BANKURAUNIVERSITY

Curriculum and Credit Framework for

Computer Science

(Basic, Honours and Honours with Research)

With effect from the Academic Year 2023-2024

SEMESTER–I										
Sl. No.	Course Code	Course Title	Credit	Marks				No. of Hours		
				IA	ESE		Total	L	T	P
1	CSC/101/ MJC-01	Introduction to Programming with C	4	10	40		50	3	0	2
					T:25	L:15				
2	CSC/102/ MN-01	Introduction to Programming with C	4	10	40		50	3	0	2
					T:25	L:15				
3	CSC/103/ MD-01	Computer Fundamentals	3	10	40		50	3	0	0
					T:40	L:00				
4	CSC/104/ SEC-01	PC Software Laboratory	3	10	40		50	0	0	6
					T:00	L:40				
5	CSC/105/ AEC-01	Compulsory English: Literature and Communication	2							
6	CSC/106/ VAC-01	Environmental Studies	4							
Total in Semester-I			**20**							

SEMESTER–II										
Sl. No.	Course Code	Course Title	Credit	Marks				No. of Hours		
				IA	ESE		Total	L	T	P
1	CSC/201/ MJC-02	Data Structure using C	4	10	40		50	3	0	2
					T:25	L:15				
2	CSC/202/ MN-02	Data Structure using C	4	10	40		50	3	0	2
					T:25	L:15				
3	CSC/203/ MD-02	Programming Methodology	3	10	40		50	3	0	0
					T:40	L:00				
4	CSC/204/ SEC-02	Python Programming	3	10	40		50	0	0	6
					T:00	L:40				
5	CSC/205/ AEC-02(A/B/C)	Modern Indian Languages: Choose any one of the following A.Bengali /B. Santali /C. Sanskrit	2							
6	CSC/206/ VAC-02	**Any one of following:** a. Health and Wellness b. Understanding India: Indian Philosophical Traditions and Value Systems c. Basics of Indian Constitution d. Arts and Crafts of Bengal e. Historical Tourism in West Bengal.	4							
Total in Semester-II			**20**							

7	CSC/207/INT-1	Summer Internship (Additional for getting **UG CERTIFICATE**)	4	10	40		50	0	0	6
					T:00	L:40				
UG CERTIFICATE		Student who opt to exit after completion of the first year (semester-I & II) and have secure 40 credits will be awarded a **UG CERTIFICATE** if , in addition ,the Summer Internship (4 credit)(SL. NO:-7) during summer vacation of the first year .								

SEMESTER–I

COMPUTER SCIENCE (MJC-01/MN-01): Introduction to Programming with C
Credit: 04 (03 Theory + 01 Practical) L/T/P: 3/0/2

Course Learning Outcomes: After successful completion of the Course a student will be able to:

- Learn about basic operations of a computer.
- Develop problem solving skills coupled with top down design principles.
- Become skilled at developing simple algorithms and flow charts.
- Convert the algorithms into simple C programs.
- Develop simple C programs for solving real life problems.

Theory

Unit I:
Introduction to Programming with C
History of C, Overview of Procedural Programming and Object-Orientation Programming, Using main() function, Compiling and Executing Simple Programs in C.

Unit II:
Data Types, Variables, Constants, Operators and Basic I/O
Declaring, Defining and Initializing Variables, Scope of Variables, Using Named Constants, Keywords, Data Types, Casting of Data Types, Operators (Arithmetic, Logical and Bitwise), Using Comments in programs, Character I/O (get c, get char, put c, put char etc), Formatted and Console I/O (printf(), scanf()), Using Basic Header Files (stdio.h , conio.h etc).

Unit III:
Expressions, Conditional Statements and Iterative Statements
Simple Expressions in C (including Unary Operator Expressions, Binary Operator Expressions), Understanding Operators Precedence in Expressions, Conditional Statements (if construct, switch-case construct), Understanding syntax and utility of Iterative Statements (while, do-while, and for loops), Use of break and continue in Loops, Using Nested Statements (Conditional as well as Iterative).

Unit IV:
Functions and Arrays
Utility of functions, Call by Value, Call by Reference, Functions returning value, Void functions, Inline Functions, Return data type of functions, Functions parameters, Differentiating between Declaration and Definition of Functions, Command Line Arguments/Parameters in Functions, Functions with variable number of Arguments. Creating and Using One Dimensional Arrays (Declaring and Defining an Array, Initializing an Array, Accessing individual elements in an Array, Manipulating array elements using loops), Use Various types of arrays (integer, float and character

arrays / Strings) Two- dimensional Arrays (Declaring, Defining and Initializing Two Dimensional Array, Working with Rows and Columns), Introduction to Multi-dimensional arrays.

