Page No:131**)

Course Faculty

Verified by HOD

MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu

DESIGNING YOUR FUTURE
Estd. 2000

LECTURE HANDOUTS

L15

MCA

I / I

Course Name with Code : **19CAB04 – Problem Solving and Python Programming**

Course Faculty : **Dr.M.Moorthy**

Unit: II - CONDITIONALS AND FUNCTIONS **Date of Lecture:** 20.02.2021

Topic of Lecture: Break Statement

Introduction : (Maximum 5 sentences)

With the break statement we can stop the loop before it has looped through all the items.

Prerequisite knowledge for Complete understanding and learning of Topic: (Max. Four important topics)

- **Knowledge on Computing and Computers**
- **Knowledge on Programming C and C++**

Detailed content of the Lecture:

Example

Exit the loop when x is "banana":

```
fruits = ["apple", "banana", "cherry"]
for x in fruits:
  print(x)
  if x == "banana":
    break
```

Exit the loop when x is "banana", but this time the break comes before the print:

```
fruits = ["apple", "banana", "cherry"]
for x in fruits:
  if x == "banana":
    break
  print(x)
```

Video Content / Details of website for further learning (if any):

https://www.tutorialspoint.com/python/index.htm

Important Books/Journals for further learning including the page nos.:
Allen B. Downey, "Think Python: How to Think Like a Computer Scientist", Second Edition, Shroff/O'Reilly Publishers, 2016. (**Page No:133**)

Course Faculty

Verified by HOD

MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu

DESIGNING YOUR FUTURE
Estd. 2000

IQAC

LECTURE HANDOUTS

L16

MCA

I / I

Course Name with Code : **19CAB04 – Problem Solving and Python Programming**

Course Faculty : **Dr.M.Moorthy**

Unit: II - CONDITIONALS AND FUNCTIONS **Date of Lecture:** 23.02.2021

Topic of Lecture: Function Call and Returning Values

Introduction : (Maximum 5 sentences)

A function is a block of code which only runs when it is called.

You can pass data, known as parameters, into a function.

A function can return data as a result.

Prerequisite knowledge for Complete understanding and learning of Topic: (Max. Four important topics)

- **Knowledge on Computing and Computers**
- **Knowledge on Programming C and C++**

Detailed content of the Lecture:

Creating a Function

In Python a function is defined using the def keyword:

Example

```
def my_function():
  print("Hello from a function")
```

Calling a Function

To call a function, use the function name followed by parenthesis:

Example

```
def my_function():
  print("Hello from a function")
```

my_function()

Arguments

Information can be passed into functions as arguments.

Arguments are specified after the function name, inside the parentheses. You can add as many arguments as you want, just separate them with a comma.

The following example has a function with one argument (fname). When the function is called, we pass along a first name, which is used inside the function to print the full name:

Example

```
def my_function(fname):
  print(fname + " Refsnes")

my_function("Emil")
my_function("Tobias")
my_function("Linus")
```

Video Content / Details of website for further learning (if any):

https://www.tutorialspoint.com/python/index.htm

Important Books/Journals for further learning including the page nos.:
Allen B. Downey, "Think Python: How to Think Like a Computer Scientist", Second Edition, Shroff/O'Reilly Publishers, 2016. (**Page No:52**)

Course Faculty

Verified by HOD

MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu

DESIGNING YOUR FUTURE
Estd. 2000

IQAC

LECTURE HANDOUTS

L17

MCA

I / I

Course Name with Code : **19CAB04 – Problem Solving and Python Programming**

Course Faculty : **Dr.M.Moorthy**

Unit: II - CONDITIONALS AND FUNCTIONS **Date of Lecture:** 24.02.2021

Topic of Lecture: Local and Global Scope

Introduction : (Maximum 5 sentences)

If you create a variable with the same name inside a function, this variable will be **local**, and can only be used inside the function.

Variables that are created outside of a function (as in all of the examples above) are known as **global** variables.

Global variables can be used by everyone, both inside of functions and outside.

Prerequisite knowledge for Complete understanding and learning of Topic:
(Max. Four important topics)

- **Knowledge on Computing and Computers**
- **Knowledge on Programming C and C++**

Detailed content of the Lecture:

Example

Create a variable inside a function, with the same name as the global variable

```
x = "awesome"

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

myfunc()

print("Python is " + x)
```

The global Keyword

Normally, when you create a variable inside a function, that variable is local, and can only be used inside that function.

To create a global variable inside a function, you can use the `global` keyword.

Example

If you use the `global` keyword, the variable belongs to the global scope:

```
def myfunc():
  global x
  x = "fantastic"

myfunc()

print("Python is " + x)
```

Also, use the `global` keyword if you want to change a global variable inside a function.

Example

To change the value of a global variable inside a function, refer to the variable by using the `global` keyword:

```
x = "awesome"

def myfunc():
  global x
  x = "fantastic"

myfunc()

print("Python is " + x)
```

Video Content / Details of website for further learning (if any):
https://www.tutorialspoint.com/python/index.htm

Important Books/Journals for further learning including the page nos.:
Allen B. Downey, “Think Python: How to Think Like a Computer Scientist”, Second Edition, Shroff/O‘Reilly Publishers, 2016 (**Page No:208**)

Course Faculty

Verified by HOD

MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu

Estd. 2000

LECTURE HANDOUTS

L18

MCA

I / I

Course Name with Code : **19CAB04 – Problem Solving and Python Programming**

Course Faculty : **Dr.M.Moorthy**

Unit: II - CONDITIONALS AND FUNCTIONS **Date of Lecture:** 26.02.2021

Topic of Lecture: Recursive Functions

Introduction : (Maximum 5 sentences)

Python Recursive Function

In Python, we know that a function can call other functions. It is even possible for the function to call itself. These types of construct are termed as recursive functions.

Prerequisite knowledge for Complete understanding and learning of Topic: (Max. Four important topics)

- **Knowledge on Computing and Computers**
- **Knowledge on Programming C and C++**

Detailed content of the Lecture:

Example of a recursive function

```
def factorial(x):
    """This is a recursive function
    to find the factorial of an integer"""

    if x == 1:
        return 1
    else:
        return (x * factorial(x-1))


num = 3
print("The factorial of", num, "is", factorial(num))
```

Output

```
The factorial of 3 is 6
```

This recursive call can be explained in the following steps.

```
factorial(3)          # 1st call with 3

3 * factorial(2)      # 2nd call with 2

3 * 2 * factorial(1)  # 3rd call with 1

3 * 2 * 1             # return from 3rd call as number=1

3 * 2                 # return from 2nd call

6                     # return from 1st call
```

Video Content / Details of website for further learning (if any):

https://www.tutorialspoint.com/python/index.htm

Important Books/Journals for further learning including the page nos.:
Allen B. Downey, "Think Python: How to Think Like a Computer Scientist", Second Edition, Shroff/O'Reilly Publishers, 2016 (**Page No:96**)

Course Faculty

Verified by HOD

MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu

Estd. 2000

LECTURE HANDOUTS

L19

MCA

I / I

Course Name with Code : **19CAB04 – Problem Solving and Python Programming**

Course Faculty : **Dr.M.Moorthy**

Unit: III - SIMPLE DATA STRUCTURES IN PYTHON **Date of Lecture:** 27.02.2021

Topic of Lecture: List data structure

Introduction : (Maximum 5 sentences)

Lists are used to store multiple items in a single variable.

Lists are one of 4 built-in data types in Python used to store collections of data, the other 3 are Tuple, Set, and Dictionary, all with different qualities and usage.

