1. Explain different types of Errors in Java.
Ans: There are 3 types of errors in Java and they are:
i. **Syntax Errors**: These are syntax-related errors in that a programmer mistakenly uses incorrect syntax that may change depending on the programming language. These are also known as Compile Time Errors because these errors are identified by the compiler during the compilation of the program itself.
ii. **Logical Errors**: These are errors that arise during runtime due to a wrong/bad logic used in the code by the programmer. It may result in an unexpected or wrong output. These errors are not identified by the compiler but only at the runtime. They are solved by using a step in the Sofware Development Life Cycle called Testing.
iii. **Runtime Errors**: In these types of errors, there is no syntax-related issue or no case of wrong/unexpected output but because of some user input/operation, the application crashes. While using the application or when it is running, it is crashing sometimes. They are identified only at runtime by the JVM during the execution of the program.

2. What is an Exception in Java?
Ans: It is an unexpected or unwanted event that disturbs the normal flow of execution of a program/application. It is similar to a runtime error where there is no compilation issue in the program and also there is no logical error but due to some user input or operation the application crashes. Since it is responsible for crashing the application, it needs to be “Handled” explicitly by the programmer.

3. How can you handle exceptions in Java? Explain with an example.
Ans: Exceptions can be handled with the following blocks:
try block: Here, the set of statements that may be Critical(there are possibilities that these may generate an Exception) is written inside this block.
catch block: This block is used for Handling the Exception, i.e., writing logic to catch all exceptions that are generated in the try block.
finally block: This block is used to close all the resources. It will be executed irrespective of whether the try or catch block is executed, hence, it is always executed whenever there is a resource leak happening in the program.
Example:

```
import java.util.Scanner;
class Test {
    public static void main(String []args) {
        int num1;
        int num2;
        int res=0;
        Scanner sc = new Scanner(System.in);
        num1 = sc.nextInt();
        num2 = sc.nextInt();
        try {
            res = num1/num2;
        }
    catch(Exception e) {
        System.out.println("Enter a valid number "  + e.getMessage());
    }
    finally {
        sc.close();
    }
    System.out.println(res);
    }
}
```

4. Why do we need exception handling in Java?
Ans: If we don't handle the exception it will result in the program/application to halt/crash/terminate then and there itself and we cannot know what may have been written further from that line where the exception occurred because none of it would get executed. This is actually a very bad thing to happen in real-life and it lands us in an unknown stage.
If we handle it, then the execution of the program/application would be smooth. Even if we get the exception in a line, it would just show us the error message and continue with the execution as it as expected.

5. What is the difference between exception and error in Java?
Ans: An error can't be handled like an exception and it cannot be controlled by a programmer like an exception also compiler doesn't force us to handle it. Errors happen when the program/application is running whereas Exceptions occur due to the application itself.

6. Name the different types of exceptions in Java.
Ans: Based on how JVM handles the exceptions, there are 2 types of exceptions:
Checked Exceptions: These occur during the compilation because the compiler knows that it will generate an Exception and explicitly tells the programmer to handle it using a try-catch block. If not handled, it will throw an Exception.
examples: IOException, SQLException, AWTException, etc.
Unchecked Exceptions: These occur while the program/application is running meaning that the compiler doesn't check for these Exceptions or it is unable to detect them. It passes the compilation stage but when the application is running it will throw an Exception.
examples: RuntimeException

7. Can we just use try instead of finally and catch blocks? Give an example.
Ans: No, it is not possible to do so because it will result in a compiler error.
Example:

```
import java.util.Scanner;
```

```
class Test {
    public static void main(String []args) {
        int num1;
        int num2;
        int res=0;
        Scanner sc = new Scanner(System.in);
        num1 = sc.nextInt();
        num2 = sc.nextInt();
        try {
            res = num1/num2;
        }
        System.out.println(res);
    }

Output:
Test.java:10: error: 'try' without 'catch', 'finally' or resource
declarations
        try {
        ^
1 error
```

This shows that using a try block without a catch or finally is not possible. catch or finally block must always accompany a try block. We can remove either finally or catch block but not both of them.