GANPAT UNIVERSITY

FACULTY OF ENGINEERING & TECHNOLOGY

Object Oriented Programming
(2CEIT302)

Prof.Y. J. Prajapati
IT Dept. UVPCE

Subject Overview

Subject Name - Object Oriented Programming

- Subject code - 2CEIT302

Teaching scheme					
(Per week)	Lecture(DT)		Practical(Lab.)		Total
	L	TU	P	TW	
Credit	3	0	2	-	5
Hours	3	0	4	-	7

Examination scheme (Marks)			
	CE	SEE	Total
Theory	40	60	100
Practical	30	20	50

Why Do We Need Object-Oriented Programming?

- Object-Oriented Programming was developed because limitations were discovered in earlier approaches to Programming

- To appreciate what OOP does, we need to understand what these limitations are and how they arose from traditional programming languages.

History

- James Gosling, **Mike Sheridan**, and **Patrick Naughton** initiated the Java language project in June 1991. The small team of sun engineers called **Green Team**.
- Initially designed for small, embedded systems in electronic appliances like set-top boxes.
- Firstly, it was called **"Greentalk"** by James Gosling, and the file extension was .gt.
- After that, it was called **Oak** and was developed as a part of the Green project.

Why Java named "Oak"?

- **Why Oak?** Oak is a symbol of strength and chosen as a national tree of many countries like the U.S.A., France, Germany, Romania, etc.
- In 1995, Oak was renamed as **"Java"** because it was already a trademark by Oak Technologies.

History

- Why Java Programming named "Java"?
- **Why had they chosen java name for Java language?** The team gathered to choose a new name. The suggested words were "dynamic", "revolutionary", "Silk", "jolt", "DNA", etc.
- Java is an island of Indonesia where the first coffee was produced (called java coffee). Java name was chosen by James Gosling while having coffee near his office.
- Initially developed by James Gosling at Sun Microsystems (which is now a subsidiary of Oracle Corporation) and released in 1995.
- Initially developed by James Gosling at Sun Microsystems (which is now a subsidiary of Oracle Corporation) and released in 1995.

History

- JDK 1.0 (January 23, 1996)
- JDK 1.1 (February 19, 1997)
- J2SE 1.2 (December 8, 1998)
- J2SE 1.3 (May 8, 2000)
- J2SE 1.4 (February 6, 2002)
- J2SE 5.0 (September 30, 2004)
- Java SE 6 (December 11, 2006)
- Java SE 7 (July 28, 2011)
- Java SE 8 (March 18, 2014)
- Java SE 9 (September 21, 2017)
- Java SE 10 (March, 20, 2018)
- Java SE 11 (September 2018)
- Java SE 12 (March 2019)
- Java SE 13 (September 2019)
- **Java SE 14 (March 2020)**

Types of Java Applications

1) Standalone Application

Standalone applications are also known as desktop applications or window-based applications. These are traditional software that we need to install on every machine. Examples of standalone application are Media player, antivirus, etc. AWT and Swing are used in Java for creating standalone applications.

2) Web Application

An application that runs on the server side and creates a dynamic page is called a web application. Currently, Servlet, JSP, Struts, Spring, Hibernate, JSF, etc. technologies are used for creating web applications in Java.

3) Enterprise Application

An application that is distributed in nature, such as banking applications, etc. is called enterprise application. It has advantages of the high-level security, load balancing, and clustering.

4) Mobile Application

An application which is created for mobile devices is called a mobile application. Currently, Android and Java ME are used for creating mobile applications.

Application

- Desktop Applications such as acrobat reader, media player, antivirus, etc.
- Web Applications such as irctc.co.in, javatpoint.com, etc.
- Enterprise Applications such as banking applications.
- Mobile
- Embedded System
- Smart Card
- Robotics
- Games, etc.

Java Platforms / Editions

1) Java SE (Java Standard Edition)

   It is a Java programming platform. It includes Java programming APIs such as java.lang, java.io, java.net, java.util, java.sql, java.math etc. It includes core topics like OOPs, String, Regex, Exception, Inner classes, Multithreading, I/O Stream, Networking, AWT, Swing, Reflection, Collection, etc.

2) Java EE (Java Enterprise Edition)

   It is an enterprise platform which is mainly used to develop web and enterprise applications. It is built on the top of the Java SE platform. It includes topics like Servlet, JSP, Web Services, etc.

3) Java ME (Java Micro Edition)

   It is a micro platform which is mainly used to develop mobile applications.

4) JavaFX

   it is used to develop rich internet applications. It uses a light-weight user interface API.

Procedure-Oriented Programming

- C, Pascal, FORTRAN, and similar languages are procedural languages.
- Each statement in the language tells the computer to do something: – get some input
  - – add these numbers
  - – divide by 6
  - – display that output
- A program in a procedural language is a list of **instructions.**