Prerequisite knowledge for Complete understanding and learning of Topic: (Max. Four important topics)

- **Knowledge on Computing and Computers**
- **Knowledge on Programming C and C++**

Detailed content of the Lecture:

How to create a list?

In Python programming, a list is created by placing all the items (elements) inside square brackets [], separated by commas.

It can have any number of items and they may be of different types (integer, float, string etc.).

```
# empty list
my_list = []

# list of integers
my_list = [1, 2, 3]

# list with mixed data types
my_list = [1, "Hello", 3.4]
```

A list can also have another list as an item. This is called a nested list.

```
# nested list
my_list = ["mouse", [8, 4, 6], ['a']]
```

How to slice lists in Python?

We can access a range of items in a list by using the slicing operator :(colon).

```
# List slicing in Python

my_list = ['p','r','o','g','r','a','m','i','z']

# elements 3rd to 5th
print(my_list[2:5])

# elements beginning to 4th
print(my_list[:-5])

# elements 6th to end
print(my_list[5:])

# elements beginning to end
print(my_list[:])
```

Video Content / Details of website for further learning (if any):

https://www.tutorialspoint.com/python/index.htm

Important Books/Journals for further learning including the page nos.:

Allen B. Downey, "Think Python: How to Think Like a Computer Scientist", Second Edition, Shroff/O'Reilly Publishers, 2016 (**Page No:173**)

Course Faculty

Verified by HOD

MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu

Estd. 2000

LECTURE HANDOUTS

L20

MCA

I / I

Course Name with Code : **19CAB04 – Problem Solving and Python Programming**

Course Faculty : **Dr.M.Moorthy**

Unit: III - SIMPLE DATA STRUCTURES IN PYTHON **Date of Lecture:** 02.03.2021

Topic of Lecture: Finding and Updating an Item

Introduction : (Maximum 5 sentences)

Lists are mutable, meaning their elements can be changed unlike string or tuple. We can use the assignment operator `=` to change an item or a range of items.

Prerequisite knowledge for Complete understanding and learning of Topic:
(Max. Four important topics)

- **Knowledge on Computing and Computers**
- **Knowledge on Programming C and C++**

Detailed content of the Lecture:

```
 # Correcting mistake values in a list
odd = [2, 4, 6, 8]

# change the 1st item
odd[0] = 1

print(odd)

# change 2nd to 4th items
odd[1:4] = [3, 5, 7]

print(odd)
```

Output

```
[1, 4, 6, 8]
[1, 3, 5, 7]
```

We can add one item to a list using the `append()` method or add several items using `extend()` method.

```
# Appending and Extending lists in Python
odd = [1, 3, 5]

odd.append(7)

print(odd)

odd.extend([9, 11, 13])

print(odd)
```

Output

```
[1, 3, 5, 7]
[1, 3, 5, 7, 9, 11, 13]
```

Video Content / Details of website for further learning (if any):

https://www.tutorialspoint.com/python/index.htm

Important Books/Journals for further learning including the page nos.:

Allen B. Downey, "Think Python: How to Think Like a Computer Scientist", Second Edition, Shroff/O'Reilly Publishers, 2016 (**Page No:183**)

Course Faculty

Verified by HOD

MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu

Estd. 2000

LECTURE HANDOUTS

L21

MCA

I / I

Course Name with Code : **19CAB04 – Problem Solving and Python Programming**

Course Faculty : **Dr.M.Moorthy**

Unit: III - SIMPLE DATA STRUCTURES IN PYTHON **Date of Lecture:** 03.03.2021

Topic of Lecture: Looping through a List

Introduction : (Maximum 5 sentences)

List is equivalent to arrays in other languages, with the extra benefit of being dynamic in size. In Python, the list is a type of container in Data Structures, which is used to store multiple data at the same time. Unlike Sets, lists in Python are ordered and have a definite count. There are multiple ways to iterate over a list in Python.

Prerequisite knowledge for Complete understanding and learning of Topic: (Max. Four important topics)

- **Knowledge on Computing and Computer**
- **Knowledge on Programming C and C++**

Detailed content of the Lecture:

```
# Python3 code to iterate over a list
list = [1, 3, 5, 7, 9]

# Using for loop
for i in list:
    print(i)
```

Output:

```
1

3

5

7

9
```

```
# Python3 code to iterate over a list
list = [1, 3, 5, 7, 9]

# getting length of list
length = len(list)

# Iterating the index
# same as 'for i in range(len(list))'
```

```
for i  in range(length):
    print(list[i])
```

Output:

```
1
3

5

7

9
```

Video Content / Details of website for further learning (if any):

https://www.tutorialspoint.com/python/index.htm

Important Books/Journals for further learning including the page nos.:

Allen B. Downey, “Think Python: How to Think Like a Computer Scientist”, Second Edition, Shroff/O‘Reilly Publishers, 2016 (**Page No:177**)

Course Faculty

Verified by HOD

LECTURE HANDOUTS

L22

MCA

I / I

Course Name with Code : **19CAB04 – Problem Solving and Python Programming**

Course Faculty : **Dr.M.Moorthy**

Unit: III - SIMPLE DATA STRUCTURES IN PYTHON **Date of Lecture:** 05.03.2021

Topic of Lecture: Sorting a List

Introduction : (Maximum 5 sentences)

The sort() method sorts the list ascending by default.

You can also make a function to decide the sorting criteria(s).

Syntax

list.sort(reverse=True|False, key=myFunc)

Prerequisite knowledge for Complete understanding and learning of Topic: (Max. Four important topics)

- **Knowledge on Computing and Computers**
- **Knowledge on Programming C and C++**

Detailed content of the Lecture:

Example 1: Sort a given list

```
# vowels list
vowels = ['e', 'a', 'u', 'o', 'i']

# sort the vowels
vowels.sort()

# print vowels
print('Sorted list:', vowels)
```

Output

```
Sorted list: ['a', 'e', 'i', 'o', 'u']
```

Example 2: Sort the list in Descending order

```
# vowels list
vowels = ['e', 'a', 'u', 'o', 'i']

# sort the vowels
vowels.sort(reverse=True)

# print vowels
print('Sorted list (in Descending):', vowels)
```

Output

```
Sorted list (in Descending): ['u', 'o', 'i', 'e', 'a']
```

Video Content / Details of website for further learning (if any):

https://www.tutorialspoint.com/python/index.htm

Important Books/Journals for further learning including the page nos.:

Allen B. Downey, "Think Python: How to Think Like a Computer Scientist", Second Edition, Shroff/O'Reilly Publishers, 2016 (**Page No:180**)

Course Faculty

Verified by HOD

LECTURE HANDOUTS **L23**

MCA **I / I**

Course Name with Code : **19CAB04 – Problem Solving and Python Programming**

Course Faculty : **Dr.M.Moorthy**

Unit: III - SIMPLE DATA STRUCTURES IN PYTHON **Date of Lecture:** 06.03.2021

Topic of Lecture: List Slices

Introduction : (Maximum 5 sentences)

Slice Notation

Indexing allows you to access/change/delete only a single cell of a list. What if we want to get a sublist of the list. Or we want to update a bunch of cells at once? Or we want to go on a frenzy and extend a list with an arbitrary number of new cells in any position?

Those and lots of other cool tricks can be done with slice notation.

