| **ITM (SLS) Baroda University** | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **School of Computer Science Engineering & Technolgy** | | | | | | | |
| **Teaching Scheme of B.Tech (Cyber Security & Networking) ALL SEMESTERS EFFECTIVE FROM 2022-23** | | | | | | | |
|  | | | | | | | |
| **Semester 1** | | | | | | | |
| **Sr#** | **Course**  **Code** | **Course Name** | **Teaching Scheme Per Week (in Hrs.)** | | | **Hours** | **Credit** |
| **L** | **T** | **P** |
| 1 | C2110C1 | Programming in Python-1 | 3 | 0 | 4 | 7 | 5 |
| 2 | S2110C1 | Discrete Mathematics with Python | 4 | 0 | 2 | 6 | 5 |
| 3 |  | C2110C2 R Programming for Data | 3 | 0 | 2 | 5 | 4 |
| 4 |  | C2110C3 Web Technologies: HTML, CSS, JS, PHP | 2 | 0 | 4 | 6 | 4 |
|  |  | **Total** | **12** | **0** | **12** | **24** | **18** |
|  |  | **Project-Based Learning** |  |  |  |  |  |
| 5 | C2110A1 | PBL1.1:- MIT Inventor | 0 | 2 | 0 | 2 | 2 |
| 6 |  | C2110A2 PBL1.2:- Intel 8085 | 0 | 2 | 0 | 2 | 2 |
|  |  | **Total** | **12** | **4** | **12** | **28** | **22** |
| **Semester 2** | | | | | | | |
| **Sr#** | **Course**  **Code** | **Course Name** | **Teaching Scheme Per Week (in Hrs.)** | | | **Hours** | **Credit** |
| **L** | **T** | **P** |
| 1 | C2210C1 | Programming in C | 3 | 0 | 4 | 7 | 5 |
| 2 | C2210C2 | Data Structures and Algorithms-1 | 3 | 0 | 2 | 5 | 4 |
| 3 | C2210C3 | Computer Graphics | 4 | 0 | 2 | 6 | 5 |

| 4 | C2210C4 | Programming in Python-2 | 2 | 0 | 4 | 6 | 4 |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | **TOTAL** | **12** | **0** | **12** | **24** | **18** |
|  |  | **Project-Based Learning** |  |  |  |  |  |
| 5 | C2230A1 | PBL 2.1: Desktop OS Security: Windows and Ubuntu | 0 | 2 | 0 | 2 | 2 |
| 6 |  | C2230A2 PBL 2.2: Application Security | 0 | 2 | 0 | 2 | 2 |
|  |  | **Total** | **12** | **4** | **12** | **28** | **22** |
| **Semester 3** | | | | | | | |
| **Sr#** | **Course**  **Code** | **Course Name** | **Teaching Scheme Per Week (in Hrs.)** | | | **Hours** | **Credit** |
| **L** | **T** | **P** |
| 1 | C2310C1 | Object-oriented Programming using Java | 3 | 0 | 4 | 7 | 5 |
| 2 | C2310C2 | Computer Architecture | 3 | 0 | 2 | 5 | 4 |
| 3 | C2310C3 | Systems Software | 3 | 0 | 2 | 5 | 4 |
| 4 | C2310C4 | Database Management Systems | 3 | 0 | 4 | 7 | 5 |
|  |  | **TOTAL** | **12** | **0** | **12** | **24** | **18** |
|  |  | **Project-Based Learning** |  |  |  |  |  |
| 5 | C2330A1 | PBL-3.1: Linux Basics and Kali Pen Testing Tools | 0 | 2 | 0 | 2 | 2 |
| 6 | C2330A2 | PBL-3.2: Basic Network Administration | 0 | 2 | 0 | 2 | 2 |
|  |  | **TOTAL** | **12** | **4** | **12** | **28** | **22** |
| **Semester 4** | | | | | | | |

**Sr# Course**

| **Teaching Scheme Per Week (in Hrs.)** |
| --- |

**Code Course NameHoursL T P**

**Credit**

**Sr# Course**

**Code Course Name Hours**

**Credit**

**L T P**

| 1 | C2410C1 | Data Structures and Algorithms-2 | 4 | 0 | 2 | 6 | 5 |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 2 | C2430C1 | Cryptography & Network Security | 4 | 0 | 2 | 6 | 5 |
| 3 | C2410C3 | Operating Systems | 4 | 0 | 2 | 6 | 5 |
| 4 | C2410C4 | Computer Networking | 4 | 0 | 2 | 6 | 5 |
|  |  | **Total** | **16** | **0** | **8** | **24** | **20** |
|  |  | **Project-Based Learning** |  |  |  |  |  |
| 5 | C2430A1 | PBL-4.1: Firewalls: Configuration and Operations | 0 | 2 | 0 | 2 | 2 |
| 6 | C2430A2 | PBL-4.2: WireShark: Installation, Configuration and Operations | 0 | 2 | 0 | 2 | 2 |
|  |  | **Total** | **16** | **4** | **8** | **28** | **24** |
| **Semester 5** | | | | | | | |
| **Sr#** | **Course**  **Code** | **Course Name** | **Teaching Scheme Per Week (in Hrs.)** | | | **Hours** | **Credit** |
| **L** | **T** | **P** |
| 1 | C2510C3 | Mobile Application Development | 4 | 0 | 2 | 6 | 5 |
| 2 |  | C2530C1 Enterprise IT Security | 4 | 0 | 2 | 6 | 5 |
| 3 |  | C2530C2 Introduction to Cyber Security & Ethical Hacking | 3 | 0 | 2 | 5 | 4 |
| 4 |  | Elective-1 | 3 | 0 | 4 | 7 | 5 |
|  |  | **TOTAL** | **14** | **0** | **10** | **24** | **19** |
|  |  | **Project-Based Learning** |  |  |  |  |  |
| 5 | C2510A1 | PBL 5.1 - Javascript Framework | 0 | 2 | 0 | 2 | 2 |
| 6 | C2530A1 | PBL 5.2 - OWASP TOP - 10 Web Applications Pen. Testing | 0 | 2 | 0 | 2 | 2 |

|  |  | **Total** | **14** | **4** | **10** | **28** | **23** |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | **Elective-1 List** |  |  |  |  |  |
| 1 | C2530D1 | Machine Learning (Same as C2520C1) |  |  |  |  |  |
| 2 |  | C2530D2 Advanced Web Technologies |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
| **Semester 6** | | | | | | | |
| **Sr#** | **Course**  **Code** | **Course Name** | **Teaching Scheme Per Week (in Hrs.)** | | | **Hours** | **Credit** |
| **L** | **T** | **P** |
| 1 |  | C2630C1 UML & Agile Methodologies | 4 | 0 | 2 | 6 | 5 |
| 2 |  | C2630C2 Vulnerability Assessment and Penetration Testing | 3 | 0 | 2 | 5 | 4 |
| 3 |  | C2630C3 Mobile and Wireless Security | 3 | 0 | 2 | 5 | 4 |
| 4 |  | Elective-2 4 | | 0 | 2 | 6 | 5 |
|  |  | **Total** | **14** | **0** | **8** | **22** | **18** |
|  |  | **Project-Based Learning** |  |  |  |  |  |
| 5 | C2630A1 | PBL 6.1 - SNORT : Installation, Configuration and Operations | 0 | 2 | 0 | 2 | 2 |
| 6 | C2630A2 | PBL 6.2 - SURRICATA : Installation, Configuration and Operations | 0 | 2 | 0 | 2 | 2 |
| 7 | C2630A3 | PBL 6.3 - Wi-Fi and Bluetooth Security | 0 | 2 | 0 | 2 | 2 |
|  |  | **Total** | **14** | **6** | **8** | **28** | **24** |
|  |  | **Elective-2 List** |  |  |  |  |  |
| 1 | C2630D1 | Software Quality Assurance and Testing |  |  |  |  |  |

| 2 | C2630D2 | Design and Implementation of API |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 3 | C2630D3 | Cyber Security Using AI |  |  |  |  |  |
| **Semester 7** | | | | | | | |
| **Sr#** | **Course**  **Code** | **Course Name** | **Teaching Scheme Per Week (in Hrs.)** | | | **Hours** | **Credit** |
| **L** | **T** | **P** |
| 1 | C2730C1 | Cyber Forensics | 4 | 0 | 4 | 8 | 6 |
| 2 | C2730C2 | Secure Software Design and Development | 4 | 0 | 4 | 8 | 6 |
| 3 | C2730C3 | Malware Analysis | 3 | 0 | 2 | 5 | 4 |
| 4 |  | Elective-3 | 3 | 0 | 2 | 5 | 4 |
|  |  | **Total** | **14** | **0** | **12** | **26** | 20 |
|  |  | **Project-Based Learning** |  |  |  |  |  |
| 5 | C2730A1 | PBL 7.1 - Information Security - Laws and Standards | 0 | 2 | 0 | 2 | 2 |
|  |  | **Total** | **14** | **2** | **12** | **28** | **22** |
|  |  | **Elective-3 List** |  |  |  |  |  |
| 1 |  | C2730D1 Cloud Security |  |  |  |  |  |
| 2 |  | C2730D2 IoT Security |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
| **Semester 8** | | | | | | | |

| **Sr#** | **Course**  **Code** | **Course Name** | **Teaching Scheme Per Week (in Hrs.)** | | | **Hours** | **Credit** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **L** | **T** | **P** |
| 1 | C2830P1 | Capstone Project | 0 | 0 | 24 | 24 | 12 |
| 2 | C2830P2 | Project Reporting and Guidance | 0 | 4 | 0 | 4 | 4 |
| 3 | C2830A\_ | Project-Based Learning/MOOC (As per NEP) | 0 | 4 | 0 | 4 | 4 |
| 4 |  | C28X0A1 Environmental Studies | 2 | 0 | 0 | 2 | 2 |
|  |  | **Total** | **2** | **8** | **24** | **34** | **22** |



**ITM (SLS) Baroda University**

**School of Engineering**

**Department of Computer Science and Engineering**

**Course Name :Programming in Python-1**

**Course Type:** Core

Course code: C2110C1

| **Teaching**  **Scheme** | | | **Credits** | **Examination Marks** | | **Total**  **Marks** |
| --- | --- | --- | --- | --- | --- | --- |
| **L** | **T** | **P** | **C** | **Theory Marks Practical Marks** | |  |
| **External Internal External** | **Internal** |  |
| **3** | **0** | **4** | **5** | **40 60 20** | **30** | **150** |

**What is Python?**

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together.

Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed.

**Course overview**

In this course, students will learn how to do programming in Python. Students will learn: how Python works and its place in the world of programming languages; to work with and manipulate strings; to perform math operations; to work with Python sequences; to collect user input and output results; flow control processing; to write to, and read from, files; to write functions; to handle exception; and work with dates and times.



**Prerequisite**

This course does not require any programming background. This course helps the students to learn programming in python.

**Learning outcomes:**

After completing the course, the student shall be able to :

|  | **Course Outcome Bloom’s Level** |
| --- | --- |
| CO1  CO2 | Understand basics concepts of Python programming Understanding  Describe the Numbers, Math functions, Strings, List, Tuples  Application  and Dictionaries in Python |
| CO3 | Express different decision making statements and functions Applying |
| CO4 | Learn implementation essentials of python Creation |

CO5 Implement the specified mini-projects Creation

**Textbook:**

| **Sr.No** | **Book Name** |
| --- | --- |
| **1** | SheetalTaneja, Naveen Kumar, “Python Programming: A Modular Approach”, Pearson (2019) |
| **2** | R. NageswaraRao, “Core Python Programming”, dreamtech |

**Reference Book:**

| **Sr.No** | **Book Name** |
| --- | --- |
| **1** | ReemaThareja, “Python Programming: Using Problem Solving Approach”, Oxford University Press (2017) |



| **2** | John V Guttag, “Introduction to Computation and Programming using Python with Application to Understanding Data”, PHI (2016) |
| --- | --- |
| **3** | Martin C Brown, “Python: The Complete Reference”, McGraw-Hill Education (2018) |
| **4** | YashavantKanetkar, “Let us Python”, BPB Publication (2019) |
| **5** | Stephen Klosterman, “Data Science Projects with Python: A case study approach to successful data science projects using Python, pandas, and scikit”, Packt Publishing (2019) Kindle edition |

**Required Software:**

1. Python Version 2.7 or 3.6 https://www.python.org/downloads/

2. Google Colab: https://colab.research.google.com/notebooks/io.ipynb

**Learning Resources:**

**TedEx Videos:**

| **Sr. No** | **TEDx Video** |
| --- | --- |
| **T1** | https://www.youtube.com/watch?v=ENWVRcMGDoU–  How algorithms shape our world.- Kevin Slavin  Kevin Slavin argues that we're living in a world designed for -- and increasingly controlled by -- algorithms. In this riveting talk from TEDGlobal, he shows how these complex computer programs determine espionage tactics, stock prices, movie scripts, and architecture. Slavin also warns that we are writing code we can't understand with implications we can't control. |

**T2** World Changing: Data Science and AI | Fred Blackburn | TEDxURL

The lecture covered the key points like tremendous increase in data, real world examples of machine learning i.eAlexa, Robot Scientist, Healthcare industry, artwork and many more.Video also covered the racing trends of living and working with human intelligence and the learning pattern of human mind and machine

