|  |  |
| --- | --- |
|  | **PIR MEHR ALI SHAH ARID AGRICULTURE UNIVERSITY**  **University Institute of Information Technology** |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Programming Fundamentals (CSC-101)** | | | | |
| **Credit Hours:** | 4(3-3) | **Prerequisites:** | None | |
| **Teacher:** | Dr. kashif Sattar | Office: R-104 | kashif@uaar.edu.pk | |
| **Course Learning Outcomes (CLOs)** | | | | |
| At the end of course the students will be able to: | | | | **Bloom Taxonomy** |
| 1. Understand basic problem solving steps and logic constructs | | | | C2 (Understand) |
| 1. Apply basic programing concepts | | | | C3 (Apply) |
| 1. Design and implement algorithms to solve real world problems. | | | | C3 (Solve) |
| C=Cognitive domain | | | | |

|  |
| --- |
| **Course Outline:** |
| Introduction to problem solving, a brief review of Von-Neumann architecture, Introduction to programming, role of compiler and linker, introduction to algorithms, basic data types and variables, input/output constructs, arithmetic, comparison and logical operators, conditional statements and execution flow for conditional statements, repetitive statements and execution flow for repetitive statements, lists and their memory organization, multi-dimensional lists, introduction to modular programming, function definition and calling, stack rolling and unrolling, string and string operations, pointers/references, static and dynamic memory allocation, File I/O operations |
| **Course Objective:** |
| * To explore the logic of programming via the algorithm concepts and implement them in programming structures including functions, arrays, strings, and pointers. * To develop the program in C++ language, dry run and test it, debug it (fix errors if any). * To develop an insight of modular and generic programming using functions and structures. * To implement input/output (I/O) functionality to read from and write to text files and understand I/O streams |
| **Teaching Methodology:** |
| Lectures, Written Assignments, Practical labs, Semester Project, Presentations |
| **Courses Assessment:** |
| Mid Exam, Home Assignments, Quizzes, Project, Presentations, Final Exam |
| **Reference Materials:** |
| 1. **Object Oriented Programming in C++ by Robert Lafore** 2. Starting out with Python, 4th Edition, Tony Gaddis. 3. **Starting out with Programming Logic & Degins, 4th Edition, Tony Gaddis,** 4. **The C Programming Language, 2ndEd by Brian W. Kernighan, Dennis M. Ritchie** 5. Introduction to Computation and Programming Using Python: With Application to Understanding Data, 2nd Edition by Guttag, John 6. Practice of Computing Using Python, 3rd Edition by William Punch & Richard Enbody 7. **C How to Program, 7th Edition by Paul Deitel & Harvey Deitel** 8. **Problem Solving and Program Design in C++, 7th Edition by Jeri R. Hanly & Elliot B. Koffman** |

|  |  |  |  |
| --- | --- | --- | --- |
| **Week/Lecture #** | | **Theory** | **Practical** |
| Week 1 | Lecture-I | * **Introduction to problem solving, a brief review of Von-Neumann architecture:** | ***Lab Contents can be updated time to time.*** |
| Lecture-II | * **Introduction to programming** |  |
| Lecture-III | * C++ Basic Compilation |  |
| Practical-I |  | **Lab1: Programming and Debugging using Dev C++ and Microsoft Visual C++** |
| Practical-II |  | Getting started with C++ language environment, Phases of C++ program |
| Practical-III |  | Program editing, compiling, executing and debugging. |
| Week 2 | Lecture-I | * **Role of compiler and linker:** * C++ Basic Program Structure |  |
| Lecture-II | * **Introduction to algorithms:** * Program Statements * Directives |  |
| Lecture-III | * Comments * Namespaces |  |
| Practical-I |  | C++ first program for the practice of directives. |
| Practical-II |  | C++ program for the practice of comments & namespaces. |
| Practical-III |  | How to compile and execute. |
| Week 3 | Lecture-I | * **Basic data types and variables:** * Real World Data &Data Types |  |
| Lecture-II | * Variables & Constants, Input/Output Constructs |  |
| Lecture-III | * **Arithmetic, comparison and logical operators:**   Operators, Expression, Precedence & Associativity |  |
| Practical-I |  | **Lab2:** **Variables and Arithmetic Operators** |
| Practical-II |  | Programming Exercises for use of different primitive data types, Variables and constants, Input/Output constructs |
| Practical-III |  | Practicing with Arithmetic operators, Precedence & Associativity. |
| Week 4 | Lecture-I | * **Conditional statements and execution flow for conditional statements:** * Conditional Statements * Conditional Operators for Statements |  |
| Lecture-II |  |  |
| Lecture-III |  |  |
| Practical-I |  | **Lab3: Decision Control Statements** |
| Practical-II |  | Comparison/Relational Operators, Logical Operators |
| Practical-III |  | If, If-else, |
| Week 5 | Lecture-I | * **Conditional statements and execution flow for conditional statements:** * Conditional Statements * Conditional Operators for Statements |  |
| Lecture-II | * If Statements * If-else Statements |  |
| Lecture-III | * Nested if-else Statements * Switch Statements |  |
| Practical-I |  | **Lab4:** Decision Control Structures and Statements |
| Practical-II |  | Writing programs that use if else, nested if-else structure like Comparing digits using if-else structure. |
| Practical-III |  | Writing programs that find out largest digit using nested if structure |
| Week 6 | Lecture-I | * **Repetitive statements and execution flow for repetitive statements:** * Loop * For loop |  |
| Lecture-II | * While Loop * Do while loop |  |
| Lecture-III | * Nested loop |  |
| Practical-I |  | **Lab5:** Practicing different types of loops |
| Practical-II |  | Writing programs which use for, while and do while loops. |
| Practical-III |  | Writing programs which use Switch, break, continue statements |
| Week 7 | Lecture-I | * **Lists and their memory organization:** * Goto statement * Exit & Continue |  |