Unit V:
Derived Data Types (Structures and Unions)
Understanding utility of structures and unions, Declaring, initializing and using simple structures and unions, Manipulating individual members of structures and unions, Array of Structures, Individual data members as structures, Passing and returning structures from functions, Structure with union as members, Union with structures as members.

Unit VI:
Pointers in C
Understanding a Pointer Variable, Simple use of Pointers (Declaring and Dereferencing Pointers to simple variables), Pointers to Pointers, Pointers to structures, Problems with Pointers, Passing pointers as function arguments, Returning a pointer from a function, using arrays as pointers, Passing arrays to functions. Pointers vs. References, Declaring and initializing references, using references as function arguments and function return values.

Unit VII:
Memory Allocation in C
Differentiating between static and dynamic memory allocation, use of malloc , calloc and free functions, storage of variables in static and dynamic memory allocation.

Unit VIII:
File I/O, Preprocessor Directives
Opening and closing a file, Reading and writing Text Files, Using put(), get(), read() and write() functions, Random access in files, Understanding the Preprocessor Directives, Macros.

Practical

Unit I. Given the problem statement, students are required to formulate problem, develop flowchart/algorithm, write code, execute and test it. Students should be given assignments on following :

a) To learn elementary techniques involving arithmetic operators and mathematical expressions, appropriate use of selection (if, switch, conditional operators) and control structures

b) Learn how to use functions and parameter passing in functions.

Unit II. Students should be given assignments on following:

a) Write Programs to learn the use of strings and string handling operations.

b) Problems which can effectively demonstrate use of Arrays. Structures and Union.

c) Write programs using pointers.

d) Write programs to implement search algorithms.

Reference Books

1. C Programming, Karnighan,&Ritchie, PHI
2. Herbtz Schildt, "C++: The Complete Reference", Fourth Edition, McGraw Hill.2003
3. E Balaguruswamy, " Programming with C", Tata McGraw-Hill Education, 2008.
4. Programming through C, Richard Johnsonbaugh and Martin Kalin, Pearson Education
5. Programming in C, B.S. Gottfried, Sahaum Series.
6. Y Kanetkar, "Let us C", BPB

COMPUTER SCIENCE (SEC-01): PC Software Laboratory
Credit: 03 L/T/P: 0/0/6

Course Learning Outcomes: After successful completion of the Course:

- Learners will be able to claim proficiency in MS-Office.
- Learners will able to independently create professional-looking documents and presentations.
- Learners will be familiar with some advanced Word Power Point and Excel functions.

Practical

Unit I: MS Windows: Windows 7 and higher version, Desk top cell user interface action, icon on desktop, closing windows, renaming icons, resizing windows(maximizing and minimizing), control panel.

Unit II: MS Word: Overview, creating, saving, opening, importing, exporting, and inserting files, formatting pages, paragraphs and sections, indents and outdents, creating lists and numbering. Headings, styles, fonts and font size, editing, positioning, viewing texts, searching and replacing text, inserting page breaks, page numbers, bookmarks, symbols, and dates. Using tabs and tables, header, footer and printing.

Unit III: MS Excel: Worksheet overview, entering information, worksheet creation, opening and saving workbook, formatting numbers and texts, protecting cells, producing charts, and printing operations. Application of Excel for obtaining statistical parameters, Mean, Median, Mode, average, co-relation, Regression.

Unit IV: MS Access: Introduction, understanding databases, creating tables, queries, forms, reports, adding graphs to your reports.

Unit V: PowerPoint: Slide creation with PowerPoint.

References:

1. A. Leon and M. Leon, Introduction to Computers with MS-Office, TMH.
2. Sushila Madan, Introduction to Essential tools, JBA, 2009.
3. Anita Goel, Computer Fundamentals, Pearson, 2012

COMPUTER SCIENCE (MD-01) Computer Fundamentals
Credit: 03 L/T/P: 3/0/0

Course Learning Outcomes: After successful completion of the Course students will be able to:

- Understand and be able to converse in basic computer terminology.
- Understand the concepts of input output devices of Computers.
- Learn the functional units and classify types of Computers.

- Understand an Operating System and its functioning principles.
- Have a basic knowledge about Computer Architecture and Organization.

Unit I: Introduction: Computer Systems, Generation of computers,uses and Types.

Unit II: Data Representation: Number System, character representation, binary arithmetic.

Unit III: Human Computer Interface: Types of software, Operating system as user interface, utility programs.

Unit IV: Devices: Input and output devices (with connections and practical demo), keyboard, mouse, joystick, scanner, OCR, OMR, bar code reader, web camera, monitor, printer, plotter

Unit V: Memory: Primary, secondary, auxiliary memory, RAM, ROM, cache memory, hard disks, optical disks.