**Other Videos:**

| **Sr.**  **No** | **About Video** | **Link** | **Topic** |
| --- | --- | --- | --- |
| **O1** | Erik Demaine, Ronald Rivest, and SriniDevadas. | https://ocw.mit.edu/courses/electrical  engineering-and-computer-science/6-006-intro duction-to-algorithms-spring-2 008/ | Introduction to Algorithm |



|  | 6.006 Introduction to  Algorithms. Spring 2008.  Massachusetts Institute of Technology: MIT  OpenCourseWare,  https://ocw.mit.edu. License: Creative Commons  BY-NC-SA |  |  |
| --- | --- | --- | --- |
| **O2** | Lecture by Professor Jerry Cain for Programming Paradigms (CS107) in the Stanford  University Computer Science department. | https://www.youtube.com/watch?v=Ps8jOj7di A0&list=PLD28639E2FFC4B  86A | Programming Paradigm |
| **O3** | Dr. Anna Bell  (MIT 6.0001 Introduction to Computer Science and  Programming in Python, Fall 2016) | https://www.youtube.com/watch?v=RvRKT-jXvk o&list=PLUl4u3cNGP63WbdFxL8giv4yhgdMGa ZNA&index=17 | List, Tuple and Dictionary |
| **O4** | Introduction to Python by Harvard University (Lecture-06 CS50 2018) | https://www.youtube.com/watch?v=mvlTSMUNQ N4&t=1243s | Basics, Data  Types, Control Statements |

**Related MOOCs courses:**

| **Sr.No** | **MOOC Courses** |
| --- | --- |
| **M1** | “The Joy of Computing Using Python” by Prof. SudarshanIyengar, IIT Ropar 12 Weeks on NPTEL. |
| **M2** | “Programming for Everybody (Getting Started with Python)”,7 week course offered by University of Michigan and Courseera. |

**Course Outline:**

| **Unit**  **#** | **Topics Lab** | **Assignment Teaching** | **Hours** |
| --- | --- | --- | --- |



| 1 | P1  **Introduction to Programming:** Introduction to Programming Fundamentals, programming environment, principles of programming, what is debugging, text editors and debuggers, introduction to Flow-Chart and Algorithm | -- | 6 |
| --- | --- | --- | --- |
| 2 | P1  **Introduction to Python:** History, Features, Versions, Applications, Setting up path, Installation and Working with Python, Fundamentals of Python, Basic Syntax, Understanding Python variables, rules for naming identifiers and variables in python, operators and expressions, print(), type() and id() functions, taking user input using input() and raw\_input() functions. **Data Types:** Integer, Float, complex numbers, Concept of Mutable and Immutable, String Manipulation. | A1 | 8 |
| 3 | P2,P3,  **Data Types:** Lists, Tuples, Sets, Dictionaries, working with data types and their in-built functions, Logical Constructs, Boolean  P4,P5,  expressions, Looping, Python for loop, Python range(), Python  P6  Nested Loop Structures, Iteration, If-else, while loops, Break-Continue, Pass. | A1,A2 | 8 |
| 4 | P7,P8,  **Functions and Recursion:** Defining a function, calling a function, Types of functions, Function Arguments, Anonymous functions,  P9,P10  Passing Collections to a Function, Keyword and Optional Parameters, Local and global variables, Defining recursion and its application, programming through recursion | A1,A3 | 8 |
| 5 | P11  **Modules:** Importing Module, The Math Library, Random Numbers, Sys Module, OS, Date and Time module with their in-built functions | A3 | 8 |
| 6 | P12,  **File Handling:** File creation, open() and close() methods, read() and write() methods, file modes, file encoding, file object  P13  attributes, renaming and deleting files, Knowing Whether a File Exists or Not, Working with Binary Files, Appending Text to a File, Reading Text Files, File Exceptions, The with Statement, Python directory, directory methods and functions. | -- | 8 |
|  | **TOTAL** |  | **46** |



**Lab Experiments:**

| **Sr. No.** | **Program Statement** |
| --- | --- |
| **P1** | 1. Write a python program to calculate the addition of two numbers without using third variable.  2. Demonstrate the use of id(), type() and size() function in python.  3. Write a program to perform basic operations in python. |
| **P2** | 1. Write a program to find greatest among three numbers entered by user. 2. Write python code to check whether the entered number by user is even or odd. 3. Write a program to check whether the entered year is leap year or not. 4. Write a Python program to find all prime numbers within a given range. |
| **P3** | Write a program to take marks of 5 subjects from user and print obtained grade. (>=90% - A+, 70%-90% - A, 60%-70% - B+, 50%-60% - B, 35%-50% - C, <35% - F ) |
| **P4** | Write a program to implement a simple calculator. |
| **P5** | 1.Write a program for various functions of string in python.  2. To add 'ing' at the end of a given string (length should be at least 3). If the given string already ends with 'ing' then add 'ly' instead. If the string length of the given string 5. is less than 3, leave it unchanged. Sample String : 'abc' Expected Result : 'abcing' Sample String : 'string' Expected Result : 'stringly' |
| **P6** | 1. Write a program for list and its various in-built methods.  2. Write a program to calculate the sum of numbers stored in a list.  3. Write a program for tuple and its various in-built methods.  4. Write a program for dictionary and its various in-built methods.  5. Write a program for set and its various in-built methods. |
| **P7** | Write a function that takes a list of numbers as input from user and produces the corresponding cumulative list where each element at index i is the sum of elements at index j <= i. For example, Input List = [3, 5, 2, 7, 9, 4]; the Output List = [3, 8, 10, 17, 26, 30]. |



| **P8** | Write a program that takes a sentence as input from the user and computes the frequency of each letter. Use a variable of dictionary type to maintain the count. |
| --- | --- |
| **P9** | Write a function that takes a number as an input argument and returns the corresponding text in words, for example, if input is 368, the function should return ‘Three’, ‘Six’, ‘Eight’. Use a dictionary for mapping digits to their string representation. |
| **P10** | 1. Write a Python program to find factorial of a given number using recursion 2. Write a Python program to print ‘n terms of Fibonacci series using recursion. |
| **P11** | 1. Write a program to perform various functions of math module.  2. Write a program to perform various functions of random module.  3. Write a program to perform various system related functions using sys and os module.  4. Write a program to retrieve the date and time related information of a system using python. |
| **P12** | 1. Write a python program that reads a text file and changes the file by capitalizing each character of file.  2. Write a python program to append data to an existing file 'python.py'. Read data to be appended from the user. Then display the contents of entire file. |
| **P13** | 1. Read a text file in Python and print no. of lines and no. of unique words. 2. Write a python program to read line by line from a given files file1 & file2 and write into file3. |



**Assignments:**

| **Sr. No.** | **Assignment Name** |
| --- | --- |
| **A1** | **Guess the Number**  The Goal: This assignment also uses the random module in Python. The program will first randomly generate a number unknown to the user. The user needs to guess what that number is. (In other words, the user needs to be able to input information.) If the user’s guess is wrong, the program should return some sort of indication as to how wrong (e.g. The number is too high or too low). If the user guesses correctly, a positive indication should appear. You’ll need functions to check if the user input is an actual number, to see the difference between the inputted number and the randomly generated numbers, and to then compare the numbers.  Concepts to keep in mind:  Random function  Variables  Integers  Input/Output  Print  While loops  If/Else statement |



| **A2** | Perform the following task using dictionary for the campus drive of a company which required 75 minimum CGPI criteria for appearing in the interview:  a) Create the dictionary for department of CSE, Automobile, Mechanical, Civil and Electrical with name as key and their current CGPI with values.  b) Combine all the department detail into one common dictionary and named it as ITM.  c) Display the total number of students of university who are eligible for the drive.  d) Display the total number of students of university who are not eligible for the drive. |
| --- | --- |
| **A3** | **Problem: An advanced Math Learning Tool**  The program will generate just one question for each run that generate five questions and after a student answer all five, report the number of correct answers. The program should also display the time spent on test and list all the questions.  **Sample Input and Output:**  What is 9-2? 7  You are correct  What is 3-0? 3  You are correct  What is 3-2? 1  You are correct  What is 7-4? 4  You answer is wrong  7-4 should be 3  What is 7-5? 3  You answer is wrong  7-5 should be 2  Correct count is: 03  Test time is 1021 seconds |



|  | 9-2=7 Correct  3-0=3 Correct  3-2=1 Correct  7-4=4 Wrong  7-5=4 Wrong |
| --- | --- |

**Course Name : R Programming for Data**

**Course Type:** Core

**Course Code:**

C2110C2

| **Teaching**  **Scheme** | | | **Credits** | **Examination Marks Total**  **Marks** |
| --- | --- | --- | --- | --- |
| **L** | **T** | **P** | **C** | **Theory Marks Practical Marks** |
| **External Internal External Internal** |
| **4** | **1** | **2** | **6** | **40 60 20 30 150** |

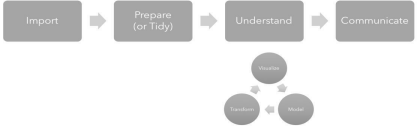
**What is R?**

R is an open source language and environment for statistical computing, datamining, modeling, and data graphics. It provides a wide variety of statistical andgraphical techniques such as linear and non-linear modeling, statistical tests, timeseries analysis, classification, and clustering.

R is one of the most used business analytics tools. For example, Facebook uses R forbehavior analysis related to status updates and profile pictures. Google uses it toanalyze advertising effectiveness and economic forecasting. Twitter leverages R fordata visualization and semantic clustering. In the 2016 data science salary surveyconducted by O'Reilly, R was ranked second in a category of programminglanguages for data science.

**Course overview**

****

In this course, students will learn how to program in R and how to use R for effectivedata analysis and visualization. “Turn raw data into understanding, insight, andknowledge” (Wickham &Grolemund, 2017, p. ix) using R to import, prepare,understand, and communicate your data findings. 

The course begins with developing a basic understanding of the R workingenvironment. Next, students will be introduced the necessary arithmetic and logicaloperators, salient functions for manipulating data, and getting help using R. Next,the common data structures, variables, and data types used in R will bedemonstrated and applied. Students will write R scripts and build R markdowndocuments to share their code others. They will utilize the various packagesavailable in R for visualization, reporting, data manipulation, and statistical analysis.

Students import data sets, transform and manipulate those datasets for variousanalytical purposes. Students will learn how to create control structures, such asloops and conditional statements to traverse, sort, merge, and evaluate data. Finally, students create interactive business applications that allow for data querying and data exploration.

**Prerequisite**

This course is designed for those who have no experience in R or programming. This class give you skills in programming in R and introduce you to 1) A new way ofthinking 2) A new language for speaking and reading (vectors, data frames, functions,objects, etc. and 3) a new syntax for writing , e.g. c(), print(), cat(), sort(), require(),subset() for data analysis and presentation.

**Course outcomes**

By the end of the course students you shall be confident and equipped with all the knowledge required to perform analytical activities in R. Specifically,

|  | **Course Outcome Bloom’s Level** |
| --- | --- |
| CO1 | **Understand** the fundamental syntax of R through  Understanding  readings, practice exercises, demonstrations, and  writing R code. |
| CO2 | **Import** a variety of data formats into R using  Understanding  RStudio |
| CO3 | **Prepare** or tidy datas for in preparation for analysis Applying |
| CO4 | **Query** data using SQL and R Application |



| CO5 | **Analyze** a data set in R and present findings using  Analyze  the appropriate R packages |
| --- | --- |
| CO6 | **Visualize** data attributes using ggplot2 and other R  Application  packages. |

**CO-PO Mapping**

|  | **PO**  **1** | **PO**  **2** | **PO**  **3** | **PO4** | **PO**  **5** | **PO**  **6** | **PO7** | **PO**  **8** | **PO**  **9** | **PO**  **PO**  **PO1**  **11**  **12**  **0** | **PO**  **PO**  **13**  **14** | **PO15** | **PO1**  **6** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| CO1 | 1 | 3 |  | 3 | 3 | 3 | 2 | 2 |  | 11 1 | 2 3 | 1 | 3 |
| CO2 | 2 | 3 | 3 | 3 | 1 | 1 | 2 | 2 | 1 |  | 2 3 | 1 | 3 |
| CO3 | 3 | 3 | 3 | 3 | 3 | 3 | 2 | 3 | 2 | 22 1 | 2 3 | 2 | 2 |
| CO4 | 1 | 3 | 3 | 3 | 3 | 3 | 3 | 2 | 2 | 1 | 3 | 1 | 2 |
| CO5 | 1 | 3 | 3 | 3 | 3 | 2 | 1 | 2 | 1 | 1 | 2 2 |  | 2 |

CO6 1 1 3 3 1 1 22 2 1



**Tutorials, video demonstrations, and exercises**

Sosulski, K. (2018). R Fundamentals.

**Textbook**

| **Sr. No** | **Book Name** |
| --- | --- |
| **1.** | Wickham, H. &Grolemund, G. (2018). for Data Science. O’Reilly: New York |
| **2.** | Norman Matloff(2011),The Art of R Programming: A Tour of Statistical Software Design ,No Starch Press, |

**Reference Book**

| **Sr. No** | **Book Name** |
| --- | --- |
| **1.** | Jared P. Lander(2013), R for Everyone: Advanced Analytics and Graphics, Addison-Wesley Data & Analytics Series |
| **2.** | Mark Gardener(2013),Beginning R – The Statistical Programming Language, Wiley |
| **3.** | Robert Knell(2013),Introductory R: A Beginner's Guide to Data Visualisation, Statistical Analysis and Programming in R, Amazon Digital South Asia Services Inc, |

**Other Videos:**

| **Sr. No** | **About Video** | **Link Topic** |
| --- | --- | --- |
| O1. | Stanford Seminar -  Expressing yourself in R,Hadley Wickham Rice University | https://www.youtube.com/watch?v=wk  Data manipulation in R  i0BqlztCo |
| O2. | MIT 15.071 The Analytics Edge, Spring 2017,MIT penCourseWare | Working with  https://www.youtube.com/watch?v=d2  CfWJkklvo  Data:History of R |

O3. MIT 15.071 The Analytics Edge, Spring 2017,MIT

penCourseWare

**Related MOOCs courses**

https://www.youtube.com/watch?v=E\_ KUHMuoPLE

Getting Started in R

| **Sr.No** | **MOOC Course** |
| --- | --- |
| **M1** | Introduction to R Software by Prof. Prof. Shalabh 8 Weeks on NPTEL. |
| **M2** | Data Science: Foundations using R, Johns Hopkins University(Coursera) |



**Required software**

• R: http://www.r-project.org/ **(FREE)**

• RStudio (additional libraries required): http://www.rstudio.com/ **(FREE)** .

