Module-I

Java is a true object oriented language every Object Oriented Programming language must satisfy three important features like Encapsulation inheritance and polymorphism.

Encapsulation: is the concept of binding the data and methods together into a single unit called class. The purpose of the feature is more security to data and methods.

Inheritance: is the process by which the objects of one class acquire the prosperities of objects of another class. Its purpose is the reusability of data and methods.

Polymorphism: is the ability to take more than one form. It is the process in which an interface with different method implementation will be done. Its purpose is to reduce the code.

Benefits of OOP:

OOP offers several advantages to both the programmer and the user. The principle advantages are:-

1 Through inheritance, redundant code can be innovated and the use of existing classes can be extended.

2 The possibility of writing a body of cod once, and than reusing that code over and over again. This leads to saving of development time and higher productivity.

3 Based on the objects work in a project can be partitioned into different modules.

4 The concept of data hiding helps to build secure programs that can not be invaded by code in other parts of the program.

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

6 Software complexity can be easily managed.

Dynamic Initialization of Variable:

Java allows variable to be initialized dynamically using any expression valid at the time the variable in declared.

```
// dynamic initialization program.

Class dynamic

{       public static void main (String args [ ])
```

```
{  int a = 3, b = 4,
    int c = math.sqrt(a*a + b*b); // dynamic initialization.
   System. out. println ("c is"+c)  ;
}
}
```

Control Statements:-

Control flow statements are used to make decisions about which statements to execute and to otherwise change the flow of execution in a program.

These are broadly of three types.

1. **Selection Statements:** if and switch (if else, nested if else, else if ladder, nested switch)
2. **Iteration (Looping) Statements:** for, while, do while.
3. **Jumping Statements:** break, continue, return.

Static: It is a keyword applied to both variables and method. These are called class members since both static variables and

static methods are accessed by the class name itself without creating an object of a class.

These static variables are common to whole class instead of a single object. Hence all objects of the class have same constant value for a static variable. A static variable may change but it is constant for whole class. It is same for different objects of the same class.

Static methods can also be accessed without creating an object of a class. main() method is a static method.

They can not refer to this or super in any way.

final: final key word can be applied to variables, method and classes. A final variable is one whose value does not change during the execution of a program. A final class is one, which cannot be extended; a final method is one, which can not be overrided by its sub class.

this: It refers to the current object. “this” is used to avoid hiding of instance variably by local variables.

Usage of super: The keyword super is used within methods and usually in constructors of a derived class to refer to the constructor of the class from which the new class was derived .super refers to the super class.

Whenever a sub class needs to refer to its, immediate super class, it can do so by the use of the key word super. super has two general forms. The first call the super class constructor. The second is used to access a members of the super class that has been hidden by a members of sub class. A sub class can call a constructor method defined by its super class by use of the following form of super.

super (parameter list);

Here parameter list specifies any parameters needed by the constructor in the super class .super ( ) must always be the first statement executed inside a sub class constructor. super refers to the super class of the sub class in which it is used. This has the general form. Super. members, here member can be either a variable or a method.

Method overriding: When a method in a sub class has the same

name, same parameters and same return type as a method in its super class,then the method in the subclass is said to override the method in the super class. when an overridden method is called from within a subclass, it will always refer to the method defined by the sub class.

Method overriding is done in different classes. It leads to polymorphism. Since the appropriate method is called at run time, it is also called **late-binding.**

Abstract class: it is a class which may or may not contain abstract and non-abstract methods. We can't create an object of an abstract class.

When a class extends an abstract class it must implement all the abstract methods or else the sub class it self be declared as abstract.

final: A final class is one, which can not be extended. The execution of a final class is very fast since there is no inheritance for a final class and hence there is no overriding of methods.

Dynamic method Dispatch: Method overriding forms the basis

for one of Java's most powerful concept called Dynamic method dispatch. Dynamic method dispatch is the mechanism by which a call to an overridden method is resolved at run time, rather than at compile time. This is important because this is how Java implements run time polymorphism.

Object class: It is the super class of all classes in Java. Hence the methods in this class are overridden by its subclasses. Some of the important methods in this class are equals( ), finalize ( ), notify ( ), notify All ( ), to String ( ), wait ( ) etc.,