2015

Test My Interview .COM

Programming Questions to get you started

V1.0


JAVA

If you have been to some Java interviews or conducted them yourselves, then you know that there are some questions which keep repeating. Here are 10 questions to give you an idea of what to ask in a ninterview. This is by no means the be all and end all on Java. So prepare your own questions.

Go through the list below and make your own set of appropriate questions. Some pointers on what responses to expect are also listed with some questions.

1. How does Java achieve platform independence?
   Answer: Java programs are not dependent on any platform, architecture or operating system such as windows or Linux. Java achieve this by using the Java virtual machine (JVM). When a Java program is compiled, it is converted to a .class file which is a collection of byte code and is directly understandable by the JVM. So the same Java program can run on any operating system that has the JVM installed. The JVM can differ for every OS but all JVM can understand converted byte code. That is how Java achieves platform independence.

2. What is a pointer and does Java support pointers?
   Answer: Pointer is a reference handle to a memory location. Improper handling of pointers leads to memory leaks and reliability issues. Because of this reason Java doesn't support the pointers.

3. What is difference between Path and Classpath?
   Answer: **Path** and **Classpath** are operating system level environment variales. While setting up a Java development environment, Path is used define where the system can find the java executables(.exe) files and classpath is used to specify the location .class files.

4. Can we override private method in Java?
   Answer: No, You can not override private method in Java, just like we can not override static method in Java. Like static methods, private method in Java is also bonded during compile time using static binding by Type information and doesn't depends on what kind of object a particular reference variable is holding. Since method overriding works on dynamic binding, its not possible to override private method in Java. private methods are not even visible to Child class, they are only visible and accessible in the class on which they are declared. private keyword provides highest level of Encapsulation in Java. Though you can hide private method in Java by declaring another private method with same name and different method signature.

5. I want to print "Hello" even before main is executed. How will you acheive that?
   Answer: To do this, you need to print the statement inside a static block of oode. Static blocks get executed when the class gets loaded into the memory and even before the creation of an object. Hence it will be executed before the main method. And it will be executed only once.

6. What is the difference between an ArrayList and a Vector in Java?
   Answer: **ArrayList** and **Vector** both implements **List** interface and maintains insertion order. But there are many differences between **ArrayList** and **Vector** classes that are given below.

ArrayList	Vector
ArrayList is **not synchronized**.	Vector is **synchronized**.
ArrayList **increments 50%** of current array size if number of element exceeds from its capacity.	Vector **increments 100%** means doubles the array size if total number of element exceeds than its capacity.
ArrayList is **not a legacy class**, it is introduced in JDK 1.2.	Vector is a **legacy** class.
ArrayList is **fast** because it is non-synchronized.	Vector is **slow** because it is synchronized i.e. in multithreading environment, it will hold the other threads in runnable or non-runnable state until current thread releases the lock of object.
ArrayList uses **Iterator** interface to traverse the elements.	Vector uses **Enumeration** interface to traverse the elements. But it can use Iterat

7. What is the difference between HashMap and Hashtable In Java?

   Answer: HashMap and Hashtable both are used to store data in key and value form. Both are using hashing technique to store unique keys. But there are many differences between HashMap and Hashtable classes that are given below.

HashMap	Hashtable
HashMap is **non synchronized**. It is not-thread safe and can't be shared between many threads without proper synchronization code.	Hashtable is **synchronized**. It is thread-safe and can be shared with many threads.
HashMap **allows one null key and multiple null values**.	Hashtable **doesn't allow any null key or value**.
HashMap is a **new class** introduced in JDK 1.2.	Hashtable is a **legacy class**.
HashMap is **fast**.	Hashtable is **slow**.
We can make the HashMap as synchronized by calling this code Map m = Collections.synchronizedMap(hashMap);	Hashtable is internally synchronized and can't be unsynchronized.
HashMap is **traversed by Iterator**.	Hashtable is **traversed by Enumerator and Iterator**.
Iterator in HashMap is **fail-fast**.	Enumerator in Hashtable is **not fail-fast**.
HashMap inherits **AbstractMap** class.	Hashtable inherits **Dictionary** class.

8. can we declare a static variable inside a method?

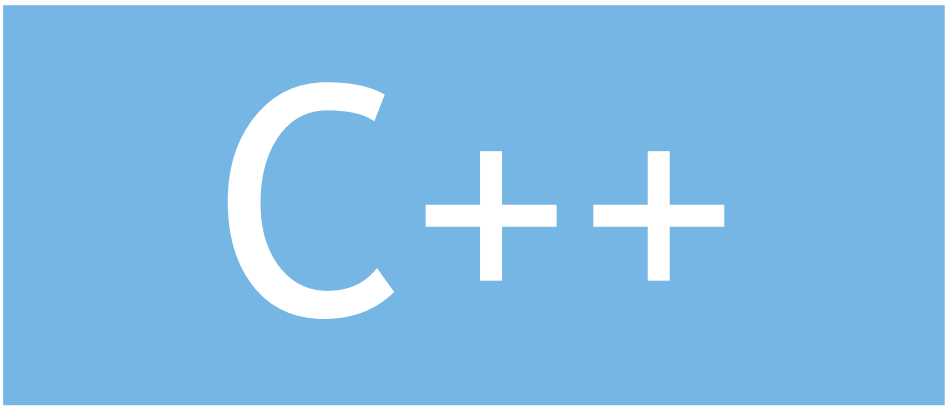
   Answer: Static varaibles are class level variables and they can't be dedared inside a method. If dedared, the class will not compile.