**Learning resources**

• R Project: http://www.r-project.org/

• RStudio (additional libraries required): http://www.rstudio.com

• Quick-R http://www.statmethods.net/

• Google’s R Style Guide: http://google-styleguide.googlecode.com/svn/trunk/ Rguide.xml

**Course Outline**

**Class Topic Readings**

| **Unit**  **No.** | **Topics Lab** | **Assignment Hours** |
| --- | --- | --- |
| **1** | **Introduction to R programming**  P1  • What is R?  • Installing R and RStudio  • RStudio Overview  • Working in the Console  • Arithmetic Operators  • Logical Operations  • Using Functions  • Getting Help in R and  Quitting RStudio | A1 4 |
| 2  3 | **Data structures, variables, and**  P2  **data types**  • Creating Variables  • Numeric, Character and Logical  Data  • Vectors  • Data Frames  • Factors  • Sorting Numeric, Character, and  Factor Vectors  • Special Values  **R packages and scripts**  P3  • Installing and loading  packages  • Setting up your working  directory  • Downloading and  importing data | A2 4  A3 4 |



|  | • Working with missing  data  • Extracting a subset of a  data frame  • Writing R scripts  • Adding comments and  documentation  • Creating reports |  |
| --- | --- | --- |
| 4 | **Descriptive statistics in R**  P4  • Measures of central  tendency  • Measures of variability  • Skewness and kurtosis  • Summary functions,  describe functions, and  descriptive statistics by  group  • Correlations | A45 |
| 5 | **Statistical graphs**  P5  • Scatter Plots  • Box Plots  • Scatter Plots and Boxand  Whisker Plots  Together  • Histograms | A52 |
| 6 | **Working with messy data**  P6  • Messy Data  • Renaming Columns  (Variable Names)  • Attaching / Detaching  • Tabulating Data:  Constructing Simple  Frequency Tables  • Ordering Factor  Variables | A64 |
| 7 | **Iteration**  P7  • while loops  • for loops | 4 |
| 8 | **Conditional Statements**  P8  • If / else  • Boolean logical  operators | 2 |



|  |  |  |
| --- | --- | --- |
| 9 | **Writing functions**  P9  **Reporting**  • Creating functions  • Calling functions | A7 5 |
| 10 | **Data Exploration and**  P10  **Visualization**  • Using the ggplot2package to  visualizedata  • Applying themes fromggthemes  to refine andcustomize charts and  graphs  • Building data graphics  for dynamic reporting | 3 |
| 11 | **Data querying: SQL and R**  P 11  • Writing SQL statements  in R  • Using the Select, From,  Where, Is, Like, Order By,  Limit, Max, Min SQL  Functions | 4 |
| 12 | **Interactive reporting with**  P12  **Rmarkdown**  • RMarkdown basics  • Text formatting  • Code chunks  • YAML header  • Preview of  notebooks,presentations ,websites,  and dashboards | A8 4 |
| Total Hours 45 | | |

and Tutorials

La

ignmen**du**

**eLab Experiments:**

| **Sr.N**  **o** | **Practical Name** |
| --- | --- |
| P1 | Compute the following:  1. (123 - 45) / 4 + 4 \* (72 / 2.34 - 3)  2. (((20\*3)-14)^3) |



|  | 3. Absolute value of -88  4. Base 10 logarithm of 72  5. Round the square root of 50 to the fourth decimal  6. e^1.45 - 2.612  7.  a. Assign a variable year\_born to 1984  b. Assign a variable year\_current to 2014  c. Assign a variable age and compute it  d. Return True / False if person is eligible to vote in US (if age is greater than or equal to 18)  6. Given: formula for area of circle is pi\*r2 Given: Area = 100  a. Write statement to find r. (Hint: utilize “sqrt” and “pi” functions)  7. Given: went to lunch and pre-tax bill was $45.90  a. Compute subtotal: add NYC tax of 8.875%  b. Compute 15% tip on subtotal  c. Compute 20% tip on subtotal  8.  a. Assign a variable customers to 500  b. Assign a variable pizza\_price to $20  c. Assign a variable todays\_revenue (customers \* pizza\_price) and compute today’s revenue  d. Is today’s revenue greater than yesterday’s revenue of $7,000 and less than tomorrow’s projected revenue of $11,000? Show the code that would answer the following question. |
| --- | --- |
| P2 | **Dataset:** Winter Olympic Medals.  Download Link: **http://becomingvisual.com/rfundamentals/winter\_olympic.csv Data Dictionary:** Review the data dictionary for the Winter Olympic Medals data set.  **Variable Description**  Rank Rank in number of medals |



|  | NOC Name of country  Gold Number of gold medals  Silver Number of silver medals  Bronze Number of bronze medals  Total Total number of medals  Region Country Region  1. Getting to know the data  a. Import the data  b. View the data  c. How many variables are in the data frame?  d. What are the names of these variables?  e. How many countries (rows) are in the data frame?  2. Printing data  a. The first row of data  b. The last row of data  c. The first 5 rows of data  3. Creating vectors  a. Create a vector called “country\_medals” from data frame  b. Create a vector called “gold” from data frame  c. What type of variable is “gold”?  4. Create a new data frame that holds data from the region Asia  a. Call the data frame “asia”  b. How many rows and columns are in this data frame? [Hint: use dim() ] 5. Create the data frame “total\_medals”  a. Create vector “country”  b. Create vector “total\_medal\_ct” |
| --- | --- |



|  | c. Use cbind() to combine the two vectors  d. What is the type of object “total\_medals”?  6. Vector data counts  a. What are the different levels of data$Region? [Hint: use levels() ]  b. Are any of the other variables factor variables? [Hint: use str() ]  7. Subsetting  a. Create a data frame that holds countries that did not win any gold medals 8.  a. Create a vector called test\_scores with the following values 92, 75, 84, 94, 88, 89, 91  b. Create a vector called students with the following values Jerry, Monica, Felix, James, April, Ruth, Tony  c. Create a data frame with these two vectors  d. It turns out that Monica’s test was regraded and was awarded five extra points - correct this in the data frame.  e. Extract the students who got above or equal to 90%  f. Sort all the students by their test score in descending order |
| --- | --- |
| P3 | **Dataset:** Basketball data from March Madness  Download Link: **http://becomingvisual.com/rfundamentals/march\_madness.csv Data Dictionary:**  **Variable Description**  Rank Team Ranking  Previous Previous Team Ranking  School Name of the College or University  Conference NCAA Conference (30 +)  Record Overall Record |



|  | Neutral Record with games in a neutral location  Home Record with games at home  Non Div I Record with non-divison 1 games  **Write a R script to do the following:**  1. Set working directory – Hint: setwd()  2. Import the csv file  3. View the file  4. Print number of rows and columns – Hint: dim()  5. Print columns names  6. Change column names to lower case so it is easier to use Hint: names(df\_name) <- tolower(names(df\_name))  7. Explore the variable types. – Hint: str()  8. How many different conferences are there?  9. Let’s look at the difference in values of first two columns:  a. Compute a new vector called “diff” and calculate the difference in rank and previous  b. Print count and list of schools that changed 3 or more places Hint: create subset that satisfies criteria  10. Import the GDP dataset and compute the difference in GDP between 2007 and 2017 for each country.  Download: http://becomingvisual.com/rfundamentals/gdp.csv  a. Create a subset of countries that saw an increase of over one trillion dollars. |
| --- | --- |
| P4 | Create a RMarkdown (.Rmd) document that answers and addresses the following requirements. 1. Getting to know the data:  a. Import the data (http://becomingvisual.com/rfundamentals/winter\_olympic.csv) b. View the data  c. Look at column names  d. Look at dimension of data (rows and columns) |



|  | 2. Data is currently sorted by Rank. Sort data by total medals and country. Assign sorted data to a new data frame. Call it sort\_total.  3. Use describe() function to look at data.  a. If function does not work, first import library: library(Hmisc)  4. Look at some statistics  a. What is median of number of gold, silver, bronze and total medals?  b. Also look at the mean and total number of G, S, B and T medals  5. More statistics  a. For Gold, look at summary stats, including: IQR, min, max, mean, var, sd, skew b. Use summary() and describe(). (May need to install library(psych) )  6. More statistics - subset  a. Redo above statistics, this time group by Region  b. Which region won the highest mean total medals?  c. How many countries are in this Geographic Region?  d. How many countries are in the EUROPE group?  e. What is the max number of medals won? What country won the max? 7. More statistics – correlations  a. Explore correlations between Total medals and number of Gold and Bronze b. What is the correlation between Rank and Total medals? Is this expected or surprising?  8. Import the GDP dataset and compute the measures of central tendency for 2017. (Divide by a trillion, and use na.rm = TRUE when computing the measures.)  a. Find the mean  b. Find the median  c. Find the range  d. Find the quantile |
| --- | --- |
| P5 | In an R Markdown document, complete the following with the movies.csv data. Download the data from http://becomingvisual.com/rfundamentals/movies.csv 1. Getting to know the data  a. Import the data  b. View the data  c. Look at column names |



|  | d. Look at dimension of data (rows and columns)  2. Scatterplots  a. Do scatter plot of Tickets Sold and Gross (Is the trend expected?)  b. Redo scatter plot, adjusting scales, divide by 1000  c. Redo scatter plot, adjusting scales, divide by 100,000  d. Redo scatter plot, adjusting scales, divide by 1,000,000  3. What is the correlation between tickets sold and sales? Is this expected? 4. Scatterplots with lines  a. Do scatter plot with millions scale, add a regression line  b. Add label to x and y axis, add plot title label  5. Other plots  a. Do boxplot  b. Do boxplot - horizontal  c. Do histogram for type of films  d. Do histogram of gross sales. How bins are shown by default?  e. Do histogram of gross sales with 10 bins.  f. Do histogram of ticket sales. Try different bin numbers.  g. Do histogram of ticket sales (use millions unit). Add frequency count to top of bars. Add titles.  h. Do barplot of genre  6. In a R Markdown document, produce plots that describe the GDP  (http://becomingvisual.com/rfundamentals/gdp.csv )and Life Expectancy (: http://becomingvisual.com/rfundamentals/life\_expectancy.csv) during 2016 You will need to create a new data frame with these columns.  a. Create a scatter plot of GDP to Life Expectancy  b. Create a histogram of GDP  c. Create a box and whisper plot of Life Expectancy |
| --- | --- |
| P6 | Create an RMarkdown document to complete the following:  1. Getting to know the data  a. Import the data  (http://becomingvisual.com/rfundamentals/summer\_winter\_olympics.csv) b. View the data |



|  | c. Look at column names  d. Look at dimension of data (rows and columns)  2. Dealing with Data  a. Look at the column names and change names to more meaningful names. b. The data represent, in order:  1. country  2. number of summer games played, gold, silver, bronze, total,  3. number of winter games played, gold, silver, bronze and total, total  4. total (Winter + Summer) games, gold, silver, bronze, total  3. Summary  a. Use table() to find frequency of total summer games played  b. Explore the data with other variables  4. Graphs  a. Do histogram of summer games (total)  b. Do histogram of winter games (total)  c. Put above two histograms on one page  d. Do two histograms on one page: total summer, total winter medals won e. Is there a correlation between number of medals given out in winter and summer? (do plot)  f. How about number of games each country competes in. Is there correlation between winter and summer?  g. Look at distribution of each of the types of medals, by season (6 histograms on one page)  h. Redo g with different number of bins (10 instead of 20)  i. Explore data on your own  5. Merge the columns for the year 2016 for  GDP (http://becomingvisual.com/rfundamentals/gdp.csv),  Life Expectancy (http://becomingvisual.com/rfundamentals/life\_expectancy.csv), and Employment (http://becomingvisual.com/rfundamentals/employment.csv) into a new data frame and clean-up the new table.  a. Rename the appropriate columns to “country”, “gdp”, “life\_expectancy”, and “employment”. |
| --- | --- |



|  | b. Convert the employment number to percentages by dividing by 100  c. Then round life expectancy to zero decimals and employment to two decimals d. Create a frequency table for each variable  e. Draw histograms for each variable |
| --- | --- |
| P7 | Import the following AirBnb data set (http://becomingvisual.com/rfundamentals/airbnb.csv) Using a for loop count the number of AirBnbs that are in a particular neighbourhood the NYU area using the Greenwich Village and West Village neighbourhoods. |
| P8 | Import the following AirBnb data set (http://becomingvisual.com/rfundamentals/airbnb.csv) Using if / else statement, count the number of AirBnbs that are in a particular neighbourhood the NYU area using the Greenwich Village and West Village neighbourhoods. |
| P9 | 1. Create a function that computes the mean, median, min, and max values. Use this function to compute those values for the attitude data set.  2. Create a new function called checkforna that checks to see if a give value is NA and prints out the row number and column name from the following data  set: http://becomingvisual.com/rfundamentals/airbnb.csv |
| P10 | Create a ggplot for following datasets:  Data = MPG, Iris, Glass (From UCR Repository)  1. How many rows are in each dataset  2. How many columns are in each dataset?  3. Make a scatter plot of hwy vs cyL variables from mpg.  4. |



|  |  |
| --- | --- |
| P11 | SELECT all applicable data  1. The players on the San Antonio Spurs in 2014  2. Top 5 blockers in 2010  3. Top 10 combination power-forwards with the most defensive rebounds 4. Top 20 Player-seasons in the NBA 50-40-90 Club (players who have hit over 50% for FG%, 40% for 3P%, 90% for FT%, 300 field goals, 55 3-pointers, and 125 free throws) ordered by their amount of points  Top 10 oldest Milwaukee Bucks players with over 1000 points. |
| P12 | Using the http://becomingvisual.com/rfundamentals/nyuclasses.csv file, create a shiny app that displays a box plot of the student grades based on the assignment selection that looks like the image below: |



|  | 1. Revise the nyuclasses app to provide a default view of most recent distributions by most recent assignment due date.  2. Revise app to include a selector by one or more students.  3. Revise app to include doughnut charts to show completion, late or incompleted assessments by assessment type. |
| --- | --- |

