MODULE 2p - Exception Throwing and Catching

FIRST TASK

Key in and try out the ArgsArrayA program and note that the ArrayIndexOutOfBoundsException is thrown when there are fewer than three arguments.

SECOND TASK

To stop the program crashing when an exception is thrown the appropriate course of action is to embed the statement or statements that might throw the exception in a try-clause and then catch the exception, including helpful println statements in an associated catch-clause. Modify ArgsArrayA to...

```
public class ArgsArrayEx
 { public static void main(String[] args)
    { System.out.println("There are " + args.length + " arguments");
      try
       { System.out.println(args[0] + " " + args[1] + " " + args[2]);
       }
      catch(ArrayIndexOutOfBoundsException e)
       { System.out.println("Reading non-existent array element");
       }
    }
 }
```

Test this program with three arguments and no arguments.

THIRD TASK

Key in the following program in which Thread.sleep(3000L); is supposed to pause the program for 3000 milliseconds. The L in the argument 3000L indicates a long integer (64-bit) constant.

```
public class HelloA
```

```
 { public static void main(String[] args)
    { System.out.println("Hello");
      Thread.sleep(3000L);
      System.out.println("World");
    }
 }
```

This program won't compile. The compiler complains:

```
HelloA.java:4: Exception java.lang.InterruptedException
must be caught, or it must be declared in the throws clause
of this method.

Thread.sleep(3000L);
           ^
1 error
```

FOURTH TASK

Embed `Thread.sleep(3000L);` in a try-clause and catch the InterruptedException as in the following...

```
public class HelloB
 { public static void main(String[] args)
    { System.out.println("Hello");
      try
       { Thread.sleep(3000L);
       }
      catch (InterruptedException e)
       { System.out.println("Interrupted while asleep");
       }
      System.out.println("World");
    }
 }
```

Compile and run this program.