**ABSTRACTION**

Abstract Classes: defined using Abstract keyword

Abstract methods: hides the implementation part.

Abstraction works on Methods(implementation hiding)

Encapsulation works on variables(data hiding)

Abstract class class\_name{

Abstract method name();

}

* To have abstract method my class should be abstract class.
* Abstraction is achieved in two ways abstract class(1 -100%) and interface(100% abstraction)
* Abstraction doesn’t applicable for variables we can’t give abstract variable\_name;
* In an abstract class we can have both abstract and non-abstract methods(final,static,instance)
* Even if we don’t have abstract methods in your abstract class we cannot create an instance for the abstract class.
* If we have only abstract methods in our class(100% abstraction)
* We cannot create instance for an abstract class directly but we can create instance for other classes in its main.
* We can access static methods in the main of abstract class as static methods belongs to Class but not instance.
* We can access abstract class by inheritance
* We need to implement the body of all the abstract methods of parent class in inherited class.
* We can use the child class OBJ to call the abstract methods .
* We can create a constructor for abstract class, even though we can’t create an instance of an abstract class ,the constructor can be called when child class instance is created .i.e whenever the constructor of child class is invoked it automatically invokes its parent class by Super();
* If we are having multilevel inheritance of abstract classes then all its abstract methods implementation can be done in any class but by the end of last class(where we are declaring our main method and creating instance ) all the implementation should be completed.
* We cannot have private, static and final for abstract classes
* We cannot override the static method so we can’t give static to abstract methods.
* We cannot inherit final classes but abstract methods need to be inherited for its implementation so abstract classes can’t be final
* We cannot decrease the visibility of methods and variables.
* If I try to create instance for the abstract class it creates anonymous class and by the time I don’t create implementations for all my abstract methods then my anonymous class throws error to fix that I need to implement the um implemented abstract method in my anonymous class but the instance will ne belonged to only my anonymous classs and not my abstract class.
* Abstract class allows constructor chaining.
* Compiler will take care of the chaining only for single inheritance i.e only for default constructor
* If we have more classes (i.e multi-level) then we need to explicitly take care of Constructor chainings.