**Assignments:**

| **Sr.**  **No.** | **Assignment Name** |
| --- | --- |
| A1 | a. Which of the following is a logical operator?  / | - ^  b. What value does R return in the statement below?  3 >= 4  c. What is the result of this calculation?  (45 + 3) \* 43 + 3^2  d. How would R evaluate the following?  carspeed = 70  speedlimit = 65  carspeed>speedlimit  e. How would R evaluate the following?  (2+2 == 4) | (2+2 == 5)  f. How would R evaluate the following? |



|  | !FALSE  g. What is the result of this function?  round(33.2321435452, 2)  h. What is the result of this function?  sqrt (64)  i. What is the result of this statement?  sqrt(64) == 64 ^.5  j. What is the result of this statement?  abs(-32)  k. Which of the following is an arithmetic operator?  \*, |, &, !  l. What is wrong with this code?  2 + 3 \*4 + sqrt[100] |
| --- | --- |
| A2 | Create a vector called unemploy\_rate with 12 values, one for each month in 2013. The values for each month are listed below (beginning with January’s rate of 7.9)  7.9 7.7 7.5 7.5 7.5 7.5 7.3 7.2 7.2 7.2 Create a vector called month and add 12 values, one for the name of each month in a year. Jan Feb Mar Apr May Jun July Aug Sep Oct  Convert month to a factor variable  Create a data frame called monthly\_rate that is comprised of unemploy\_rate and month. How would you extract the unemployment rate for March?  Extract only those months where unemployment was below 7.5%.  What is a factor variable? When would you want to use a factor variable?  What is unique about a numeric variable?  Why would you use a data frame over a vector to store your data? |
| A3 | Create a new R Script in RStudio named fed\_stimulus.R  Add comments to your script that include your name and date  Go to NYC Open Data and export the Federal Stimulus dataset as a CSV file  from https://data.cityofnewyork.us/Business/Federal-Stimulus-Data/ivix-m77e  Review the details of the variables included in the dataset by selecting the manage button on the NYC Open Data site for the Federal Stimulus data. |



|  | Move the Federal\_Stimulus\_Data.csv file to your mydata folder on your desktop  Import the dataset in RStudio. Change the name of the data frame  from Federal\_Stimulus\_Data to fed\_stimulus  Compute the sum and mean for the payment value column  Create a subset of your data that returns those projects with project status is equal to the completed 50% or more. Do not include fully completed projects.  Review your R Script and add appropriate explanatory comments  Try creating a knitr report (you do not need to hand this in). |
| --- | --- |
| A4 | For this assignment use a pre-loaded dataset in R named attitude.  This is from a survey of the clerical employees of a large financial organization, the data are aggregated from the questionnaires of the approximately 35 employees for each of 30 (randomly selected) departments. The numbers give the percent proportion of favorable responses to seven questions in each department. attitude is already pre-loaded in R. To view it, type  >View(attitude).  Create an R script that computes the measures of central tendency and measures of variability and the relationships for each of the seven variables in the attitude dataset. Use the functions below:  mean, median, mode, max, min, range, quantile, IQR, var( ), sd( ), and cor( ) Check your work by using the summary and/or describe functions. |
| A5 | Return to the attitude dataset. Produce at least one scatter plot, histogram, and box-and-whisker plot for each variable. Complete this as a R Markdown document.  Optional: To save time, explore creating a matrix of histograms, a matrix of scatter plots, and a matrix of boxplots. |
| A6 | Use the undersgraduate survey data from http://becomingvisual.com/rfundamentals/undergrad.csv to create ordered factor variables for the excel, statistics and programming variables. In a R Markdown draw histograms for your new ordered factor variables. |
| A7 | Import the following AirBnb data set (http://becomingvisual.com/rfundamentals/airbnb.csv) 1. Create a set of functions that compute specific metrics by neighborhood:  a. average\_number\_of\_reviews  b. average\_price |



|  | 2. Use the functions created in part 1 to compute these metrics by room type and neighborhood. |
| --- | --- |
| A8 | 1. Identify your own data set and create a shiny app that allows the user to explore it (similar to the movie explorer).  2. Host the app on the shinyapps.io website.  3. Submit the URL to your published app. |

**Course Name : Discrete Mathematics with Python**

**Type your text**

****

**Course Type:**

**Course Code:** S2110C1

**SYLLABUS**

| **Teaching**  **Scheme** | | | **Credits** | **Examination Marks** | **Total**  **Marks** |
| --- | --- | --- | --- | --- | --- |
| **L** | **T** | **P** | **C** | **Theory Marks Practical Marks** |  |
| **External Internal External Internal** |  |
| **4** | **0** | **2** | **5** | **40 60 20 30** | **150** |

**What is Discrete Mathematics?**

Discrete Mathematics is a branch of mathematics involving discrete elements that uses algebra and arithmetic. It is increasingly being applied in the practical fields of mathematics and computer science. It is a very good tool for improving reasoning and problem-solving capabilities. This course explains the fundamental concepts of Sets, Relations and Functions, Mathematical Logic, Counting Theory, Probability, Mathematical Induction and Recurrence Relations, Graph Theory, Trees and Boolean Algebra.

**What is Python?**

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance.

Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed.

Python has several built in libraries which can be used to understand the concept of Sets, Algebraic structures and Matrix representation of Graphs.

**Course Overview**

The purpose of this course is to understand and use (abstract) discrete structures that are backbones of computer science. In particular, this class is meant to introduce logic, proofs, sets, relations, functions, counting, and probability, with an emphasis on applications in computer science.

Students can also determine whether a mathematical argument is logically correct, study the relationship between finite sets, count the number of ways to arrange objects in a certain pattern and analyze processes that involve a finite number of steps.



Throughout the teaching of various topics in this subject, teachers discuss how a statement, a function or a complete Python program can help visualize and concretize that topic. It is a fond hope and desire of the designers of this course that many students will directly be helped by this and they will be better computer scientists/engineers having learned Discrete Mathematics with Python. Actually the Python language is separately taught in its full glory under a separate subject title.

Learning how to think mathematically is far more important than knowing how to do all the computations. Consequently, the principal objective of this course is to help you develop the analytic skills you need to learn mathematics. To achieve this goal, we will show you the motivation behind the ideas, explain the results, and dissect why some solution methods work while others do not.

This course will introduce you to the wonderful world of Python programming. You will learn about the essential elements of programming and how to construct basic Python programs for solving the examples in Discrete Mathematics. The topic becomes more understandable if the student is able to write a computer program in python and relate it with the manual solution. This will also develop the logical ability of the student. You will learn expressions, variables, functions, logic, and conditionals, which are used to solve the problems for most of the topic. You will also learn how to use Python modules, which enable you to benefit from the vast array of functionality that is already a part of the Python language. These concepts and skills will help you to begin to think like a computer programmer and to understand how to go about writing Python programs. By the end of the course, you will be able to write short Python programs that are able to accomplish real, practical tasks. This course is the foundation for understanding the python modules which are used in discrete mathematics for understanding the concepts of sets, objects, graphs, matrices which are fundamental for writing a program for several applications in real world life such as optimization problem, finding shortest path image processing etc.

**Prerequisite:** The students are required to have a reasonable mastery over Algebra, Logic.

**Learning outcome**-By the end of the course students shall be confident and equipped with elementary knowledge of Discrete Mathematics and how python is used for solving the problems.

|  | **Course Outcome** | **Bloom’s**  **Level** |
| --- | --- | --- |
| CO  1 | **Understand** the basic principles of sets and operations in sets and apply counting principles to determine probabilities, domain and range of a function, identify one-to one functions, perform the composition of functions and apply the properties of functions to application problems. | Understanding |
| CO  2 | **Write** an argument using logical notation and determine if the argument is or is not valid. To simplify and evaluate basic logic statements including compound statements, implications, inverses, converses, and contrapositives using truth tables and the properties of logic. To express a logic sentence in terms of predicates, quantifiers, and logical connectives. | Understanding |
| CO  3  CO  4  CO  5 | **Apply** relations and to determine their properties. Be familiar with recurrence relations.  **Interpret** different traversal methods for trees and graphs. Model problems in Computer Science using graphs.  **Analyze** how to write functions and pass arguments in Python. | Applying  Understanding Analyze |

CO6 **Create** problems of Discrete mathematics and solve them using Python programs. Creation



**CO-PO Mapping**

|  | **PO**  **1** | **PO**  **2** |  | **PO3 PO4** |  |  | **PO5 PO6 PO7 PO8 PO 9 PO PO**  **PO**  **10**  **1 1**  **1 2** | **PO**  **13** | **PO**  **1 4** | **PO**  **1 5** | **PO1 6** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| CO1 | 1 | 2 |  | 3 | 3 | 3 2 | 2 1 1 2 | 2 | 3 | 1 | 3 |
| CO2 | 1 | 3 | 3 | 3 | 2 | 1 2 | 2 1 | 2 | 3 | 1 | 3 |
| CO3 | 3 | 3 | 3 | 3 | 3 | 3 2 | 3 2 2 2 1 | 2 | 1 | 2 | 2 |
| CO4 | 1 | 3 | 3 | 3 | 3 | 3 3 | 2 1 1 |  | 3 | 2 | 3 |
| CO5 | 2 | 3 | 3 | 3 | 3 | 2 1 | 2 2 1 | 2 | 3 |  | 2 |
| CO6 |  | 2 | 1 | 3 | 3 | 2 2 | 2 2 |  | 2 |  | 1 |

**Tutorials, video demonstration, and exercise**

T1: Set theory, Function & Counting.

T2: Propositional Logic & Predicate Logic

T3: Relations, Partial ordering & Recurrence T4:

Graph theory

T5: Algebraic Structure

T6: Finite State Automata: Deterministic and Non Deterministic Finite State Automata.

https://youtu.be/XOH1wxrBMpE

Students will learn about sets, functions and relations.

https://youtu.be/mrCrjeqJv6U

Students will learn about the basic principles of counting.

https://youtu.be/E40r8DWgG40

Students will get good knowledge of graph theory.



Textbook

J. P. Tremblay and R. Manohar, Discrete Mathematical Structures with Applications to

Computer Science, Tata McGraw-Hill,1997

K. H. Rosen, Discrete Mathematics and its applications, Tata McGraw-Hill, 6th Ed., 2007.

Reference book

S. Lipschutz and M. L. Lipson, Schaum’s Outline of Theory and Problems of Discrete

Mathematics, 2nd Ed., Tata McGraw-Hill,1999.

David Liben-Nowell, Discrete Mathematics for Computer Science, Wiley publication, July

2017.

Eric Gossett, Discrete Mathematics with Proof, 2nd Edition,Wiley publication, July 2009.

Al Doerr and Ken Levasseur, "Applied Discrete Structures" 2020, 3rd Edition - version 7

https://faculty.uml.edu/klevasseur/ADS2

**Required software**

https://www.python.org/downloads/: Python Software.

**Learning resources**

https://swayam.gov.in/nd1\_noc20\_cs82/preview

Discrete Mathematics - By Prof. SudarshanIyengar, Prof, Neeldhara – IIT Ropar, IIT Gandhinagar. https://www.edx.org/course/probability-the-science-of-uncertainty-and-data

Probability – The Science of Uncertainty and data by Massachusetts Institute of

Technology.

https://youtu.be/h\_9WjWENWV8 Course on Discrete Mathematics byMIT.

https://www.py4e.com/book.php

https://Py4e.com

https://colab.research.google.com/

**Pedagogy:**

The real value of Discrete Mathematics is in the abstract, mathematical models which help a Computer Scientist or Engineer to think clearly about complex systems, keeping away mundane details. However, study of such models requires the student to have background in more basic topics like Set theory, combinatorics, propositional logic, and basic matrix algebra. Thus the time spent in studying these basic topics will pay out well in the more abstract topics which are studied later on.

Another factor which helps the student is properly selected examples, both in terms of quality and quantity, that are relevant to the material being introduced.

