**COURSE DESCRIPTION FORM: CS-1002: Programming Fundamentals**

**INSTITUTION**  FAST School of Computing, National University of Computer and Emerging Sciences, Islamabad

BS-DS: **Fall 2023**

**PROGRAM TO BE EVALUATED**

**Course Description**

|  |  |
| --- | --- |
| **Course Code** | CS-1002 |
| **Course Title** | Programming Fundamentals |
| **Credit Hours** | 3+1 |
| **Course Instructors** | Mr. Adil Majeed, Mr. Hassan Raza |
| **Grading Policy** | Absolute Grading |
| **Policy about missed assessment items in the course** | Retake of missed assessment items (other than midterm/ final exam) will not be held.  For a missed midterm/ final exam, an exam retake/ pre-take application and necessary evidence must be submitted to the department secretary The examination assessment and retake committee decides the exam retake/ pre-take cases. |
| **Course Plagiarism Policy** | Plagiarism in project or midterm/ final exam may result in an F grade in the course.  Plagiarism in an assignment will result in zero marks in the whole assignments category. |
| **Prerequisites by Course(s) or Topics** | N/A |
| **Assessment Instruments with Weights**  (Homework, quizzes, sessional exams, final exam, assignments, etc.) | Assessment items of Theory Part   |  |  |  | | --- | --- | --- | | **Assessment Item** | **Number** | **Weight (%)** | | Assignments | >= 4 | 10 | | Sessional-I | 1 | 10 | | Sessional-II | 1 | 15 | | Quizzes | >= 4 | 5 | | Project | 1 | 15 | | Final Exam | 1 | 45 |   Assessment items of Lab Part   |  |  |  | | --- | --- | --- | | **Assessment Item** | **Number** | **Weight (%)** | | Lab Tasks | 15 | 25 | | Assignments | >= 4 | 5 | | Sessional | 1 | 15 | | Lab Quiz | 1 | 10 | | Project | 1 | 10 | | Final Exam | 1 | 35 | |
| **Course Coordinator** | Mr. Adil Majeed |
| **Lab Instructors**  **(if any)** | Ms. Mubbrra Asma |
| **URL (if any)** |  |
| **Current Catalog Description** | The course aims to equip students with the basic computing concepts and to provide them the ability to analyze the given requirements for solving problems in different domains while implementing the solutions on a computer system. It emphasizes on developing an algorithm and applying the basic programming constructs like control structures, arrays, functions, pointers, dynamic memory allocation, etc. for its development. The students will learn the syntax of the C++ programming language for the implementation. |
| **Textbook** (or **Laboratory Manual** for Laboratory Courses) | Tony Gaddis "STARTING OUT WITH C++" 9th Edition |
| **Reference Material** | Paul Deitel, Harvey Deitel "C++ How to Program" 10th Edition  Walter Savitch "Problem Solving with C++" 10th Edition |
| **Course Goals** | |  |  |  | | --- | --- | --- | | **A. Course Learning Outcomes (CLOs)** | | | | After completion of the course, the students shall be able to:   1. Understand basic problem-solving steps and logic constructs. 2. Apply basic programming concepts. 3. Design and implement algorithms to solve real-world problems. | | | | **B. Program Learning Outcomes (PLOs)** | | | | **PLO 1** | Computing and Data Science Knowledge | Apply knowledge of mathematics, statistics, natural sciences, computing fundamentals, and a data specialization to the solution of complex data science problems. | | **PLO 2** | Problem Analysis | Identify, formulate, research literature, and analyze complex data problems, reaching substantiated conclusions using first principles of mathematics, statistics, natural sciences, computing, and data sciences. | | **PLO 3** | Design/Develop Solutions | Design solutions for complex data science problems and design systems, components, and processes that meet specified needs with appropriate consideration for public health and safety, cultural, societal, and environmental considerations. | | **PLO 4** | Investigation & Experimentation | Conduct investigation of complex data science problems using research-based knowledge and research-based methods. | | **PLO 5** | Modern Tool Usage | Create, select, and apply appropriate techniques, resources, and modern data science tools, including prediction and modeling for complex data science problems. | | **PLO 6** | Society Responsibility | Apply to reason informed by the contextual knowledge to assess societal, health, safety, legal, and cultural issues relevant to the context of complex data science problems. | | **PLO 7** | Environment and Sustainability | Understand and evaluate the sustainability and impact of data professional work in solving complex data science problems. | | **PLO 8** | Ethics | Apply ethical principles and commit to professional ethics and responsibilities and norms of computing practice. | | **PLO 9** | Individual and Team Work | Function effectively as an individual, and as a member or leader in diverse teams and in multi-disciplinary settings. | | **PLO 10** | Communication | Communicate effectively on complex data science activities with the data professionals’ community and with society at large. | | **PLO 11** | Project Management and Finance | Demonstrate knowledge and understanding of management principles and economic decision-making and apply these to one's work as a member or team member. | | **PLO 12** | Life Long Learning | Recognize the need for, and have the preparation and ability to engage in independent and lifelong learning in the broadest context of technological changes. | |  | | | | **C. Mapping of CLOs to PLOs**  (CLO: Course Learning Outcome, PLOs: Program Learning Outcomes) | | | | |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | |  | | **PLOs** | | | | | | | | | | | | | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | | **CLOs** | 1 | ✓ | ✓ |  |  |  |  |  |  |  |  |  |  | | 2 |  | ✓ | ✓ |  |  |  |  |  |  |  |  |  | | 3 |  |  | ✓ |  |  |  |  |  | ✓ |  |  |  | | | | |  | | | |  | | | |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Topics Covered in the Course**  (Assume 15-week instruction and 3 contact hours per week) | |  |  |  | | --- | --- | --- | | List of Topics | No. of Weeks | Contact Hours | | - Problem-solving, Basic flowchart, block diagram, and programming languages.  - Primitive data types, input/output (hello world).  - Signed and unsigned data types, constants, and variables. | **1** | **3** | | - Arithmetic operators (+, -, \*, /, % and their compound counterparts) with their associativity and precedence.  - Bitwise operators | **2** | **6** | | - Conditional/selection structures.  - Comparison and logical operators.  - if, if. . .else and if else if structure.  - Switch statement, *break* statement.  - Ternary operator. | **2** | **6** | | **1st Sessional Exam** | | | | - Repetition structures.  - Pre/post increment/decrement operators.  - while loop (sentinels + condition).  - Loop with *for.*  - Loop with *do-while.*  - Nesting of *while*, *for* loop, and *continue* statement. | **3** | **9** | | - Function prototypes, definition, and calling. | **1** | **3** | | - Introduction to Arrays.  - Array initialization and representation.  - Char arrays. | **1.33** | **4** | | **2nd Sessional Exam** | | | | - Multi-Dimensional Arrays (MDA).  MDA representation in memory.  - parameters passing by value and by reference (passing arrays).  - Function calling order | **1.66** | **5** | | - Pointers.  - const. vs. non-const. pointers, a pointer to const. data vs. a pointer to non-constant data.  - Using pointers.  - Dynamic memory allocation.  - Array of pointers. | **2** | **6** | | - Header files (creating own file).  - Files handling  - Opening flags (app mode). | **1** | **3** | | **Total** | **15** | **45** | | | | |
| **Laboratory Projects/Experiments Done in the Course** | Yes, there are lab tasks with every lab of three hours. | | | |
| **Programming Assignments Done in the Course** | Yes, there are four programming assignments and a project. | | | |
| **Class Time Spent**  (In percentage) | **Theory** | **Problem Analysis** | **Solution Design** | **Social and Ethical Issues** |
| 40 | 40 | 15 | 5 |
| **Oral and Written Communications** | Every student is required to submit at least \_\_1\_\_\_ written reports of typically \_\_\_5\_\_ pages and to make \_\_1\_\_\_ demonstration of typically \_\_\_\_10\_\_\_ minutes duration. | | | |

