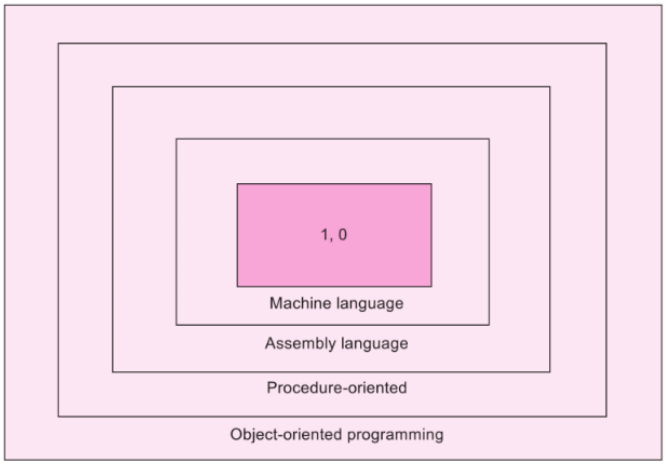
Introduction:

Programmers write instructions in various programming languages to perform their computation tasks such as:

(i) Machine-level Language
(ii) Assembly-level Language
(iii) High-level Language



Machine-level Language: Machine code or machine language is a set of instructions executed directly by a computer's central processing unit (CPU). Each instruction performs a specific task, such as a load, a jump, or an ALU operation on a data unit in a CPU register or memory. Every program directly executed by a CPU is made up of a series of such instructions.

Assembly level Language: An assembly language (or assembler language) is a low-level programming language for a computer, or other programmable devices, in which there is a very strong (generally one-to-one) correspondence between the language and the architecture's machine code instructions. Assembly language is converted into executable machine code by a utility program referred to as an assembler; the conversion process is referred to as assembly, or assembling the code.

High-level Language: High-level language is any programming language that enables the development of a program in a much simpler programming context and is generally independent of the computer's hardware architecture. High-level language has a higher level of abstraction from the computer and focuses more on the programming logic rather than the underlying hardware components such as memory addressing and register utilization.

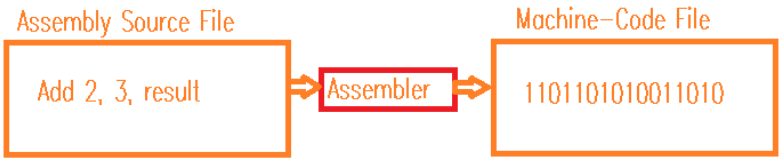
The first high-level programming languages were designed in the 1950s. Now there are dozens of different languages, including Ada, Algol, BASIC, COBOL, C, C++, JAVA, FORTRAN, LISP, Pascal, and Prolog. Such languages are considered high-level because they are closer to human languages and farther from machine languages. In contrast, assembly languages are considered low-level because they are very close to machine languages.

For example, adding two numbers together in machine language would look like this:

```
1101101010011010
```

If you want to add the numbers 2 and 3 in assembly language, it would look like this:

```
add 2, 3, result
```



High-Level Language statement that calculates the area of a circle with a radius of 5:

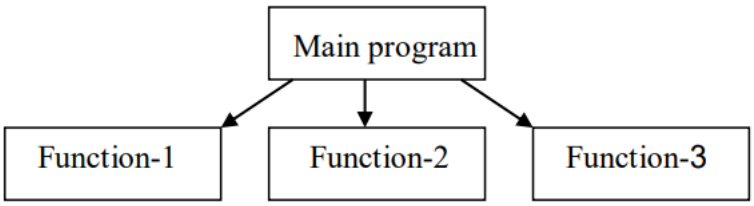
```
area = 5 * 5 * 3.14159;
```

The high-level programming languages are broadly categorized into two categories:

(i) Procedure-oriented programming (POP) language.
(ii) Object-oriented programming (OOP) language.

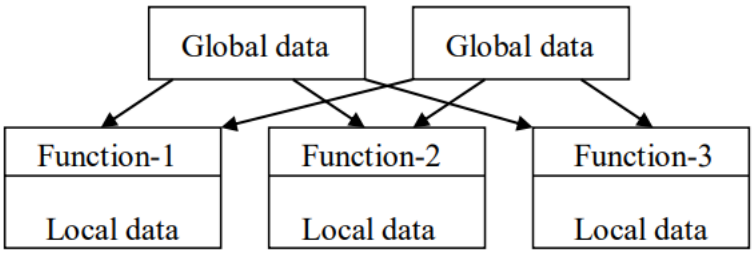
Procedure Oriented Programming Language

In the procedure-oriented approach, the problem is viewed as a sequence of things to be done such as reading, calculation and printing. Procedure-oriented programming basically consists of writing a list of instructions or actions for the computer to follow and organizing these instructions into functions.