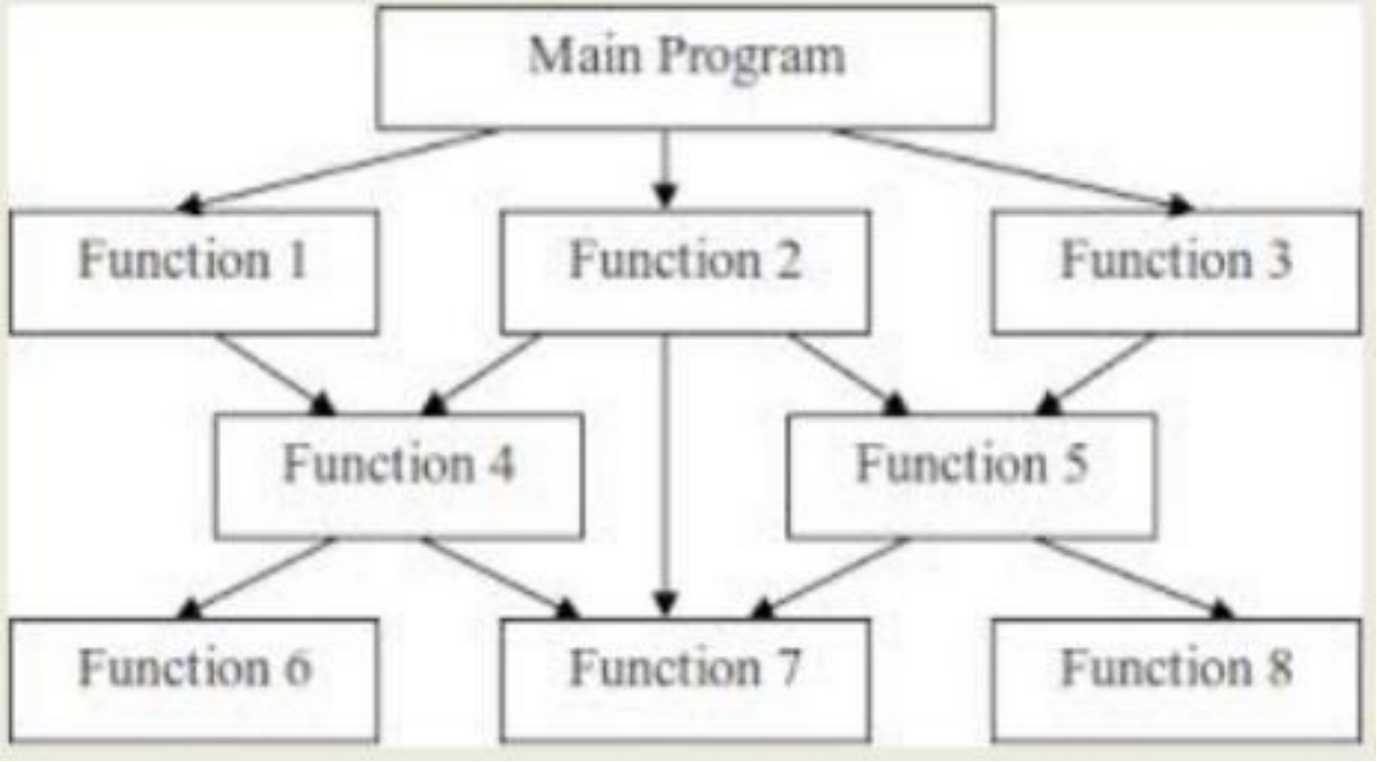
Procedure-Oriented Programming

- Procedural program is divided into functions:

- Each function has a clearly defined purpose and a clearly defined interface to the other functions in the program.
- The idea of breaking a program into functions can be further extended by grouping a number of functions together into a larger entity called a module.

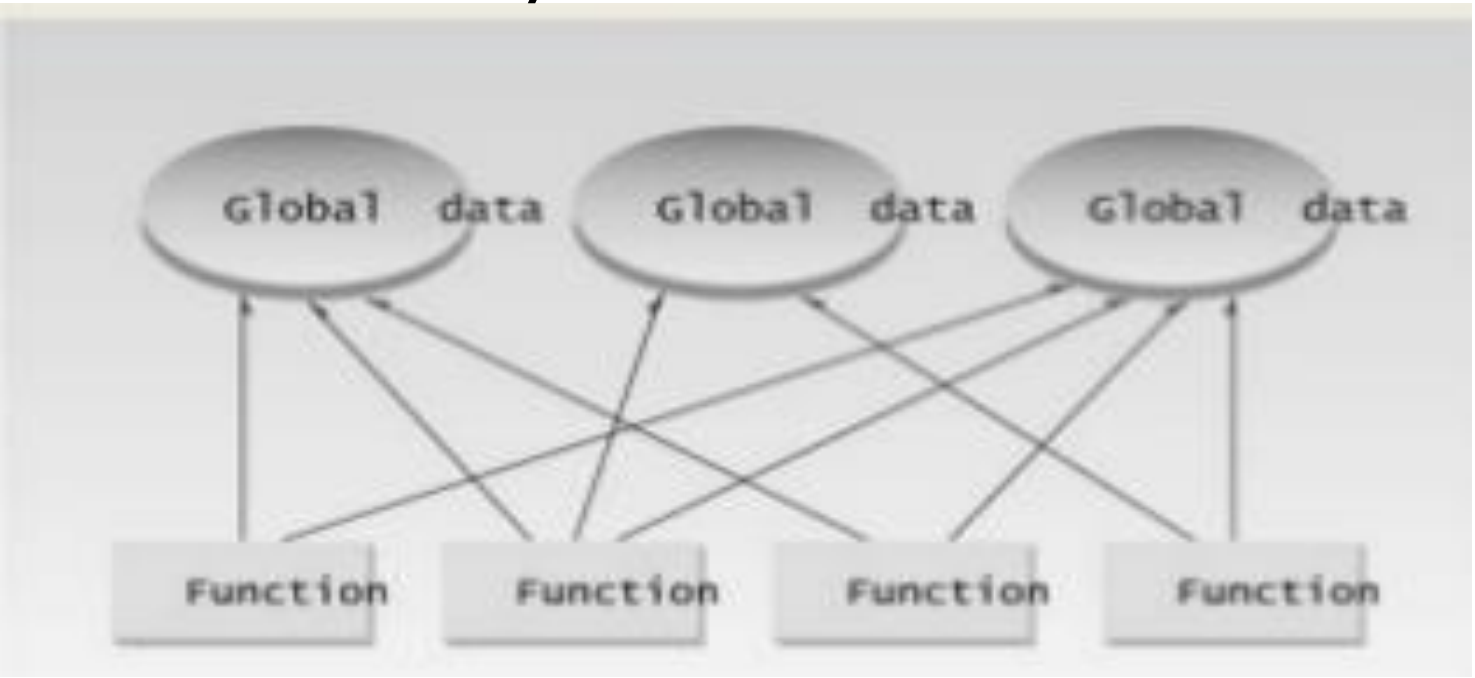
Procedure-Oriented Programming

- Procedure-Oriented Programming Division into Functions:



Procedure-Oriented Programming

- In Multi-function program important data items are placed as global so that they may be accessed by all functions
- Each function may have its own local data



Procedure-Oriented Programming

Drawbacks

- Since every function has complete access to the global variables, the new programmer can corrupt the data accidentally by creating function
- We can access the data of one function from other since, there is no protection
- In large program it is very difficult to identify what data is used by which function
- Similarly, if new data is to be added, all the function needed to be modified to access the data.
- Does not model real world problem very well

How is Java different from C

- **C Language:**
  - Major difference is that C is a structure oriented language and Java is an object oriented language and has mechanism to define classes and objects.
  - Java does not support an explicit pointer type
  - Java does not have preprocessor, so we cant use #define and #include statements.
  - Java does not include structures, unions and enum data types.
  - Java does not include keywords like goto, sizeof and typedef.
  - Java adds labeled break and continue statements.
  - Java adds many features required for object oriented programming.

How is Java different from C++

- C++ language

  Features removed in java:

  - Java doesn't support pointers to avoid unauthorized access of memory locations.
  - Java does not include structures, unions and enum data types.
  - Java does not support operator over loading.
  - Preprocessor plays less important role in C++ and so eliminated entirely in java.
  - Java does not perform automatic type conversions that result in loss of precision.

How is Java different from C++

- Java does not support global variables. Every method and variable is declared within a class and forms part of that class.
- Java does not support inheritance of multiple super classes by a sub class (i.e., multiple inheritance). This is accomplished by using 'interface' concept.
- In java objects are passed by reference only. In C++ objects may be passed by value or reference

Object-Oriented Programming

- OOP was introduced to overcome flaws in the procedural approach to programming
- Such as lack of reusability & maintainability
- Fundamental idea behind object-oriented languages is to combine into a single unit both data and the functions that operate on that data.
- Such a unit is called an object

Object-Oriented Programming

- In OOP, problem is divided into the number of entities called object and then build data and functions around these objects.
- It ties the data more closely to the functions that operate on it, and protects it from accidental modification from the outside functions
- Data of an object can be accessed only by the functions associated with that object
- Communication of the objects done through function

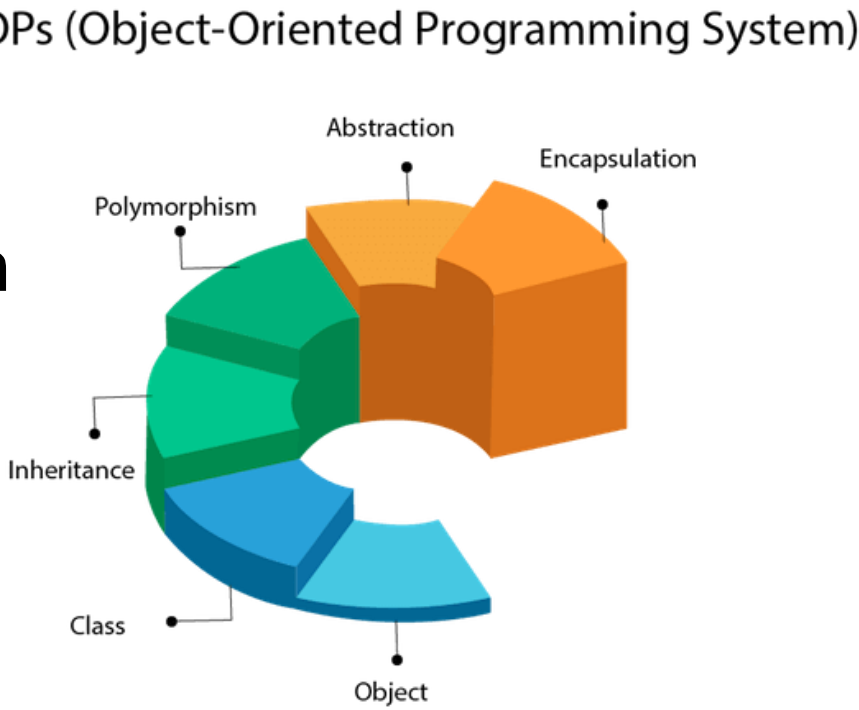
Object-Oriented Programming

Characteristics

- Emphasis on data rather than procedure
- Programs are divided into entities known as objects
- Data Structures are designed such that they characterize objects
- Functions that operate on data of an object are tied together in data structures
- Data is hidden and cannot be accessed by external functions
- Objects communicate with each other through functions
- New data and functions can be easily added whenever necessary

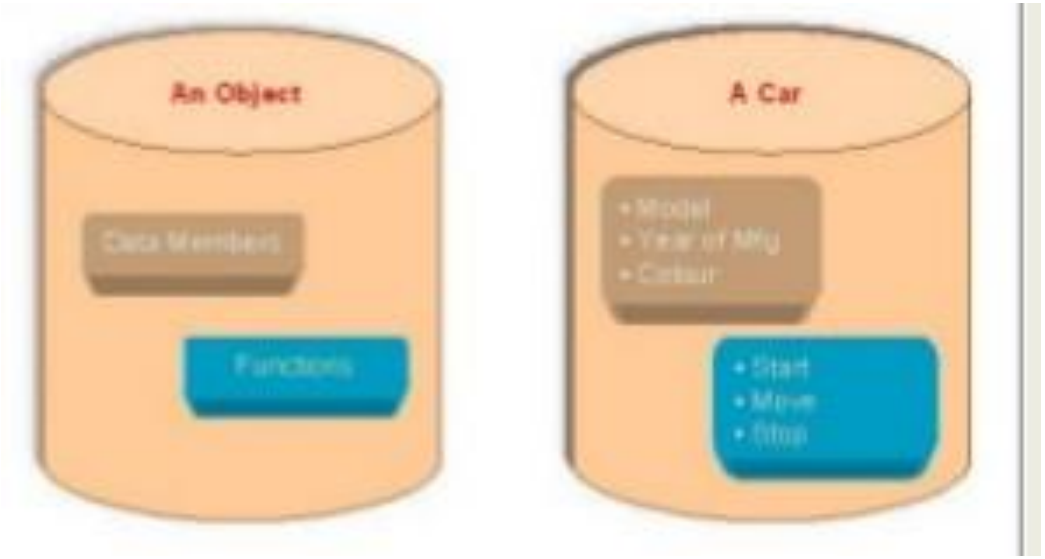
Basic Concepts of oops

1. Objects
2. Classes
3. Data Abstraction
4. Data Encapsulation
5. Inheritance
6. Polymorphism
7. Dynamic binding
8. Message Passing



