**A. Course Handout (Student & Faculty)**

|  |  |  |  |
| --- | --- | --- | --- |
| **Institute/School Name** | Chitkara University Institute of Engineering and Technology | | |
| **Department Name** | Department of Computer Applications | | |
| **Programme Name** | Bachelor of Computer Applications | | |
| **Course Name** | Programming Paradigms | **Session** | July-Dec 2024 |
| **Course Code** | 23CA001 | **Semester/Batch** | 3rd /2023 |
| **L-T-P (Per Week)** | 3-0-4 | **Course Credits** | 5 |
| **Course Coordinator** | Mr. Vikas Rattan | | |

1. **Any pre-requisites for the course:**

## Basic Knowledge of problem-solving skills using any of the procedural language particularly ‘C++‘.

1. **Scope and objective of the course**:

## Object-oriented programming is an important paradigm shift that is followed in almost all modern computer languages. Object Oriented Programming is based upon theories from the field of cognitive science about how information is represented in the human mind. The main goal of this course is to make students understand the importance of object-oriented programming features enabling students to write generalized code expressing an algorithm or data structure in a way that may be used in a variety of real-world situations.

**The main objectives of the course are:** The students should be able to

## To make students learn the implementation of generic programming using templates.

## To enable the students to acknowledge the advantages of standard Template Libraries (STL).

## To understand storing and manipulating objects using STL, which makes the program reusable and robust.

## To strengthen the ability of the students to solve problems computationally by utilizing an understanding of algorithm analysis and data structures.

## To enable the learners to segregate and classify the available data structures into various classes depending upon storage and their access-associated characteristics.

1. **Course Learning Outcomes:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Course Learning Outcome** | **POs** | **CL** | **KC** | **Sessions** |
| **CLO01** | To implement efficient and excellent code with the data structures and algorithms in the standard template library. | PO1, PO2 | K2 | Conceptual | 7 |
| **CLO02** | To analyze different techniques for solving problems like sorting and searching. | PO1 | K4 | Factual,  Conceptual | 6 |
| **CLO03** | Understand the importance of data structures in the context of writing efficient programs. | PO1, PO2 | K3 | Fundamental,  Conceptual | 5 |
| **CLO04** | Plan the optimum solution, keeping in mind programming language ethics, among various solutions using various data structures. | PO1, PO2 PO5, PO6, PO9 | K4 | Conceptual Procedural | 5 |
| **Total Contact Hours** | |  |  |  | **48** |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **CLO** | **PO** | **PO1** | **PO2** | **PO3** | **PO4** | **PO5** | **PO6** | **PO7** | **PO8** | **PO9** | **PO10** | **PO11** | **PO12** | **PO13** | **PO14** |
| **CLO1** | | M | L |  |  |  |  |  |  |  |  |  |  |  |  |
| **CLO2** | | M |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **CLO3** | | M | M |  |  |  |  |  |  |  |  |  |  |  |  |
| **CLO4** | | M | H |  |  | L | M |  |  | M |  |  |  |  |  |
| L: Low, M: Medium and H: High | | | | | | |  | |  |  |  | |  |  | |

1. **ERISE Grid Mapping**

|  |  |
| --- | --- |
| **Feature Enablement** | **Level(1-5, 5 being highest)** |
| Entrepreneurship | 1 |
| Research | 2 |
| Innovation | 2 |
| Skills | 2 |
| Employability | 3 |

1. **Recommended Books (Reference Books/Text Books):** 
   * **B1:** The Complete Reference C++ by Herbert Schildt, Mcgraw Hill Education, Fourth Edition, 2008.
   * **B2:** Object Oriented Programming with C++ by E Balaguruwamy, Mcgraw Hill Education, Seventh Edition, 2018.
   * **B3:** Data Structures through C++ by Yashwant Kanetkar, BPB Publications, Fourth Edition, 2022
   * **B4:** Data Structures, Seymour Lipschutz, Nineteenth Reprint, Tata McGraw Hill, 2010
2. **Other readings & relevant websites:**

|  |  |
| --- | --- |
| **S. No** | **Link of journals, Magazines, websites, and Research Papers** |
| Link 1 | https://www.simplilearn.com/tutorials/cpp-tutorial/cpp-standard-template-library |
| Link 2 | https://www.geeksforgeeks.org/cpp-stl-tutorial/ |
| Link 3 | https://byjus.com/gate/introduction-to-data-structure-notes/ |
| Link 4 | https://www.hackerearth.com/practice/data-structures |

