**Main() Method in Java**

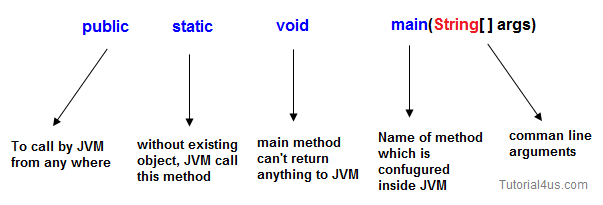
main() method is starting execution block of a java program or any java program start their execution from main method. If any class contain main() method known as main class.

**Syntax of main() method:**

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

*.......*

*}*



**Public**

public is a keyword in a java language whenever if it is preceded by main() method the scope is available anywhere in the java environment that means main() method can be executed from anywhere. main() method must be accessed by every java programmer and hence whose access specifier must be public.

**Static**

static is a keyword in java if it is preceded by any class properties for that memory is allocated only once in the program. Static method are executed only once in the program. main() method of java executes only once throughout the java program execution and hence it declare must be static.

**Void**

void is a special datatype also known as no return type, whenever it is preceded by main() method that will be never return any value to the operating system. main() method of java is not returning any value and hence its return type must be void.

**String args[]**

String args[] is a String array used to hold command line arguments in the form of String values.

**In case of main() method following changes are acceptable**

1. We can declare String[] in any valid form.

*String[] args*

*String args[]*

*String []args*

2. Instance of String[] we can take var-arg String parameter is String...

**Syntax**

main(String[] args) --> main(String... args)

3. We can change the order of modifiers i.e Instead of

**Syntax**

public static we can take static public

4. Instead of args we can take any valid java identifier.

**Syntax**

public static void main(String a[])

**We can overload main() method ?**

Yes, We can overload main() method. A Java class can have any number of main() methods. But run the java program, which class should have main() method with signature as "public static void main(String[] args). If you do any modification to this signature, compilation will be successful. But, not run the java program. we will get the run time error as main method not found.

**Example of override main() method**

*public class mainclass{*

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

*System.out.println("Execution starts from Main()");*

*}*

*void main(int args){*

*System.out.println("Override main()");*

*}*

*double main(int i, double d){*

*System.out.println("Override main()");*

*return d;*

*}*

*}*

**Output**

**Execution starts from Main()**

**What is main method in Java**

main method in Java is an standard method which is used by JVM to start execution of any Java program. main method is referred as entry point of Java application which is true in case of core java application but in case of container managed environment like Servlet, EJB or MIDlet this is not true as these Java programs have there own life-cycle methods like init(), service() or destory() for Servlet's which is used by container. main method in Java is run by main thread which is a non daemon thread and Java program runs until main method finishes or any other user thread is running. When we start JVM by running java command we also provide name of class which contains main method, which is later invoked by JVM to start Java program execution. for example in below command :

*C:\Documents and Settings\JavaTutorial>edit Calculator.java*

**code of Calculator class**

*public class Calculator{*

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

*System.out.println("I am main class which contains main method");*

*}*

*}*

*C:\DOCUME~1\JavaTutorial>javac Calculator.java*

*C:\DOCUME~1\JavaTutorial>java Calculator*

**I am main class which contains main method**

Calculator class must contain public static void main(String args[]) method. Now if we change the signature of main method and try to run same Java program we will get Error as shown below :

*public class Calculator{*

*public static void main(String args){*

*System.out.println("I am main class which contains main method");*

*}*

*}*

*C:\DOCUME~1\sharma>javac Calculator.java*

*C:\DOCUME~1\sharma>java Calculator*

**Exception in thread "main" java.lang.NoSuchMethodError: main**

because we have changed String[] to String which means JVM not able to find standard main method which is required to start execution of Java program

**Valid Signature of main method in Java**

main method is a standard method and has pre specified signature, if you change the signature of main method JVM will not be able to locate main method will throw Exception at runtime as shown in above example. main method is public, static and void and accept an String[] as argument and from Java 5 onwards it can also accept variable arguments instead of array. following signatures are valid main method signature in Java :

*public static void main(String args[]) {}*

*public static void main(String[] args){}*

*public static void main(String... args){}*

You can also use certain modifier like final, synchronized and strictfp along with main method in Java.