**Lab/ Practical Component of the course**

|  |  |  |
| --- | --- | --- |
| **Weeks** | **Contents/Topics** | **Assessment Items  (Case Study/ Exercise Assignment/ Quiz etc.)** |
| ***Week- 01*** | Linux, Linux shell, and compiler installation |  |
| ***Week-02*** | Introduction to pseudo-code, algorithms, and flow chart with scratch | Assignment-1 (Problem Solving) |
| ***Week-03*** | Basic program writing in C++ and stream insertion/extraction  operators | Assignment-2 (Operators) |
| ***Week-04*** | Control structures-I (If else) |  |
| ***Week-05*** | Control structures-II (Switch cases) | Assignment-3 (Control Structures) |
| ***Week-06*** | Repetitions–I (while loop) | **1st Sessional Exam** |
| ***Week-07*** | Repetitions-II (do-while and for loops) |  |
| ***Week-08*** | Repetitions-III (nested loops) | Assignment-4 (Repetitions) |
| ***Week-09*** | Open Lab |  |
| ***Week-10*** | Functions (definition, calling, forward declaration, and parameter  passing by value/reference) | Assignment-6 (Functions) |
| ***Week-11*** | Arrays-I (1D arrays) | **2nd Sessional Exam** |
| ***Week-12*** | Arrays-II (char and multi-dimensional arrays) | Assignment-5 (Arrays) |
| ***Week 13*** | Introduction to pointers and dynamic memory allocation (for 1D) |  |
| ***Week-14*** | Dynamic memory allocation (2D, and 3D) | Project |
| ***Week-15*** | Project-based lab |  |
| ***Week-16*** | Basic File Handling |  |

**Practical/ Programming Work/ Tools:** Ubuntu, g++, Scratch