Objects

- Objects are the basic run-time entities of an object oriented system
- They may represent a person, a place or any item that the program must handle
- Example:-



Objects

- When a program is executed, the objects interact by sending messages to one another.
- For e.g. if "customer" and "account" are two objects in a program, then the customer object may send a message to the account object requesting for the bank balance.
- Each object contains data, and code to manipulate the data

Classes

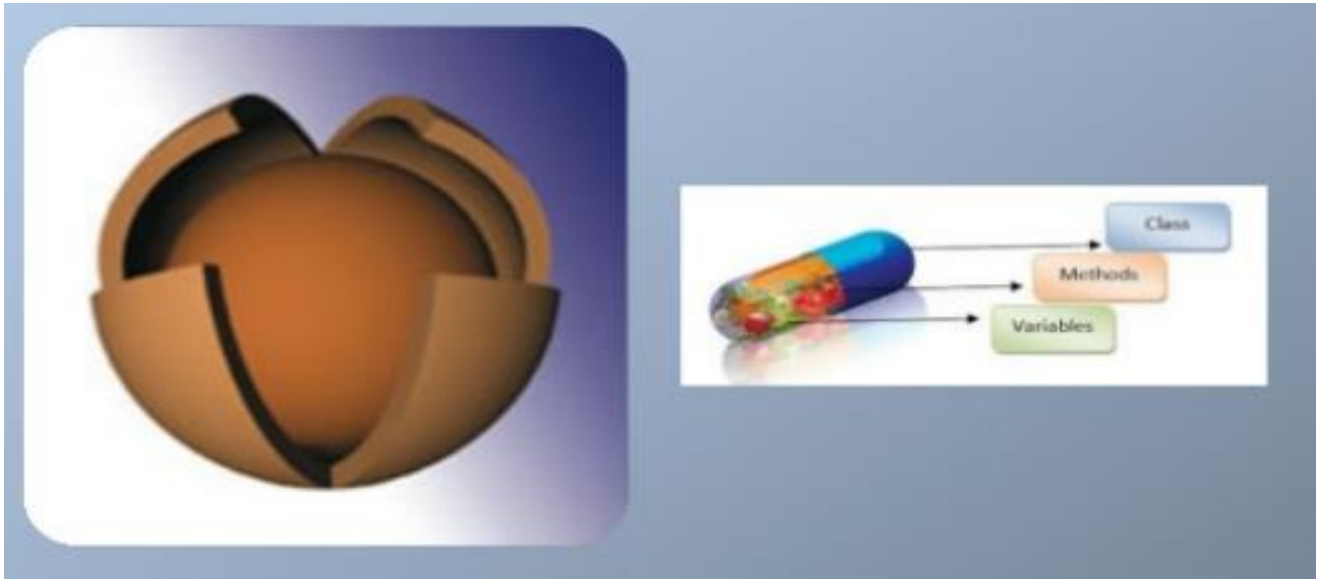
- Classes are user-defined data types and it behaves like built in types of programming language
- Object contains code and data which can be made user define type using class
- Objects are variables of class
- Once a class has been defined we can create any number of objects for that class
- A class is collections of objects of similar type

Classes

- We can create object of class using following syntax
- Syntax: class-name object-name
- Here class name is class which is already created Object name is any user define name. For example, if Student is class
- Example: Student ram, sham
- In example ram and sham are name of objects for class Student. We can create any number of objects for class

Encapsulation

- Encapsulation is the first pillar or principle of object-oriented programming
- In simple words, "Encapsulation is a process of binding data members (variables, properties) and member functions (methods) into a single unit"
- Class is the best example of encapsulation



Encapsulation

- For example: Medical store
- Lets say you have to buy some medicines. You go to the medical store and ask the chemist for the medicines
- Only the chemist has access to the medicines in the store based on your prescription
- The chemist knows what medicines to give to you
- This reduces the risk of you taking any medicine that is not intended for you
- Encapsulation In this example:-

MEDICINES → Member Variables

CHEMIST → Member Methods

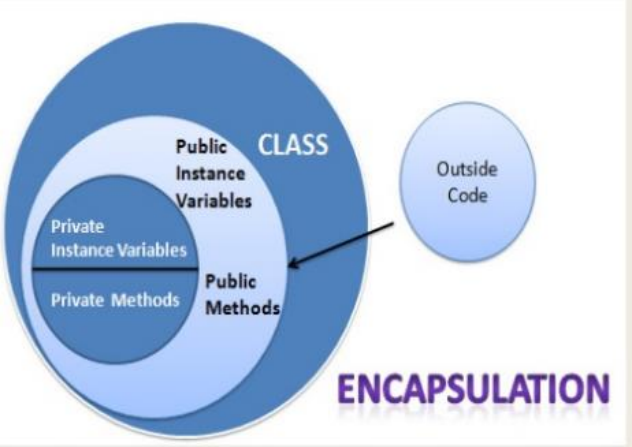
You → External Application or piece of Code

Encapsulation

- Through Encapsulation, Data is not accessible to the outside world, and only those functions which are wrapped in the class can access it