1. **Recommended Tools and Platforms:**

CodeChef, HackerRank and Leetcode

1. **Complete Theory course coverage plan:**

|  |  |  |
| --- | --- | --- |
| **Session No.** | **Topic(s)** | **Recommended**  **Books (B)** |
| 1 | **Unit-1: Templates/Generic Programming**  Introduction to generic functions and generic classes | B2 |
| 2 | Function with two generic types, explicitly overloading a generic function, using standard parameters with template functions generic function restrictions. | B1 |
| 3 | A class with one and two generic data types, default arguments with template classes | B1 |
| 4 | **Unit-2: Standard Template Library (STL)**  Overview of STL: Containers, Algorithms, Iterators | B1 and B2 |
| 5 | Container Classes: vector, stack, queue, deque, list, map | B1 and B2 |
| 6 | **Unit-3: Introduction to data structures**  Classification of data structures, Various operations associated with each data structure, Algorithm characteristics, Introduction to time-space complexity of algorithms | B4 |
| 7 | Array Data Structure: Memory Representation of one- and two-dimensional array, Address calculation of element, | B4 |
| 8 | Bubble Sort, Selection Sort, Insertion Sort | B4 |
| 9 | Linear Search, Binary Search | B4 |
| 10 | **Unit 4: Stack and Queues**  Application of Stacks, Operations on stacks, | B3 |
| 11 | Polish Notations, Conversion of Polish Notation to infix and vice versa, | B4 |
| 12 | Evaluation of Polish expressions, Recursion | B1 |
| 13 | Types of Queues, Applications, and Operations on Simple and Circular Queues | B3 |
| 14 | **Unit 5: Link List and Trees**  Applications of link list, classification of link list, Memory Representation, traversing, | B3 |
| 15 | Insertion and deletion operations on the Singly list | B3 |
| 16 | Implementation of doubly list | B3 |
| 17 | Implementation of Circular linked list | B4 |
| 18 | Stack and Queue implementation using a linked list | B4 |
| 19 | Introduction to tree terminology and Memory representations of binary tree | B4 |
| 20 | Tree Traversal Algorithms | B4 |
| 21 | Tree Construction for tree traversals | B4 |
| 22 | Insertion and deletion in binary search tree | B4 |
| 23 | Merge sort and Quick sort | B4 |
|  | **End Term** |  |

1. **Delivery/Instructional Resources:**

|  |  |  |  |
| --- | --- | --- | --- |
| **S. No.** | **Topic(s)** | **Web References** | **Audio/Video** |
| 1 | Generic Programming | http://users.cis.fiu.edu/~weiss/Deltoid/vcstl/templates | <https://archive.nptel.ac.in/courses/106/105/106105234/> |
| 2 | STL | https://www.geeksforgeeks.org/the-c-standard-template-library-stl/ | <https://archive.nptel.ac.in/courses/106/105/106105234/> |
| 3 | Introduction to data structures | https://archive.nptel.ac.in/courses/106/102/106102064/ | <https://nptel.ac.in/courses/106102064> |
| 4 | Stack and Queues | https://betterprogramming.pub/stacks-and-queues-7c322b5f4e35 | <https://www.youtube.com/watch?v=mB5HXBb_HY8> |
| 5 | Linked List | https://www.programiz.com/dsa/linked-list | <https://www.youtube.com/watch?v=QN9hnmAgmOc> |
| 6 | Trees | https://www.scholarhat.com/tutorial/datastructures/trees-in-data-structures | <https://www.youtube.com/watch?v=G_VDV9fYGsU> |
| 7 | Merge sort and quick sort | https://www.geeksforgeeks.org/cpp-program-for-merge-sort/ | https://www.youtube.com/watch?v=tWCaFVJMUi8 |

