

DEPARTMENT OF COMPUTER SCIENCE
PROGRAMME OUTCOME, PROGRAMME SPECIFIC OUTCOME,
LEARNING OUTCOME AND COURSE OUTCOME

PROGRAMME OUTCOME	<p>Formulate and develop computational arguments in a logical manner. Also when there is a need for information, the student will be able to identify, locate, evaluate, and effectively use that information for handling issues or solving problems at hand.</p> <p>Acquire good knowledge and understanding in advanced areas of programming and its applications.</p>
PROGRAMME SPECIFIC OUTCOME	Will be able to apply critical thinking skills to solve problems , to critically interpret numerical and graphical data, to read and construct arguments and proofs, to use computer technology appropriately to solve problems and to promote understanding, to apply Computational and Logical knowledge to a career related to Computer Science thus cultivating a proper attitude for higher learning .
LEARNING OUTCOME	Students will be well equipped to critically analyse a given problem, understand and write a program to represent the problem and interpret the resulting solution. Students are well prepared for higher studies in their chosen field.
COURSE OUTCOMES	
C-PROGRAMMING:	<p>C programming is utilized for the Development of system software and Desktop application. Some C programming applications are given below.</p> <ul style="list-style-type: none"> • Building Compilers of various Languages that can take data from different programming and transform them into lower-level machine-dependent language. • To assess any numerical equation, use c programming. • UNIX Kernel is entirely produced in C programming. • C programming can be utilized to create Network Devices. • To improve software applications like databases and spreadsheets. • C programming can be utilized to create system software like the compiler and operating system. • C programming can be utilized to design an Operating System. • This language can be utilized to design compilers. • It is used for developing Graphical applications for computers and mobile.
C-PROGRAMMING PRACTICALS – I	Develop skill to create simple programs in TURBO C, to generate particular sequences , to find largest or smallest of the given numbers, to check for palindromes... , Matrix addition, Matrix multiplication, GCD of two numbers....using string handling functions...
DATA STRUCTURES USING C	<p>Data structures using C gives solutions to standard problems in detail and gives an insight to students as how efficient it is to use each one of the algorithms and methods. It also helps to learn the science of evaluating the efficiency of an algorithm and how to best choose an algorithm.</p> <ul style="list-style-type: none"> ▪ Introduction and Overview of Data Structures, data structures operations, Abstract data types, algorithms complexity, time-space tradeoff.

	<p>Preliminaries: Mathematical notations and functions, Algorithmic notations, control structures, Complexity of algorithms, asymptotic notations for complexity of algorithms. String Processing: Definition, Storing Strings, String as ADT, String operations, word/text processing, Pattern Matching algorithms.</p> <ul style="list-style-type: none"> ▪ Arrays: Definition, Linear arrays, arrays as ADT, Representation of Linear Arrays in Memory, Traversing Linear arrays, Inserting and deleting, Sorting: Bubble sort, Insertion sort, Selection sort, Searching: Linear Search, Binary search, Multidimensional arrays, Matrices and Sparse matrices. ▪ Linked list: Definition, Representation of Singly linked list in memory, Traversing a Singly linked list, Searching a Singly linked list, Memory allocation, Garbage collection, Insertion into a singly linked list, Deletion from a singly liked list; Doubly liked list, Header liked list, Circular linked list. ▪ Stacks – Definition, Array representation of stacks, Linked representation of stacks, Stack as ADT, Arithmetic Expressions: Polish Notation, Application of Stacks, Recursion, Towers of Hanoi, Implementation of recursive procedures by stack. Queues – Definition, Array representation of queue, Linked list representation of queues Types of queue: Simple queue, Circular queue, Double ended queue, Priority queue, Operations on Queues, Applications of queues. <p>Graphs: Graph theory terminology, Sequential representation of Graphs: Adjacency matrix, traversing a Graph. Tree – Definitions, Binary trees, Representing binary trees in memory, Traversing Binary Trees, Binary Search Trees, Searching, Inserting and Deleting in a Binary Search Tree.</p>
DATA STRUCTURES USING C PRACTICALS	Students develop skills in implementations and applications of data structures. Implementing basic algorithms for sorting and searching. Implementing basic data structures such as stacks, queues and trees. Applying algorithms and data structures in various real-life software problems.
DBMS	<ul style="list-style-type: none"> • Introduction, Characteristics of database approach, Advantages of using the DBMS approach, History of database applications. • Data Models, Schemas, and Instances. Three schema architecture and data independence, database languages, and interfaces, The Database System environment. • Entity types, Entity sets, attributes, roles, and structural constraints, Weak entity types, ER diagrams, examples, Specialization and Generalization. • Relational Model Concepts, Relational Model Constraints and relational database schemas, Update operations, transactions, and dealing with constraint violations. • Unary and Binary relational operations, additional relational operations (aggregate, grouping, etc.) Examples of Queries in relational algebra. • Relational Database Design using ER-to-Relational mapping. • SQL data definition and data types, specifying constraints in SQL, retrieval queries in SQL, INSERT, DELETE, and UPDATE statements in SQL, Additional features of SQL.
DBMS LAB	Data Models, Schemas, and Instances. Three schema architecture and data independence, database languages, and interfaces, The Database System environment.SQL data definition and data types, specifying