The disadvantage of procedure-oriented programming languages is:

1. Global data access
2. It does not model real word problems very well
3. No data hiding

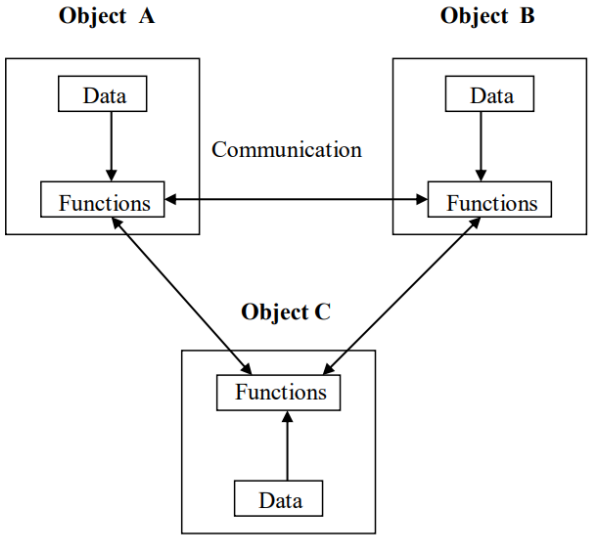


Characteristics of procedure-oriented programming:

1. Emphasis is on doing things(algorithm).
2. Large programs are divided into smaller programs known as functions.
3. Most of the functions share global data.
4. Data move openly around the system from function to function.
5. Function transforms data from one form to another.
6. Employs a top-down approach in program design.

Object Oriented Programming

“Object-oriented programming is an approach that provides a way of modularizing programs by creating partitioned memory area for both data and functions that can be used as templates for creating copies of such modules on demand”.



Features of the Object-Oriented Programming

1. Emphasis is on doing rather than procedure.
2. Programs are divided into what are known as objects.
3. Data structures are designed such that they characterize the objects.
4. Functions that operate on the data of an object are tied together in the data structure.
5. Data is hidden and can’t be accessed by external functions.
6. Objects may communicate with each other through functions.
7. New data and functions can be easily added.
8. Follows bottom-up approach in program design.

Benefits of OOP

OOP offers several benefits to both the program designer and the user. Object-oriented contributes to the solution of many problems associated with the development and quality of software products. The principal advantages are:

1. Through inheritance we can eliminate redundant code and extend the use of existing classes.

2. We can build programs from the standard working modules that communicate with one another, rather than having to start writing the code from scratch. This leads to saving of development time and higher productivity.

3. This data-hiding principle helps the programmer build secure programs that can’t be invaded by code in other parts of the program.

4. It is possible to have multiple instances of an object co-exist without any interference.

5. It is easy to partition the work in a project based on objects.

6. Object-oriented systems can be easily upgraded from small to large systems.

7. Message-passing techniques for communication between objects makes the interface description with external systems much simpler.

8. Software complexity can be easily managed.

Procedure Oriented Programming

- Conventional programming language using high level language such as Cobol

and C, is commonly known as Procedure Oriented Programming (POP).

- In Procedure Oriented Programming, the problem is viewed as a sequenceof a thing to be done.
- The primary focus is on functions.
- Procedure Oriented Programming basically consists of writing a list of instructions for the computer to follow, and organizing these functions into groups known as functions.
- In a multi-function program, many important data items are placed as global so that they may be accessed by all the functions.
- Each function may have its own local data.

Some characteristics of Procedure Oriented Programming are:

1. Large programs are divided into smaller programs known as functions.
2. Most of the functions share global data.
3. Data move openly around the system from function to function.
4. Functions transform data from one form to another.
5. Employs top-down approach in program design.

Need of Object Oriented Programming

- Developments in software technology continue to be dynamic. New tools and techniques are announced in quick succession.
- This has forced the software engineers and industry to continuously look for new approaches to software design and development, and they are becomingmore and more critical in view of the increasing complexity of software system as well as the highly competitive nature of the industry.
- This rapid advance appears to have created a situation of crises within the industry.

The following issues needs to be solved:

- To represent real life entities of problems in system design.
- To design system with open interface.
- To ensure reusability and extensibility of modules.
- To develop modules that is tolerant to any changes in future
- To improve software productivity and decrease cost.
- To improve the quality of the software.
- To manage time schedule.
- To industrialize the software development process.

To overcome these problems and major motivation factor in the invention of object oriented approach is to remove some of the flows encountered in the procedure oriented approach.

Object Oriented Programming

Object Oriented Programming is the most recent concept among programming model. The motivating factor in the invention of object oriented approach is to remove some of the flows encountered in the procedural approach. OOPS treats data as a critical element in the program development and does not allow it to flow freely around the system. It binds data more closely to the functions that operate on it, and protects it from accidental modification from outside functions.