Prerequisite knowledge for Complete understanding and learning of Topic: (Max. Four important topics)

- **Knowledge on Computing and Computers**
- **Knowledge on Programming C and C++**

Detailed content of the Lecture:

Basic Usage of Slices

Let’s create a basic list:

```
>>> nums = [10, 20, 30, 40, 50, 60, 70, 80, 90]
```

What if we want to take a sublist from the `nums` list? This is a snap when using slice:

```
>>> nums = [10, 20, 30, 40, 50, 60, 70, 80, 90]
>>> some_nums = nums[2:7]
>>> some_nums
[30, 40, 50, 60, 70]
```

So, here is our first example of a slice: *2:7*. The full slice syntax is: *start:stop:step*. `start` refers to the index of the element which is used as a start of our slice. `stop` refers to the index of the element we should stop just before to finish our slice. `step` allows you to take each nth-element within a *start:stop* range.

In our example `start` equals `2`, so our slice starts from value `30`. `stop` is `7`, so the last element of the slice is `70` with index `6`. In the end, slice creates a new list(we named it `some_nums`) with selected elements.

Video Content / Details of website for further learning (if any):

https://www.tutorialspoint.com/python/index.htm

Important Books/Journals for further learning including the page nos.:

Allen B. Downey, “Think Python: How to Think Like a Computer Scientist”, Second Edition, Shroff/O‘Reilly Publishers, 2016 (**Page No:179**)

Course Faculty

Verified by HOD

LECTURE HANDOUTS

L24

MCA

I / I

Course Name with Code : **19CAB04 – Problem Solving and Python Programming**

Course Faculty : **Dr.M.Moorthy**

Unit: III - SIMPLE DATA STRUCTURES IN PYTHON **Date of Lecture:** 09.03.2021

Topic of Lecture: Tuples data structure

Introduction : (Maximum 5 sentences)
A tuple is a collection of objects which ordered and immutable. Tuples are sequences, just like lists. The differences between tuples and lists are, the tuples cannot be changed unlike lists and tuples use parentheses, whereas lists use square brackets.

Prerequisite knowledge for Complete understanding and learning of Topic: (Max. Four important topics)

- **Knowledge on Computing and Computers**
- **Knowledge on Programming C and C++**

Detailed content of the Lecture:

A tuple is a collection of objects which ordered and immutable. Tuples are sequences, just like lists. The differences between tuples and lists are, the tuples cannot be changed unlike lists and tuples use parentheses, whereas lists use square brackets.

Creating a tuple is as simple as putting different comma-separated values. Optionally you can put these comma-separated values between parentheses also. For example −

```
tup1 = ('physics', 'chemistry', 1997, 2000);
tup2 = (1, 2, 3, 4, 5 );
tup3 = "a", "b", "c", "d";
```

The empty tuple is written as two parentheses containing nothing −

```
tup1 = ();
```

To write a tuple containing a single value you have to include a comma, even though there is only one value −

```
tup1 = (50,);
```

Like string indices, tuple indices start at 0, and they can be sliced, concatenated, and so on.

Accessing Values in Tuples

To access values in tuple, use the square brackets for slicing along with the index or indices to obtain value available at that index. For example −

```
#!/usr/bin/python

tup1 = ('physics', 'chemistry', 1997, 2000);
tup2 = (1, 2, 3, 4, 5, 6, 7 );
print "tup1[0]: ", tup1[0];
print "tup2[1:5]: ", tup2[1:5];
```

When the above code is executed, it produces the following result −

```
tup1[0]:  physics
tup2[1:5]:  [2, 3, 4, 5]
```

Video Content / Details of website for further learning (if any):

https://www.tutorialspoint.com/python/index.htm

Important Books/Journals for further learning including the page nos.:

Allen B. Downey, “Think Python: How to Think Like a Computer Scientist”, Second Edition, Shroff/O‘Reilly Publishers, 2016 (**Page No:217**)

Course Faculty

Verified by HOD

MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu

Estd. 2000

LECTURE HANDOUTS

L25

MCA

I / I

Course Name with Code : **19CAB04 – Problem Solving and Python Programming**

Course Faculty : **Dr.M.Moorthy**

Unit: III - SIMPLE DATA STRUCTURES IN PYTHON **Date of Lecture:** 10.03.2021

Topic of Lecture: Manipulating Tuples

Introduction : (Maximum 5 sentences)
Tuples are immutable which means you cannot update or change the values of tuple elements. You are able to take portions of existing tuples to create new tuples.

Prerequisite knowledge for Complete understanding and learning of Topic: (Max. Four important topics)

- **Knowledge on Computing and Computers**
- **Knowledge on Programming C and C++**

Detailed content of the Lecture:

Updating Tuples

Tuples are immutable which means you cannot update or change the values of tuple elements. You are able to take portions of existing tuples to create new tuples as the following example demonstrates −

```
#!/usr/bin/python

tup1 = (12, 34.56);
tup2 = ('abc', 'xyz');

# Following action is not valid for tuples
# tup1[0] = 100;

# So let's create a new tuple as follows
tup3 = tup1 + tup2;
print tup3;
```

When the above code is executed, it produces the following result −

```
(12, 34.56, 'abc', 'xyz')
```

Video Content / Details of website for further learning (if any): **https://www.tutorialspoint.com/python/index.htm**
Important Books/Journals for further learning including the page nos.: Allen B. Downey, “Think Python: How to Think Like a Computer Scientist”, Second Edition, Shroff/O‘Reilly Publishers, 2016 (**Page No:217**)

Course Faculty

Verified by HOD

MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu

Estd. 2000

LECTURE HANDOUTS

L26

MCA **I / I**

Course Name with Code : **19CAB04 – Problem Solving and Python Programming**

Course Faculty : **Dr.M.Moorthy**

Unit: III - SIMPLE DATA STRUCTURES IN PYTHON **Date of Lecture:** 12.03.2021

Topic of Lecture: Nested Tuples

Introduction : (Maximum 5 sentences)

Sometimes, while working with tuples, we can have a problem in which we need to convert individual records into a nested collection yet remaining as separate element. Usual addition of tuples, generally adds the contents and hence flattens the resultant container, this is usually undesired.

Prerequisite knowledge for Complete understanding and learning of Topic: (Max. Four important topics)

- **Knowledge on Computing and Computers**
- **Knowledge on Programming C and C++**

Detailed content of the Lecture:

Using `+ operator + ", " operator` **during initialization**

In this method, we perform the usual addition of tuple elements, but while initializing tuples, we add a comma after the tuple so that they don't get flattened while addition.

```
# Python3 code to demonstrate working of

# Concatenating tuples to nested tuples

# using + operator + ", " operator during initialization

  # initialize tuples

test_tup1 = (3, 4),

test_tup2 = (5, 6),

  # printing original tuples
```

```
print("The original tuple 1 : " + str(test_tup1))

print("The original tuple 2 : " + str(test_tup2))

  # Concatenating tuples to nested tuples

# using + operator + ", " operator during initialization

res = test_tup1 + test_tup2

  # printing result

print("Tuples after Concatenating : " + str(res))
```

Output :

```
The original tuple 1 : ((3, 4), )

The original tuple 2 : ((5, 6), )

Tuples after Concatenating : ((3, 4), (5, 6))
```

Video Content / Details of website for further learning (if any):

https://www.tutorialspoint.com/python/index.htm

Important Books/Journals for further learning including the page nos.:

Allen B. Downey, “Think Python: How to Think Like a Computer Scientist”, Second Edition, Shroff/O‘Reilly Publishers, 2016 (**Page No:224**)

Course Faculty

Verified by HOD

DESIGNING YOUR FUTURE
Estd. 2000

MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu

IQAC

LECTURE HANDOUTS

L27

MCA

I / I

Course Name with Code : **19CAB04 – Problem Solving and Python Programming**

Course Faculty : **Dr.M.Moorthy**

Unit: III - SIMPLE DATA STRUCTURES IN PYTHON **Date of Lecture:** 13.03.2021

Topic of Lecture: Sets data structure

Introduction : (Maximum 5 sentences)

A *set* in Python is an unordered collection of objects used when membership and uniqueness in the set are main things you need to know about that object. Like dictionary keys, the items in a set must be immutable and hashable. This means that ints, floats, strings, and tuples can be members of a set, but lists, dictionaries, and sets themselves can't.

