Principles of Programming Language

[BE SE-6th Semester]

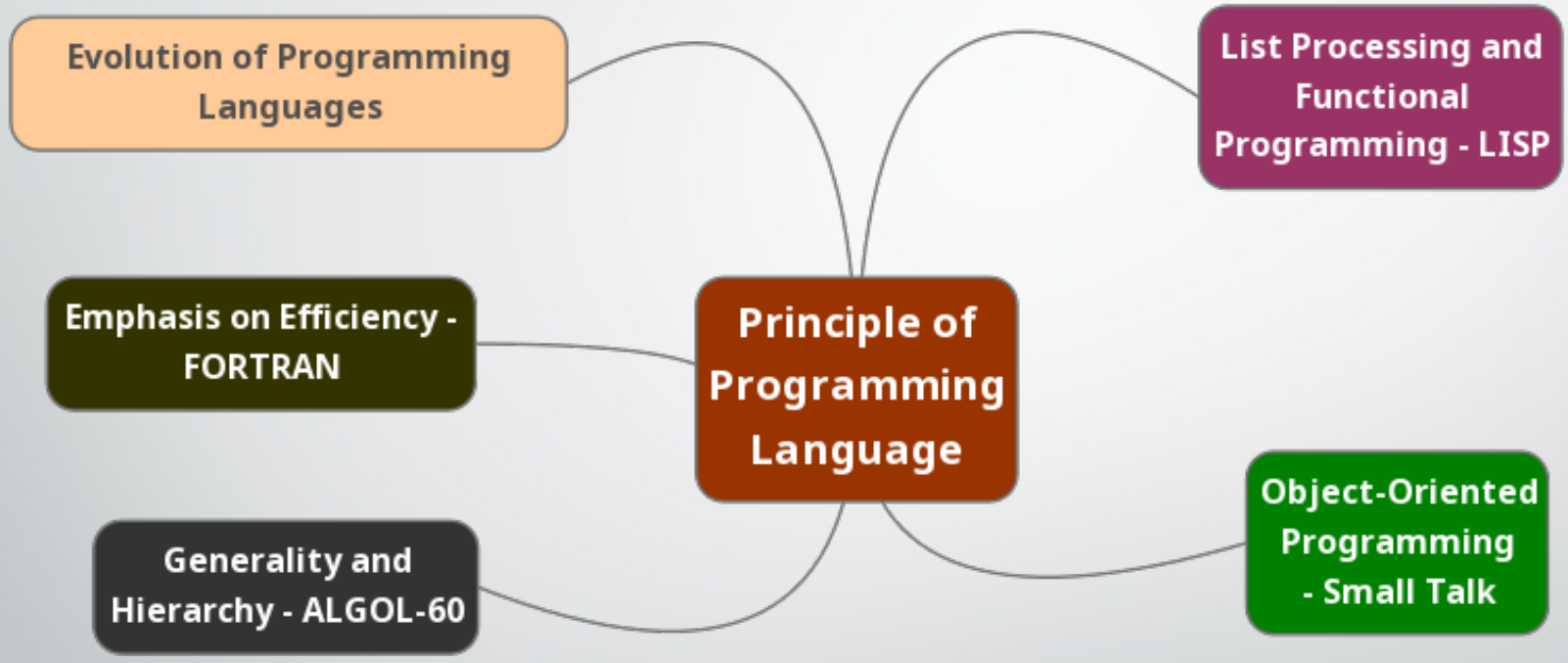
Rishi K. Marseni

Textbook:

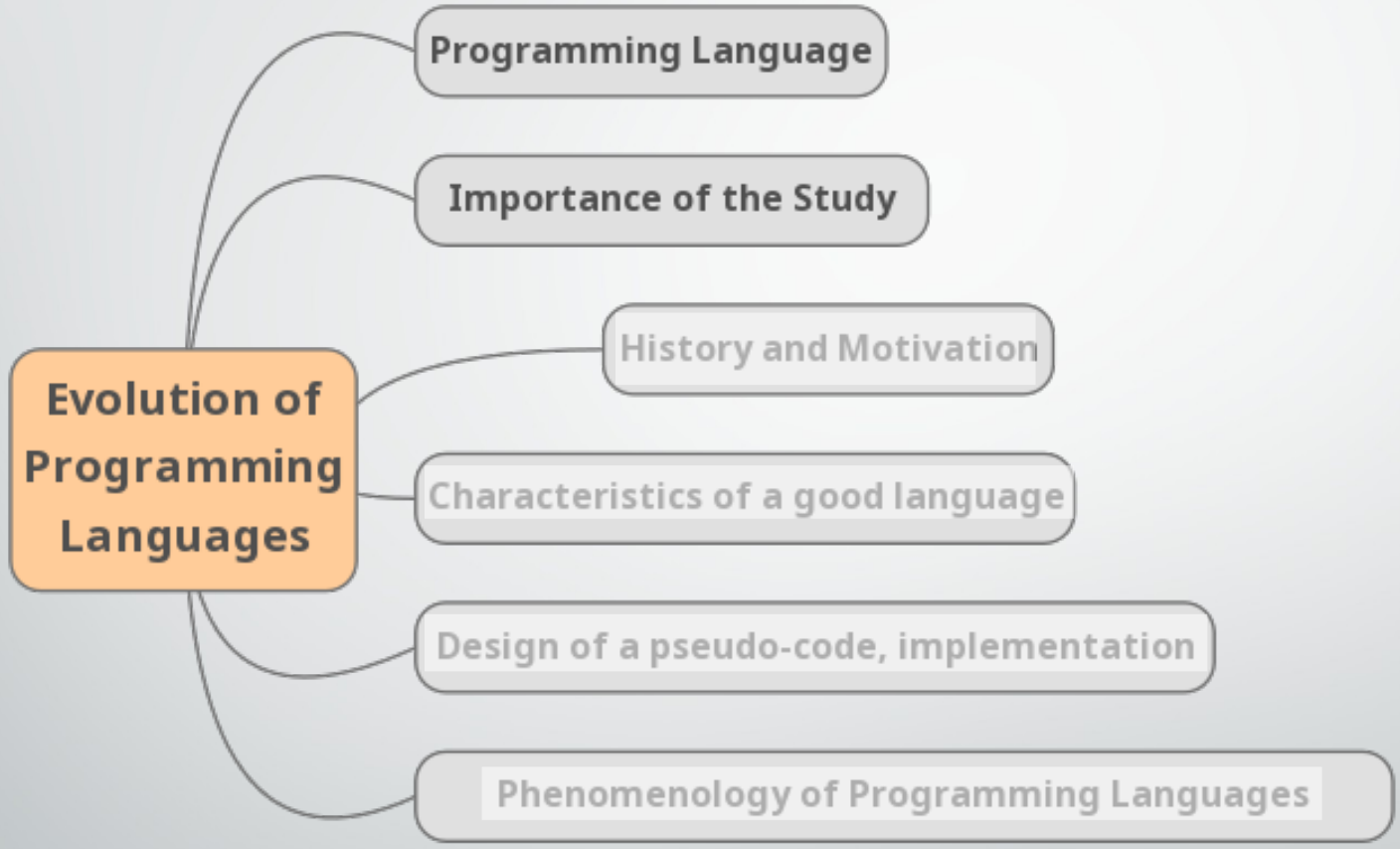
Principles of programming languages: design, evaluation, and implementation.

Author: Bruce J. MacLennan

Principle of Programming Language



Unit 1: Evolution of Programming Language



Programming Language

- Programming languages are notations, used for specifying, organizing, and reasoning about computations
- Syntactic rules, keywords, naming structures, data structures, expression and control structures
- A language that is intended for the expression of computer programs and that is capable of expressing any computer program

1) Machine Language - 0’s & 1’s that represent high & low voltage
2) Assembly Language - uses symbolic operation code
3) High Level Language - uses English like statements

Programming Paradigms

1) **Imperative Programming**

   A program is a sequence of state-changing actions

2) **Object-Oriented Programming**

   A program is the interactions between abstract objects

3) **Logic Programming**

   A program is the formal logical specification of the problem

4) **Functional Programming**

   A program is the composition of operations on data

Imperative Programming

- A program is a sequence of state-changing actions
- Uses variables, arithmetic-logic operators and control flow statements
- Based on the Von Neumann architecture of computers
- The design of the imperative languages:

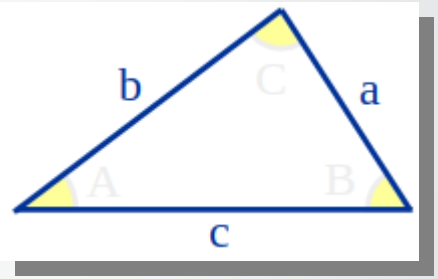
1) States — representing memory cells with changing values
2) Sequential orders — reflecting the single sequential CPU
3) Assignment statements — reflecting piping

FORTRAN(1954) | Cobol (1959) | Pascal (1970) | C (1971) | Ada (1979)

Imperative Programming: FORTRAN Code

```
1   PROGRAM Triangle
2   IMPLICIT NONE
3   REAL :: a, b, c, Area
4   PRINT *, 'Welcome, please enter the&
5   |       |   &lengths of the 3 sides.'
6   READ *, a, b, c
7   PRINT *, 'Triangle''s area:  ', Area(a,b,c)
8   END PROGRAM Triangle
9   FUNCTION Area(x,y,z)
10  IMPLICIT NONE
11  REAL :: Area              ! function type
12  REAL, INTENT( IN ) :: x, y, z
13  REAL :: theta, height
14  theta = ACOS((x**2+y**2-z**2)/(2.0*x*y))
15  height = x*SIN(theta); Area = 0.5*y*height
16  END FUNCTION Area
```

http://www.mrao.cam.ac.uk/~pa/f90Notes/HTMLNotesnode40.html



$$\cos(\text{theta}) = \frac{x^2 + y^2 - z^2}{2xy}$$

Object-Oriented Programming

- A program is the interactions between abstract objects
- Object has static and dynamic properties as data and methods
- Interaction using message passing or method invocation

