1. **What does the “static” keyword mean?**

Ans.: In java, the ‘static’ keyword is used for memory management. The ‘static’ keyword can be used with:

1. **Variable**
2. **Block**
3. **Method**
4. **Static Variable:**

If we declare any variable as static, it is known static variable. It makes your program **memory efficient.**

1. **Static Block:**

**It is used to initialize the static data member which is executed before main method at the time of class loading.**

1. **Static Method:**

If you apply static keyword with any method, it is known as static method. A static method belongs to the class rather than object of a class. It can be invoked without the need for creating an instance of a class. It can access static data member and can change the value of it.

**2. Can you override private or static method in Java?**

**Ans.** No, we can’t override private or static method in java since private or static method are declared once and are inaccessible outside a class.

**3. Can you access non-static variable in static context?**

**Ans.** Yes we can access non-static variable in a static context by creating instance of that class where static variable is defined.

**4. What are the Data Types supported by Java?**

**Ans:** Datatype is a special keyword used to allocate sufficient memory space for the data, in other words Data type is used for representing the data in main memory (RAM) of the computer.

The different types of data types supported in java are:

• Primitive Data Types

• Reference/Object Data Types

**Primitive Data Types**

There are eight primitive datatypes supported by Java. Primitive datatypes are predefined by the language and named by a keyword.

Byte, short, int, char, float, long, double and Boolean

**Reference Data Types**

Reference variables are created using defined constructors of the classes. They are used to access objects. These variables are declared to be of a specific type that cannot be changed. For example, Employee, Puppy, etc.

Class objects and various type of array variables come under reference datatype.

Default value of any reference variable is null.

A reference variable can be used to refer any object of the declared type or any compatible type.

**5. What is Autoboxing and Unboxing?**

**Ans.** The automatic conversion of primitive data types into its equivalent Wrapper type is known as boxing and opposite operation is known as unboxing.

Advantage of Autoboxing and Unboxing:

No need of conversion between primitives and Wrappers manually so less coding is required.

Simple Example of Autoboxing in java:

class BoxingExample1{

public static void main(String args[]){

int a=50;

Integer a2=new Integer(a);//Boxing

Integer a3=5;//Boxing

System.out.println(a2+" "+a3);

}

}

Output:50 5

Simple Example of Unboxing in java:

class UnboxingExample1{

public static void main(String args[]){

Integer i=new Integer(50);

int a=i;

System.out.println(a);

}

}

Output:50

**6. Difference between an Interface and an Abstract class?**

**Ans.** The difference between an Interface and an Abstract class is:

|  |  |
| --- | --- |
| **Abstract class** | **Interface** |
| 1. Abstract class can **have abstract and non-abstract** methods. | 1. Interface can have **only abstract** methods. |
| 1. Abstract class **doesn't support multiple inheritance**. | 1. Interface **supports multiple inheritance**. |
| 1. Abstract class **can have final, non-final, static and non-static variables**. | 1. Interface has **only static and final variables**. |
| 1. Abstract class **can have static methods, main method and constructor**. | 1. Interface **can't have static methods, main method or constructor**. |
| 1. Abstract class **can provide the implementation of interface**. | 1. Interface **can't provide the implementation of abstract class**. |
| 1. The **abstract keyword** is used to declare abstract class. | 1. The **interface keyword** is used to declare interface. |

**7. What is the difference between pass by reference and pass by value?**

**Ans**: The difference between Pass by Value and Pass by Reference are:

a> In Pass by Value Type, a copy of actual arguments is passed to formal arguments of the called function and any change made to the formal arguments in the called function have no effect on the values of actual arguments in the calling function.

In Pass by Reference Type, the location (address) of actual arguments is passed to formal arguments of the called function. This means by accessing the addresses of actual arguments we can alter them within from the called function.

b> In Pass by Value Type, actual arguments will remain safe, they cannot be modified accidentally.

In Pass by Reference Type, alteration to actual arguments is possible within from called function; therefore, the code must handle arguments carefully else you get unexpected results.

**8. What are the basic interfaces of Java Collections Framework?**

**Ans.** Java Collection Framework is an interface which is inherited by all the classes interfaces and is used to implement the concept of sets, list, queue etc.

The basic interfaces of Java Collection Framework are:

**a. Sets**

**b. Lists**

**c. Map**

**a. Sets**

Set is a structure which helps us to store the data in an unordered manner as well as in ordered manner properly. Sets never allows duplicate data.

**b. List**

List is a structure which helps the programmer to store data in a sequential manner or in an ordered manner. In list, duplicate data can’t be stopped from being added. Each data can be used in list by its position like adding new data or removing the old one.

**c. Map**

Map is a special kind of structure. It contains values based on key i.e. key and value pair. Each key and value pair is known as an entry. Map contains only unique keys.

Map is useful if we must search, update or delete elements based on key.

**9. Why do we use an Iterator while using Java Collections?**

**Ans.**  An Iterator is an object that enables programmer to traverse through a collection and to remove elements from the collection selectively.

It allows updating the collection i.e. addition and removal of data to the collection.

Iterator can also be used at points where there is no clue what type of collections will be used as all collections implement Iterator interface.