Prerequisite knowledge for Complete understanding and learning of Topic: (Max. Four important topics)

- **Knowledge on Computing and Computers**
- **Knowledge on Programming C and C++**

Detailed content of the Lecture:

Set operations

In addition to the operations that apply to collections in general, such as in, len, and iteration in for loops, sets have several set-specific operations:

```
>>> x = set([1, 2, 3, 1, 3, 5])
>>> x
{1, 2, 3, 5}

>>> x.add(6)
>>> x
{1, 2, 3, 5, 6}

>>> x.remove(5)
>>> x
{1, 2, 3, 6}

>>> 1 in x
True

>>> 4 in x
False

>>> y = set([1, 7, 8, 9])
>>> x | y
{1, 2, 3, 6, 7, 8, 9}

>>> x & y
{1}

>>> x ^ y
{2, 3, 6, 7, 8, 9}
>>>
```

- You can create a set by using set on a sequence, such as a list
- When a sequence is made into a set, duplicates are removed
- After creating a set by using the set function, you can use add and remove to change the elements in the set.
- The in keyword is used to check for membership of an object in a set
- You can also use | to get the union, or combination, of two sets, & to get their intersection and ^ to find their symmetric difference—that is, elements that are in one set or the other but not both.

Video Content / Details of website for further learning (if any):

https://www.tutorialspoint.com/python/index.htm

Important Books/Journals for further learning including the page nos.:
https://www.geeksforgeeks.org/sets-in-python/

Course Faculty

Verified by HOD

MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu

Estd. 2000

LECTURE HANDOUTS

L28

MCA

I / I

Course Name with Code : **19CAB04 – Problem Solving and Python Programming**

Course Faculty : **Dr.M.Moorthy**

Unit: IV - STRINGS, DICTIONARIES, MODULES **Date of Lecture:** 16.03.2021

Topic of Lecture: Indexing and Traversing Strings

Introduction : (Maximum 5 sentences)

A string is a list of characters in order.

A character is anything you can type on the keyboard in one keystroke,
like a letter, a number, or a backslash.

```
word = "Hello World"
>>> print word
Hello World
```

Prerequisite knowledge for Complete understanding and learning of Topic: (Max. Four important topics)

- **Knowledge on Computing and Computers**
- **Knowledge on Programming C and C++**

Detailed content of the Lecture:

Creation

```
word = "Hello World"
>>> print word
Hello World
```

Accessing

Use [ ] to access characters in a string

```
word = "Hello World"
letter=word[0]
>>> print letter
```

```
H
```

Length

```
word = "Hello World"
>>> len(word)
11
```

Finding

```
word = "Hello World">>> print word.count('l') # count how many times l is in the string
3
>>> print word.find("H") # find the word H in the string
0
>>> print word.index("World") # find the letters World in the string
6
```

Count

```
s = "Count, the number of spaces"
>>> print s.count(' ')
8
```

Slicing

Use [# : #] to get set of letter

Keep in mind that python, as many other languages, starts to count from 0!!

```
word = "Hello World"
print word[0] #get one char of the word
print word[0:1] #get one char of the word (same as above)
print word[0:3] #get the first three char
print word[:3] #get the first three char
print word[-3:] #get the last three char
print word[3:] #get all but the three first char
print word[:-3] #get all but the three last character
```

Video Content / Details of website for further learning (if any):
https://www.tutorialspoint.com/python/index.htm

Important Books/Journals for further learning including the page nos.:

Allen B. Downey, “Think Python: How to Think Like a Computer Scientist”, Second Edition, Shroff/O‘Reilly Publishers, 2016 (**Page No:145**)

Course Faculty

Verified by HOD

MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu

DESIGNING YOUR FUTURE
Estd. 2000

IQAC

LECTURE HANDOUTS

L29

MCA

I / I

Course Name with Code : **19CAB04 – Problem Solving and Python Programming**

Course Faculty : **Dr.M.Moorthy**

Unit: IV - STRINGS, DICTIONARIES, MODULES **Date of Lecture:** 17.03.2021

Topic of Lecture: String Concatenating and Appending

Introduction : (Maximum 5 sentences)

String concatenation means add strings together.

Use the + character to add a variable to another variable.

Python string object is immutable. So every time we use + operator to concatenate two strings, a new string is created.

If we have to append many strings, using + operator will unnecessarily create many temporary strings before we have the final result.

Prerequisite knowledge for Complete understanding and learning of Topic: (Max. Four important topics)

- **Knowledge on Computing and Computers**
- **Knowledge on Programming C and C++**

Detailed content of the Lecture:

Merge variable a with variable b into variable c:

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

Python String Append

Let’s look at a function to concatenate a string ‘n’ times.

```
def str_append(s, n):

    output = "
    i = 0
    while i < n:
       output += s
       i = i + 1
   return output
```

Video Content / Details of website for further learning (if any):

https://www.tutorialspoint.com/python/index.htm

Important Books/Journals for further learning including the page nos.:

Kenneth A. Lambert, “Fundamentals of Python: First Programs”, Cengage Learning, 2012. (**Page No:65**)

Course Faculty

Verified by HOD

MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu

Estd. 2000

LECTURE HANDOUTS

L30

MCA

I / I

Course Name with Code : **19CAB04 – Problem Solving and Python Programming**

Course Faculty : **Dr.M.Moorthy**

Unit: IV - STRINGS, DICTIONARIES, MODULES **Date of Lecture:** 19.03.2021

Topic of Lecture: String Multiplying and Formatting

Introduction : (Maximum 5 sentences)
String multiplication in python is achieved using string with multiplication operator as follows
s='hello'
Print(s*3)
Will print hellohellohello
and string formatting is done using format function.

Prerequisite knowledge for Complete understanding and learning of Topic:
(Max. Four important topics)

- **Knowledge on Computing and Computers**
- **Knowledge on Programming C and C++**

Detailed content of the Lecture:
String multiplication in python is achieved using string with multiplication operator as follows
s='hello'
Print(s*3)
Will print hellohellohello

Syntax : { } .format(value)
Parameters :
(value) : Can be an integer, floating point numeric constant, string, characters or even variables.
Returntype : Returns a formatted string with the value passed as parameter in the placeholder position.

Output :
GeeksforGeeks, A computer science portal for geeks.
This article is written in Python

Hello, I am 18 years old!

Video Content / Details of website for further learning (if any): **https://www.tutorialspoint.com/python/index.htm**
Important Books/Journals for further learning including the page nos.: Kenneth A. Lambert, “Fundamentals of Python: First Programs”, Cengage Learning, 2012. (**Page No:93**)

Course Faculty

Verified by HOD

LECTURE HANDOUTS

L31

MCA

I / I

Course Name with Code : **19CAB04 – Problem Solving and Python Programming**

Course Faculty : **Dr.M.Moorthy**

Unit: IV - STRINGS, DICTIONARIES, MODULES **Date of Lecture:** 20.03.2021

Topic of Lecture: Built-in String Functions

Introduction : (Maximum 5 sentences)

Python has a set of built-in methods that you can use on strings.

Note: All string methods returns new values. They do not change the original string.