**Why main method is public static and void in Java**

This is very famous core Java interview question, and appeared on many fresher and experience level interviews. Though ever java programmer uses main method not every one is familiar with reason why main is static or why main is public in Java. Here are few reasons which make sense to me to think of Why main is public, static and void in java

main method in Java is public so that its visible to every other class, even which are not part of its package. if its not public JVM classes might not able to access it.

main method is static in Java, so that it can be called without creating any instance. While JVM tries to execute Java program it doesn't know how to create instance of main class as there is no standard constructor is defined for main class.

main method is void in Java because it doesn't return any thing to caller which is JVM .

**Difference between C,C++ and Java main method**

If you come from C and C++ programming language than you know what is main method as both of these program also uses main() as entry point for program execution but main method in C and C++ is quite different than main method in Java. Main method in java doesn't return anything and has return type void while main method in C and ++ return int.

That's all on What is main method in Java, What are valid signature of main method in Java, What happens if you put incorrect signature of main method, Why main method is public, static, and void in Java and finally we saw difference between C, C++ and Java main methods. This is one of the most important fundamental of Java programming language and every Java programmer should be familiar with it.

**Running The main() Method**

When you start a Java program you usually do so via the command line (console). You call the java command that comes with the JRE, and tells it what Java class to execute, and what arguments to pass to the main() method. The Java application is then executed inside the JVM (or by the JVM some would claim). Here is a diagram illustrating this:

A command line executing the java command, which in turn executes a Java main program.

A command line executing the java command, which in turn executes a Java main program.

**Here is an example command line:**

*java -cp classes myjavacode.MyClass*

The first part of this command is the java command. This command starts up the JVM. In some cases you may have to specify the full path to where the java command is located on your computer (typically inside the bin subdirectory of the Java install dir).

The second and third arguments (-cp classes) tells the JVM in what directory the compiled Java classes are located (cp means class path). In this case the compiled Java classes are located in a directory named classes.

The fourth argument is the name of the Java class the JVM is to execute. Notice how the class name also contains the name of the package the class is located in (the "fully qualified class name").

**Passing Arguments to the main() Method**

You can pass arguments from the command line to the main() method. This command line shows how:

*java -cp classes myjavacode.MyClass Hello World*

When the JVM executes the main() method of the myjavacode.MyClass, the String array passed as parameter to the main() method will contain two Strings: "Hello" and "World".

**The main() method can access the arguments from the command line like this:**

*package myjavacode;*

*public class MyClass {*

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

*System.out.println( args[0] );*

*System.out.println( args[1] );*

*}*

*}*

Notice the references to element 0 and element 1 in the args array (args[0] and args[1]). args[0] will contain the String (text) Hello and args[1] will contain the String World.

Compiling and running Java source code is explained in more detail in the text Java Project Overview, Compilation and Execution.

Variables and arrays will be explained in more detail in later texts. Don't worry if you don't fully understand them at this point.

**The Java Main Class**

If only a single Java class in your Java program contains a main() method, then the class containing the main() method is often referred to as the main class.

You can have as many classes as you want in your project with a main() method in. But, the Java Virtual Machine can only be instructed to run one of them at a time. You can still call the other main() methods from inside the main() method the Java Virtual Machine executes (you haven't seen how yet) and you can also start up multiple virtual machines which each execute a single main() method.

**Can We Overload main() method?**

Yes, We can overload main() method. A Java class can have any number of main() methods. But to run the java class, class should have main() method with signature as “public static void main(String[] args)”. If you do any modification to this signature, compilation will be successful. But, you can’t run the java program. You will get run time error as main method not found.