Some of the striking features of Object Oriented Programming are:

- Importance on data rather than procedure.
- Programs are divided into what are known as objects.

Data structures are designed as such that they characterize the objects.

Introduction to OOPs

- Functions that operate on the data of an object are tied together in the datastructure.
- Data is hidden and cannot be accessed by external functions.
- Objects may communicate with each other through functions.
- New data and functions can be easily added whenever necessary.
- Follow bottom-up approach in program design.

Comparison of Procedural and Object Oriented Approach

There are two different approaches to write a program, i.e., Procedure Oriented Programming and Object Oriented Programming. Basic aim of these methods is nothing but to make programming efficient. We can write the program using any of the way but there are notable differences between both approaches.

Sr. No	Procedure Oriented Programming	Object Oriented Programming
	POP is divided into small parts called as functions	In OOP, program is divided into parts called objects
	In POP, Importance is not given to data but to functions as well as sequence of actions to be done.	In OOP, Importance is given to the data rather than procedures or functions because it works as a real world.
	POP follows top-down approach.	OOP follows bottom-up approach.
	POP does not have any access specifier.	OOP has access specifiers named Public, Private, Protected, etc.
	In POP, data can move freely from function to function in the system.	In OOP, objects can move and communicate with each other through member functions.
	To add new data and function in POP is not so easy.	OOP provides an easy way to add new data and function.
	In POP, most function uses global data for sharing that can be accessed freely from function to function in the system.	In OOP, data cannot move easily from function to function, it can be kept public or private so we can control the access of data.
	POP does not have any proper way for hiding data so it is less secure.	OOP provides data hiding so provides more security.
	In POP, overloading is not possible.	In OOP, overloading is possible in the form of Function Overloading and Operator Overloading.
	Example of POP are: C, VB, FOR-TRAN, Pascal.	Example of OOP are: C++, JAVA, VB.NET, C#.NET.

Benefits of OOP

1) As OOP is closer to the real world phenomena, hence, it is easier to map realworld problems onto a solution in OOP.
2) The objects in OOP have the state and behaviour that is similar to the realworld objects.
3) It is more suitable for large projects.
4) The projects executed using OOP techniques are more reliable.
5) It provides the bases for increased testability (automated testing) and

hencehigher quality.

6) Abstraction techniques are used to hide the unnecessary details and focus isonly on the relevant part of the problem and solution.
7) Encapsulation helps in concentrating the structure as well as the behaviour ofvarious objects in OOP in a single enclosure.
8) The enclosure is also used to hide the information and to allow strictly controlledaccess to the structure as well as the behaviour of the objects.
9) OOP divides the problems into collection of objects to provide services forsolving a particular problem.
10) Object oriented systems are easier to upgrade/modify.
11) The concepts like inheritance and polymorphism provide the extensibility ofthe OOP languages.
12) The concepts of OOP also enhance the reusability of the code written.
13) Software complexity can be better managed.
14) The use of the concept of message passing for communication among the objectsmakes the interface description with external system much simpler.
15) The maintainability of the programs or the software is increased manifold. If designed correctly, any tier of the application can be replaced by another provided the replaced tier implements the correct interface(s). The applicationwill still work properly.

1.1 Advantages of OOPs

1. Code reusability in terms of inheritance.
2. Object-oriented system can be easily upgraded from one platform to another.
3. Complex projects can be easily divided into small code functions.
4. The principle of abstraction and encapsulation enables a programmer to buildsecure programs.
5. Software complexity decreases.
6. Principle of data hiding helps programmer to design and develop safe programs.
7. Rapid development of software can be done is short span of time.
8. More than one instance of same class can exist together without any interference.

1.2 Object Oriented Languages

Some of the most popular Object-oriented Programming languages are :

a)C++

b)Ruby

c)Java.

d)Delphi

e) smalltalk

f) Charm++

g)Eiffle.

h)Simula.

1.3 Applications of OOPS

Object concept helps to translate our thoughts to a program. It provides a way of solving a problem in the same way as a human being perceives a real world problem and finds out the solution. It is possible to construct large reusable components using object-oriented techniques. Development of reusable components is rapidly growing in commercial software industries. If there is complexity in software development, object-oriented programming is the best

paradigm to solve the problem. The following areas make the use of OOP:

1. Image Processing
2. Pattern Recognition
3. Computer Assisted Concurrent Engineering
4. Computer Aided Design and Manufacturing
5. Computer Aided Teaching
6. Intelligent Systems
7. Database Management Systems
8. Web-based Applications
9. Distributed Computing and Applications
10. Component-based Applications
11. Business Process Re-engineering
12. Enterprise Resource Planning

13. Data security and management
14. Mobile Computing
15. Data Warehouse and Data Mining
16. Parallel Computing