Unit VI: Computer Organisation and Architecture: C.P.U., registers, system bus, main memory unit, cache memory, Inside a computer, SMPS, Motherboard, Ports and Interfaces, expansion cards, ribbon cables, memory chips, processors.

References:

1. U. Rajaraman and N. Adabala, Fundamentals of Computers, PHI.
2. P. K. Sinha and Preeti Sinha, Computers Fundamentals, BPB Publications.
3. Sanders H. Donald, Computer Concepts and Applications, McGraw-Hill.

Semester-II

COMPUTER SCIENCE (MJC-02/MN-02): Data Structure Using C
Credit: 04 (03 Theory + 01 Practical) L/T/P: 3/0/2

Course Learning Outcomes:

(After the completion of course, the students will have ability to):

- *To be familiar with fundamental data structures and with the manner in which these data structures can best be implemented; become accustomed to the description of algorithms in both functional and procedural styles*
- *To have knowledge of complexity of basic operations like insert, delete, search on these data structures.*
- *Ability to choose a data structure to suitably model any data used in computer applications.*
- *Design programs using various data structures Binary and general search trees, heaps etc.*
- *Ability to assess efficiency tradeoffs among different data structure implementations.*
- *Implement and know the applications of algorithms for sorting, searching etc.*

Theory

UNIT I. Basic concepts- Algorithm Specification-Introduction, Recursive algorithms, Data Abstraction, Performance analysis, Linear and Non Linear data structures, Singly Linked Lists-Operations, Concatenating, Circularly linked lists-Operations for Circularly linked lists, Doubly Linked Lists- Operations. Representation of single, two dimensional arrays.

UNIT II. Stack- Definition and Operations, Array and Linked Implementations, Applications - Valid Expression Checking (Parenthesis matching), Reversal of string, Infix to Postfix Conversion, Postfix Expression Evaluation, Recursion Implementation.

UNIT III. Queue - Definition and Operations, Array and Linked Implementations, Applications, Circular Queues - Insertion and Deletion Operations, Dequeue (Double Ended Queue) - Introduction.

UNIT IV. Sorting Methods – Bubble, Insertion, Selection, Using Divide-Conquer Approach (Quick and Merge sort), Comparison of Sorting Methods, Searching Methods – Linear and Binary.

UNIT V. Trees, Representation of Trees, Binary tree, Properties of Binary Trees, Binary Tree Representations- Array and Linked Representations, Binary Tree Traversals, Threaded Binary Trees, Binary Search tree - Creation, Insertion, Deletion and Search, Heap- Definition, Min heap, Max heap, Insertion and Deletion.

Practical

Students are required to write and practically execute programs to solve problem using various data structures. The teacher can suitably device problems which help students experiment using the suitable data structures and operations. Some of the problems are indicated below.

1. Write program that uses functions to perform the following:
   a) Creation of list of elements where the size of the list, elements to be inserted and deleted is dynamically given as input.
   b) Implement the operations, insertion, deletion at a given position in the list and search for an element in the list
   c) To display the elements in forward / reverse order.

2. Write a program to implement stack data structure and basic operations on it (Insertion, deletion). Write a program that demonstrates the application of stack operations (Eg: infix expression to postfix conversion, postfix evaluation).

3. Write a program to implement queue data structure and basic operations on it (Insertion, deletion, find length) and code at least one application using queues.

4. Write program that implements linear and binary search methods of searching for an elements in a list.

5. Write and trace programs to understand the various phases of sorting elements using the methods a) Bubble sort b) Insertion Sort c) Quicksort etc.

6. Write a program to create a Binary search tree and insert and delete from the tree. Write recursive and non-recursive routines to traverse a binary tree in preorder, inorder and postorder.

7. Write programs for recursion (e.g. Fibonacci numbers).

References:

1. Fundamentals of Data structures in C, 2nd Edition, E. Horowitz, S. Sahni and Susan Anderson-Freed, Universities Press.
2. Data structures and Algorithm Analysis in C, 2nd edition, M. A. Weiss, Pearson.
3. Lipschutz: Schaum's outline series Data structures Tata McGraw-Hill
4. Data Structure through C in Depth. S.K. Srivastava and Deepali Srivastava, B.P.B Publication.

COMPUTER SCIENCE (SEC-02): Python Programming
Credit: 03 L/T/P: 0/0/6

Practical

Course Learning Outcomes: After successful completion of the Course a student will be able to:
Learn the basic knowledge of Python.
• Students will be able to acquire programming skills in core Python.
• Students will be able to acquire Object Oriented Skills in Python.
• Students will be able to solve problems requiring the writing of well-documented programs in The Python language, including use of the logical constructs of that language.