1. **Action plan for different types of learners:**

|  |  |  |
| --- | --- | --- |
| **Slow Learners** | **Average Learners** | **Fast Learners** |
| * Remedial Classes on Saturdays * Encouragement for improvement using Peer Tutoring * Use of Audio and Visual Materials * Use of Real-Life Examples | * Formative Exercises used to highlight concepts * E-notes and E-exercises to read ahead of the pedagogic material. * Doubt session on the Saturdays and according the slot free in time table. | * Design complex problem based on real life problem. * Presentation on topics beyond those covered in CHO. * Engaging students to hold hands of slow learners by creating a Peer Tutoring Group. |

1. **Evaluation Scheme & Components:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Evaluation**  **Component** | **Type of Component** | **No. of Assessments** | **Weightage of Component** | **Mode of Assessment** |
| #Component 1 | Lab Evaluations and Internal Viva | 03\* | 20% | Offline |
| Component 2 | Sessional Tests (STs) | 02\*\* | 30% | Offline |
| #Component 3 | End Term Examination | 01\*\*\* | 50% | Offline |
|  | Total |  | 100% |  |

#Component 1 and component 3 are mandatory components and require 40% marks in each for clearing the subject.

\*Lab Evaluation have 2 Lab performances- LP1 and LP2 and one Internal viva which shall be considered for evaluation based on experiments.

\*\*Out of 02 STs, the ERP system automatically picks the best 01 ST.

\*\*\*Further, as per Academic Guidelines, minimum 75% attendance is required to become eligible for appearing in the End Semester Examination.

1. **Complete Lab Course Coverage plan**

|  |  |  |
| --- | --- | --- |
| **Session No.** | **Lab Session**  (all the programs must be written using classes and objects etc. i.e. OOPS paradigms) | **Virtual lab link/ Multisim link** |
| 1 | Write down a program to sort an array using function templates. | https://www.geeksforgeeks.org/templates-cpp/ |
| 2 | Given an [array](https://www.geeksforgeeks.org/array-data-structure/) **V[]** of size **R**, the task to [reverse the array](https://www.geeksforgeeks.org/write-a-program-to-reverse-an-array-or-string/) using [Stack](https://www.geeksforgeeks.org/stack-data-structure/).  **Input:** arr[] = { 10, 20, 30, 40, 50 }  **Output:** 50 40 30 20 10  **Explanation:** Reversing the array modifies arr[] to { 50, 40, 30, 20, 10 } Therefore, the required output is 50 40 30 20 10.  There are 2 approaches to solve this problem use both the approaches (stack / STL ) | https://www.geeksforgeeks.org/reverse-an-array-using-stack/ |
| 3 | Given an expression (a+(b-c)) \*({d-e}/ [f+g-h]) , write a program to examine whether the pairs and the orders of “{“, “}”, “(“, “)”, “[“, “]” are correct in the given expression. | https://www.geeksforgeeks.org/check-for-balanced-parentheses-in-an-expression/ |
| 4 | Implement a simple queue using an array and perform the following operations.  Enqueue(): For adding elements in the queue  Dequeue(): For Deleting elements in queue  isEmpty(): returns 1 if the queue is empty else 0.  isFull(): returns 1 if the queue is full else 0.  printContent(): For printing elements of the queue |  |
| 5 | Implementation of statement at Sr. No 4 via Standard Template libraries. |  |
| 6 | Implementation of the circular queue of any data types. | https://www.programiz.com/dsa/circular-queue |
| 7 | Write down a program to create and traverse a link list of n nodes. | https://www.javatpoint.com/program-to-create-a-singly-linked-list-of-n-nodes-and-count-the-number-of-nodes |
| 8 | Given a linked list of length N and an integer K, append the last K elements of a linked list to the front. Note that K can be greater than N.  **Input Format**  First line contains a single integer N denoting the size of the linked list. Second line contains N space separated integers denoting the elements of the linked list. Third line contains a single integer K denoting the number of elements that are to be appended.  **Constraints**  1 <= N <= 10^4  1 <= K <= 10^4  **Output Format**  Display all the elements in the modified linked list.  Sample Input  7  1 2 2 1 8 5 6  3  **Sample Output**  8 5 6 1 2 2 1 |  |
| 9 | Given a linked list with n nodes. Find the kth element from last without computing the length of the linked list.  **Input Format**  First line contains space separated integers representing the node values of the linked list. The list ends when the input comes as '-1'. The next line contains a single integer k.  **Constraints**  n < 10^5  Output Format Output a single line containing the node value at the kth element from last.  **Example**:  **Sample Input**  1 2 3 4 5 6 -1  3  **Sample Output**  4  Note: The linked list is 1 2 3 4 5 6. -1 is not included in the list. So the third element from the last is 4 |  |
| 10 | Insert nodes from the second list into the first list at alternate positions in the first list, given two linked lists.  For instance, if the first list is 15->17->117->113->111 and the second list is 22->20->22->24->26, the first list should be replaced with 15->22->17->20->117->22->113->24->111->26 and the second list should be left empty. The nodes of the second list should be inserted only when there are available positions. | https://www.prepbytes.com/blog/linked-list/merge-a-linked-list-into-another-linked-list-at-alternate-positions/ |
| 11 | Write down a program to calculate the factorial of a number greater than 20. | https://www.programiz.com/c-programming/examples/factorial |
| 12 | Implement a binary tree and write its traversals. | https://www.programiz.com/dsa/tree-traversal |