- Encapsulation solves the problem at the implementation level

- A class can specify how accessible each of its members (variables, properties, and methods) is to code outside of the class



Encapsulation

- So encapsulation means hiding the important features of a class which is not been needed to be exposed outside of a class and exposing only necessary things of a class
- Here hidden part of a class acts like Encapsulation and exposed part of a class acts like Abstraction

Abstraction

- Abstraction
  refers representation of necessary features without including more details or explanations
- Data abstraction
  is a programming (and design) technique that relies on the separation of interface and implementation
- When you
  press a key on your keyboard the character appears on the screen,
  you need to know only this, but How exactly it works based on the electronically is not needed

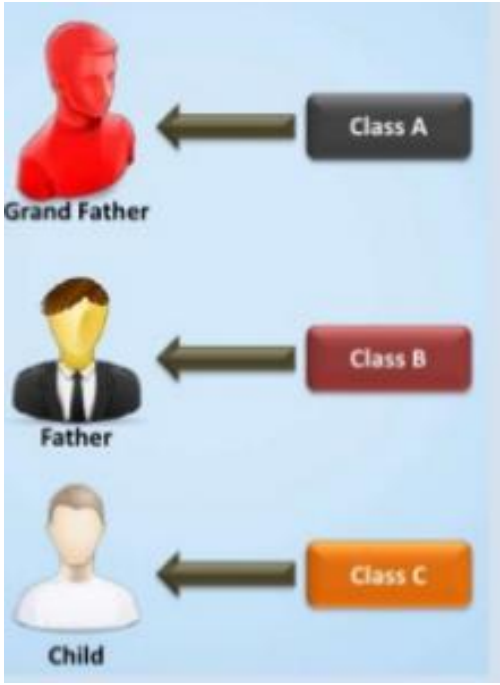
This is called Abstraction

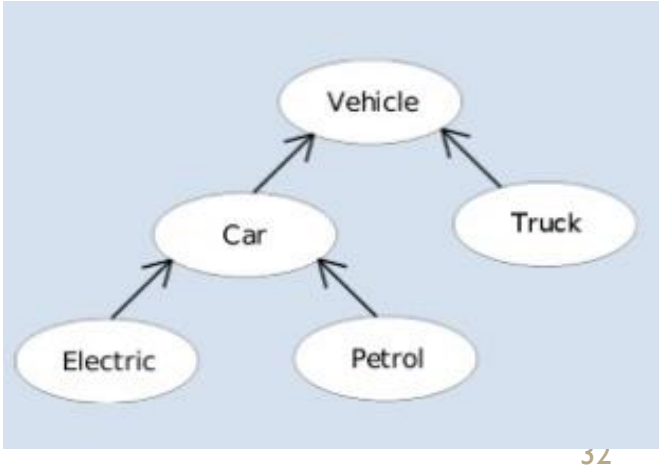
Abstraction

- Another Example when you use the remote control of your TV, you do not bother about how pressing a key in the remote changes the channel on the TV. You just know that pressing the +volume button will increase th e volume

Inheritance

- The mechanism of deriving a new class from an old class is called inheritance or derivation
- The old class is known as base class while new class is known as derived class or sub class.
- The inheritance is the most powerful features of OOP





Inheritance

For Example:

Consider an example of family of three members having a mother, father & son named Jacky.

- Jacky father : tall and dark
- Jacky Mother : Short and fair
- Jacky is tall and fair , so he is said to have inherited the features of his father and mother respectively

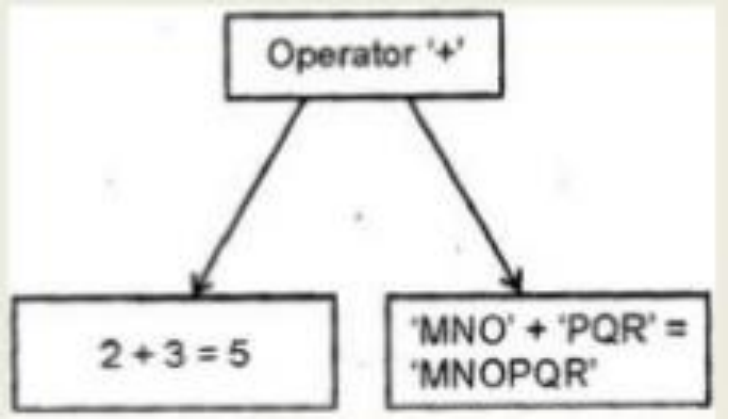
Inheritance

- Through effective use of inheritance, you can save lot of time in your programming and also reduce errors
- Which in turn will increase the quality of work and productivity

- The different types of Inheritance are:
  1. Single Inheritance
  2. Hierarchical Inheritance
  3. Multiple Inheritance
  4. Multi Level Inheritance

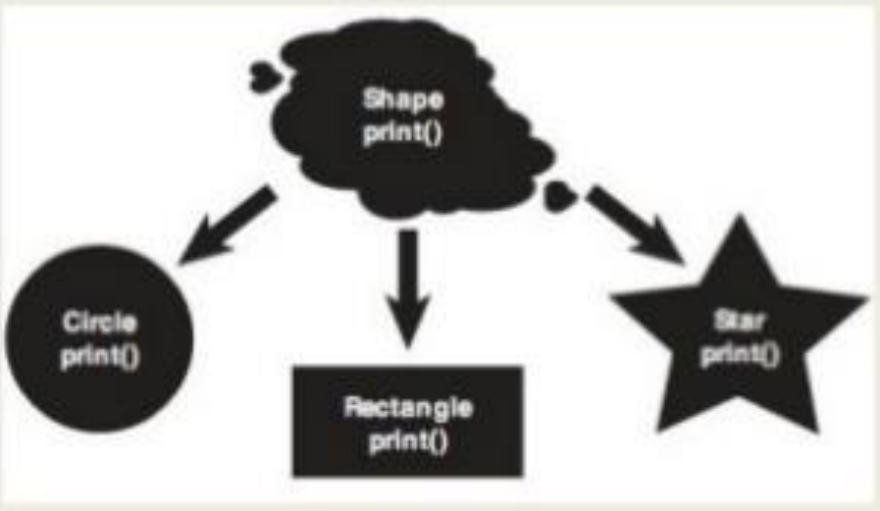
Polymorphism

- Polymorphism is a Greek term which means ability to take more than one form
- For example, + is used to make sum of two numbers as well as it is used to combine two strings
- This is known as operator overloading because same operator may behave differently on different instances