*public class MainMethod{*

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

*System.out.println("Execution starts from this method");*

*}*

*void main(int args){*

*System.out.println("Another main method");*

*}*

*double main(int i, double d){*

*System.out.println("Another main method");*

*return d;*

*}*

*}*

**Can we declare main() method as private or protected or with no access modifier?**

No, main() method must be public. You can’t define main() method as private or protected or with no access modifier. This is because to make the main() method accessible to JVM. If you define main() method other than public, compilation will be successful but you will get run time error as no main method found.

*public class MainMethod{*

*private static void main(String[] args){*

*//Run time error*

*}*

*}*

**Can We Declare main() Method As Non-Static?**

No, main() method must be declared as static so that JVM can call main() method without instantiating it’s class. If you remove ‘static’ from main() method signature, compilation will be successful but program fails at run time.

*public class MainMethod{*

*public void main(String[] args){*

*System.out.println(1);* ***//Run time error***

*}*

*}*

**Why main() method must be static?**

Suppose, If main() is allowed to be non-static, then while calling the main method JVM has to instantiate it’s class. While instantiating it has to call constructor of that class. There will be an ambiguity if constructor of that class takes an argument. For example, In the below program what argument JVM has to pass while instantiating class “MainMethod”?.

*public class MainMethod{*

*public MainMethod(int i){*

***//Constructor taking one argument***

*}*

*public void main(String[] args){*

***//main method as non-static***

*}*

*}*

That’s why main() method must be static.

**Can we change return type of main() method?**

No, the return type of main() method must be void only. Any other type is not acceptable.

*public class MainMethod{*

*public static int main(String[] args){*

*return 1; //****run time error : No main method found***

*}*

*}*

**Can main() method take an argument other than string array?**

No, argument of main() method must be string array. But, from the introduction of var args you can pass var args of string type as an argument to main() method. Again, var args are nothing but the arrays.

*public class MainMethod{*

*public static void main(String... args){*

***//Var args as an argument***

*}*

*}*

**Can we run java class without main() method?**

No, you can’t run java class without main method. But, there are some scenarios like if super class has main() method, then sub class can be run without defining main() method in it. For example, if you run class B in the below program, you will get 1 as output.

*class A{*

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

*System.out.println(1);*

*}*

*}*

*public class B extends A{*

*}*

**Note:** Before Java 7, you can run java class by using static initializers. But, from Java 7 it is not possible.

----------------------------------------------------------------------------------

**Top 10 Java Interview Questions On main() Method**

**1.Can we define a class without main method?**

No, you can’t run java class without main method.

Before Java 7, you can run java class by using static initializers. But, from Java 7 it is not possible.

**2.Can main() method take an argument other than string array?**

No, argument of main() method must be string array.

But, from the introduction of var args you can pass var args of string type as an argument to main() method. Again, var args are nothing but the arrays.

*package com.instanceofjava;*

*public class MainMethod{*

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

*}*

*}*

**3.Can we change return type of main() method?**

No, the return type of main() method must be void only. Any other type is not acceptable.

*package com.instanceofjava;*

*public class A{*

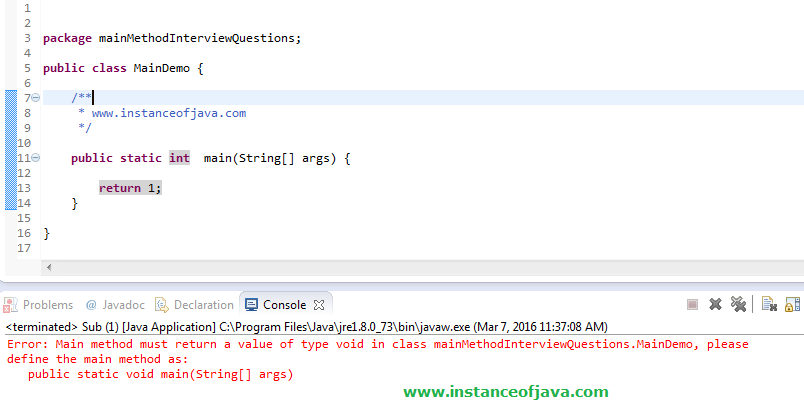
*public static int main(String[] args){*

*return 1;*

***//run time error : No main method found***

*}*

*}*



**4.Why main() method must be static?**

main() method must be static.