	<p>constraints in SQL, retrieval queries in SQL, INSERT, DELETE, and UPDATE statements in SQL, Additional features of SQL.</p>
SHELL PROGRAMMING USING UNIX	<p>The shell is a program within a Linux or Unix operating system which allows you to enter commands for execution by the system. When a terminal window is opened on a linux computer, it starts the shell program which presents an interface to enter commands. This interface is known as the command line interface. When a command is entered, it is executed by the shell and the output is displayed on the screen.</p> <p>Shell script is utilized for the Development of Desktop application. Some shell programming benefits are given below.</p> <ul style="list-style-type: none"> • Shell scripting is an important skill to become a better developer or programmer. • some simple control and loop constructs along with bash variables makes a lot of difference • To automate the frequently performed operations. • To run sequence of commands as a single command. • To improve software applications like databases and spreadsheets. • Easy to use • Portable (It can be executed in any Unix-like operating systems without any modifications) • Tool kit
SHELL PROGRAMMING PRACTICALS – I	<p>Develop skill to create simple programs in shell, to generate particular sequences , to find largest or smallest of the given numbers, to check for palindromes... , GCD of two numbers....using string handling functions, create a payroll system, generating greeting message...</p>
OBJECT ORIENTED PROGRAMMING USING JAVA.	<p>Java™ has significant advantages over other languages and environments that make it suitable for just about any programming task. The advantages of Java are as follows: Java is easy to learn. Java was designed to be easy to use and is therefore easy to write, compile, debug, and learn than other programming languages.</p> <p>Few of the advantages of Java Programming Language are:</p> <ul style="list-style-type: none"> • Platform Independence : Java is platform independent, which means that if a program is written and compiled in Java on any platform(underlying hardware and software), it can run on any other platform of similar capabilities subjected to available hardware (computing power, memory and disk space) and software(installed JRE) requirements. This is achieved by the virtue of the software component which is called Java Virtual Machine which is an abstract computing machine . • Automatic Storage Management : This is done using a garbage collector, which avoids the safety problems of explicit de-allocations. • Avoids Unsafe Constructs This helps in running the program correctly, else imagine a scenario where the language does not check the array index and returns a value which is stored in the memory location, but the memory location is not within the boundaries of the array. This could pose a big question on the correctness of your program.

- **No explicit declaration order required :** It does not require types (Classes or Interfaces) or their members(fields and methods) to be declared before they are used. It only becomes significant when we declare local variables, local classes and the order of initializers of fields in a class or interface. All other variables are automatically initialized to a default value, the Java programming language does not automatically initialize local variables.
- **Type Safety :** Java is a safe programming language because of its static typed and strongly typed nature. **Statically typed** means every variable is declared and has a type and the type is known at the compile time. **Strongly typed** means that a variable can only be assigned a set of values which are compatible with the type of the variable. This also helps in identifying most of the errors at compile time and clearly distinguishes them with the runtime errors.
- **Exception Handling Support :** Java has defined an Exception Hierarchy and has a great support for Exception Handling. This offers a lot of robustness and helps the developers to handle compile time exceptions without fail and write a defensive program.
- **Multi-threading Support :** Java encapsulates the underlying system's thread infrastructure and offers a great and easy support for multi threading. There is no explicit need of writing a monitor code for acquiring a locks on objects.
- **Applet Programming:** Introduction, How Applets Differ from Applications, Preparing to Write Applets, Building Applet Code, Applet Life Cycle, Creating an Executable applet, Designing a Web Page, Applet Tag, Adding Applet to HTML File, running the Applet, More About HTML Tags, Displaying Numerical Values, Getting Input from the User.
- **Graphics Programming:** Graphics programming: Introduction, The Graphics Class, Lines and rectangles, circles, and Ellipses, Drawing Arcs, Drawing Polygons, Lines Graphs, Using Control Loops in Applets, Drawing Bar Charts.
- **Input/Output Files in JAVA:** Introduction, Concept of Streams, Stream Classes, Byte Stream Classes, Character Stream Classes, Using Streams, Other Useful I/O Classes, Using the File Class, Input / Output Exceptions, Creation of Files, Reading / Writing Characters, Reading / Writing Bytes, Handling Primitive Data Types, Concatenating and Buffering Files, Interactive Input and output, Other Stream Classes.