Polymorphism

- Same way functions can be overloaded
- For example, sum () function may takes two arguments or three arguments etc. i.e. sum (5, 7) or sum (4, 6, 8)
- Single function print() draws different objects



Dynamic binding

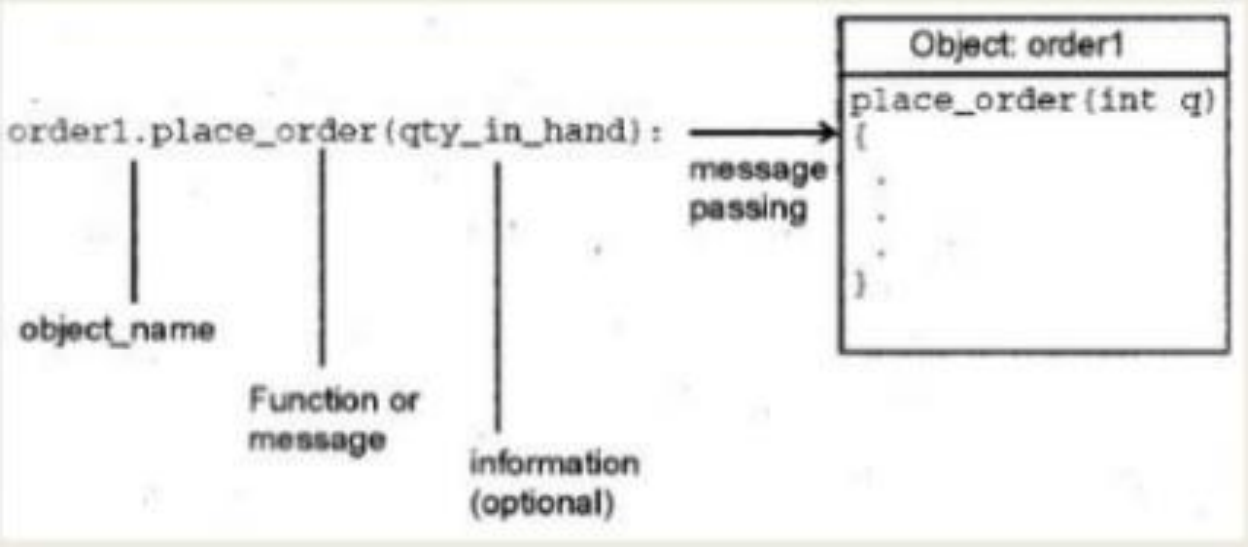
- Binding means link between procedure call and code to be execute
- It is the process of linking of a function call to the actual
  code of the function at run- time
- For example, complier comes to know at runtime that which function of sum will be call either with two arguments or with three arguments

Message Passing

- Objects can communicate with each others by passing message same as people passing message with each other
- Objects can send or receive message or information
- Message passing involves the following basic steps:
  - Creating classes that define objects & their behavior.
  - Creating objects from class definitions.
  - Establishing communication among objects
- Concept of message passing makes it easier to talk about building systems that directly model or simulate their real world counterparts

Message Passing

- For example, consider two classes Product and Order. The object of the Product class can communicate with the object of the Order class by sending a request for placing order



Benefits of OOP

- Code can be reuse by using inheritance
- Data can be hiding from outside world by using encapsulation
- Operators or functions can be overloaded by using polymorphism, so same functions or operators can be used for multitasking
- Object oriented system can be easily upgrade from small system to large system
- It is easy to partition work for same project
- Message passing techniques make communication easier
- Software complexity can be easily managed
- Maintenances cost is less
- Simple to implement

Areas for applications of OOP

- Real time systems
- Simulation and modeling
- Object oriented database
- Hypertext, hypermedia and expertext
- AI and expert systems
- Neural network and parallel programming
- Decision support system
- Office automation system
- CIM / CAM / CAD systems

Features of Java

- Java is simple
- Java is object-oriented
- Java is distributed
- Java is interpreted
- Java is robust
- Java is portable
- Java's performance
- Java is multithreaded
- Java is dynamic
- Java is Secure

Java is simple:

- Derived from C(syntax) & C++(features)
- Easy to write & Compile

Java is object-oriented

- Everything is an object
- Java supports all features of OOP:
  - Class
  - Object
  - Inheritance
  - Polymorphism
  - Encapsulation

Java is distributed:

- Java is known as distributed language for creating application on networks
- Java supports TCP/IP & UDP Protocol.

- RMI [The **RMI** (Remote Method Invocation) is an API that provides a mechanism to create distributed application in java. The RMI allows an object to invoke methods on an object running in another JVM.]

- Support for Internet communication primitives

Java is Interpreted

- Compiling the java source code into an intermediate representation called BYTECODE
- It can run on every platform having Java Virtual Machine installed
- Java Virtual Machine
  - Runtime Environment to execute java programs
  - Interprets the byte code and converts into platform specific code
  - Secure execution of the programs without any side effects

Java is Portable:

- Write Ones Run Anywhere

Java is Robust:

•The program should run effectively on multiple platforms
•Strictly typed language
•Code checking at compile time as well as run time
•Memory management using the Garbage Collector
•Automatic deallocation of memory with unused Objects
•Exception Handling
•Prevent program from crashing due to runtime error

Java is Multithreaded

- Multithreaded means handling multiple tasks simultaneous.
- Multithreading means a single program having different threads executing independently at the same time.