If main() is allowed to be non-static, then while calling the main method JVM has to instantiate it’s class.

While instantiating it has to call constructor of that class. There will be an ambiguity if constructor of that class takes an argument.

For example, In the below program what argument JVM has to pass while instantiating class “A”?.

*package com.instanceofjava;*

*public class A{*

*public A(int i){*

***//Constructor taking one argument***

*}*

*public void main(String[] args){*

***//main method as non-static***

*}*

**5.Can We Declare main() Method As Non-Static?**

No, main() method must be declared as static so that JVM can call main() method without instantiating it’s class.

If you remove ‘static’ from main() method signature, compilation will be successful but program fails at run time.

*package com.instanceofjava;*

*public class A{*

*public void main(String[] args){*

*System.out.println("indhu");*  ***//Run time error***

*}*

*}*

**6.Can We Overload main() method?**

Yes, We can overload main() method. A Java class can have any number of main() methods. But to run the java class, class should have main()

method with signature as “public static void main(String[] args)”. If you do any modification to this signature, compilation will be successful.

But, you can’t run the java program. You will get run time error as main method not found.

*package com.instanceofjava;*

*public class A{*

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

*System.out.println("Indhu");*

*}*

*void main(int args){*

*System.out.println("Sindhu");*

*}*

*long main(int i, long d){*

*System.out.println("Saidesh");*

*return d;*

*}*

*}*

**7.Can we declare main() method as private or protected or with no access modifier?**

No, main() method must be public. You can’t define main() method as private or protected or with no access modifier.

This is because to make the main() method accessible to JVM. If you define main() method other than public, compilation will be successful but you will get run time error as no main method found.

*package com.instanceofjava;*

*public class A{*

*private static void main(String[] args){*

***//Run time error***

*}*

*}*

**8.Can we override main in Java ?**

No you can not override main method in Java, Why because main is static method and in Java static method is bonded during compile time and you can not override static method in Java.

**9.Can we make main final in Java?**

you can make main method final in Java. JVM has no issue with that. Unlike any final method you can not override main in Java.

**10.Can we make main synchronized in Java?**

Yes, main can be synchronized in Java, synchronized modifier is allowed in main signature and you can make your main method synchronized in Java.

----------------------------------------------------------------------------------------

**Why is the Java main method static?**

The method signature of a Java main() method is:

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

*...*

*}*

**Is there a reason for this method to be static?**

A.

The method is static because otherwise there would be ambiguity: which constructor should be called? Especially if your class looks like this:

*public class JavaClass{*

*protected JavaClass(int x){}*

*public void main(String[] args){*

*}*

*}*

**Should the JVM call new JavaClass(int)? What should it pass for x?**

If not, should the JVM instantiate JavaClass without running any constructor method? I think it shouldn't, because that will special-case your entire class - sometimes you have an instance that hasn't been initialized, and you have to check for it in every method that could be called.

There are just too many edge cases and ambiguities for it to make sense for the JVM to have to instantiate a class before the entry point is called. That's why main is static.

I have no idea why main is always marked public though.

-------------------------------------------------------------------------------------

**What will happen if main method is declared as private?**

But if you declare main method as private, you would not be able to execute the class as a standalone java program. Any java class that needs to be executed as a standalone file needs to have a main method that is public, static and returns a void.

**Can you declare the main method as final?**

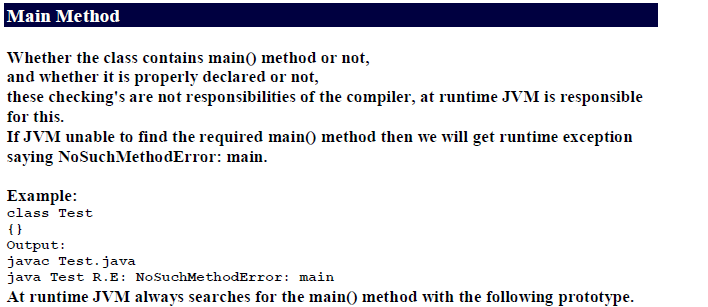
Sure, it can be declared final! It doesn't matter if it is declared final, the JVM will still find it and run it, and the compiler doesn't care. Yes We can declare main method final. If we make this final it can not be override.Hence we can't use it in its child class.