JAVA PROGRAMMING PRACTICALS

Develop core java programs, to implement command line arguments , inbuilt classes like String Math,Vector etc
Multiple threading, Applet programming, Graphics programming and I/O stream programming etc.

VISUAL PROGRAMMING

- Customizing a Form - Writing Simple Programs - Toolbox - Creating Controls - Name Property - Command Button - Access Keys - Image Controls - Text Boxes - Labels - Message Boxes - Grid - Editing Tools - Variables - Data Types - String - Numbers.
- Displaying Information - Determinate Loops - Indeterminate Loops - Conditionals - Built-in Functions - Functions and Procedures.
- Lists - Arrays - Sorting and Searching - Records - Control Arrays - Combo Boxes - Grid Control - Projects with Multiple forms - Do Events and Sub Main - Error Trapping.
- VB Objects - Dialog Boxes - Common Controls - Menus - MDI Forms - Testing, Debugging and Optimization - Working with Graphics.
- Monitoring Mouse activity .- File Handling - File System Controls - File

	System Objects - COM/OLE - automation - DLL Servers - OLE Drag and Drop.
VISUAL PROGRAMMING LAB	Writing Simple Programs - Toolbox - Creating Controls - Name Property - Command Button - Access Keys - Image Controls - Text Boxes - Labels - Message Boxes - Grid - Editing Tools - Variables - Data Types - String - Numbers. VB Objects - Dialog Boxes - Common Controls - Menus - MDI Forms - Testing, Debugging and Optimization - Working with Graphics.
WEB PROGRAMMING	<p>Web programming refers to the writing, markup and coding involved in Web development, which includes Web content, Web client and server scripting and network security. The most common languages used for Web programming are XML, HTML, JavaScript. Web programming is different from just programming,</p> <ul style="list-style-type: none"> • Fundamentals of Web: Internet, WWW, Web Browsers, and Web Servers, URLs, MIME, HTTP, Security, The Web Programmers Toolbox. XHTML: Origins and evolution of HTML and XHTML, Basic syntax, Standard XHTML document structure, Basic text markup, Images, Hypertext Links, Lists, Tables. • HTML and XHTML: Forms, Frames in HTML and XHTML, Syntactic differences between HTML and XHTML. CSS: Introduction, Levels of style sheets, Style specification formats, Selector forms, Property value forms, Font properties, List properties, Color, Alignment of text, The Box model, Background images. • Java Script: Overview of JavaScript; Object orientation and JavaScript; General syntactic characteristics; Primitives, Operations, and expressions; Screen output and keyboard input; Control statements; Object creation and Modification; Arrays; Functions; Constructor; Pattern matching using expressions; Errors in scripts; Examples. • Java Script and HTML Documents: The JavaScript execution environment; The Document Object Model; Element access in JavaScript; Events and event handling; Handling events from the Body elements, Button elements, Text box and Password elements; The DOM 2 event model; The navigator object; DOM tree traversal and modification. • Dynamic Documents with JavaScript: Introduction to dynamic documents; Positioning elements; Moving elements; Element visibility; Changing colors and fonts; Dynamic content; Stacking elements; Locating the mouse cursor; Reacting to a mouse click; Slow movement of elements; Dragging and dropping elements. XML: Introduction; Syntax; Document structure; Document Type definitions; Namespaces; XML schemas; Displaying raw XML documents; Displaying XML documents with CSS; XSLT style sheets; XML Processors; Web services.
Web PROGRAMMING PRACTICALS	Develop Web programs to implement many tags in web page. JavaScript code to Evaluates the expression, dynamic effects, applying CSS, Create a web page to display mouse position and use key events.
PROJECT	Final Year students will be doing Project on Visual Basic/Visual Programming.