surface removal; Interactive Mesh Displays; Parallel-projection matrices; Perspective-projection matrices; Projections and Shadows.

UNIT – 7   **6 Hours**
Lighting and Shading: Light and Matter; Light Sources; The Phong Lighting model; Computation of vectors; Polygonal Shading; Approximation of a sphere by recursive subdivisions; Light sources in OpenGL; Specification of materials in OpenGL; Shading of the sphere model; Global Illumination.

UNIT – 8   **8 Hours**
Implementation: Basic Implementation Strategies; Four major tasks; Clipping; Line-segment clipping; Polygon clipping; Clipping of other primitives; Clipping in three dimensions; Rasterization; Bresenham's algorithm; Polygon Rasterization; Hidden-surface removal; Antialiasing; Display considerations.

Text Books:
1. Edward Angel: Interactive Computer Graphics A Top-Down Approach with OpenGL, 5th Edition, Pearson Education, 2008. (Chapters 1 to 7)

Reference Books:
1. Donald Hearn and Pauline Baker: Computer Graphics- OpenGL Version, 3rd Edition, Pearson Education, 2004.
2. F.S. Hill Jr.: Computer Graphics Using OpenGL, 3rd Edition, PHI, 2009.
3. James D Foley, Andries Van Dam, Steven K Feiner, John F Hughes, Computer Graphics, Pearson Education 1997.

PROGRAMMING LANGUAGES

Subject Code: 10IS666	**I.A. Marks : 25**
Hours/Week : 04	**Exam Hours: 03**
Total Hours : 52	**Exam Marks: 100**

PART - A

UNIT – 1   **7 Hours**
Introduction; Names, Scopes, and Bindings: The art of language design; Programming language spectrum; Why study programming languages? Compilation and interpretation; Programming environments.
Names, scope, and bindings: The notion of binding time; Object lifetime and storage management; Scope rules; Implementing scope; The meaning of names within a scope; The binding of referencing environments; Macro expansion.

UNIT – 2   **7 Hours**
Control Flow: Expression evaluation; Structured and unstructured flow; Sequencing; Selection; Iteration; Recursion; Non-determinacy

UNIT – 3   **6 Hours**
Data Types: Type systems; Type checking; Records and variants; Arrays; Strings; Sets; Pointers and recursive types; Lists; Files and Input/Output; Equality testing and assignment.

UNIT – 4   **6 Hours**
Subroutines and Control Abstraction: Review of stack layout; Calling sequences; Parameter passing; Generic subroutines and modules; Exception handling; Coroutines; Events.

PART – B

UNIT – 5   **6 Hours**
Data Abstraction and Object Orientation: Object oriented programming; Encapsulation and Inheritance; Initialization and finalization; Dynamic method binding; Multiple inheritance; Object oriented programming revisited.

UNIT – 6   **7 Hours**
Functional Languages, and Logic Languages: Functional Languages: Origins; Concepts; A review/overview of scheme; Evaluation order revisited; Higher-order functions; Functional programming in perspective. Logic Languages: Concepts; Prolog; Logic programming in perspective.

UNIT – 7   **6 Hours**
Concurrency: Background and motivation; Concurrency programming fundamentals; Implementing synchronization; Language-level mechanisms; Message passing.

UNIT – 8   **7 Hours**
Run-Time Program Management: Virtual machines; Late binding of machine code; Inspection/introspection.
Text Books:
1. Michael L. Scott: Programming Language Pragmatics, 3rd Edition, Elsevier, 2009.
   (Chapters 1.1 to 1.5, 3.1 to 3.7, 6 excluding the sections on CD, 7 excluding the ML type system, 8, 9, 10 excluding the sections on CD, 11 excluding the sections on CD, 12, 15. Note: Text Boxes titled Design & Implementation are excluded)

Reference Books:
1. Ravi Sethi: Programming languages Concepts and Constructs, 2nd Edition, Pearson Education, 1996.
2. R Sebesta: Concepts of Programming Languages, 8th Edition, Pearson Education, 2008.
3. Allen Tucker, Robert Nonan: Programming Languages, Principles and Paradigms, 2nd Edition, Tata McGraw-Hill, 2007.

FILE STRUCTURES LABORATORY

Subject Code: 10ISL67	**I.A. Marks : 25**
Hours/Week : 03	**Exam Hours: 03**
Total Hours : 42	**Exam Marks: 50**

PART - A

Design, develop, and implement the following programs

1. Write a C++ program to read series of names, one per line, from standard input and write these names spelled in reverse order to the standard output using I/O redirection and pipes. Repeat the exercise using an input file specified by the user instead of the standard input and using an output file specified by the user instead of the standard output.

2. Write a C++ program to read and write student objects with fixed-length records and the fields delimited by "|". Implement pack ( ), unpack ( ), modify ( ) and search ( ) methods.

3. Write a C++ program to read and write student objects with Variable - Length records using any suitable record structure. Implement pack ( ), unpack ( ), modify ( ) and search ( ) methods.

4. Write a C++ program to write student objects with Variable - Length records using any suitable record structure and to read from this file a student record using RRN.

5. Write a C++ program to implement simple index on primary key for a file of student objects. Implement add ( ), search ( ), delete ( ) using the index.

6. Write a C++ program to implement index on secondary key, the name, for a file of student objects. Implement add ( ), search ( ), delete ( ) using the secondary index.