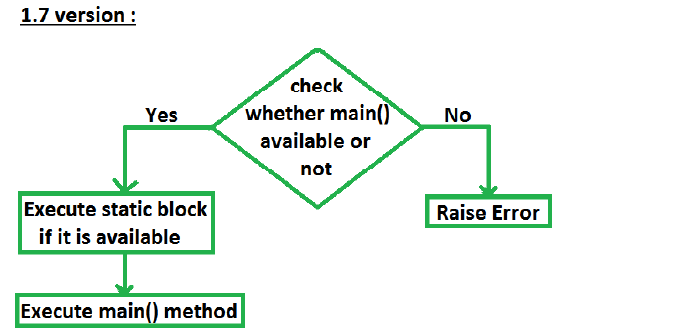
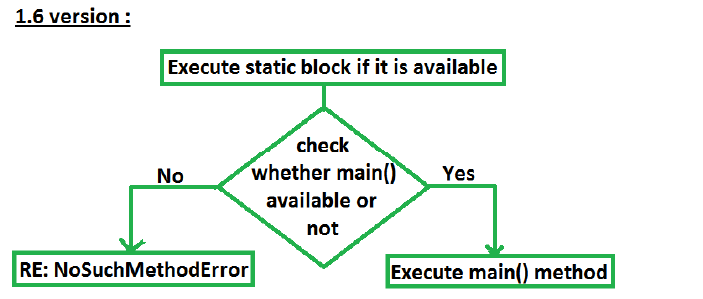
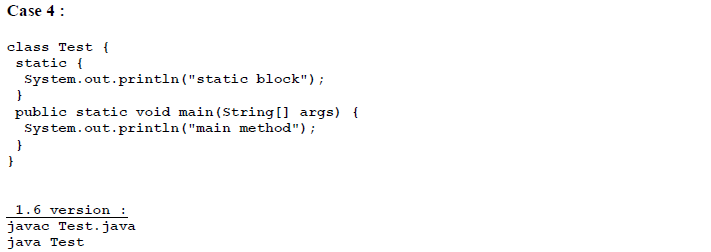
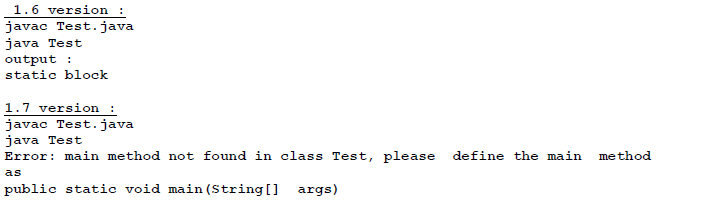
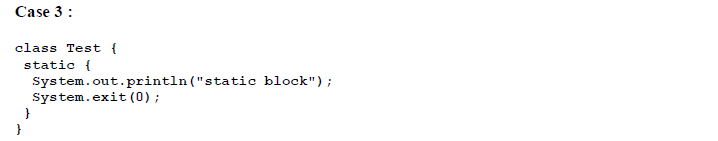
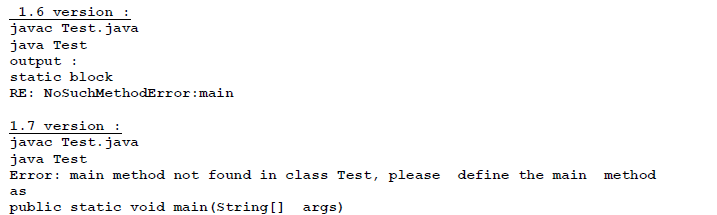
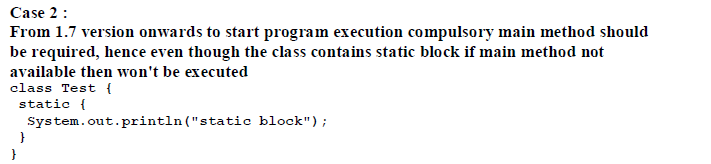
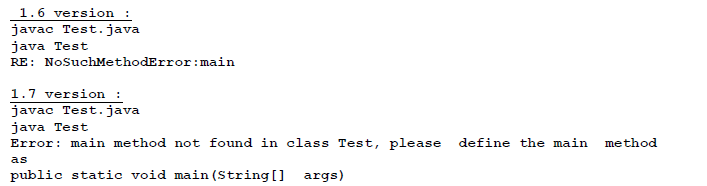
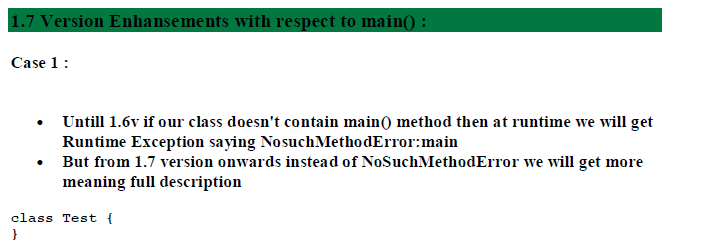
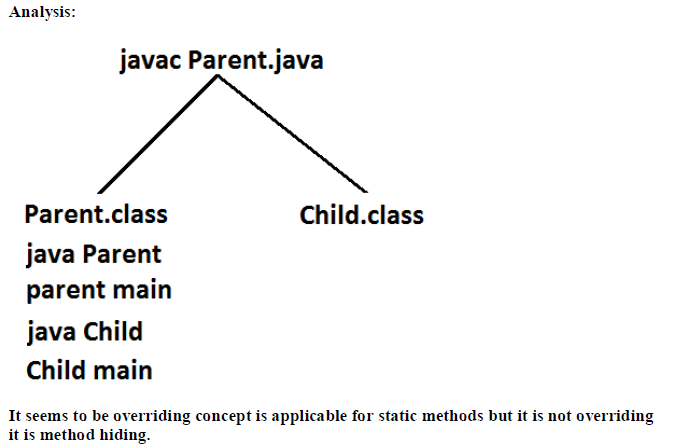
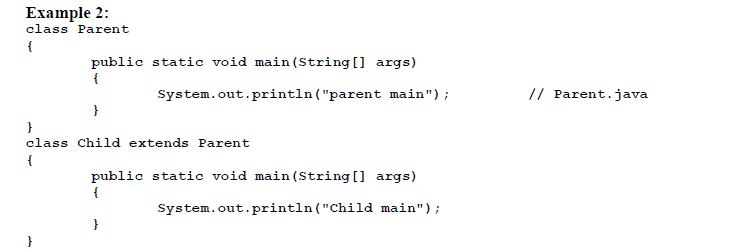
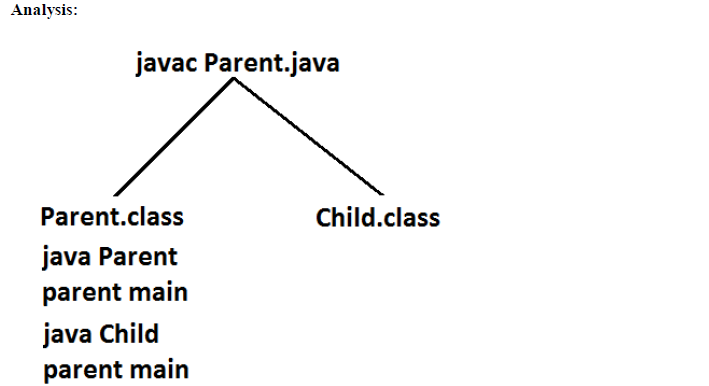
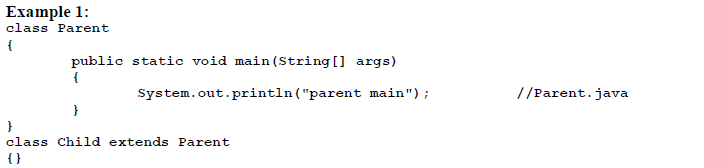
