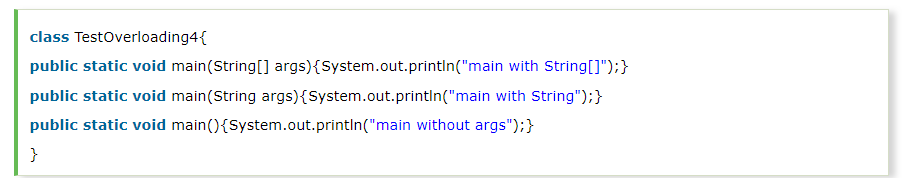
**CSE 310**

**Week 2**

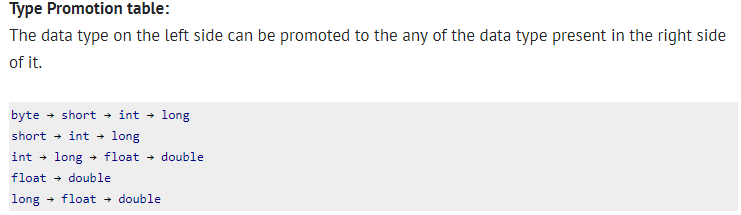
1. **Method overloading:**
   1. Function Signature:
      1. Parameter types
      2. Parameter order
      3. Number of parameters.

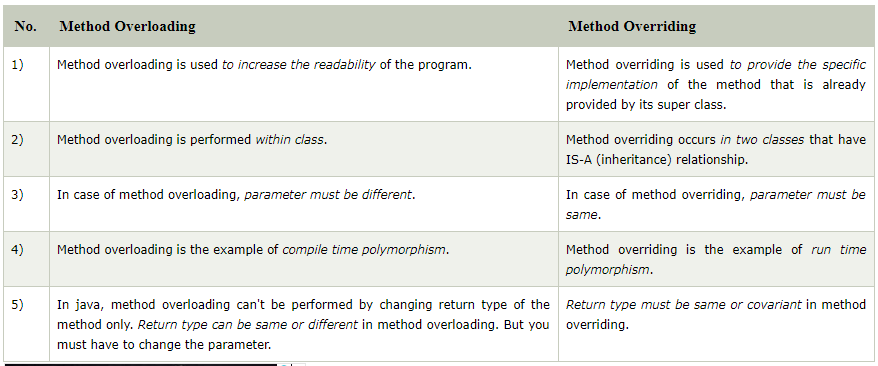
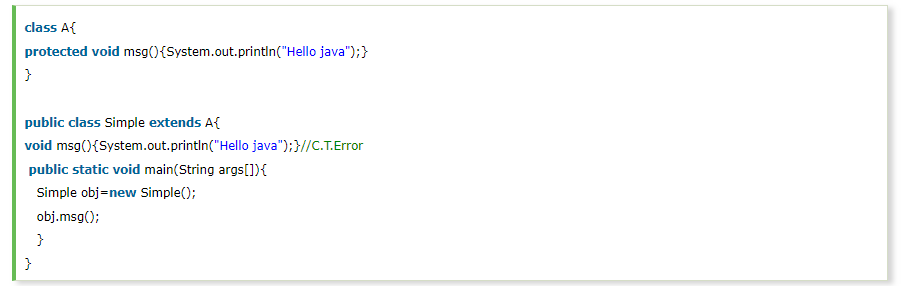
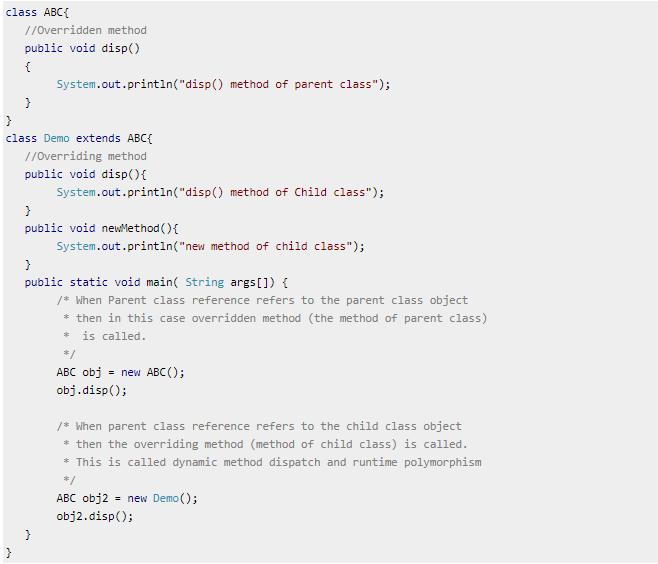
N.B: Parameter type is not function signature. (check error)

* 1. Can we overload main() method?
     + - Yes, by method overloading. You can have any number of main methods in a class by method overloading. But JVM calls main() method which receives string array as arguments only.