Prerequisite knowledge for Complete understanding and learning of Topic: (Max. Four important topics)

- **Knowledge on Computing and Computers**
- **Knowledge on Programming C and C++**

Detailed content of the Lecture:

Function	Description
format()	It’s used to create a formatted string from the template string and the supplied values.
split()	Python string split() function is used to split a string into the list of strings based on a delimiter.
join()	This function returns a new string that is the concatenation of the strings in iterable with string object as a delimiter.
strip()	Used to trim whitespaces from the string object.
format_map()	Python string format_map() function returns a formatted version of the string using substitutions from the mapping provided.
upper()	We can convert a string to uppercase in Python using str.upper() function.
lower()	This function creates a new string in lowercase.
replace()	Python string replace() function is used to create a new string by replacing some parts

	of another string.
find()	Python String find() method is used to find the index of a substring in a string.
translate()	Python String translate() function returns a new string with each character in the string replaced using the given translation table.

Video Content / Details of website for further learning (if any):

https://www.tutorialspoint.com/python/index.htm

Important Books/Journals for further learning including the page nos.:

Kenneth A. Lambert, “Fundamentals of Python: First Programs”, Cengage Learning, 2012. (**Page No:138**)

Course Faculty

Verified by HOD

MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu

Estd. 2000

LECTURE HANDOUTS

L32

MCA

I / I

Course Name with Code : **19CAB04 – Problem Solving and Python Programming**

Course Faculty : **Dr.M.Moorthy**

Unit: IV - STRINGS, DICTIONARIES, MODULES **Date of Lecture:** 23.03.2021

Topic of Lecture: Dictionary Creation and Access

Introduction : (Maximum 5 sentences)

Dictionaries are used to store data values in key:value pairs.

A dictionary is a collection which is ordered*, changeable and does not allow duplicates.

As of Python version 3.7, dictionaries are *ordered*. In Python 3.6 and earlier, dictionaries are *unordered*.

Dictionaries are written with curly brackets, and have keys and values:

Prerequisite knowledge for Complete understanding and learning of Topic: (Max. Four important topics)

- **Knowledge on Computing and Computers**
- **Knowledge on Programming C and C++**

Detailed content of the Lecture:

Accessing Elements from Dictionary

While indexing is used with other data types to access values, a dictionary uses keys. Keys can be used either inside square brackets [] or with the get() method.

If we use the square brackets [], KeyError is raised in case a key is not found in the dictionary. On the other hand, the get() method returns None if the key is not found.

```
# get vs [] for retrieving elements
my_dict = {'name': 'Jack', 'age': 26}

# Output: Jack
print(my_dict['name'])

# Output: 26
```

```
print(my_dict.get('age'))

# Trying to access keys which doesn't exist throws error
# Output None
print(my_dict.get('address'))

# KeyError
print(my_dict['address'])
```

Output

```
Jack
26
None
Traceback (most recent call last):
  File "<string>", line 15, in <module>
    print(my_dict['address'])
KeyError: 'address'
```

Video Content / Details of website for further learning (if any):

https://www.tutorialspoint.com/python/index.htm

Important Books/Journals for further learning including the page nos.:

Allen B. Downey, “Think Python: How to Think Like a Computer Scientist”, Second Edition, Shroff/O‘Reilly Publishers, 2016 (**Page No:198**)

Course Faculty

Verified by HOD

MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu

DESIGNING YOUR FUTURE
Estd. 2000

IQAC

LECTURE HANDOUTS

L33

MCA

I / I

Course Name with Code : **19CAB04 – Problem Solving and Python Programming**

Course Faculty : **Dr.M.Moorthy**

Unit: IV - STRINGS, DICTIONARIES, MODULES **Date of Lecture:** 24.03.2021

Topic of Lecture: Built-in Dictionary functions

Introduction : (Maximum 5 sentences)
Python has a set of built-in methods that you can use on dictionaries. Some of the functions are keys(), values(), and items().

Prerequisite knowledge for Complete understanding and learning of Topic:
(Max. Four important topics)

- **Knowledge on Computing and Computers**
- **Knowledge on Programming C and C++**

Detailed content of the Lecture:

Method	Description
clear()	Removes all the elements from the dictionary
copy()	Returns a copy of the dictionary
fromkeys()	Returns a dictionary with the specified keys and value
get()	Returns the value of the specified key

items()	Returns a list containing a tuple for each key value pair
keys()	Returns a list containing the dictionary's keys
pop()	Removes the element with the specified key
popitem()	Removes the last inserted key-value pair
setdefault()	Returns the value of the specified key. If the key does not exist: insert the key, with the specified value
update()	Updates the dictionary with the specified key-value pairs
values()	Returns a list of all the values in the dictionary

Video Content / Details of website for further learning (if any):

https://www.tutorialspoint.com/python/index.htm

Important Books/Journals for further learning including the page nos.:

Kenneth A. Lambert, “Fundamentals of Python: First Programs”, Cengage Learning, 2012. (**Page No:178**)

Course Faculty

Verified by HOD

MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu

Estd. 2000

LECTURE HANDOUTS

L34

MCA

I / I

Course Name with Code : **19CAB04 – Problem Solving and Python Programming**

Course Faculty : **Dr.M.Moorthy**

Unit: IV - STRINGS, DICTIONARIES, MODULES **Date of Lecture:** 26.03.2021

Topic of Lecture: Module Loading and Execution

Introduction : (Maximum 5 sentences)

Modules which are designed to be imported by other code won't execute any code, but only expose its top-level names as attributes to the imported object.

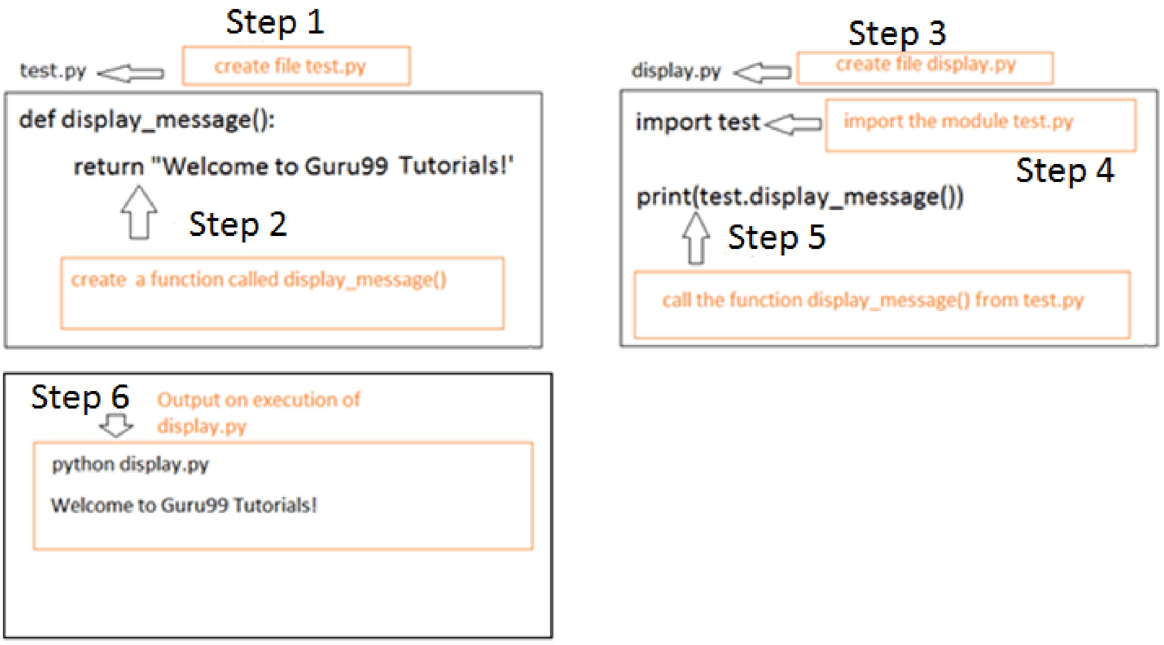
It is also possible to design *dual-mode code* Python modules which could be used for both - importing and running as a top-level script.

