D. Conditional and iterative control structures
E. Functions and parameter passing
F. Structured decomposition

Learning Outcomes

A. Analyze and explain the behavior of simple programs involving the fundamental programming constructs covered by this unit;
B. Modify and expand short programs that use standard conditional and iterative control structures and functions;
C. Design, implement, test, and debug a program that uses each of the following fundamental programming constructs: basic computation, simple I/O, standard conditional and iterative structures, and the definition of functions;
D. Choose appropriate conditional and iteration constructs for a given programming task;
E. Apply the techniques of structured (functional) decomposition to break a program into smaller pieces; and
F. Describe the mechanics of parameter passing.

PF2. Algorithms and problem-solving
Minimum coverage time: 6 hours

Topics

A. Problem-solving strategies
B. The role of algorithms in the problem-solving process
C. Implementation strategies for algorithms
D. Debugging strategies
E. The concept and properties of algorithms

Learning outcomes

A. Discuss the importance of algorithms in the problem-solving process;
B. Identify the necessary properties of good algorithms;
C. Create algorithms for solving simple problems;
D. Use pseudocode or a programming language to implement, test, and debug algorithms for solving simple problems; and
E. Describe strategies that are useful in debugging.

II. Programming Languages (PL)
PL1. Overview of programming languages
Minimum coverage time: 2 hours

Topics

A. History of programming languages
B. Brief survey of programming paradigms
C. Procedural languages
D. Object-oriented languages

Learning outcomes

A. Summarize the evolution of programming languages illustrating how this history has led to the paradigms available today; and
B. Identify at least one distinguishing characteristic for each of the programming paradigms covered in this unit.

PL4. Declarations and types
Minimum coverage time: 3 hours

Topics

A. The conception of types as a set of values together with a set of operations Declaration models (binding, visibility, scope, and lifetime)
B. Overview of type-checking

Learning outcomes

A. Explain the value of declaration models, especially with respect to programming-in-the-large;
B. Identify and describe the properties of a variable such as its associated address, value, scope, persistence, and size;
C. Discuss type incompatibility;
D. Demonstrate different forms of binding, visibility, scoping, and lifetime management; and
E. Defend the importance of types and type-checking in providing abstraction and safety.

Topics fulfilling these tasks and outcomes could include OOPS and other programming elements. This course is recommended to contain hands-on programming and problem solving tasks.

4. Methods of Instruction:

Discussion: