Course content

This course focuses on object-oriented programming (OOP) principles and techniques using C++. It provides students with a solid foundation for object-oriented system development and prepares them for future coursework and professional application development. Emphasis is placed on hands-on programming practice to develop problem-solving and system development skills.

The following topics are covered in the course:

1. Introduction to Object-Oriented Programming: Understanding the principles and methodologies of OOP, including encapsulation, inheritance, and polymorphism.
2. C++ Programming Basics: Learning the syntax, semantics, and core features of C++ as an object-oriented programming language.
3. Class Design and Implementation: Creating and managing classes, objects, and data abstraction in C++.
4. Object-Oriented Problem Solving: Applying OOP principles to analyze programming requirements and develop solutions.
5. Advanced OOP Concepts: Exploring advanced features of C++, such as operator overloading, templates, and exception handling.
6. Debugging and Optimization: Techniques for identifying key algorithms, debugging, and optimizing object-oriented programs.
7. Practical Applications: Building object-oriented systems and solving complex problems through project-based learning.

Course objectives

Knowledge

1. Understand the fundamental concepts and principles of object-oriented programming.
2. Learn the syntax and features of C++ for object-oriented development.
3. Grasp advanced OOP techniques such as operator overloading and exception handling.

Skills

1. Analyze programming requirements and propose solutions using object-oriented methodologies.
2. Design, implement, and debug object-oriented programs using C++.
3. Apply OOP principles to identify critical algorithms and code components for solving programming problems.
4. Optimize and refine C++ programs to improve performance and maintainability.

Competencies

1. Develop proficiency in using C++ for object-oriented programming and system development.
2. Translate problem requirements into well-structured and efficient C++ programs.
3. Build a solid foundation for future coursework and professional application development in object-oriented programming.
4. Solve complex engineering problems using object-oriented design and programming techniques.