  Abstraction | Encapsulation | Inheritance | Dynamic binding or Polymorphism

- Simulation the real-world events using active objects
- Most used paradigm by software industries

  Smalltalk (1969) | C++ (1983) | Java (1995)

Object-Oriented Programming: C++ Code

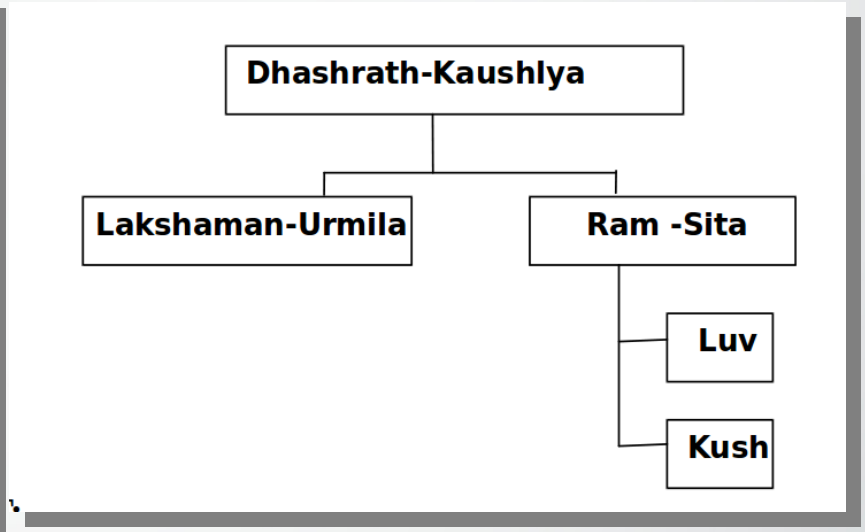
```cpp
#include <iostream>
using namespace std;
class Room {
    public:
        double length;
        double breadth;
        double height;
        double calculateArea(){
            return length * breadth;
        }

        double calculateVolume(){
            return length * breadth * height;
        }
};
int main() {
    Room room1;
    room1.length = 18.5;
    room1.breadth = 20.0;
    room1.height = 10.5;
    cout << "Area of Room =  " << room1.calculateArea() << endl;
    cout << "Volume of Room =  " << room1.calculateVolume() << endl;
    return 0;
}
```

Logic Programming

- A program is the formal logical specification of the problem
- About what properties the solution must have rather than how to find
- Use the concept of axioms, rules and knowledge-base to reach the goal state
- The problem description is used by an inference engine to find a solution
- Based on FOPL - First Order Predicate Logic
- Prolog (1970), and Godel (1994)

Logic Programming: Prolog Code

```
1   male(dhasharath).
2   male(ram).
3   male(lakshaman).
4   male(luv).
5   male(kush).
6   female(kaushlya).
7   female(urmila).
8   female(sita).
9   father(dashrath, ram).
10  father(dashrath, lakshaman).
11  father(ram, luv).
12  father(ram, kush).
13  father(lakshaman, hari).
14  wife(kaushyla, dashrath).
15  wife(sita, ram).
16  wife(urmila, lakshaman).
17  sibling(ram, lakshaman).
18  sibling(luv, kush).
19
20  grandfather( A,C):-father( A, B ), father( B, C ).
21  grandmother( A,C):-wife( A, B ), grandfather( B, C ).
22  uncle(A,C):- (sibling(A,B);sibling(B,A)), father(B,C).
23  aunty(P, R):- wife(P,Q), uncle(Q,R).
```




```
?- female(ram).
     False
?- father(X,ram).
     X=dashrath
?- grandfather(Who, luv).
     Who=dashrath
```

Functional Programming

- A program is the composition of operations on data
- Based on the theory of mathematical functions, lambda-calculus
- Closer to the nature of the problem rather than the machine
- Recursion rather than iteration
- Components [data objects, built-in functions and functional forms]

LISP (1958) | ML (1973)| Scheme (1975) | Miranda (1982) | Haskell (1987)

```
1   (defun add-numbers (num1 num2)
2       (+ num1 num2))
3
4
5   (add-numbers 5 6)
```

Importance of the study(1)

- Familiarization with the fundamental concepts of computer science
- Theoretical backgrounds of various programming languages and their pragmatic structure
- Develop proficiency in an engineering problem solving and design methodology
- Understand the importance of advanced information technologies
- Use computers and application software as tools to solve problems.
- Analyze, design, build and test operational solutions

Importance of the study(2)

- Increased ability to express ideas
- Improved background for choosing appropriate languages
- Provides greater ability to learn new languages
- Understand significance of implementation
- Ability to design new languages

Principle Vs Practice

Principle

- Fundamental truth a that serves as the foundation for a system
- Focus on optimization and efficiency
- Suggest an ideal solution

Practice

- Actual application or use of the concept
- Focus on ease of use and programmer productivity
- Provide a workable solution

Unit 1: Evolution of Programming Language