* 1. Static Polymorphism is also known as compile time binding or early binding.
  2. [Static binding](https://beginnersbook.com/2013/04/java-static-dynamic-binding/) happens at compile time. Method overloading is an example of static binding where binding of method call to its definition happens at Compile time.
  3. Type promotion:

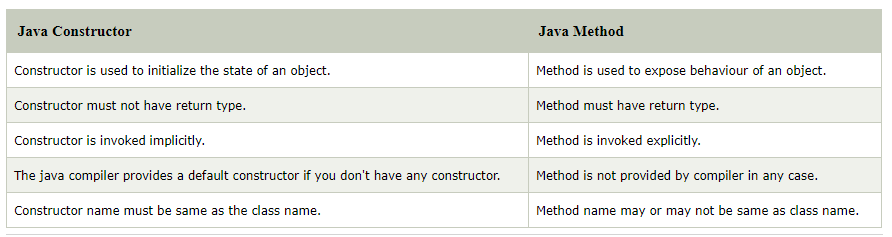


1. **Method overriding:**
   1. Rules for method overriding:
      1. Method must have the same name as in the parent class.
      2. Method must have same parameter as in the parent class.
      3. Must be IS-A relationship (inheritance).
   2. Real life scenario: Bank interest in different banks
   3. Can we override static method?
      * + No, static method cannot be overridden.
   4. Why we cannot override static method?
      * + Because static method is bound with class whereas instance method is bound with object. Static belongs to class area and instance belongs to heap area.
   5. Difference between method overloading and overriding?
   6. If you are overriding any method, overridden method (i.e. declared in subclass) must not be more restrictive. The following code will generate an error.
   7. Method overriding and dynamic method dispatch:
      * + When a parent class reference points to the child class object then the call to the overridden method is determined at runtime, because during method call which method(parent class or child class) is to be executed is determined by the type of object. This process in which call to the overridden method is resolved at runtime is known as dynamic method dispatch. Please write, run and check the output of following code:

**N.B: In dynamic method dispatch the object can call the overriding methods of child class and all the non-overridden methods of base class but it cannot call the methods which are newly declared in the child class. In the above example the object obj2 is calling the disp(). However if you try to call the newMethod() method (which has been newly declared in Demo class) using obj2 then you would give compilation error**

* 1. Binding of overridden methods happen at runtime which is known as dynamic binding.
  2. If a class is extending an abstract class or implementing an interface then it has to override all the abstract methods unless the class itself is an abstract class.

1. **Static variable:**
   1. The static variable can be used to refer the common property of all objects (that is not unique for each object) e.g. company name of employees, college name of students etc.
   2. The static variable gets memory only once in class area at the time of class loading.
2. **Static Method:**
   1. A static method belongs to the class rather than object of a class.
   2. A static method can be invoked without the need for creating an instance of a class.
   3. Static method can access static data member and can change the value of it.
   4. Static method cannot use non static data member or call non-static method directly.
   5. this and super cannot be used in static context.
   6. Why main() method is static?
      * + Because object is not required to call static method if it were non-static method, jvm create object first then call main() method that will lead the problem of extra memory allocation.
3. **Static Block:**
   1. Is used to initialize the static data member.
   2. It is executed before main method at the time of classloading.
   3. Can we execute a program without main() method?
      * + Yes, one of the way is static block but in previous version of JDK not in JDK 1.7.
4. **Constructor:**
   1. In Java, constructor is a block of codes similar to method. It is called when an instance of object is created and memory is allocated for the object.
   2. Every time an object is created using new() keyword, at least one constructor is called. It is called a default constructor.
   3. It is called constructor because it constructs the values at the time of object creation. It is not necessary to write a constructor for a class. It is because java compiler creates a default constructor if your class doesn't have any.
   4. Rules for creating java constructor:
      1. Constructor name must be same as its class name.
      2. Constructor must have no explicit return type.
   5. There are two types of constructors: Default and Parameterized.
   6. Difference between constructor and method:



1. **Java Copy constructor:**
   1. There is no copy constructor in java. But, we can copy the values of one object to another like copy constructor in C++.
   2. There are many ways to copy the values of one object into another in java. They are:
      * + By using constructor
        + By assigning the values of one object into another
        + By clone() method of Object class
   3. Does constructor return any value?
      * + Yes, current class instance.

### Can constructor perform other tasks instead of initialization?

### Yes, like object creation, starting a thread, calling method etc. You can perform any operation in the constructor as you perform in the method.