1. **Syllabus of the Course:**

|  |  |
| --- | --- |
| **Name of the course:** Programming Paradigms | **Subject Code:**23CA001 |

|  |  |  |
| --- | --- | --- |
| **Contents** | **No. of Sessions** | **Weightage** |
| **Unit-1: Templates/Generic Programming**  Introduction to generic functions and generic classes, Function with two generic types, explicitly overloading a generic function, using standard parameters with template functions generic function restrictions | 03 | 10% |
| **Unit-2: Standard Template Library (STL)**  Overview of STL: Containers, Algorithms, Iterators, Container Classes: vector, stack, queue, deque, list, map | 02 | 15% |
| **Unit 3: Introduction to data structures**  Classification of data structures, Various operations associated with each data structure, Algorithm characteristics, Introduction to time-space complexity of algorithms, Array Data Structure: Memory Representation of one- and two-dimensional array, Address calculation of elements, Linear Search, Binary Search | 04 | 20% |
| **Unit 4: Stack and Queues**  Application of Stacks, Operations on stacks, Polish Notations, Conversion of Polish Notation to infix and vice versa, Evaluation of Polish expressions, Recursion, Types of Queues, Applications, and Operations on Simple and Circular Queues | 04 | 25% |
| **Unit 5: Link List and Trees**  Applications of link list, classification of link list, Memory Representation, traversing, Insertion and deletion operations on the Singly list, Implementation of doubly list, Implementation of Circular linked list, Stack and Queue implementation using a linked list, Introduction to tree terminology and Memory representations of binary tree, Tree Traversal Algorithms, Tree Construction from tree traversals, Insertion and deletion in binary search tree, Merge sort and Quick sort | 10 | 30% |

1. **Academic Honesty policy:**

Chitkara University ensures the implementation of the highest level of academic integrity in all the documents being prepared / adopted by its Faculty members and students.

Any branch of the same will be tantamount to severe academic penalties.

**This Document is approved by:**

|  |  |  |
| --- | --- | --- |
| **Designation** | **Name** | **Signature** |
| **Course Coordinator** | Mr. Vikas Rattan |  |
| **Program Incharge** | Dr. Preetinder Singh Brar |  |
| **Pro Vice Chancellor** | Dr. Jaiteg Singh |  |
| **Date (DD/MM/YYYY)** | 03/07/2024 |  |