1. **What is Checked Exception ?**

Java forces you to handle these error scenarios in some manner in your application code. They will come immediately into your face, once you start compiling your program. You can definitely ignore them and let them pass to JVM, but it’s bad habit. Ideally, you must handle these exceptions at suitable level inside your application so that you can inform the user about failure and ask him to retry/ come later.

Checked exceptions are subclasses of **Exception** class.

**Example of checked exceptions : ClassNotFoundException**, **IOException, SQLException**

public static void main(String[] args)

{

    try

    {

        FileReader file = new FileReader("somefile.txt");

    }

    catch (FileNotFoundException e)

    {

        //Alternate logic

        e.printStackTrace();

    }

}

1. **What is Unchecked Exception ?**

Java also provides UncheckedExceptions, the occurrences of which **are not checked by the compiler.** They will come into life/occur into your program, once any buggy code is executed.

A method is not forced by compiler to declare the unchecked exceptions thrown by its implementation. Generally, such methods almost always do not declare them, as well.

Unchecked Exceptions are subclasses of **RuntimeException. Example of unchecked exceptions**are : ArithmeticException, ArrayStoreException, ClassCastException and so on.

1. **Who is responsible to handle checked and unchecked exception?**

When a super class method (overridden method) declares that it can throw an exception then sub class method (overriding method) must also declare that it can throw the same kind of exception or a sub type of that exception.   
  
To handle the exception while you overriding a method in Java, you will have to follow three important rules. They are as follows.

1. **If an overridden method does not throw an exception using throws clause then**

**➲**The overriding method can not throw any checked or compile-time exception.

**➲** The overriding method can throw any unchecked or runtime exception.  
  
2. **If an overridden method throws an unchecked or runtime exception then**  
**➲** The overriding method can throw any unchecked or runtime exception.  
     **➲** The overriding method can throw the same exception which is thrown by the overridden method.  
     **➲** The overriding method can ignore the method level exception.  
  
3. **If the superclass method throws checked or compile-time exception then**  
**➲** Subclass method can throw the exception which is a subclass of the super class method's exception.   
**➲** Subclass method cannot throw the exception which is a super class of the super class method's exception.  
**➲** Subclass method can throw any unchecked or runtime exception.

**4. Can we write only try block without catch and finally blocks?**

Yes, wecan have trywithoutcatchblock by using finallyblock. You can use try with finally. As you know finallyblock always executes even if you have exception or return statement in **try** block **except** in case of System

**5. Differentiate between error and exception?**

An **Error** "indicates serious problems that a reasonable application should not try to catch." An **Exception** "indicates conditions that a reasonable application might want to catch." **Error** along with RuntimeException & their subclasses are unchecked **exceptions**. All other **Exception** classes are checked **exceptions**

You can use the MyUncheckedBusinessException in the same way as any other unchecked exception. You can throw it in your code and catch it in a catch clause. And you can but don't need to specify if your method throws it.

**6. What is Rethrow Exception ?**

**Java rethrow exception** allows you to specify more specific **exception** types in the throws clause of a method declaration.

**7. Can we throw an exception manually? If yes, how?**

Yes, we can throw an exception manually using throw keyword. 125.. **Exceptions** raised in the try block are handled in the catch block. **If** it is unable to handle that **exception**, it can rethrow that exception using throw keyword.