Prerequisite knowledge for Complete understanding and learning of Topic: (Max. Four important topics)

- **Knowledge on Computing and Computers**
- **Knowledge on Programming C and C++**

Detailed content of the Lecture:

A module is a file with python code. The code can be in the form of variables, functions, or class defined. The filename becomes the module name.



Step 1
test.py
create file test.py
Step 2
create a function called display_message()
Step 3
display.py
create file display.py
import the module test.py
Step 4
Step 5
call the function display_message() from test.py
Step 6
Output on execution of display.py

```
def display_message():
    return "Welcome to Guru99 Tutorials!'
```

```
import test
print(test.display_message())
```

```
python display.py
Welcome to Guru99 Tutorials!
```

Video Content / Details of website for further learning (if any):

https://www.tutorialspoint.com/python/index.htm

Important Books/Journals for further learning including the page nos.:

Allen B. Downey, "Think Python: How to Think Like a Computer Scientist", Second Edition, Shroff/O'Reilly Publishers, 2016 (**Page No:262**)

Course Faculty

Verified by HOD

MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu

DESIGNING YOUR FUTURE
Estd. 2000

IQAC

LECTURE HANDOUTS

L35

MCA

I / I

Course Name with Code : **19CAB04 – Problem Solving and Python Programming**

Course Faculty : **Dr.M.Moorthy**

Unit: IV - STRINGS, DICTIONARIES, MODULES **Date of Lecture:** 27.03.2021

Topic of Lecture: Packages

Introduction : (Maximum 5 sentences)
A package is a hierarchical file directory structure that defines a single Python application environment that consists of modules and subpackages and sub-subpackages, and so on.

Prerequisite knowledge for Complete understanding and learning of Topic:
(Max. Four important topics)

- **Knowledge on Computing and Computers**
- **Knowledge on Programming C and C++**

Detailed content of the Lecture:

Create a directory named Phone

Enter into that directory and create a file Pots.py in it
with the following code

```
def Pots():
   print("I'm Pots Phone")
```

Similar way, we have another two files having different functions with the same name as above −

Phone\Isdn.py file having function Isdn()
as follows

```
def Isdn():
   print("I'm ISDN phone")
```

Phone\G3.py file having function G3()
as follows

```
def G3():
   print("I'm 3G phone")
```

Now, create one more file __init__.py in Phone directory −
Phone__init__.py
To make all of your functions available when you've imported Phone, you need to put explicit

import statements in __init__.py as follows −

```
from Pots import Pots
from Isdn import Isdn
from G3 import G3
```

After you add these lines to __init__.py, you have all of these classes available when you import the Phone package.

```
# Now import your Phone Package.
import Phone
Phone.Pots()
Phone.Isdn()
Phone.G3()
```

When the above code is executed, it produces the following result −

```
I'm Pots Phone
I'm 3G Phone
I'm ISDN Phone
```

Video Content / Details of website for further learning (if any):

https://www.tutorialspoint.com/python/index.htm

Important Books/Journals for further learning including the page nos.:
https://www.tutorialsteacher.com/python/python-package

Course Faculty

Verified by HOD

LECTURE HANDOUTS

L36

MCA

I / I

Course Name with Code : **19CAB04 – Problem Solving and Python Programming**

Course Faculty : **Dr.M.Moorthy**

Unit: IV - STRINGS, DICTIONARIES, MODULES **Date of Lecture:** 30.03.2021

Topic of Lecture: Python Standard Libraries

Introduction : (Maximum 5 sentences)
The Python Standard Library contains a wide range of modules to deal with everyday programming and is included with the standard version of Python, meaning no additional installation is required. It provides modules for such tasks as interacting with the operating system, reading and writing CSV files, generating random numbers, and working with dates and time.

Prerequisite knowledge for Complete understanding and learning of Topic:
(Max. Four important topics)

- **Knowledge on Computing and Computers**
- **Knowledge on Programming C and C++**

Detailed content of the Lecture:

1. Zipfile

The **zipfile** library provides tools to easily work with zip files. It allows you to create, read, and write zip files directly in **Python**, without being necessary to use an external program.

2. Random

Random numbers play an important role in artificial intelligence and data science. We use random numbers to shuffle the training data prior to each epoch, to set initial weights in a neural network, to separate training and testing data, or to conduct A/B testing.

3. Os

The **os** module provides functions for interacting with the **operating system**. It contains numerous tools for working with **directories**, **paths**, and **files**. In this article, we will only cover some of the functionalities provided by **os**.

4. Time

Measure execution time

We can measure the execution time of a block of code using the **time.time()** function. This function returns the number of seconds passed since the epoch. The epoch is the point where time begins (**time.time()** would return 0) and is platform dependent, being on Windows systems January 1, 1970, 00:00:00 (UTC).

5. Datetime

The **datetime** library provides many tools for working with dates and times in **Python**. We can easily get the current time, subtract two dates, or convert dates into custom-formatted strings using the **datetime** module.

6. Csv

A **comma-separated values** (CSV) file is a common format used to transfer information. The information is structured as a **table** where each **row** contains a **record** and each **column** a **field**, being the **fields** separated by **commas**. Although commas are the most common **separator**, we can use other delimiters such as spaces or tabs.

7. Glob

The **glob** module allows you to find a list of files and paths matching a given pattern by using **wildcard** notation. A **wildcard** is a special character we can use to select multiple similar names.

8. Difflib

The **difflib** module contains a variety of functions and classes for comparing sequences, being especially helpful for computing differences between texts and strings.

Video Content / Details of website for further learning (if any):

https://www.tutorialspoint.com/python/index.htm

Important Books/Journals for further learning including the page nos.:
https://docs.python.org/3/library/

Course Faculty

Verified by HOD

LECTURE HANDOUTS

L37

MCA

I / I

Course Name with Code : **19CAB04 – Problem Solving and Python Programming**

Course Faculty : **Dr.M.Moorthy**

Unit: V - FILE HANDLING AND EXCEPTION HANDLING Date of Lecture: 31.03.2021

Topic of Lecture: File Path

Introduction: (Maximum 5 sentences)
A file has two key properties: a *filename* (usually written as one word) and a *path*. The path specifies the location of a file on the computer.

Prerequisite knowledge for Complete understanding and learning of Topic: (Max. Four important topics)

- **Knowledge on Computing and Computers**
- **Knowledge on Programming C and C++**

Detailed content of the Lecture:

ABSOLUTE VS. RELATIVE PATHS

There are two ways to specify a file path.

- An *absolute path*, which always begins with the root folder
- A *relative path*, which is relative to the program's current working directory

There are also the *dot* (.) and *dot-dot* (..) folders. These are not real folders but special names that can be used in a path. A single period ("dot") for a folder name is shorthand for "this directory." Two periods ("dot-dot") means "the parent folder."

