Date-19/2/19

OOPs: - 4concepts of OOPs.

1.Encapsulations-Binding and wrapping code and data together into single unit.

2.Polymorphism-When one task is performed by different ways. In java we use overloading and overriding to achieve polymorphism.

3.Inheritance-one object acquire all the properties and behaviours of parent’s object.

4.Abstractions-Hiding internal details and showing functionality.

Classes and Objects: - Class contains code.

Classes-data member (objects are also known as instance).

Instance variables :-Inside the class and outside the method.

Local variable :- Variable distinguish by brown color. The variable that is declared with in psvm and method class.

JVM Memory-

Heap: -Heap contains objects.

Stack: -Stack contains memory contains methods, local and reference variables.

Code: -Code contains byte code.

Static: -Static contains Static data/methods.

Methods: - It is collection of statements that are grouped together to perform one operation.

Public void result known as signature.

1.What is void?  
<access specifiers><return type><method name> :this is fixed ,we can’t move return type and method name. Only move keyword and access specifiers.

Constructors

Constructor is a special method.It is used for initialization. It is used for instance variables.

This is reference as well as keyword and method.This refer to the current object.

Public static int a;

Static public int a;

Public static void main()

Void static public main()

Static public void main()

Static block priority is greater than the instance variable. Static block execute without written in main method.Till java 1.5 program execute without using psvm block.

This or super keyword cannot use in same time in static method.

Static method belongs to class and not to the object. It access only static data. It call only static method It accessed directly class name. It cannot refer “this” or “super” keyword.

Inheritance

Extends keyword is used to achieve inheritance.java doesn’t support multiple inheritance with the use of classes but support with the help of interface.

Polymorphism: -2types of polymorphism: overloading and overriding.

Overloading: compile, static binding.

Overriding: runtime, dynamic binding.

Prototype:3 factors effect prototype are numbers of arguments, type of argument and sequence of argument.

We can overload: const, methods,static methods.

OL can be used in inheritance.

Overriding: same signature using in parent as well as child class. Overloading is possible in same class but whereas overriding is not possible in same class.

Parent ob.=new child ();

Ob. Show ();

Can be override and overload constructor of parent’s class into the child class?

Can be override static methods of parent into child?

This ()-default constructor;

Super ()-constructor;

Super. Show ()-calling for parent class.

Return type is used for initialization getting value on run TIME.

1 Overriding always exist inheritance. It is not possible in same class.

All the method of parents defines by child.

Final Keyword-Final put restrictions. We can’t change value in final with the help of using final front of the declaration.

When final use in static then it gives memory management it saves memory.

We can’t change the value of final class. We can change final to static but not change from static to final. Final method can be overridden to the child class. There is no sub class in a final class.

PACKAGES

A way to organize files in java.it is used when a project consists of multiple module. Package keyword is used to create new package.

Import package. \*- for all my pack classes.

Import package. ClassName-Class name-by default.

Fully qualified name;

Access Specifier:It is also known as visibility specifiers which regulate access to classes,fields and methods.

Public(everyone):from same class.from any class in same package. . from subclass in same package.from subclass outside the same package.from any non subclass outside the package

Private(class): from same class.

Protected: It is deal with inheritance. from same class.from any class in same package. . from subclass in same package. from subclass outside the same package

Default(package): no access specifier.

from same class.from any class in same package. . from subclass in same package.