**10. What is the difference between HashMap and Hashtable ?**

**Ans.** The difference between HashMap and Hashtable are:

|  |  |
| --- | --- |
| ***HashMap*** | ***Hashtable*** |
| 1) HashMap is non synchronized. It is not-thread safe and can't be shared between many threads without proper synchronization code.  2) HashMap allows one null key and multiple null values.  3) HashMap is a new class introduced in JDK 1.2.  4) HashMap is fast.  5) We can make the HashMap as synchronized by calling this code  Map m = Collections.synchronizedMap(hashMap);  6) HashMap is traversed by Iterator.  7) Iterator in HashMap is fail-fast.  8) HashMap inherits AbstractMap class. | 1. Hashtable is synchronized. It is thread-safe and can be shared with many threads. 2. Hashtable doesn't allow any null key or value. 3. Hashtable is a legacy class. 4. Hashtable is slow. 5. Hashtable is internally synchronized and can't be unsynchronized. 6. Hashtable is traversed by Enumerator and Iterator. 7. Enumerator in Hashtable is not fail-fast. 8. Hashtable inherits Dictionary class. |

**11. What is difference between Array and ArrayList ? When will you use Array over ArrayList?**

**Ans.** The difference between Array and ArrayList are:

|  |  |
| --- | --- |
| Array | ArrayList |
| 1. Array is a fixed length structure. 2. Arrays can be single or multi-dimensional. 3. In array we insert elements using the assignment operator. 4. In Array we use loop and for each loop to iterate through array. 5. Array can contain both primitives and Objects in Java | 1. ArrayList is a variable length [Collection class](http://java67.blogspot.sg/2012/09/java-collection-interview-questions.html) 2. ArrayList is always single dimensional. 3. We can insert elements into the arraylist object using the add() method. 4. In ArrayList we use Iterator to iterate through the list. 5. We **cannot store primitives in ArrayList**, it can only contain Objects. |

**12. When to use parseInt method?**

**Ans.** We use parseInt method in java whenever we need to parse the string argument as a signed decimal integer.

**13. What is the difference between StringBuffer and StringBuilder class?**

**Ans.**

|  |  |
| --- | --- |
| **StringBuffer** | **StringBuilder** |
| 1. StringBuffer is *synchronized* i.e. thread safe. It means two threads can't call the methods of StringBuffer simultaneously. | 1. StringBuilder is *non-synchronized* i.e. not thread safe. It means two threads can call the methods of StringBuilder simultaneously. |
| 1. StringBuffer is *less efficient* than StringBuilder. | 1. StringBuilder is *more efficient* than StringBuffer. |

**14. What is finalize method?**

**Ans**. The finalize is a method used to perform clean up processing just before object is garbage collected.

**15. What is the difference between final, finally and finalize method?**

**Ans.** The difference between final, finally and finalize is:

Final is a keyword, finally is a block and finalize is a method.

|  |  |  |
| --- | --- | --- |
| **Final** | **Finally** | **Finalize** |
| 1. final is a keyword. 2. final is used to apply restrictions on class, method and variable. Final class can't be inherited; final method can't be overridden and final variable value can't be changed. | 1. finally is a block. 2. finally is used to place important code, it will be executed whether exception is handled or not. | 1. finalize is a method. 2. finalize is used to perform clean up processing just before object is garbage collected. |

**16. What is the difference between throw and throws keyword?**

**Ans.** The difference between throw and throws is:

|  |  |
| --- | --- |
| **Throw** | **throws** |
| 1. Java throw keyword is used to explicitly throw an exception. | 1. Java throws keyword is used to declare an exception. |
| 1. Checked exception cannot be propagated using throw only. | 1. Checked exception can be propagated with throws. |
| 1. Throw is followed by an instance. | 1. Throws is followed by class. |
| 1. Throw is used within the method. | 1. Throws is used with the method signature. |
| 1. We cannot throw multiple exceptions. | 1. We can declare multiple exceptions e.g. public void method()throws IOException,SQLException. |

**17. When and why to use “super” keyword ?**

**Ans.** The super keyword in java is a reference variable that is used to refer immediate parent class object.

Whenever you create the instance of subclass, an instance of parent class is created implicitly i.e. referred by super reference variable.

Program example:

class Vehicle{

int speed=50;

}

class Car extends Vehicle{

int speed=100;

void display()

{

System.out.println(super.speed);//will print speed of Vehicle now

}

public static void main(String args[]){

Car c=new Car();

b.display();

}

}

Output:

50

**18. What is “jar” file and how to create and use it?**

**Ans.** “Jar” file stands for Java ARchive file. It is a package file format typically used to aggregate many Java class files, associated metadata and resources such as text, images, etc. into one file to distribute application software or libraries on the Java platform.

Java Archive files “JAR” are built on the [ZIP (file format)](https://en.wikipedia.org/wiki/ZIP_%28file_format%29) and typically have the .jar [file extension](https://en.wikipedia.org/wiki/File_extension).

We can create a JAR file using following Steps:

1. Start *Command Prompt*.
2. Navigate to the folder that holds your class files:

C:\>cd \mywork

1. Set path to include JDK’s bin.  For example:

C:\desktop> path c:\Program Files\Java\jdk1.7.0\_25\bin;%path%

1. Compile your class(es):

C:\desktop> javac \*.java

1. Create a manifest file and your jar file:

C:\desktop> echo Main-Class: Assignment >output.txt

C:\desktop> jar cvfm Assignment.jar output.txt \*.class

To test or run the jar file we use the command

C:\mywork> Assignment.jar

C:\mywork> java -jar Craps.jar

**19. What is the purpose of fileInputStream and fileReader class?**

**Ans.** FileInputStream Class is the class package contain various predefined methods which help us to read the data such as images, audio files etc in the form of bytes.

FileReader Class is the class package we use to read character data from the text files as input.

20. What is the purpose of fileOutputStream and fileWriter class?

Ans. FileOnputStream Class is the class package contain various predefined methods which help us to write the data to files such as images, audio files etc in the form of bytes.

FileWriter Class is the class package we use to write character data to the text files as output.