9. What is an Abstract Class and what is it's purpose?

   Answer: A Class that doesn't provide complete implementation is defined as an abstract class. Abstract classes enforce abstraction.

10. How is final different from finally and finalize?

    Answer: **final** is a modifier which can be applied to a dass or a method or a variable. final class can't be inherited, final method can't be overridden and final variable can't be changed. **finally** is an exception handling code section which gets executed whether an exception is raised or not by the try block code segment. **finalize()** is a method of Object class which will be executed by the JVM just before garbage collecting object to give a final chance for resource releasing activity.


C++

If you have been to some C++ interviews or conducted them yourselves, then you know that there are some questions which keep repeating. Here are 10 questions to give you an idea of what to ask in a ninterview. This is by no means the be all and end all on C++. So prepare your own questions.

Go through the list below and make your own set of appropriate questions. Some pointers on what responses to expect are also listed with some questions.

1. Explain what is the use of void main () in C++ language?
   Answer: To run a C++ application it involves two steps. The first step is a compilation where conversion of C++ code to object code take place. While second step includes linking, where combining of object code from the programmer and from libraries takes place. This function is operated by main () in C++ language.

2. What is namespace std; and what is consists of?
   Answer: Namespace std; defines your standard C++ library, it consists of classes, objects and functions of the standard C++ library. You can specify the library by using namespace std or std: : throughout the code. Namespace is used to differentiate the same functions in a library by defining the name.

3. What will i and j equal after the code below is executed? Explain your answer.

```
int i = 5;
int j = i++;
```

   Answer: After the above code executes, i will equal 6, but j will equal 5.

   Understanding the reason for this is fundamental to understanding how the unary increment (++) and decrement (--) operators work in C++.

   When these operators precede a variable, the value of the variable is modified first and then the modified value is used. For example, if we modified the above code snippet to instead say int j = ++i;, i would be incremented to 6 and then j would be set to that modified value, so both would end up being equal to 6.

   However, when these operators follow a variable, the unmodified value of the variable is used and then it is incremented or decremented. That's why, in the statement int j = i++; in the above code snippet, j is first set to the unmodified value of i (i.e., 5) and then i is incremented to 6.

4. Explain how functions are classified in C++ ?
   Answer: In C++ functions are classified as
   - Return type
   - Function Name
   - Parameters
   - Function body

5. Explain what is a reference variable in C++?

   Answer: A reference variable is just like a pointer with few differences. It is declared using & Operator. In other words, reference is another name for an already existing variable.

6. Explain what is data abstraction in C++?

   Answer: Data abstraction is a technique to provide essential information to the outside world while hiding the background details. In the below example, you don't have to understand how `cout` displays the text "Hello candidate" on the user screen and at the same time implementation of cout is free to change.

   For example:

```
#include
Using namespace std;

int main ( )
{
Cout << "Hello candidate" <<endl;
return 0 ;
}
```

7. What is Polymorphism?

   Answer: In object-oriented programming, polymorphism is a generic term that means 'many shapes'. (from the Greek meaning "having multiple forms"). Polymorphism is briefly described as "one interface, many implementations."

   Polymorphism is a characteristic of being able to assign a different meaning or usage to something in different contexts - specifically, to allow an entity such as a variable, a function, or an object to have more than one form.

   There are two types of polymorphism one is compile time polymorphism and the other is run time polymorphism. Compile time polymorphism is functions and operators overloading. Runtime time polymorphism is done using inheritance and virtual functions. Here are some ways how we implement polymorphism in Object Oriented programming languages.

   Compile time polymorphism -> Operator Overloading, Function Overloading
   Run time polymorphism -> Interface and abstract methods, Virtual member functions.

8. How virtual functions are implemented C++?

   Answer: Virtual functions are implemented using a table of function pointers, called the vtable. There is one entry in the table per virtual function in the class. This table is created by the constructor of the class. When a derived class is constructed, its base class is constructed _rst which creates the vtable. If the derived class overrides any of the base classes virtual functions, those entries in the vtable are overwritten by the derived class constructor. This is why you should never call virtual functions from a constructor: because the vtable entries for the object may not have been set up by the derived class constructor yet, so you might end up calling base class implementations of those virtual functions.

9. What are the differences between a C++ struct and C++ class?
   Answer: The default member and base class access specifies are different. This is one of the commonly misunderstood aspects of C++. Believe it or not, many programmers think that a C++ struct is just like a C struct, while a C++ class has inheritance, member functions, overloaded operators, and so on. Actually, the C++ struct has all the features of the class. The only differences are that a struct defaults to public member access and public base class inheritance, and a class defaults to the private access specified and private base-class inheritance.

10. What happens when a function throws an exception that was not specified by an exception specification for this function?
    Answer: **Unexpected()** is called, which, by default, will eventually trigger **abort()**.