Core Java selective Question & Answer

Q-1: What do you mean by encapsulation?

Answer:

Encapsulation is a process that allows selective hiding of properties and methods within objects. The main advantage of encapsulation is the ability to hide the implementation of a class. Therefore, it is also known as **data hiding**

Q-2: What do you mean by " super( ) " and " this "?

Answer:

" super( ) " refers to invoking of super class constructor. And
" this " refers to current class object

Q-3: What do you mean by method overloading and method overriding?

Answer:

Overloading: Define two or more methods with the same name but different number, type and sequence of parameter in a class is called method overloading.

Example:

```
int sum(int x, int y) {
        //bodypart;
  }
Int sum(int x, int y, int z){

            }
```

Overriding:

Define two or more methods with the same name and same number, type and sequence of parameter in a derived class is called method overriding.

Example:

```
interface A{
int sum(int x, int y);
}
class B implements A{
public int sum(int x, int y){
        }
}
```

Q-4: What do you mean by polymorphism?

Answer:

Polymorphism enables the same method to behave differently on different class.

Q-5: What do you mean by logical and short-circuit operator
Or, What is the difference between & and && ?

Answer:

The difference between & and && is that the conditional or short-circuit operator (&&) will not bother to evaluate the right hand operand if the left hand operand false.

But the logical operator (&) will evaluate both the operands.

Q-6: What do you mean by instanceof operator?

Answer:

The instanceof operator is used to check the type of an instance of an object.

Eg:

```
String s = "XYZ";
if (s instanceof java.lang.String)
        returns TRUE.
```

***Q-7: Define package. What are the advantages of package? Write down the name of* default name of package.**

Answer:

A package in java is an encapsulation mechanism that can be used to group related class, interfaces and sub packages.

Advantages of package:d

a. Package allows us to organize our class into smaller units and make it easy to locate and use the appropriate class file.
b. It helps us to about naming conflict.
c. It protects our class and methods in a larger way than a class to class basis.
d. Package name can be use to identify our classes.

The default package name in java is java.lang

Q-8: What is a constructor? What do you mean by default constructor?

Answer:

Constructor:

A constructor is a special method that is used to initialize a newly created object. A **constructor** doesn't have a return type. The primary purpose of a constructor is initializing the instant variable.

Default constructor:

If we do not define any constructor for a class the compiler will supply a default constructor in the class, which actually does nothing. The default constructor is also described as no argument constructor. Because it requires no argument to be specified when it called. When we create an object of a class the default constructor is automatic invoked.

```
Ex:     TestClass tc = new TestClass();
```

Q-9: What is an Array? How many ways can create an Array in java?

Answer:

Array:

An **array** is a container object that holds a fixed number of values of a single type. The length of an **array** is established when the **array** is created. After creation, its length is fixed..

We can declare an array in several ways.

```
int[] arr = new int[5];
int arr[] = new int[5];
int[] arr = {0,1,2,3,4};                    //dense array
int[][] arr2 = new int[length1][length2];   //2D array
Object list1 = new Object[5];               //An array of objects
```

Q-10: What is the difference between ‘ equals ( ) ’ methods and ‘ == ’ operator.?

Answer:

equals ( ):

equals ( ) method checks the equality of the content.

== :

“==” checks the equality of object reference.

Q-11: What is type casting?

Answer:

Type casting means to explicitly convert one type to another.

For example, the following lines:

```
double x = 5.0;
int y = x;
```

will produce an error message, because of a possible loss of precision (any decimals will get lost when converting to int). The following will work however:

```
double x = 5.0;
int y = (int) x;
```

The programmer is forcing the Java compiler to accept the conversion; saying, in effect: "Please do this anyway, I know what I am doing".

Q-12: What do you mean by Class and Object?

Answer:

A class, acts like a blue print for creating objects. That means a class is a collection of objects that have common properties and behavior and relationship.

Instance of a class is called an object. An object is a collection of variable or properties and methods.

Object = variable/properties + methods.

Q-13: Checked Exceptions vs. Unchecked Exceptions.

Answer:

Checked Exceptions

In **Java exceptions** under Error and RuntimeException classes are unchecked **exceptions**

- A checked exception is subclass of Exception excluding class RuntimeException and its subclasses.
- Compiler checks to see if these exceptions have been properly caught or not. Else the code doesn't compile. Thus, a program is forced to deal with the situations where an exception can be thrown.
- Checked exceptions must be either declared or caught at compile time.

Unchecked Exceptions

The compiler checks to make sure that this is done. There are two things a method can do with a checked exception. Such as catch and throw.

- Unchecked exceptions are RuntimeException and all of its subclasses along with class java.lang.Error and its subclasses also are unchecked.
- A program does compile without these exceptions being handled during compile time.

Q-14: What is the use of finalize ( ) and finally block?

Answer:

The automatic garbage collector calls the finalized() method that is eligible for garbage collector before actual destroying the object.

A finally block is an optional block that exist after the last catch block and always executed whether or not exception is caught.

Q-15: What do you mean by auto boxing?

Answer:

The automatic conversion of primitive int type into a wrapper class object is called autoboxing. It does not require to type cast the int value. The modification of primitive wrapper objects is done directly. The following example illustrates autoboxing:

```
int number;
Integer intObject;
number = 1;
intObject = 2;
number = intObject;
intObject = number;
```