1. Write a program to demonstrate the use of try, catch, finally throw and throws keywords and demonstrate the following points in the program.

* 1. Multiple catch blocks.
  2. try-catch-finally combination.
  3. try-finally combination.
  4. Exception propagation among many methods.
  5. Nested try blocks

package Assignment7;  
  
class ExceptionDemo {  
 static void method1() throws Exception {  
 throw new Exception("From method1");  
 }  
  
 static void method2() throws Exception {  
 *method1*();  
 }  
// a)  
 static void multipleCatch() {  
 try {  
 int i = 10/0;  
 } catch (ArithmeticException e) {  
 System.*out*.println("Caught ArithmeticException in multipleCatch: " + e);  
 } catch (Exception e) {  
 System.*out*.println("Caught Exception in multipleCatch: " + e);  
 }  
 }  
// b)  
 static void tryCatchFinally() {  
 try {  
 int i = 10/2;  
 System.*out*.println("Result: " + i);  
 } catch (Exception e) {  
 System.*out*.println("Caught Exception in tryCatchFinally: " + e);  
 } finally {  
 System.*out*.println("Finally block in tryCatchFinally");  
 }  
 }  
// c)  
 static void tryFinally() {  
 try {  
 int i = 10/2;  
 System.*out*.println("Result: " + i);  
 } finally {  
 System.*out*.println("Finally block in tryFinally");  
 }  
 }  
//e)  
 static void nestedTry() {  
 try {  
 int i = 10/2;  
 try {  
 int j = i/0;  
 } catch (Exception e) {  
 System.*out*.println("Caught Exception in nested try: " + e);  
 }  
 } catch (Exception e) {  
 System.*out*.println("Caught Exception in nested try: " + e);  
 }  
 }  
  
 public static void main(String[] args) {  
 try {  
 *method2*();  
 } catch (Exception e) {  
 System.*out*.println("Caught Exception in main: " + e);  
 }  
 *multipleCatch*();  
 *tryCatchFinally*();  
 *tryFinally*();  
 *nestedTry*();  
 }  
}

1. Write a program to throw a checked exception explicitly using 'throw' keyword and

* 1. Handle the exception in same method.
  2. use throws clause and handle the exception in some other method (calling method)
  3. Don't either handle or use the throws clause.

package Assignment7;  
  
public class CheckedException {  
 public static void main(String[] args) throws Exception {  
 try {  
 *throwException*();  
 } catch (Exception e) {  
 System.*out*.println("Exception caught in main method: " + e);  
 }  
  
 try {  
 *throwExceptionWithThrows*();  
 } catch (Exception e) {  
 System.*out*.println("Exception caught in main method: " + e);  
 }  
  
 *throwExceptionWithoutHandling*();  
 }  
  
 public static void throwException() throws Exception {  
 try {  
 throw new Exception("Exception thrown using throw keyword");  
 } catch (Exception e) {  
 System.*out*.println("Exception caught in throwException method: " + e);  
 }  
 }  
  
 public static void throwExceptionWithThrows() throws Exception {  
 throw new Exception("Exception thrown using throw keyword and handled in calling method");  
 }  
  
 public static void throwExceptionWithoutHandling() throws Exception {  
 try {  
 throw new Exception("Exception thrown without handling");  
 } finally {  
 System.*out*.println("This is a finally block");  
 }  
 }  
}

1. Write a program to throw an unchecked exception explicitly using 'throw' keyword and

* 1. Handle the exception in same method.
  2. use throws clause and handle the exception in some other method (calling method)
  3. Don't either handle or use the throws clause.

1. package Assignment7;  
     
   public class UncheckedException {  
    public static void main(String[] args) {  
    try {  
    *throwException*();  
    } catch (RuntimeException e) {  
    System.*out*.println("Exception caught in main method: " + e);  
    }  
     
    try {  
    *throwExceptionWithThrows*();  
    } catch (RuntimeException e) {  
    System.*out*.println("Exception caught in main method: " + e);  
    }  
     
    *throwExceptionWithoutHandling*();  
    }  
     
    public static void throwException() {  
    try {  
    throw new RuntimeException("Exception thrown using throw keyword");  
    } catch (RuntimeException e) {  
    System.*out*.println("Exception caught in throwException method: " + e);  
    }  
    }  
     
    public static void throwExceptionWithThrows() {  
    throw new RuntimeException("Exception thrown using throw keyword and handled in calling method");  
    }  
     
    public static void throwExceptionWithoutHandling() {  
    throw new RuntimeException("Exception thrown without handling");  
    }  
   }
2. Write a program in which main method calls the foo method which calls the bar method. Bar method can throw a checked exception. Use throws for throwing the exception from bar. Don’t handle exception in bar using try catch. Let the calling function handle the same.

package Assignment7;  
  
public class CheckedExceptionPropagation {  
 public static void main(String[] args) {  
 try {  
 *foo*();  
 } catch (Exception e) {  
 System.*out*.println("Exception caught in main method: " + e);  
 }  
 }  
  
 public static void foo() throws Exception {  
  
 *bar*();  
 }  
  
 public static void bar() throws Exception {  
 throw new Exception("Checked exception thrown from bar method");  
 }  
}