Video Content / Details of website for further learning (if any):
https://www.tutorialspoint.com/python/index.htm

Important Books/Journals for further learning including the page nos.:
Allen B. Downey, "Think Python: How to Think Like a Computer Scientist", Second Edition, Shroff/O'Reilly Publishers, 2016 (**Page No:256**)

Course Faculty

Verified by HOD

MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu

Estd. 2000

LECTURE HANDOUTS

L38

MCA

I / I

Course Name with Code : **19CAB04 – Problem Solving and Python Programming**

Course Faculty : **Dr.M.Moorthy**

Unit: V - FILE HANDLING AND EXCEPTION HANDLING Date of Lecture: 02.04.2021

Topic of Lecture: Opening and Closing Files

Introduction : (Maximum 5 sentences)

Opening a file refers to getting the file ready either for reading or for writing. This can be done using the `open()` function.

Python has a `close()` method to close a file. The close() method can be called more than once and if any opertion is performed on a closed file it raises a `ValueError`.

Prerequisite knowledge for Complete understanding and learning of Topic: (Max. Four important topics)

- **Knowledge on Computing and Computers**
- **Knowledge on Programming C and C++**

Detailed content of the Lecture:

Opening a file refers to getting the file ready either for reading or for writing. This can be done using the `open()` function.

Python has a `close()` method to close a file. The close() method can be called more than once and if any opertion is performed on a closed file it raises a `ValueError`.

Syntax: *File_object = open("File_Name", "Access_Mode")*

This function returns a file object and takes two arguments, one that accepts the file name and another that accepts the mode(Access Mode).

```
# open the file using open() function
file = open("sample.txt")

# Reading from file
print(file.read())

# closing the file
file.close()
```

```
# open the file using open() function
file = open("sample.txt", 'w')
```

```
# Overwrite the file
file.write(" All content has been overwritten !")
```

Video Content / Details of website for further learning (if any):

https://www.tutorialspoint.com/python/index.htm

Important Books/Journals for further learning including the page nos.:

Allen B. Downey, “Think Python: How to Think Like a Computer Scientist”, Second Edition, Shroff/O‘Reilly Publishers, 2016 (**Page No:253**)

Course Faculty

Verified by HOD

MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu

LECTURE HANDOUTS

L39

MCA

I / I

Course Name with Code : **19CAB04 – Problem Solving and Python Programming**

Course Faculty : **Dr.M.Moorthy**

Unit: V - FILE HANDLING AND EXCEPTION HANDLING **Date of Lecture:** 03.04.2021

Topic of Lecture: Reading Files

Introduction : (Maximum 5 sentences)

There are three ways to read data from a text file. They are read(), readline(), and readines()

Prerequisite knowledge for Complete understanding and learning of Topic:
(Max. Four important topics)

- **Knowledge on Computing and Computers**
- **Knowledge on Programming C and C++**

There are three ways to read data from a text file.

1. **read() :** Returns the read bytes in form of a string. Reads n bytes, if no n specified, reads the entire file.
   File_object.read([n])
2. **readline() :** Reads a line of the file and returns in form of a string.For specified n, reads at most n bytes. However, does not reads more than one line, even if n exceeds the length of the line.
   File_object.readline([n])
3. **readlines() :** Reads all the lines and return them as each line a string element in a list.
   File_object.readlines()

Note: '\n' is treated as a special character
For reading a file the modes are

1. **Read Only ('r') :** Open text file for reading. The handle is positioned at the beginning of the file. If the file does not exists, raises I/O error. This is also the default mode in which file is opened.

2. **Read and Write ('r+') :** Open the file for reading and writing. The handle is positioned at the beginning of the file. Raises I/O error if the file does not exists.

There are two types of files that can be handled in python, normal text files and binary files (written in binary language,0s and 1s).

- **Text files:** In this type of file, Each line of text is terminated with a special character called EOL (End of Line), which is the new line character ('\n') in python by default.
- **Binary files:** In this type of file, there is no terminator for a line and the data is stored after converting it into machine understandable binary language.

Video Content / Details of website for further learning (if any):

https://www.tutorialspoint.com/python/index.htm

Important Books/Journals for further learning including the page nos.:

Allen B. Downey, "Think Python: How to Think Like a Computer Scientist", Second Edition, Shroff/O'Reilly Publishers, 2016 (**Page No:253**)

Course Faculty

Verified by HOD

LECTURE HANDOUTS

L40

MCA

I / I

Course Name with Code : **19CAB04 – Problem Solving and Python Programming**

Course Faculty : **Dr.M.Moorthy**

Unit: V - FILE HANDLING AND EXCEPTION HANDLING Date of Lecture: 06.04.2021

Topic of Lecture: Writing Files

Introduction : (Maximum 5 sentences)

There are two ways to write in a file.

1. **write() :** Inserts the string str1 in a single line in the text file.
   File_object.write(str1)
2. **writelines() :** For a list of string elements, each string is inserted in the text file.Used to insert multiple strings at a single time.
   File_object.writelines(L) for L = [str1, str2, str3]

Prerequisite knowledge for Complete understanding and learning of Topic: (Max. Four important topics)

- **Knowledge on Computing and Computers**
- **Knowledge on Programming C and C++**

Detailed content of the Lecture:

There are two ways to write in a file.

1. **write() :** Inserts the string str1 in a single line in the text file.
   File_object.write(str1)
2. **writelines() :** For a list of string elements, each string is inserted in the text file.Used to insert multiple strings at a single time.
   File_object.writelines(L) for L = [str1, str2, str3]

```
file1 = open("myfile.txt","w")
L = ["This is Delhi \n","This is Paris \n","This is London \n"]

# \n is placed to indicate EOL (End of Line)
file1.write("Hello \n")
file1.writelines(L)
file1.close() #to change file access modes
```

Video Content / Details of website for further learning (if any):

https://www.tutorialspoint.com/python/index.htm

Important Books/Journals for further learning including the page nos.:

Allen B. Downey, “Think Python: How to Think Like a Computer Scientist”, Second Edition, Shroff/O‘Reilly Publishers, 2016 (**Page No:253**)

Course Faculty

Verified by HOD

LECTURE HANDOUTS **L41**

MCA **I / I**

Course Name with Code : **19CAB04 – Problem Solving and Python Programming**

Course Faculty : **Dr.M.Moorthy**

Unit: V - FILE HANDLING AND EXCEPTION HANDLING Date of Lecture: 07.04.2021

Topic of Lecture: File Position

Introduction : (Maximum 5 sentences)
Python file method seek() sets the file's current position at the offset.

Syntax

Following is the syntax for **seek()** method −

fileObject.seek(offset[, whence])

Prerequisite knowledge for Complete understanding and learning of Topic: (Max. Four important topics)

- **Knowledge on Computing and Computers**
- **Knowledge on Programming C and C++**

Detailed content of the Lecture:

Syntax

Following is the syntax for **seek()** method −

fileObject.seek(offset[, whence])

Parameters

- **offset** − This is the position of the read/write pointer within the file.
- **whence** − This is optional and defaults to 0 which means absolute file positioning, other values are 1 which means seek relative to the current position and 2 means seek relative to the file's end.

Return Value

This method does not return any value.

The `tell()` method returns the current file position in a file stream.

Tip: You can change the current file position with the seek() method.

Syntax

file.tell()

Video Content / Details of website for further learning (if any):

https://www.tutorialspoint.com/python/index.htm

Important Books/Journals for further learning including the page nos.:

https://www.geeksforgeeks.org/python-seek-function/

Course Faculty

Verified by HOD

MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu

Estd. 2000

LECTURE HANDOUTS

L42

MCA

I / I

Course Name with Code : **19CAB04 – Problem Solving and Python Programming**

Course Faculty : **Dr.M.Moorthy**

Unit: V - FILE HANDLING AND EXCEPTION HANDLING **Date of Lecture:** 09.04.2021

Topic of Lecture: Errors in Files

Introduction : (Maximum 5 sentences)
File exceptions in Python
If an exception of type IOError occurs, it executes the statements in the except branch and then continues. You can use multiple except blocks to handle different kinds of exceptions. If your program detects an error condition, you can make it raise an exception.