Unit I. Introduction to Python, Python, Features of Python, Execution of a Python, Program, Writing Our First Python Program, Data types in Python. Python Interpreter and Interactive Mode; Values and Types: int, float, boolean, string, and list; Variables, Expressions, Statements, Tuple Assignment, Precedence of Operators, Comments; Modules and Functions, Function Definition and use, Flow of Execution, Parameters and Arguments

Unit II. Operators in Python, Input and Output, Control Statements. Boolean Values and operators, Conditional (if), Alternative (if-else), Chained Conditional (if-elif-else); Iteration: state, while, for, break, continue, pass; Fruitful Functions: Return Values, Parameters, Local and Global Scope, Function Composition, Recursion

Unit III. Arrays in Python, Strings and Characters. Strings: String Slices, Immutability, String Functions and Methods, String Module; Lists as Arrays. Illustrative Programs: Square Root, gcd, Exponentiation, Sum an Array of Numbers, Linear Search, Binary Search.

Unit IV. Functions, Lists and Tuples. List Operations, List Slices, List Methods, List Loop, Mutability, Aliasing, Cloning Lists, List Parameters; Tuples: Tuple Assignment, Tuple as Return Value; Dictionaries: Operations and Methods.

Unit V. Files and Exception: Text Files, Reading and Writing Files, Format Operator; Command Line Arguments, Errors and Exceptions, Handling Exceptions, Modules, Packages; Illustrative Programs: Word Count, Copy File.

The students are required to verify their ability to use core programming basics and program design with functions using Python programming language. The teacher shall program to strengthen the practical expertise of the students. The following is an indicative list of programs that can be practised.

1. Write a program to demonstrate different number data types in Python.
2. Write a program to perform different Arithmetic Operations on numbers in Python.
3. Write a program to create, concatenate and print a string and accessing sub-string from a given string.
4. Write a python script to print the current date in the following format "Sat Oct 11 02:26:23 IST 2020"
5. Write a program to create, append, and remove lists in python.
6. Write a program to demonstrate working with tuples in python.
7. Write a program to demonstrate working with dictionaries in python.
8. Write a python program to find largest of three numbers.
9. Write a Python program to construct the different pattern, using a nested for loop, Like

```
*
* *
* * *
* *
 *
```

10. Write a Python script that prints prime numbers less than 20.
11. Write a python program to define a module to find Fibonacci Numbers and import the module to another program.
12. Write a python program to define a module and import a specific function in that module to another program.
13. Write a program that inputs a text file. The program should print all of the unique words in the file in alphabetical order.
14. Write a Python class to convert an integer to a roman numeral.
15. Write a Python class to reverse a string word by word.

References:

1. Allen Downey, Think Python, Green Tea Press.
2. Wesley J. Chun, Core Python Programming, Pearson Education.
3. Mark Lutz, Learning Python, Oreilly Publication.
4. Kenneth A. Lambert, Fundamentals of Python: First Programs, Course Technology Inc.

COMPUTER SCIENCE (MD-02) : Programming Methodology
Credit: 03 L/T/P: 3/0/0

Course Learning Outcomes: After successful completion of the Course a student will be able to:

- Learn about basic operations of a computer.
- Become skilled at developing simple algorithms and flow charts.
- Convert the algorithms into simple Python programs.

Theory

Unit I: Planning the Computer Program: Concept of problem solving, Problem definition, Program design, Debugging, Types of errors in programming, Documentation.

Unit II: Techniques of Problem Solving: Flowchart, decision table, algorithms, Structured programming concepts, Programming methodologies viz. top-down and bottom-up programming.

Unit III: Overview of Programming: Structure of a Python Program, Elements of Python

Unit IV: Introduction to Python: Python Interpreter, Using Python as calculator, Python shell, Indentation. Atoms, Identifiers and keywords, Literals, Strings, Operators (Arithmetic operator, Relational operator, Logical or Boolean operator, Assignment, Operator, Ternary operator, Bit wise operator, Increment or Decrement operator).

Unit V:Creating Python Programs: Input and Output Statements, Control statements (Looping-while Loop, for Loop , Loop Control, Conditional Statement- if...else, Difference between break, continue and pass).

Unit VI: Structures: Numbers, Strings, Lists, Tuples, Dictionary, Date & Time, Modules, Defining Functions, Exit function, default arguments.

References:

1. Allen Downey, Think Python, Green Tea Press.
2. Wesley J. Chun, Core Python Programming, Pearson Education.
3. Mark Lutz, Learning Python, Oreilly Publication.
4. Kenneth A. Lambert, Fundamentals of Python: First Programs, Course Technology Inc.