With each major topic being covered is accompanied by at least one application in the computer field. One of the group of most immediate applications would be in present day complex computer software and that is where dovetailing a very convenient programming language like Python is justified. The teacher demonstrates in the class links to Python program constructs for a particular topic and the students later on develop and test the complete programs, which helps manifestation of abstract concepts in form of tangible results of a running program.





| B.Tech (Discrete  Mathematics with Python) | | Pedagogy | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| S.N  o. | Topic | Lecture | Workshop ABL  Tutorial  / PBL | Practical Indust | rial Visit | Expert  Lecture | MOOC/ NPT EL | Any  other  Tool |
| 1 | Set theory,  Function & Counting | 6 | - - 1 | P-1, P-2, P3 | - | - |  |  |
| 2 | Propositional Logics,  Predicate  Logics | 7 | - - 2 | P-4, P-5, P-6 | - | - |  |  |
| 3 | Relations,  Partial  Ordering,  Recurrence Relation | 9 | - - 3 | P-7, P-8, P-9 | - | - |  |  |
| 4 | Graphs, Typesof Graph,  Subgraph,  Connectednes s | 9 | - - 4 | P-10 | - | - |  |  |
| 5 | Algebraic  Structures | 4 | - - 5 |  | - | - |  |  |
| 6 | Deterministic Finite  Automata,  Nondetermini sti  c Finite  Automata | 4 | 6 | P-11 |  |  |  |  |



Course Outline

| Class | Topic | | Reading s | La b | Tutorial |
| --- | --- | --- | --- | --- | --- |
| 1 | Set Theory | Definitions- Inclusion, Equality of Sets, Cartesian product, The Power Set, Some operations on Sets, Venn Diagrams  Basic Concepts of Set Theory  Some Basic Set Identities  Understanding set operations using Python | Lesson 1 | P-1 | T1 |
| 2 | Function | Introduction & definition  Co-domain, range, image, value of a function, Examples  Surjective, injective, bijective; examples Composition of functions, examples  Inverse function, Identity map  Condition of a function to be invertible, examples Inverse of composite functions  Properties of Composition of functions  Defining the mathematical expressions in python | Lesso n 2-3 P-1 |  | T1 |
| 3 | Counting | The Basics of Counting  The Pigeonhole Principle  Permutations and Combinations  Binomial Coefficients  Generalized Permutations and Combinations Generating Permutations and Combinations.  Solving different problem related to Counting using python | Lesso n 4-6 P-2 |  | T1 |



| 4 | Propositio n al Logics | Definition, Statements & Notation  Truth Values, Connectives  Statement Formulas & Truth Tables  Well-formed Formulas  Tautologies  Equivalence of Formulas  Duality Law,  Tautological Implications, Examples  Formation of truth table using python. | Lesson 6-9 P-3, P-4 T2 |  |  |
| --- | --- | --- | --- | --- | --- |
| 5 | Predicat e Logics | Definition of Predicates  Statement functions, Variables, Quantifiers Predicate Formulas, Free & Bound Variables The Universe of Discourse, Examples  Valid Formulas & Equivalences, Examples. | Lesso n 10- 12 |  | T2 |
| 6 | Relations | Definition, Binary Relation  Representation, Domain, Range, Universal Relation,Void Relation, Union, Intersection, | Lesson 13 P-8 |  | T2 |



|  |  | and Complement Operations on Relations  Properties of Binary Relations in a Set: Reflexive, Symmetric, Transitive, Anti- symmetric Relations  Understanding the properties of Relations using python. |  |  | |  | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 7  8 | Matrices & Equivalenc e Classes  Partial  Ordering | Definition of Matrices  Addition, subtraction, transpose  Multiplication by a scalar, multiplication of two matrices, special matrices like Identity, determinant and inverse of a 2x2 matrix.  Relation Matrix and Graph of a Relation  Partition and Covering of a Set, Equivalence Relation, Equivalence Classes  Compatibility Relation, Maximum Compatibility Block, Composite Relation  Converse of a Relation, Transitive Closure of a Relation R inSet X .  Performing the matrix operation using Python. Definition, Examples  Simple or Linear Ordering  Totally Ordered Set (Chain), Frequently Used Partially Ordered Relations  Representation of Partially Ordered Sets  Hasse Diagrams, Least & Greatest Members, Minimal & Maximal Members, Least Upper Bound (Supremum), Greatest Lower Bound (infimum)  Well-ordered Partially Ordered Sets (Posets)  Lattice as Posets, complete  Distributive modular and complemented lattices Boolean and pseudo Boolean lattices.  Finding the Least, Maxima element from the POSET using python. | Lesson  14-16  Lesson  17-19 | | P-5  P-6 | | T3 |



| 9 | Recurrenc e Relation | Introduction, Recursion  Recurrence Relation  Solving, Recurrence Relation  Solving the recurrence relations using python. | Lesso n 20- 21 | P-7, P-9 | T3 |
| --- | --- | --- | --- | --- | --- |
| 10 | Graph | Introduction, definition, examples  Nodes, edges, adjacent nodes, directed and undirected edge Directed graph, undirected graph, examples  Initiating and terminating nodes, Loop (sling)  Distinct edges, Parallel edges | Lesson 22 |  | T4 |



| 11 |  | Types of GraphMulti-graph, simple graph, weighted graphs, examples Lesso n 23-  Isolated nodes  24  Null graph; Isomorphic graphs, examples  Degree, In-degree, out-degree, total degree of a node,  examples |  | T4 |
| --- | --- | --- | --- | --- |
| 12  13 | Subgraph | Definition, examples  Lesson  25-26  Converse (reversal or directional dual) of a digraph,  examples  Path: Definition, Paths of a given graph, length of path, examples  Simple path (edge simple), elementary path (node simple), examples  Cycle (circuit), elementary cycle, examples  Connected nessDefinition, weakly connected, strongly connected,  Lesson  unilaterally connected, examples  27-30  Strong, weak, and unilateral components of a graph,  examples  Matrix representation of graph: Definition, Adjacency  matrix, Boolean (or bit) matrix, examples  Determine number of paths of length n through Adjacency matrix, examples; Path (Reachability) matrix of a graph, examples  Warshall’s algorithm to produce Path matrix, Flowchart.  Obtaining Boolean matrices, Adjacency matrix using  python. | P-10 | T4  T4 |
| 14 | Algebrai c  Structure s | Algebraic structures with one binary operation –  Lesso n 31-  semigroups  34  Monoids and Groups  congruence relation and quotient structures  Free and cyclic monoids and groups  permutation groups  substructures  Normal subgroups |  | T5 |



|  |  | Understanding Algebraic structures using Python |  |  |
| --- | --- | --- | --- | --- |
| 15 | Determinist ic Finite  Automata | Alphabets, Strings, Languages  Lesso n 35-  Finite Automata (FA), acceptance of strings, and languages 36  Deterministic Finite Automata (DFA) |  | T6 |
| 16 | Nondeterm  inistic Finite Automata | An informal view of Nondeterministic Finite Automata Lesso n 37-  Definition of Nondeterministic Finite Automata, The  38  extended transition function  The language of an NFA, Equivalence of Deterministic and Nondeterministic Finite Automata. | P-11 | T6 |

**List of Program using Python for Discrete Mathematics:**

P-1.Write a python program to define a Set, List the elements and perform the basic operations such as : Union, Intersection, Complement, Cartesian product of two sets.

P-2. Write a program to find the factorial of a number.

P-3. Write a program to find scalar multiply, matrix multiply of matrix.

P-4. Write a Python function that takes a list of pairs as an argument and determines if the argument constitutes a function or a general relation.

P-5. Write a program to denote Relation: as subsets of a power set, matrix representation. P-6. Write a program for Propositional Logic, Logic operators, Truth Tables.

P-7. Write a program for Equivalence and implication.

P-8. Write a program for Functions: functional Python programming.

P-9. Write a program for Recursion: functions, generators.

P-10. Write a program for Graphs: matrix representation, association.

P-11 Write a program for Regular Expressions, Finite State Machine.

**Web Technologies: HTML, CSS, JavaScript, PHP**

**Course Type:Core**

**Course code:** C2110C3



| Teaching Scheme | | | Credits | Examination Marks | | Total Marks |
| --- | --- | --- | --- | --- | --- | --- |
| L | T | P | C | Theory Marks Practical Marks | |  |
| External | Internal External Internal |  |
| 2 | 0 | 4 | 4 | 40 | 60 20 30 | 150 |

**What is Web Technology?**

Learning Web Technology is essential today because Internet has become the number one source to information, and many of the traditional software applications have become Web Applications. Web Applications have become more powerful and can fully replace desktop application in most situations.

That’s why you need to know basic Web Programming, including HTML, CSS and JavaScript. It all started with Internet (1960s) and the World Wide Web - WWW (1991). The first Web Browser, Netscape, came in 1994. This was the beginning of a new era, where everything is connected on internet, the so-called Internet of Things (IoT).

**Course overview**

In this course students will start on a path toward future studies in web development and design, no matter how little experience or technical knowledge you currently have. The web is a very big place, and if you are the typical internet user, you probably visit several websites every day, whether for business, entertainment or education. But have you ever wondered how these websites actually work? How are they built? How do browsers, computers, and mobile devices interact with the web? What skills are necessary to build a website? With almost 1 billion websites now on the internet, the answers to these questions could be your first step toward a better understanding of the internet and developing a new set of internet skills.





By the end of this course you’ll be able to describe the structure and functionality of the world wide web, create dynamic web pages using a combination of HTML, CSS, and JavaScript, apply essential programming language concepts when creating HTML forms, select an appropriate web hosting service, and publish your WebPages for the world to see. Finally, you’ll be able to develop a working model for creating your own personal or business websites in the future and be fully prepared to take the next step in a more advanced web development.

**Learning outcomes**

By the end of the course students you shall be confident and equipped with all the knowledge required to design an effective web site

|  | **Course Outcome Bloom’s Level** |
| --- | --- |
| CO1 | **Remember** the concepts of WWW including  Understand  browser and HTTP protocol. |
| CO2 | **Understand** various HTML Tags to develop user  Understand  friendly web pages |
| CO3 | Understand, analyze and build dynamic and interactive  Understanding, Analyze  web sites |

CO4 **Apply** CSS with its types to provide the styles to

Application

the web pages at various levels



| CO5 | **Apply** JavaScript to develop the dynamic web pages Analyze |
| --- | --- |
| CO6 | Use server-side scripting with PHP to generate the web  Application  pages dynamically. |
| CO7 | Be able to write the schema for the given XML  Creation  documents in both DTD and XML Schema languages |
| CO8 | **Create** their own personal and professional websites.Creation |

**TextBooks:**

| Sr.  No | Book Name |
| --- | --- |
| **1** | Ralph Moseley and M. T. Savaliya, “Developing Web Applications”, Wiley-India |
| **2** | Bai and Ekedhi, The Web Warrior Guide to Web Programming, 3rd Edition, Thomson, 2008 |
| **3** | Web Technologies, Black Book, dreamtech Press |
| **4**  **5** | Knuckles, “Web Applications: Concepts and Real World Design”, Wiley-India Developing Web Applications in PHP and AJAX, Harwani, McGrawHill |
| **6** | Internet and World Wide Web How to program, P.J. Deitel & H.M. Deitel, Pearson |
| **7** | The complete reference PHP- Steven Holzner, Tata mcGraw-Hill |



**Required software**

• Notepad++

• VS Code

• Codepen (Online Tool



**Learning Resources**

**TEDx Videos:**

| **Sr. No** | **TEDx Video** |
| --- | --- |
| **T1** | https://www.ted.com/talks/kevin\_kelly\_how\_technology\_evolves?language=en#t-17539How technology works.- Kevin Kelly  Tech enthusiast Kevin Kelly asks "What does technology want?" and discovers that its movement toward ubiquity and complexity is much like the evolution of life. |
| **T2** | https://www.youtube.com/watch?v=Va07q3HFEZQ – The Dark Web – Alan Pearce Alan talks about the dark web and how it can be used for worthwhile purposes. |

**Other Videos**

| **Sr.**  **No** | **About Video Link** | **Topic** |
| --- | --- | --- |
| O1 | Stanford Lecture - Ben Galbraith: Web Skills:  https://www.youtube.co  m/watch?v=r4F5tY\_6u7  Introduction to Web Technologies and HTML 5  0&list=PLDD1386C4D  784B56B&index=3&t=  0s | Introductio n to web |
| O2  O3 | James Mickenss. Securing Web Applications MIT  https://www.youtube.co  OpenCourseWare  m/watch?v=WlmKwIe9  z1Q  Prof.I.Sengupta, Department of Computer Science &  https://www.youtube.co  Engineering ,IIT Kharagpur On HTML  m/watch?v=QEtWL4lW  lL4 | Securing  Web  Application s  HTML |



| O4 | Introductory lecture on Basic concepts of web  https://www.youtube.co  applications, how they work and the HTTP protocol  m/watch?v=RsQ1tFLwl  dY | HTTP and WWW |
| --- | --- | --- |
| O5 | Douglas Crockford: The JavaScript Programming  https://www.youtube.co  Language  m/watch?v=v2ifWcnQs  6M&list=PL62E185BB  Douglas Crockford explores not only the language as  8577B63D  it is today but also how the language came to be the  way it is. | JavaScript |
| O6 | Building Dynamic websites. Harvard  https://www.youtube.co  OpencourseWare. CS E-75  m/watch?v=h2Nq0qv0K  8M | How tom create  dynamic  website |
| O7 | David Malan : PHP Harvard Web Development https://www.youtube.co m/watch?v=gwUEjgbs0  2w&list=PLTney8JqQk  kk2oGFQONHHKDzvs  ChJMKEY&index=2 | PHP |

**Related MOOCs courses**

| **Sr. No.** | **MOOC Courses** |
| --- | --- |
| **M1** | Introduction to HTML5 by University of Michigan 4 Weeks on Coursera. |
| **M2** | Advanced Styling with Responsive Design By University of Michigan on (coursera) |
| **M3** | https://www.edx.org/professional-certificate/w3cx-front-end-web-developer |
| **M4** | https://www.edx.org/course/javascript-introduction |



| **M5** | https://www.udemy.com/courses/development/web-development/ |
| --- | --- |
| **M6** | https://www.edx.org/course/css-basics |
| **M7** | https://www.edx.org/course/programming-for-the-web-with-javascript |