Java is dynamic

- By connecting to the Internet, a user immediately has access to thousands of programs and other computers.
- During the execution of a program, Java can dynamically load classes that it requires either from the local hard drive, from another computer on the local area network or from a computer somewhere on the Internet.
- Java is capable dynamically linking in new class libraries, methods, and objects.

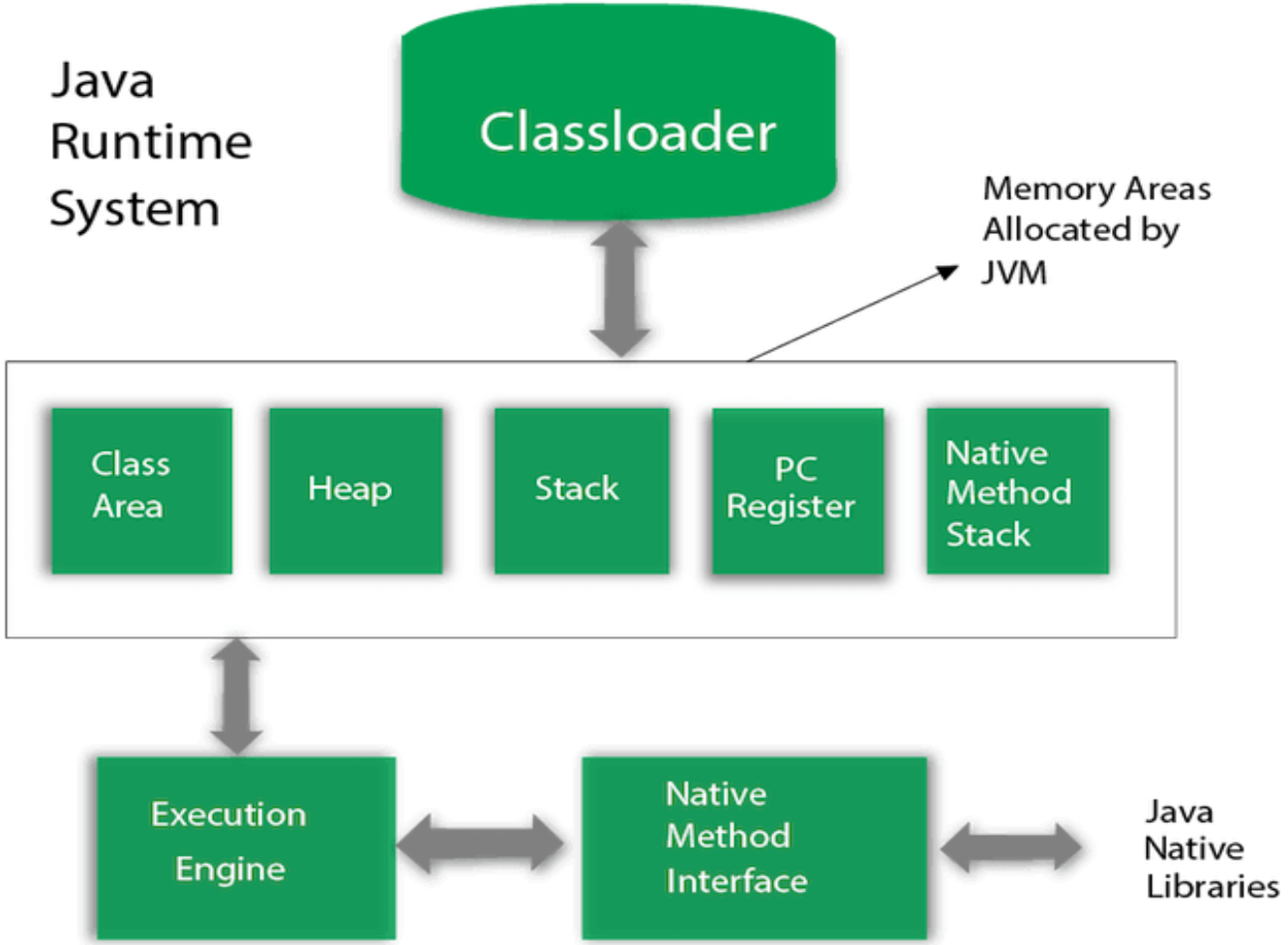
Java is Secure

- The Java language has built-in capabilities to ensure that violations of security do not occur.
- Threat or viruses and abuse of resources are everywhere. Java system not only verifies all memory access but also ensure that no viruses are communicated with an Applet.
- Absence of pointer in java ensures that program cannot gain access to memory locations without proper authorization.

Java Architecture(JDK, JRE and JVM)