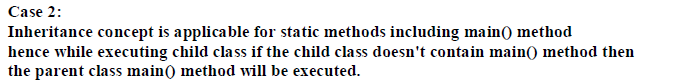
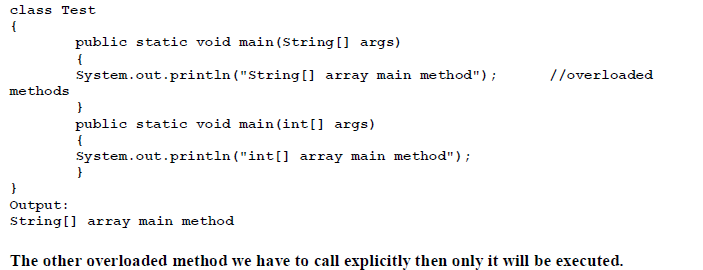
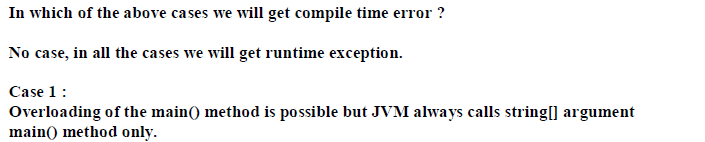
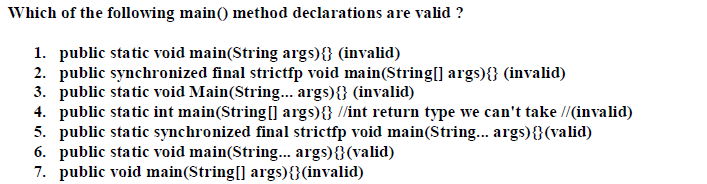
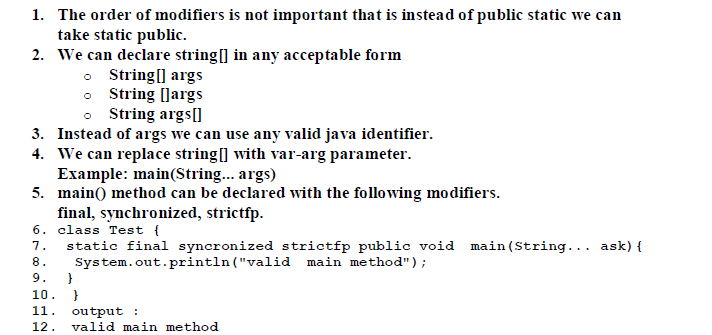
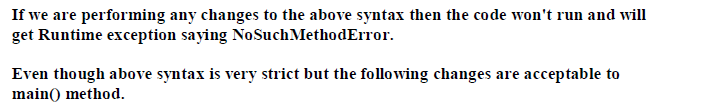
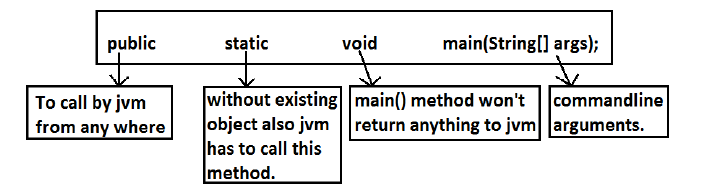
**Can an abstract class have a main method in Java?**

Abstract just means you can't instantiate the class directly. You can have constructors if you want - they might be needed for subclasses to initiate the object state. You can have static methods, including main() and they don't need an object so calling them is fine.

**Why the main method that is present in some Java classes is a static method?**

This is neccesary because main() is called by the JVM before any objects are made. Since it is static it can be directly invoked via the class. Similarly, we use static sometime for user defined methods so that we need not to make objects. void indicates that the main() method being declared does not return a value.

******

******