**Course Outline:**

| **Unit**  **No** | **Topic** | **Lab** | **Hours** |
| --- | --- | --- | --- |
| **1** | **Introduction to WWW**  ● Concept of WWW  ● Internet and WWW  ● HTTP protocol: Request and Response ● Web Browser and Web Servers  ● Features of Web 2.0  ● Setting up Unix and Linux Web Servers ● Dynamic IP  ● Concepts of effective Web Design ● Web Design Issues | **-** | **04** |
|  |  |  |  |
| **2.** | HTML  ● Basics of HTML  ● Formatting and Fonts  ● Color and Hyperlinks  ● List, Table, Images, Forms  ● HTML5 New Elements  ● HTML5 : Drag/Drop, Video, Audio, Input types  ● XHTML: Meta Tags | P1, P2, P3, P4,  P5 | **07** |
|  |  |  |  |
| **3.** | CSS  ● Need for CSS  ● Introduction to CSS  ● Syntax and Structure  ● Background images, Colors and  properties  ● Manipulation of Texts: Font, Border and Boxes  ● CSS positioning  ● CSS2, Overview and Features of CSS3 | P1, P2,P3,P4,P5 | **04** |
|  |  |  |  |



| **4.** | Introduction to JavaScript and JQuery  ● Client side Scripting with JavaScript: Variables, conditions, Loops,  Functions, Pop up Boxes  ● Advance JavaScript : JavaScript and objects,  ● DOM  ● Forms and Validations  ● DHTML: Combining HTML , CSS and JavaScript  ● Events and Buttons  ● AJAX: Introduction.  ● AJAX Based WebApplications  ● JQuery: Introduction | P6,P7,P8,P9,P10, P11,P16 | **15** |
| --- | --- | --- | --- |
|  |  |  |  |
| **5** | XML  ● Introduction to XML  ● XML Key Components  ● DTD and schemas  ● Transforming XML using XSL and XSLT | P12, p13 | **08** |
|  |  |  |  |
| **6** | PHP  ● Introduction to Server Side Scripting ● Introduction and Basic Syntax of PHP ● String Processing and Regular  expression  ● Web Development Frameworks | P14, P15 | **10** |

**Lab Experiments:**

**P1 Creating a Food Drive Web Page**

*Problem:* You did volunteer work for the Community Food Drive in your city. You would like to recruit other friends to volunteer for community service. You have been asked to create a Web page to display information about why you choose to volunteer and let people know how they also can help



| **P2.** | **Creating a Web Page with Links:**Create a Web page demonstrating your knowledge of link targets |
| --- | --- |
| **P3** | *Problem:* Your Communications instructor has asked each student in the class to create a two-page Web site to help students in the class get to know more about the area in which you are majoring in school. The first Web page is a home page that includes basic information about your major. If you can, add an image related to your chosen field somewhere on the Web page. Add a link to the second Web page. The second Web page includes a paragraph of text and numbered lists with links. |
| **P4** | **Creating a School Bookstore Survey**  *Problem:* The staff of the school bookstore wants to survey students about their book-buying habits to determine where they purchase their books. The staff has asked you to create a Web page form that contains the questions (create a form containing the basis information like name, address, city, zip, Email id ask the questions containing the survey. |
| **P5** | **Frames**  **Creating a Two-Frame Structure for a Soccer Web Site**  Problem: The Director of the Schererville Soccer league has asked you to create a new Web site with information on the upcoming soccer season. After reviewing the content with the Director, you suggest using a two-frame structure with two horizontal frames. The top frame will display a header and menu bar for navigation, while the bottom frame will display schedules, standings, rules, and other information. |
| **P6** | Write a JavaScript program for performing Arithmetic Operations |
| **P7** | Text growing and shrinking in Java script |
| **P8** | Design HTML form which includes two fields username and password. Write JavaScript code to show and hide password. |
| **P9** | Design a login form using HTML & JavaScript with following validations on username and password fields: Also, password length must be of 6 to 12 characters.  Username should not start with \_, @ or number and both the fields should not be blank. |
| **P10** | Write JS to demonstrate various built-in string functions  Develop and demonstrate a HTML5 file that includes JavaScript script that uses functions for the following problems: a. Parameter: A string Output: The position in |



|  | the string of the left-most vowel b. Parameter: A number Output: The number with its digits in the reverse order |
| --- | --- |
| **P11** | Write JS to search an element in an array of size N. |
| **P12** | Create an XML document that contains 10 users information. Write a java program, which takes user id as input and returns the user details by taking the user information from XML document using a)DOM parser b)SAX parser |
| **P13** | Introduction **XSLT**: Prepare a XSL file to transform Students data from XML to tabular form in XHTML. |
| **P14** | Write a PHP program named states.py that declares a variable states with value "Mississippi Alabama Texas Massachusetts Kansas". write a PHP program that does the following:  a. Search for a word in variable states that ends in xas. Store this word in element 0 of a list named statesList.  b. Search for a word in states that begins with k and ends in s. Perform a caseinsensitive comparison. [Note: Passing re.Ias a second parameter to method compile performs a case-insensitive comparison.] Store this word in element1 of statesList.  c. Search for a word in states that begins with M and ends in s. Store this word in element 2 of the list.  d. Search for a word in states that ends in a. Store this word in element 3 of the list. |
| **P15** | Create a XHTML form with Name, Address Line 1, Address Line 2, and E-mail text fields. |
| **P16** | Using AJAX retrieve data from a TXT file and display it. Create XSL file to convert XML file into XHTML file |

**ITM (SLS) Baroda University**

**School of Computer Science, Engineering and Technology**

**B.Tech - Semester II**

**Course Name: Programming in C**

**Course Code: C2210C1**

**Course Type: Core**

**Teaching Scheme:**

| **Teaching**  **Scheme** | | **Credits** | **Examination Marks** | | **Total**  **Marks** |
| --- | --- | --- | --- | --- | --- |
| **L** | **T P** | **C** | **Theory Marks** | **Practical Marks** |  |
| **External Internal** | **External Internal** |  |
| **3** | **0 4** | **5** | **40 60** | **20 30** | **150** |

**Preamble:**

For a given problem, developing logic, translating that logic into a programming language statements and implementation in a programming language is the skill required by programmers. Basic constructs of most of the programming languages are common. Under this backdrop and having gone through one programming language (Python-I during Semester I), the orientation in this course will be to learn the skills to develop logic for a given problem, select proper C language constructs and data elements and structures required to express that logic and its implementation in C Programming Language.

**What is C?**

**C** is a general-purpose programming language that is extremely popular, simple, and flexible to use. It is a structured programming language that is machine-independent and extensively used to write various applications, Operating Systems like Windows, and many other complex programs like Oracle database, Git, Python interpreter, and more. C was originally developed at Bell Labs by Dennis Ritchie between 1972 and 1973.

Many later languages have borrowed syntax/features directly or indirectly from C language. Like syntax of Java, PHP, JavaScript, and many other languages are mainly based on C language.

**Course overview:**

In this course, students will learn how to do programming in C. Students will learn: how C works and its place in the world of programming languages; to work with and manipulate strings; to perform math operations; flow control processing in C; to write functions; to work

with pointers and arrays in C.

**Understand the**

**Problem Prerequisite: Draw**

**Flowchart**

**Write**

**Psuedcode**

**ProgramOutcome Development**

This course assumes a basic understanding of the principles of programming, python programming-1 course in semester I.

**Learning outcomes:**

After completing the course, the student shall be able to:

|  | **Course Outcome Bloom’s Level** |
| --- | --- |
| CO1 | Formulate algorithm/flowchart for given arithmetic and  Understanding  logical problem. |
| CO2 | Translate algorithm/flowchart into C program using correct  Understanding  syntax and execute it. |
| CO3 | Write programs using conditional, branching, iteration,  Applying  and recursion. |
| CO4 | Decompose a problem into function Application |
| CO5 | Develop an application using the concepts of array, and pointer  Application  to solve engineering and/or scientific problems. |

**Programme Outcomes:**

|  | **POs** | **Bloom’s Level** |
| --- | --- | --- |
| PO1 | Design, develop and maintain computing systems using concepts from mathematics, science and engineering and program core courses | Creation |
| PO2 | To implement software design systems, components, or processes to meet the desired needs and specifications. | Creation |
| PO3 | Design algorithms for real world computational problems and analyze their complexities | Creation |

To adapt to the usage of modern

recent software

PO4

tools and technologies and

Application

| PO5 | Analyze and interpret data and discover knowledge to provide solutions to engineering problems | Analyze |
| --- | --- | --- |
| PO6 | Assess the security, privacy, quality and cost parameters in developing software systems | Evaluate |
| PO7 | Ability to work effectively in multi-disciplinary teams using common tools and standards to achieve project objectives | Application |
| PO8 | Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice. | Creation |
| PO9 | Proficiency in oral, written and visual means for technical presentations and documentations | Application |
| PO10 | Engage in lifelong learning through self-directed and independent study of new techniques and tools | Creation |

**CO-PO Mapping:**

|  | **PO1** | **PO2** | **PO3** | **PO4** | **PO5** | **PO6 PO7** | **PO8** | **PO9** | **PO1**  **0** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **CO1** | **2** | **3** | **3** | **2** |  | **3 2** |  |  | **2** |
| **CO2** | **2** | **3** | **3** | **3** |  | **3 2** |  |  | **2** |
| **CO3** | **3** | **3** |  | **2** | **1** | **3 3** |  |  | **2** |
| **CO4** |  | **2** |  | **3** | **2** | **2 2** |  |  | **2** |
| **CO5** | **2** | **2** | **2** | **2** | **1** | **1 2** |  |  | **2** |

**Course Outline:**

| **Unit**  **#** | **Topics Lab** | **Assign ment** | **Teachi**  **ng**  **Hours** |
| --- | --- | --- | --- |

| 1 | **Fundamentals of C:** Features of C language, structure of  P1,P2,  C Program, comments, header files, data types, constants  P3  and variables, operators (Arithmetic, relational, logical, increment, decrement, bitwise, assignment) expressions, conditional expressions, evaluation of expressions, type conversion, Flowchart and algorithm design. Overview of all concepts using C. | A1 | 8 |
| --- | --- | --- | --- |
| 2 | **Control structure in C:** Simple statements, Decision  P4,P  making statements (if-else, switch), looping statements  5,  (for, while, do-while), Nesting of control structures, break  P6  and continue, goto statement. | A2 | 8 |
| 3 | **Functions and Recursion:** Concepts of user defined  P7,P8  functions, prototypes, definition of function, parameters, parameter passing, calling a function, scope and lifetime of variable, Recursion, as a different way of solving problems. Example programs, such as Finding Factorial, Fibonacci series | A3 | 8 |
| 4 | **Pointers and Arrays:** Basics of pointers, pointer to  P9,P1  pointer, pointer and array, pointer to array, array to  0,P1  pointer, function returning pointer,  1  Concepts of array, one- and two-dimensional arrays, declaration and initialization of arrays, string, string storage, Initialization of pointer, pointers vs multidimensional arrays, command line arguments, pointers to functions | A4,A  5  ,A6 | 12 |
| 5 | **Structures:** Basics of structures, structures and functions,  P12,P  13  arrays of structures, pointers to structures, self-referential structures, table lookup, typedef, unions, bit-fields |  | 8 |
| 6 | **Input and output:** standard input and output, formatted  P14,P  15  output, variable length argument list, formatted input, file access, Error handling-stderr and exit, line input and output, miscellaneous functions. | A7 | 8 |
| 7 | **The UNIX System interface:** File Descriptor, Low level  P16  I/O- Read and Write, open, create, close, unlink, random access using Lseek, Fopen and Getc, Listing Directories, A storage Allocator |  | 4 |
| 8 | **Standard libraries:** string.h, math.h, stdlib.h, assert.h,  P17  time.h |  | 2 |
|  | **TOTAL** |  | **58** |

**Textbooks:**

| **Sr. No.** | **Book Name** |
| --- | --- |
| 1. | Brian W. Kernigham, Dennis Ritchie, “The C Programming Language”, Pearson |
| 2. | R. G. Dromey, “How to Solve it by Computer”, Pearson |

**Reference Books:**

| 1. | YeshvantKanetkar, “Let Us C”, BPB Publication |
| --- | --- |
| 2. | Reema Thareja, “Computer Fundamentals and Programming in C”, Oxford Publication |
| 3. | Herbert Schildt, “C: The Complete Reference”, McGrawHill |

**Required Software:**

1. GNU C Compiler: https://gcc.gnu.org/

2. For working with embedded system corresponding architecture tool chain : Assembler, Link editor, etc.

**Learning Resources:**

1. https://ocw.mit.edu/courses/

2. https://www.edx.org/learn/c-programming

**TedEx Videos:**

| **Sr. No** | **TEDx Video** |
| --- | --- |
| **T1** | **The Right Way to Learn Code | Brian Underwood, Max Ptacek& Ben Makarechian | TEDxYouth@SHC**  **https://www.youtube.com/watch?v=PdTEXWtd3HY**  Ben Makarechian, Brian Underwood, and Max Ptacek tell the story of their experiences trying to create a virtual reality dodgeball game with little knowledge of computer code. They share the methods they used to teach themselves coding so students around the world can learn on their own. |
| **T2**  **T3** | **How algorithms shape our world. -Kevin Slavin**  **https://www.youtube.com/watch?v=ENWVRcMGDoU**  Kevin Slavin argues that we're living in a world designed for --and increasingly controlled by --algorithms. In this riveting talk from TEDGlobal, he shows how these complex computer programs determine espionage tactics, stock prices, movie scripts, and architecture. Slavin also warns that we are writing code we can't understand with implications we can't control.  **Don't Just Learn to Code, Learn to Create | Justin Richards | TEDxYouth@ColumbiaSC** |

|  | **https://youtu.be/6rxWc-TNIJI**  Students around the world are told they must ‘Learn to Code!’ This certainly sounds great, but why should you learn to code? Will it really help you accomplish your personal goals or land your dream job? We’ll see how learning to code is only a part of joining the digital revolution, but we’ll go further to explore how you can harness technology to your unique way of impacting the world. Learn how you can shape the future by not just learning to code, but learning to create. |
| --- | --- |