Prerequisite knowledge for Complete understanding and learning of Topic: (Max. Four important topics)

- **Knowledge on Computing and Computers**
- **Knowledge on Programming C and C++**

Detailed content of the Lecture:

trying to open a nonexistent file:

```
>>> f = open("Idontexist", "r")
IOError: [Errno 2] No such file or directory: 'Idontexist'
```

```
filename = raw_input('Enter a file name: ')
try:
  f = open (filename, "r")
except IOError:
  print 'There is no file named', filename
```

```
create_new = "y"

try:
    infile = open("menu.txt", "r")
    print("The current menu on file is:")
    for menuitem in infile:
        print(menuitem.rstrip())
    infile.close()
    print()
    create_new = input("Would you like to throw away this menu " +
                      "and create a new one? (y/n): ")
except:
    pass

if create_new == "y":
    outfile = open("menu.txt", "w")
    food = input("What do you want to eat? (Enter nothing to quit.) ")
    while food != "":
        outfile.write(food + "\n")
        food = input("What do you want to eat? (Enter nothing to quit.) ")
    outfile.close()
```

Video Content / Details of website for further learning (if any):

https://www.tutorialspoint.com/python/index.htm

Important Books/Journals for further learning including the page nos.:

https://www.greenteapress.com/thinkpython/thinkCSpy/html/chap11.html

Course Faculty

Verified by HOD

LECTURE HANDOUTS

L43

MCA

I / I

Course Name with Code : **19CAB04 – Problem Solving and Python Programming**

Course Faculty : **Dr.M.Moorthy**

Unit: V - FILE HANDLING AND EXCEPTION HANDLING **Date of Lecture:** 10.04.2021

Topic of Lecture: Exceptions in Files

Introduction : (Maximum 5 sentences)

An exception can be defined as an unusual condition in a program resulting in the interruption in the flow of the program.

Whenever an exception occurs, the program stops the execution, and thus the further code is not executed.

Therefore, an exception is the run-time errors that are unable to handle to Python script.

An exception is a Python object that represents an error

Prerequisite knowledge for Complete understanding and learning of Topic: (Max. Four important topics)

- **Knowledge on Computing and Computers**
- **Knowledge on Programming C and C++**

Detailed content of the Lecture:

Common Exceptions

Python provides the number of built-in exceptions, but here we are describing the common standard exceptions. A list of common exceptions that can be thrown from a standard Python program is given below.

1. **ZeroDivisionError:** Occurs when a number is divided by zero.
2. **NameError:** It occurs when a name is not found. It may be local or global.
3. **IndentationError:** If incorrect indentation is given.
4. **IOError:** It occurs when Input Output operation fails.
5. **EOFError:** It occurs when the end of the file is reached, and yet operations are being performed.

Video Content / Details of website for further learning (if any):

https://www.tutorialspoint.com/python/index.htm

Important Books/Journals for further learning including the page nos.:

Allen B. Downey, “Think Python: How to Think Like a Computer Scientist”, Second Edition, Shroff/O‘Reilly Publishers, 2016 (**Page No:258**)

Course Faculty

Verified by HOD

MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu

DESIGNING YOUR FUTURE
Estd. 2000

IQAC

LECTURE HANDOUTS

L44

MCA

I / I

Course Name with Code : **19CAB04 – Problem Solving and Python Programming**

Course Faculty : **Dr.M.Moorthy**

Unit: V - FILE HANDLING AND EXCEPTION HANDLING Date of Lecture: 16.04.2021

Topic of Lecture: Exception Handling

Introduction : (Maximum 5 sentences)

An exception is an event, which occurs during the execution of a program that disrupts the normal flow of the program's instructions.
If you have some *suspicious* code that may raise an exception, you can defend your program by placing the suspicious code in a try: block.
After the try: block, include an except: statement, followed by a block of code which handles the problem as elegantly as possible.

Prerequisite knowledge for Complete understanding and learning of Topic: (Max. Four important topics)

- **Knowledge on Computing and Computers**
- **Knowledge on Programming C and C++**

Detailed content of the Lecture:

An exception is an event, which occurs during the execution of a program that disrupts the normal flow of the program's instructions.
If you have some *suspicious* code that may raise an exception, you can defend your program by placing the suspicious code in a try: block.
After the try: block, include an except: statement, followed by a block of code which handles the problem as elegantly as possible.

Syntax

Here is simple syntax of *try....except...else* blocks –

```
try:
  You do your operations here;
  ......................
except ExceptionI:
  If there is ExceptionI, then execute this block.
except ExceptionII:
  If there is ExceptionII, then execute this block.
  ......................
else:
  If there is no exception then execute this block.
```

Here are few important points about the above-mentioned syntax −

- A single try statement can have multiple except statements. This is useful when the try block contains statements that may throw different types of exceptions.
- You can also provide a generic except clause, which handles any exception.
- After the except clause(s), you can include an else-clause. The code in the else-block executes if the code in the try: block does not raise an exception.
- The else-block is a good place for code that does not need the try: block's protection.

Example

This example opens a file, writes content in the, file and comes out gracefully because there is no problem at all −

```
try:
  fh = open("testfile", "w")
  fh.write("This is my test file for exception handling!!")
except IOError:
  print "Error: can\\'t find file or read data"
else:
  print "Written content in the file successfully"
  fh.close()
```

This produces the following result −

```
Written content in the file successfully
```

Video Content / Details of website for further learning (if any):

https://www.tutorialspoint.com/python/index.htm

Important Books/Journals for further learning including the page nos.:

Allen B. Downey, “Think Python: How to Think Like a Computer Scientist”, Second Edition, Shroff/O‘Reilly Publishers, 2016 (**Page No:258**)

Course Faculty

Verified by HOD

MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu

DESIGNING YOUR FUTURE
Estd. 2000

IQAC

LECTURE HANDOUTS

L45

MCA

I / I

Course Name with Code : **19CAB04 – Problem Solving and Python Programming**

Course Faculty : **Dr.M.Moorthy**

Unit: V - FILE HANDLING AND EXCEPTION HANDLING **Date of Lecture:** 17.04.2021

Topic of Lecture: Multiple Exceptions

Introduction : (Maximum 5 sentences)

You can also use the same *except* statement to handle multiple exceptions. If you can handle different exceptions all using a single block of code, they can be grouped together in a tuple.

Prerequisite knowledge for Complete understanding and learning of Topic: (Max. Four important topics)

- **Knowledge on Computing and Computers**
- **Knowledge on Programming C and C++**

Detailed content of the Lecture:

You can also use the same *except* statement to handle multiple exceptions as follows −

```
try:
   You do your operations here;
   ......................
except(Exception1[, Exception2[,...ExceptionN]]]):
   If there is any exception from the given exception list,
   then execute this block.
   ......................
else:
   If there is no exception then execute this block.
```

```
Example:
try:
    client_obj.get_url(url)
except (URLError, ValueError):
    client_obj.remove_url(url)
except SocketTimeout:
    client_obj.handle_url_timeout(url)
```

Video Content / Details of website for further learning (if any):

https://www.tutorialspoint.com/python/index.htm

Important Books/Journals for further learning including the page nos.:

https://www.geeksforgeeks.org/multiple-exception-handling-in-python/

Course Faculty

Verified by HOD