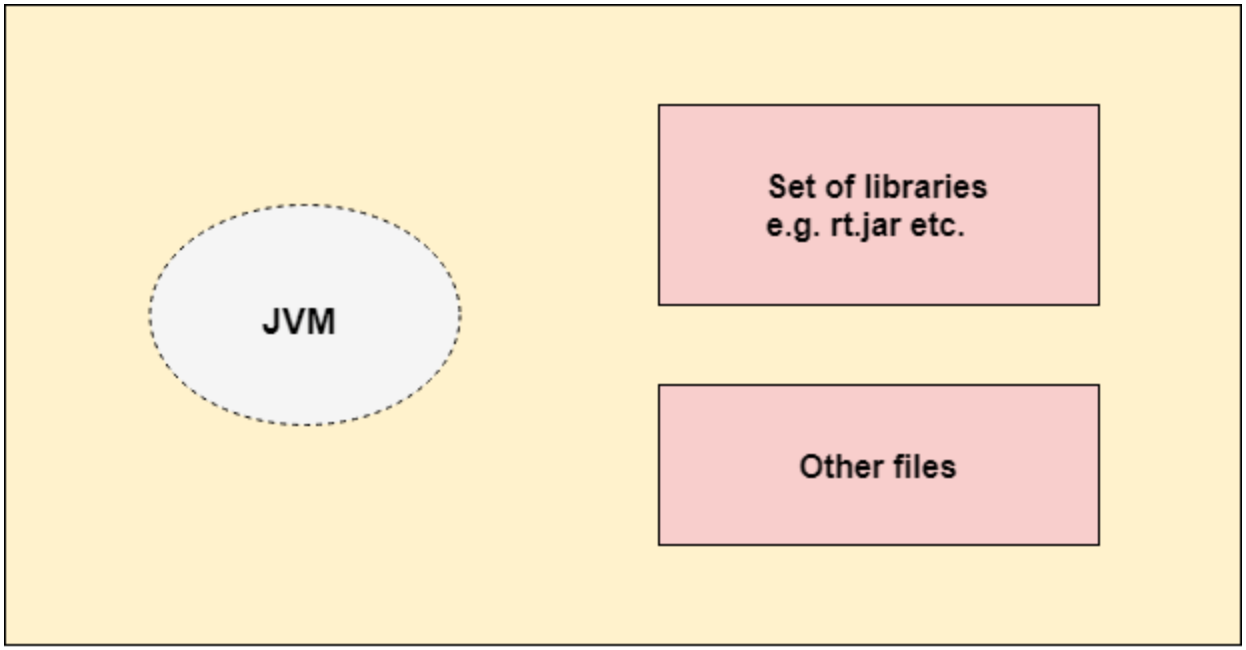
- JVM (Java Virtual Machine)
  - JVM (Java Virtual Machine) is an abstract machine. It is a specification that provides runtime environment in which java bytecode can be executed.
  - JVMs are available for many hardware and software platforms (i.e. JVM is platform dependent).
- What is JVM ?
  - **A specification** where working of Java Virtual Machine is specified. But implementation provider is independent to choose the algorithm. Its implementation has been provided by Oracle and other companies.
  - **An implementation** Its implementation is known as JRE (Java Runtime Environment).
  - **Runtime Instance** Whenever you write java command on the command prompt to run the java class, an instance of JVM is created.

JVM Architecture



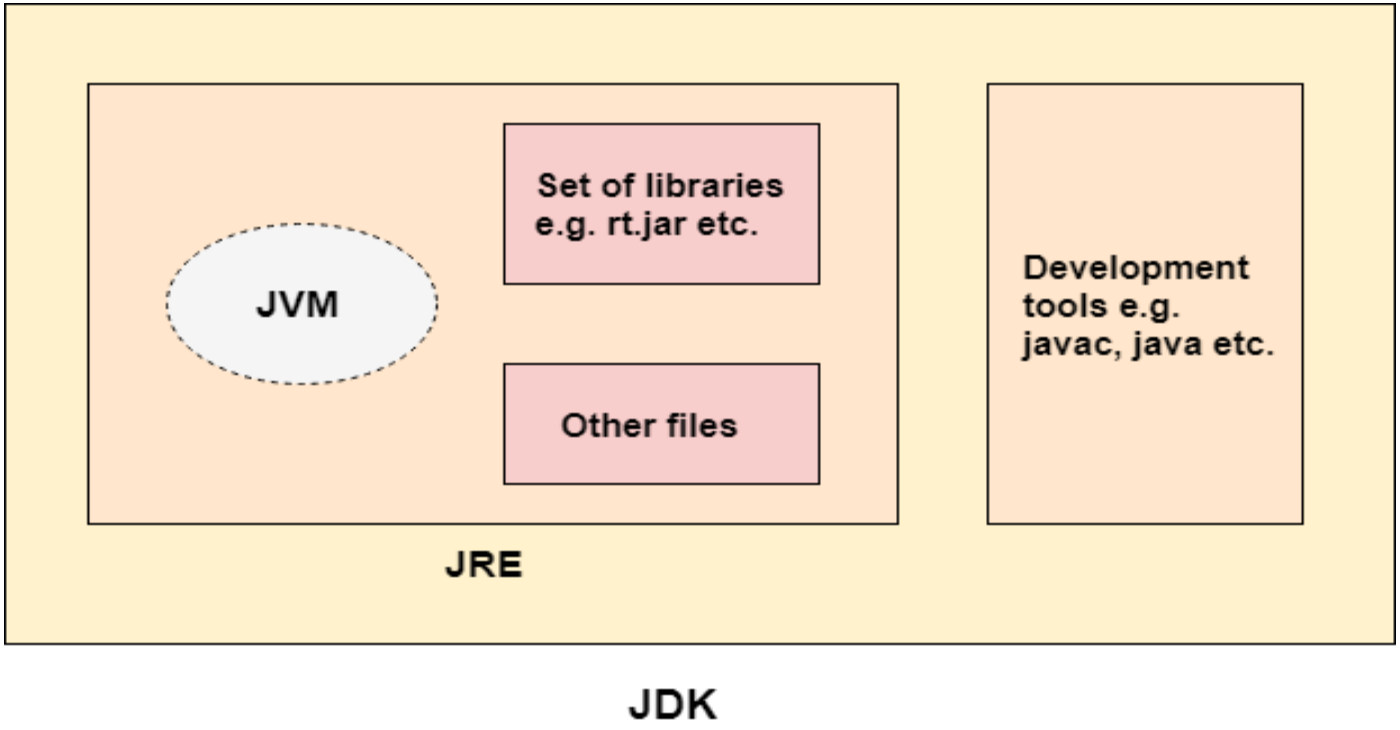
JRE

- JRE (Java Runtime Environment). It is also written as Java RTE. The Java Runtime Environment is a set of software tools which are used for developing Java applications. It is used to provide the runtime environment. It is the implementation of JVM. It physically exists. It contains a set of libraries + other files that JVM uses at runtime.



JDK: Java Development Kit

- JDK (Java Development Kit). The Java Development Kit (JDK) is a software development environment which is used to develop java applications and applets. It contains JRE + development tools.




THANK YOU!
ANY QUESTIONS?