**Other Videos:**

| **Sr.**  **No** | **About Video** | **Link** | **Topic** |
| --- | --- | --- | --- |
| **O**  **1** | Erik Demaine, Ronald  Rivest, and Srini  Devadas. 6.006  Introduction to  Algorithms. Spring 2008. Massachusetts Institute of Technology: MIT  OpenCourseWare,  https://ocw.mit.edu.  License: Creative  Commons BY-NC SA | https://ocw.mit.edu/courses/electrical engineering-and-computer-science/6-0 06-  introduction-to-algorithms-spring-2008 / | Introducti  on to  Algorithm |
| **O**  **2** | Lecture on what is  computation; introduction to data types, operators, and variables by Prof.  Eric  Grimson, Prof. John Guttag | https://www.youtube.com/watch?v=k6 U  i4gXkLM&list=PLo7g9OE1yqEKY4 Kx AiPpqA9gp7XOcPQ2V | What is  computatio  n;  introduction to data  types,  operators,  and  variables |
| **O**  **3** | Lecture by Professor  Jerry Cain for  Programming  Paradigms (CS107) in  the Stanford University Computer Science  department. | https://www.youtube.com/watch?v=P s8j  Oj7diA0&list=PLD28639E2FFC4B8 6A | Programmi  ng  Paradigm |

| **O**  **4** | Lecture Series on  Programming and Data  Structure by Dr.P.P.  Chakraborty, Department of Computer Science and Engineering, IIT Kharagpur | https://www.youtube.com/watch?v=5D 5i  erCsAkM&list=PLeCxvb23g7hrw27X lek HtfygUTQ0TmFfP&index=9 | Decomposi  tio n using  Recursion |
| --- | --- | --- | --- |
| **O**  **5** | Lecture Series on  Programming and Data  Structure by  Dr.P.P.Chakraborty,  Department of Computer | https://www.youtube.com/watch?v=3X o6 P\_V  qns&list=PLeCxvb23g7hrw27XlekHtf yg UTQ0TmFfP&index=13 | Array and  Addresses |

|  | Science and Engineering, IIT Kharagpur |  |  |
| --- | --- | --- | --- |

**Related MOOCs courses:**

| **Sr.No MOOC Courses** |
| --- |
| **M1** “Introduction to Programming in C” by Dr. Satyadev Nandakumar, IIT Kanpur 8 weeks on NPTEL. |
| **M2** “Art of C Programming” by Dr. Lajisj V.L., Department of Computer Science, University of Calicut. |

**Lab Experiments:**

| **Sr. No.** | **Program Statement** |
| --- | --- |
| **P1** | 1. Write a C program to print “Hello World…!!!” and find out what happens when printf’s argument string contains \c.  2. Write a C program to do following:  a. Print corresponding Celsius to Fahrenheit table.  b. Modify the program to print heading above the table.  c. Print Celsius to Fahrenheit table in reverse order.  d. Implement the program using user defined function for temperature conversion 3. Write a C program to print the value of EOF. |
| **P2** | 1. Write a C program to copy input file to output file, replacing multiple blanks with a single one.  2. Write a C program to copy input file to output file, replacing each tab with ‘\t’ 3. Write a C program to print one word per line from the input file. 4. Write a C Program to print a histogram of the following from the input file a. lengths of words  b. the frequencies of different characters in its input. |

| **P3** | 1. Write a C program to print all input lines from the file that are longer than 80 characters.  2. Write a C program to remove all comments from a C program. Don't forget to handle quoted strings and character constants properly.  3. Write a C program to remove all trailing blanks and tabs from each line of input file, and to delete entirely blank lines. |
| --- | --- |
| **P4**  **P5** | 1. Write a C program to read marks of a student from the keyboard whether the student is pass or fail (using if else).  2. Write a C program to read three numbers from the keyboard and find the maximum out of these three. (nested if else).  3. Write a C program to check whether the entered character is capital, small letter, digit or any special character.  1. Write a C program to read no 1 to 7 and print relatively day Sunday to Saturday (using switch case).  2. Write a C program to find out the Maximum and Minimum number from given 10 numbers. |

| **P6**  **P7**  **P8**  **P9** | 1. Write a C program to reverse a number.  2. Write a C program to input an integer number and check if the last digit of number is even or odd.  3. Write a C program to print the following pattern:  i. \*  \* \*  \* \* \*  \* \* \* \*  ii. 1  2 3  4 5 6  1. Write a C program to create a function named sum which calculates the sum of three given numbers.  2. Write a C program to convert decimal number to binary number using function. 3. Write a C program to find the prime numbers between given intervals using function.  1. Write a C program to find the factorial of a given number. (recursion) 2. Write a C program to generate the first n number of Fibonacci series (recursion). 3. Write a C program to reverse a sentence using recursion.  1. Write a C program to reverse an array using a pointer.  2. Write a C program to reverse string using pointer.  3. Write a C program to search an element in an array using a pointer. |
| --- | --- |

| **P10** | 1. Write a C program to store n elements in an array and print the elements using pointer.  2. Write a C program to print all the permutations of the given string using pointer. 3. Write a C program having a pointer version of the function strcat(s,t) which copies the string t to the end of s. |
| --- | --- |
| **P11** | 1. Write a C program for the versions of the library functions strncpy, strncat, and strncmp, which operate on at most the first n characters of their argument strings. For example, strncpy(s,t,n) copies at most n characters of t to s.  2. Write a C program expr , which evaluates a reverse Polish expression from the command line, where each operator or operand is a separate argument. For example, expr 2 3 4 + \* evaluates 2 X (3+4). |
| **P12**  **P13** | 1. Write a C program to store information of N students using structure. 2. Write a C program to add two distances in inch-feet using structure. The values of the distance are to be taken from the user.  3. Write a C program to enter the marks of 5 students in Python, C and Discrete Mathematics (each out of 100) using a structure named Marks having elements roll no., name, py\_marks, c\_marks and dm\_marks and then display the percentage of each student.  1. Write a C program that prints the distinct words in its input sorted into decreasing |

|  | order of frequency of occurrence. Precede each word by its count.  2. Write a C program to create a function undef that will remove a name and definition from the table maintained by lookup and install. |
| --- | --- |
| **P14** | 1. Write a C program to write multiple lines in a text file.  2. Write a C program to read the file and store the lines into an array. 3. A file named data contains a series of integer numbers. Write a c program to read all numbers from file and then write all odd numbers into file named “odd” and write all even numbers into file named “even”. Display all the contents of these files on screen. |
| **P15**  **P16** | 1. Write a C program that converts upper case to lower or lower case to upper, depending on the name it is invoked with, as found in argv[0].  2. Write a C program to print a set of files, starting each new one on a new page, with a title and a running page count for each file.  3. Write a C program to compare two files, printing the first line where they differ.  1. Write a C program \_flushbuf , fflush , and fclose*.*  2. The standard library function  int fseek(FILE \*fp, long offset, int origin)  is identical to lseek except that fp is a file pointer instead of a file descriptor and the return value is an int status, not a position. Write a C program for fseek . Make sure that your fseek coordinates properly with the buffering done for the other functions of the library.  3. The standard library function calloc(n,size) returns a pointer to n objects of size size , with the storage initialized to zero. Write calloc , by calling malloc or by modifying it. |

| **P17** | 1. Write a C program to perform various functions of string.h, math.h, stdlib.h, assert.h, time.h |
| --- | --- |

**Mini Project:**

Apart from the above laboratory work, each student will implement a teacher specified individual small project requiring implementation through more than one source file and one header file, algorithm selection, determining the test data sets and a documentation for a 10- 15 minutes presentation of his/her work. Some of the examples are listed below :

| **MP1** | **Snake Game**  Implement a snake game in which users can use up, down, right or left arrows to move the snake. There will be 3 lives and life will be decreased if you hit the snake's body or the wall. You have to provide food at several coordinates by which score and length of the snake will be increased. |
| --- | --- |
| **MP2** | **Bus Reservation System**  Implement a system that can reserve seats on the bus. It can display the availability of seats and also can find the booking information. |

**Assignments:**

| **Sr. No.** | **Assignment Name** |
| --- | --- |

**A1** Write a C program which takes name, basic , daper ( ie, percentage of D.A), bonper (ie, percentage bonus) and loandet ( loan amount to be debited) for an employee. Calculate the salary using the following relation:

salary = basic + basic \* daper /100 +bonper \* basic/100 - loandet

Data is as following:

**Name Basic daper bonper loadnet** Amit 2500 55 33.3 250.0

Calculate salary and then print the result under the following headings:

**Name Basic Salary**

**A2** Write a C program to generate a calendar for a given month and for a given year. (month and year is to be taken as input from user).

| **A3** | Write 2 different C functions to compute the area and perimeter of a triangle whose sides a, b, and c are given by user as inputs.  Formula to compute perimeter = a + b + c  Formula to compute area = [s(s-a)(s-b)(s-c)]0.5  Where s = 0.5 \* (a+b+c)  Function prototypes are:  double perim(double a, double b, double c)  double area(double a, double b, double c)  Your program should read the input data and print the output data via separate functions. The prototypes are: double read\_input() double print\_value(double val) |
| --- | --- |
| **A4**  **A5** | Write a C program to perform the following operations on an integer array of 10 elements. Accept the values from the user.  1. Sort an array in ascending order.  2. Display sum of all odd values stored in an array.  3. Display number of even values stored in an array.  Write a C program to implement an inventory system. Store the item number, name, rate and quantity on hand in a structure. Accept the details for five items into a structure array and display the item name and its total price. At the end, display the grand total value of the inventory. |
| **A6**  **A7** | Write a C program that takes 2 integer sets A[] and b[] as input and prints results of following set operations: i. A union B (Write function set\_union()) ii. A intersection B (Write function set\_intersection()) iii. A-B and B-A (Write function set\_difference()).  Write a C program for the following:  1. A file name is command line argument. Display the contents of the file where each word will be displayed on a new line. Display a proper message if the file does not exist.  2. Display no. of vowels stored in the file.  3. Display no. of “the” stored in the file.  4. Copy contents of the file to another file. |

| **Name** | **Basic daper** | **bonper** | **loadnet** |  |
| --- | --- | --- | --- | --- |
| Amit | 2500 55 | 33.3 | 250.0 |  |

| **Name** | **Basic** | **Salary** |  |
| --- | --- | --- | --- |

**ITM(SLS) Baroda University**

**School of Computer Science, Engineering and Technology**

**B.Tech - Semester II**

**Course Name : Data Structures and Algorithms-1**

**Course Code: C2210C2**

**Course Type: Core**

**Teaching Scheme:**

| **Teaching Scheme** | | | **Credits** | **Examination Marks** | | | **Total**  **Marks** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **L** | **T** | **P** | **C** | **Theory Marks** | | **Practical Marks** |  |
| **3** | **0** | **4** | **5** | **External** | **Internal** | **External Internal** |  |
| **40** | **60** | **20 30** | **150** |

**Course Outcome:**

After completing the course, the student shall be able to :

| **Sr. No. Course Outcome Bloom’s Level** |
| --- |
| CO1 Understand Asymptotic Notation and concept of  Understanding  ADT |
| CO2 Identify data structures suitable to solve problems Understanding |
| CO3 Develop, analyze and implement algorithms for  Applying  linear data structures like arrays and linked list |
| CO4 Develop, analyze and implement algorithms for  Application  non-linear algorithms like graphs and trees |
| CO5 Implement searching and sorting algorithms along  Application  with  Specified mini project.  CO6 Understand the concept of Time Complexity using  Application  different sorting algorithms. |

**Programme Outcomes:**

| **Sr. No.** | **POs** | **Bloom’s Level** |
| --- | --- | --- |
| PO1 | Design, develop and maintain computing systems using concepts from mathematics, science and engineering and program core courses | Creation |
| PO2 | To implement software design systems, components, or processes to meet the desired needs and specifications. | Creation |
| PO3 | Design algorithms for real world computational problems and analyze their complexities | Creation |
| PO4 | To adapt to the usage of modern tools and technologies and recent software | Application |
| PO5 | Analyze and interpret data and discover knowledge to provide solutions to engineering problems | Analyze |
| PO6 | Assess the security, privacy, quality and cost parameters in developing software systems | Evaluate |
| PO7 | Ability to work effectively in multi-disciplinary teams using common tools and standards to achieve project objectives | Application |
| PO8 | Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice. | Creation |
| PO9 | Proficiency in oral, written and visual means for technical presentations and documentations | Application |
| PO10 | Engage in lifelong learning through self-directed and independent study of new techniques and tools | Creation |

**Competencies:**

After completion of the this course, our students shall acquire the following generic competencies:

| **Sr. No** | **Competencies** |
| --- | --- |
| 1 | To implement the searching techniques, for example, in stock market applications for buying and selling of different stocks. |

| 2 | To implement games based on Tree and Graph data structures, for example, the maze game using white block problem. |
| --- | --- |
| 3 | To implement applications involving sorting of structured data, with efficiency and stability requirements |
| 4 | To implement linked data structure based applications, for example, Implementations of polynomials operations using stack (Addition , subtraction) |
| 5 | To implement applications, using applicable Data Structures, starting with implementation of their ADT’s, for example, the Queue ADT |