| Lecture-II | * **Lists and their memory organization:** * Array Fundamentals |  |
| Lecture-III | * Array operations, Searching, Sorting |  |
| Practical-I |  | **Lab6:** Arrays, practicing with loops |
| Practical-II |  | Searching array elements using binary search algorithm.  Sorting array elements using bubble sort algorithm |
| Practical-III |  | Practicing goto and exit |
| Week 8 | Lecture-I | * **Multi-dimensional lists, introduction to modular programming:** * Character Array * Dynamic Array |  |
| Lecture-II | * Multidimensional Array |  |
| Lecture-III | * Introducing Built-in functions for array |  |
| Practical-I |  | **Lab7:** Practicing Multi-dimensional Arrays.  Finding maximum and minimum elements from multi-dimensional array |
| Practical-II |  | Addition of 2D Matrices |
| Practical-III |  | Strings using multi-dimensional character arrays |
| **Midterm Exam** | | | |
| Week 9 | Lecture-I | * Introducing Abstract Data Type * Defining Structures * Declaring Structure Variables * Initializing and Accessing Members of structures | **Formation of Groups and Project Allocation** |
| Lecture-II | * Arrays of Structure * Using Nested Structure * Initializing Nested Structure |  |
| Lecture-III | * Union * Enumeration |  |
| Practical-I |  | **Lab8:** Structures in C++  Writing programs that input data into members of structure and then print data from the members of structure.  Writing programs that copy one structure variable to another variable. |
| Practical-II |  | Writing programs that swap two structure type variables, Print the results before and after swapping.  Writing programs that define structure within a structure. |
| Practical-III |  | Writing program that defines variables of type enum, initialize them, apply arithmetic and comparison operators and examine output |
| Week 10 | Lecture-I | * **Function definition and calling:** * Scope of variable * Introduction to Functions * Declaring, calling and Defining Function * Passing arguments (constant & Variable) to a function |  |
| Lecture-II | * Pass by value & Pass by ref. * Returning value from function |  |
| Lecture-III | * **Stack rolling and unrolling:** * optimize the execution time * remove or reduce iterations |  |
| Practical-I |  | **Lab9:** Functions (Call by value and reference)  Writing programs, which divide the previously implemented codes into functions. |
| Practical-II |  | Writing programs in which arguments (constants, variables and arrays) are passed to functions. |
| Practical-III |  | Writing programs in which arguments (constants, variables and arrays) are passed to functions. |
| Week 11 | Lecture-I | * **String and string operations:** * String header file |  |
| Lecture-II | * String functions |  |
| Lecture-III | * String functions |  |
| Practical-I |  | **Lab10:** Implementation of cstring and its functions  Practical Exercises related to string and their functions |
| Practical-II |  | Practical Exercises related to string and their functions |
| Practical-III |  | Program for the verification of username and password |
| Week 12 | Lecture-I | * Pointers/references: * Computer Memory * Pointer Basics. * Pointer variables and its  initialization * Void type pointer |  |
| Lecture-II | * Pointers and Arrays * Pointers and Strings |  |
| Lecture-III | * Passing & Returning array to a function * Passing & Returning structure to a function * Passing & Returning pointer to a function |  |
| Practical-I |  | **Lab11:** Implementation of Pointers  Writing programs that print the memory address of variables through their pointer variables. |
| Practical-II |  | Writing programs that input data into an array and then print data using pointer notation. |
| Practical-III |  | Writing programs that find out the maximum/minimum value in an array through pointer notation. |
| Week 13 | Lecture-I | * **Static and dynamic memory allocation:** |  |
| Lecture-II | * Function overloading |  |
| Lecture-III | * Recursion, Inline functions |  |
| Practical-I |  | **Lab12:** Static and Dynamic Memory Allocation  Writing program that passes structure as an argument to function.  Writing program that passes structure by reference as an argument to function.  Writing program that return structure from a function. |
| Practical-II |  | Writing program to swap two values by passing pointers to function.  Writing program to find out the length of a string using pointers.  Writing program to copy one string to another string using pointers.  Practicing programs which define overloaded function call them and examine output. |
| Practical-III |  | Calculating integer powers of a variable using recursive approach Calculating factorial of a number using recursive approach Calculating Fibonacci series using recursive approach  Writing programs which define local, global and static variables & their scope. |
| Week 14 | Lecture-I | * **File I/O operations** * Data Files and Streams * Reading from a file |  |
| Lecture-II | * Adding data to the end of file |  |
| Lecture-III | * Removing and Renaming file |  |
| Practical-I |  | **Lab13:** File handling in C++  Writing program which read and write file using ifstream and ofstream. |
| Practical-II |  | Writing program which copies contents from one file to other file using ifstream and ofstream |
| Practical-III |  | Writing programs that use different file operation modes. |
| Week 15 | Lecture-I | * Object Oriented Paradigm: |  |
| Lecture-II | * Introduction to Classes and Objects |  |
| Lecture-III | * Advantages of OOP |  |
| Practical-I |  | * Project Demos |
| Practical-II |  | * Project Demos |
| Practical-III |  | * Project Demos |
| Week 16 | Lecture-I | * Course Revision |  |
| Lecture-II | * Course Revision |  |
| Lecture-III | * Course Revision |  |
| Practical-I |  | * Labs Revision |
| Practical-II |  | * Labs Revision |
| Practical-III |  | * Labs Revision |
| **Final term Exam** | | | |