**Mapping:**

|  | **PO1** | **PO2** | **PO3** | **PO4 PO5** | **PO6** | **PO7 PO8** | **PO9** | **PO10** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **CO1** | **3** | **3** | **3** |  | **3** | **2** |  | **2** |
| **CO2** | **3** | **3** | **3** |  | **3** | **2** |  | **2** |
| **CO3** | **3** | **3** |  | **3** | **3** | **2** |  | **2** |
| **CO4** |  | **2** | **3** | **2** | **2** | **2** |  | **2** |
| **CO5** |  | **2** | **3** | **2** | **2** | **2** |  | **2** |
| **CO6** | **3** | **3** | **3** | **2 2** | **2** | **2** | **2** | **2** |

**Course Contents:**

| **Unit** | **Topics** | **time(in hr)** | **% evaluation** |
| --- | --- | --- | --- |
| 1 | **Introduction**: Pseudocode, Abstract Data Type, Data types **–** Primitive and non-primitive, Types of Data Structures**-** Linear & Non Linear Data Structures, ADT model and ADT implementation,Algorithm efficiency-Asymptotic Notations: Big O, Big Omega and Big Theta; Properties of Asymptotic Notations | 6 | 5 |
| 2 | **Recursion-** designing recursive algorithms,defining recursive algorithms,examples-factorial of a number, tower of hanoi , fibonacci series | 4 | 5 |

| 3 | **Linear Lists:**  **Array:** Representation of arrays, Operations on Arrays, Applications of arrays  **Stack:** Stack-Definitions & Concepts, Operations On Stacks, Applications of Stacks-Polish Expression, Reverse Polish Expression. **Queue:** Representation Of Queue, Operations On Queue, Circular Queue, Priority Queue, Array representation of Priority Queue, Double Ended Queue, Applications of Queue  **Linked List:** Singly Linked List, Doubly Linked list,Circular linked list ,Linked implementation of Stack, Linked implementation of Queue, Applications of linked list | 10 | 20 |
| --- | --- | --- | --- |
| 4 | **Non-linear Lists:**  **Tree**-Definitions,Representation of binary tree, Tree traversal -Inorder, postorder, preorder, Threaded binary tree, Binary search trees, Conversion of General Trees To Binary Trees. | 6 | 20 |
| 5 | Applications Of Trees Some balanced tree mechanism; e.g. Heap, AVL trees; 2-3 trees; Height Balanced, Weight Balance, Red black tree; Multi-way search tree: B and B+ tree; | 5 | 15 |
| 6 | **Graph:** Graph Terminologies, Adjacency Matrices and List Representations of Graphs; Elementary Graph Operations: Depth First Search & Breadth first Search, Spanning Trees: Shortest path, Minimal spanning tree using graphs,networks. | 6 | 20 |
| 7 | **Sorting**: basics, selection, insertion, exchange, QuickSort, external sort | 4 | 10 |
| 8 | **Searching**: Linear search, Binary search,list, hashed lists | 4 | 5 |
| **Total Hours** | | **45 Hours** |  |

**Reference Text:**

| Sr. No | Book |
| --- | --- |
| 1 | Forouzan and Gilberg ,“Data Structures - A pseudocode Approach with C” 2nd Ed. |

Note: Python implementation of the various data structures and corresponding algorithms

will be provided by the teacher. The students will be encouraged to study the C implementations given in the Reference Text (1).

**Simulations/Animation:**

**TEDx Videos:**

| **Sr. No** | **TEDx Video** |
| --- | --- |
| **T1** | https://www.youtube.com/watch?v=ENWVRcMGDoU - How algorithms shape our world.- Kevin Slavin  Kevin Slavin argues that we're living in a world designed for -- and increasingly controlled by -- algorithms. In this riveting talk from TEDGlobal, he shows how these complex computer programs determine espionage tactics, stock prices, movie scripts, and architecture. Slavin also warns that we are writing code we can't understand with implications we can't control. |
| **T2** | https://www.youtube.com/watch?v=6hfOvs8pY1k - What’s an algorithm - David J. Malan  An algorithm is a method of solving problems both big and small. Though computers run algorithms constantly, humans can also solve problems with algorithms. David J. Malan explains how algorithms can be used in seemingly simple situations and also complex ones. |

**Other Videos:**

| **Sr.**  **No** | **About Video** | **Link** | **Topic** |
| --- | --- | --- | --- |
| O1. | Stanford Lecture - Don  Knuth: The Analysis of  Algorithms (2015,  recreating 1969) | https://www.youtube.com/watch?v=v kU NH9r6UCI | Algorithm |

O2. Erik Demaine, Ronald Rivest, and SriniDevadas.

*6.006*

*Introduction to*

*Algorithms.* Spring 2008.

Massachusetts

Institute of Technology: MIT OpenCourseWare,https://ocw. mi t.edu. License:Creative

Commons BY-NC-SA.

https://ocw.mit.edu/courses/electrica l

engineering-and-computer-science/ 6-

006-introduction-to-algorithms-spri ng 2008/

Algorithm

| O3. | Dr. Rob Edwards from San Diego State University  provides an introduction to complexity measures | https://www.youtube.com/watch?v=z gC  nMvvw6Oo&list=PLpPXw4zFa0uK Kh aSz87IowJnOTzh9tiBk | Stack ,  Queue ,  Linked list |
| --- | --- | --- | --- |
| O4. | MIT 6.0002 Introduction to Computational Thinking  and Data Science, Fall  2016 View the complete  course:http://ocw.mit.edu/  6- 0002F16 Instructor:  Eric  Grimson | https://www.youtube.com/watch?v=B 7h VxCmfPtM | Heap and  Heaps sort |
| O5. | MIT 6.006 Introduction to Algorithms, Fall 2011 View the complete  course:http://ocw.mit.edu/6- 006F11 Instructor: Victor  Costan | https://www.youtube.com/watch?v=r 5p Xu1PAUkI | Recursion  Tree,  Binary  search tree |
| O6. | How to Construct a Tree |  Stanford University | https://www.youtube.com/watch?v= QjF HWUsoZBw | Construct  a Tree |
| O7. | Hashing-based data  structures and applications - Michael Mitzenmacher,  Harvard  University | https://www.youtube.com/watch?v=g 4Z yjRaYe3k | Hashing |

**Related MOOCs courses**

| **Sr. No** | **MOOC Course** |
| --- | --- |
| M1 | Programming, Data Structures And Algorithms Using Python by Prof. MadhavanMukund 8 Weeks on NPTEL. |
| M2 | Data Structures and Algorithms Specialization Offered By University of California San Diego National Research University Higher School of Economics(coursera) |

**Lab Experiments:**

| P1 Write a python code to demonstrate Call by Value and Call by reference. |
| --- |
| P2 Write a python program to implement Bubble Sort and find the complexity. |
| P3 Write a python program to implement Linear search and Binary Search and find its time complexity. |
| P4 Implement a python program for stack that performs the following operations using array.  (a) PUSH (b) POP (c) PEEP (d) CHANGE (e) DISPLAY |
| P5 Implement a python program to convert infix notation to postfix notation using stack. |
| P6 Write a python program to implement QUEUE using arrays that performs following operations  (a) Insert  (b) Delete  (c) Display |
| P7 Write a python program to implement Circular Queue using arrays that perform the following operations.  a. Insert  b. Delete  c. Display |
| P8 Write a menu driven python program to implement following operations on the singly linked list.  (a) Insert a node at the front of the linked list.  (b) Insert a node at the end of the linked list.  (c) Insert a node such that the linked list is in ascending order(according to the info. Field)  (d) Delete a first node of the linked list.  (e) Delete a node before a specified position.  (f) Delete a node after specified position. |
| P9 Write a python program to implement the following operations on the circular |

| linked list.  Insert a node at the end of the linked list.  Insert a node before the specified position.  Delete a first node of the linked list.  Delete a node after specified position. |
| --- |
| P10 Write a program python to implement following operations on the doubly linked list.  (a) Insert a node at the front of the linked list.  (b) Insert a node at the end of the linked list.  (c) Delete a last node of the linked list.  (d) Delete a node before a specified position. |
| P11 Write a python program which creates binary search trees and  implements recursive methods for inorder,preorder and post order  traversal. |
| P12 Write a python program to implement stack using linked lists. |
| P13 Write a python program to implement a queue using a linked list. |
| P14 Write a python program to implement Merge Sort |
| P15 Write a program python to implement Quick Sort |
| P16 Write a python program to implement DFS and BFS. |
| P17 Write a python program MID(Key,Hash)which uses the mid square method to find a 2-digit hash address HASH of 4 digit employee number key. |

**Mini Projects :**

Apart from the above laboratory work, each student will implement a teacher specified individual small project requiring data structure selection, algorithm selection, determining the test data sets and a documentation for a 10-15 minutes presentation of his/her work. Some of the examples of projects are mentioned below.

| **Sr. No** | **Project** |
| --- | --- |
| MP1 | Design and Develop the index for a text book of at least 100 pages using alphabets. |
| MP2 | Design a Student Prerequisite Subjects Management System requires the use of linked list or tree to store different courses and their prerequisites and based on this list it will allow any student to take any course or not. |

| MP3 | Simulate a simple dictionary. Assume each character contains at least 10 vocabularies. Create an index page for all characters. Retrieve the word using index value. Assume that the index characters from a to z. |
| --- | --- |

**ITM(SLS) Baroda University**

**School of Computer Science, Engineering and Technology**

**B.Tech – Semester II**

**Course Name: Computer Graphics**

**Course Code:C2210C3**

**Course Type: Core**

**Teaching Scheme:**

| **Teaching Scheme** | | | **Credits** | **Examination Marks** | | | | **Total Marks** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **L** | **T** | **P** | **C** | **Theory Marks** | | **Practical Marks** | |
| **External** | **Internal** | **External** | **Internal** |
| **4** | **0** | **2** | **5** | **40** | **60** | **20** | **30** | **150** |

**About Computer Graphics:**

Computer graphics is a sub-field of computer science which studies methods for digitally synthesizing and manipulating visual content. Although the term often refers to the study of three dimensional computer graphics, it also encompasses two-dimensional graphics and image processing.

Basically there exists two kinds of computer graphics - raster (composed of pixels) and vector (composed of paths). Raster images are more commonly called bitmap images. A bitmap image uses a grid of individual pixels where each pixel can be a different color or shade.

The importance of computer graphics lies in its applications. Interactive computer graphics allows the physician to interpret this large volume of data in new and useful ways. Computer graphics has also expanded the boundaries of art and entertainment.

**Course Overview:**

In this course the students will learn about the concepts of computer graphics. It starts with an overview of interactive computer graphics, two dimensional system and mapping, then it presents the most important drawing algorithm, two-dimensional transformation; Clipping, filling and an introduction to 3-dimensional graphics.

**Prerequisite:**

This course does not require any programming background. This course helps the students to learn programming in python.

**Course Outcome :**

At the end of the course the student will be able to:

| **CO's Course Outcome** | **Bloom Taxonomy** |
| --- | --- |

|  | **Level** |
| --- | --- |
| CO1 Understand Random and Raster scan systems, Graphics software and standards | Remember |
| CO2 Demonstrate Points, lines, circles and ellipses as graphics primitives | Understanding |
| CO3 Illustrate Fill area primitives including scan-line polygon filling | Understanding |
| CO4 Ability to draw basic 2-D geometric shapes | Applying |
| CO5 Ability to apply various 2-D transformations on geometric shapes | Applying |
| CO6 Ability to draw simple 3-D geometric shapes | Applying |

CO7 Understand 3-D transformations and projections Understanding CO8 Demonstrate Points, lines, and polygon clippings Applying

**CO-PO Mapping:**

|  | **PO1** | **PO2** | **PO3** | **PO4** | **PO5** | **PO6 PO7** | **PO8** | **PO9** | **PO10** | **PO11** | **PO12** | **PO13** | **PO14** | **PO15** | **PO16** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **CO1** | 1 | 3 |  | 3 | 3 | 3 2 | 2 |  | 1 | 1 | 1 | 2 | 3 | 1 | 3 |
| **CO2** | 2 | 3 | 3 | 3 | 1 | 1 2 | 2 | 1 |  |  |  | 2 | 3 | 1 | 3 |
| **CO3** | 3 | 3 | 3 | 3 | 3 | 3 2 | 3 | 2 | 2 | 2 | 1 | 2 | 3 | 2 | 2 |
| **CO4** | 1 | 3 | 3 | 3 | 3 | 3 3 | 2 | 2 |  | 1 |  |  | 3 | 1 | 2 |
| **CO5** | 1 | 3 | 3 | 3 | 3 | 2 1 | 2 | 1 |  |  | 1 | 2 | 2 |  | 2 |
| **CO6** | 2 | 1 | 1 | 3 | 3 | 1 1 | 2 |  |  | 2 |  |  | 2 |  | 1 |
| **CO7** | 1 | 1 | 2 | 1 | 2 | 1 2 | 1 | 1 |  |  | 1 |  | 2 | 1 |  |

**CO8** 2 1 3 1 3 1 1 2 1 2 2

**Course Outline :**

| **Unit**  **Topics**  **#** | **Hours L+P** |
| --- | --- |
| 1 **Basic of Computer Graphics**:  Basic of Computer Graphics: Coordinate Systems, Graphics APIs; Display  devices, Random and Raster scan systems, Graphics software and standards; Color models: properties of light, XYZ, RGB, YIQ and CMY color models | 8 + 2 |
| 2 **Graphics Primitives**:  Points, lines, circles and ellipses as primitives, scan conversion algorithms for primitives, Fill area primitives including scan-line polygon filling, inside-outside test, boundary and flood-fill | 10+10 |