COMP0004 Object-Oriented Programming
Example Answer for Programming Exercises 3 Question 2

Q2.

A Range represents a sequence, or range, of integers, for example 1 to 10, 3 to 15, or -6 to 3. Write a `class Range` in Java with the following methods:

- a constructor
  `public Range(int lower, int upper)`
  The range includes all the integers from the lower limit to the upper limit inclusive. For example, the range 2 to 4 includes 2, 3 and 4. The lower limit must be lower than the upper limit otherwise the constructor will throw an exception.
- A method `int getLower()` to return the lower limit.
- A method `int getUpper()` to return the upper limit.
- A method `boolean contains(int n)` that returns true if the parameter value n is in the range, false otherwise.
- A method getValues() that returns an `ArrayList<Integer>` containing all the values in the range in order.

Write a second class, with a main method, to use `Range` objects and confirm they work correctly. Note that your Range class should not do any input or output, or have a main method.

Example Answer:

The first thing is to review the specification given in the question to make sure the right kind of range gets implemented. A range can be exclusive meaning that the upper limit is not included in the range, or inclusive so the upper limit is included. For example, the exclusive range 1 to 4 contains 1, 2, 3 but not 4, while the inclusive range contains 1, 2, 3 and 4. Range representations in programming are typically exclusive by default, the range class in Python being a good example. Interestingly, the standard Java class library does not have a Range class, although there is a way to define a range using streams.

Fortunately the question makes clear that an inclusive range is being implemented as it gives a clear example of the range 2 to 4 containing 2, 3 and 4. Note, also, that the question specifies that the lower limit must be *lower* than the upper limit, meaning that a range must have at least two values.

The actual Range class is straightforward to write, with two instance variables tom store the lower and upper limits. The constructor and method bodies are then easy to fill in.

```
import java.util.ArrayList;

public class Range
{
  private int lower;
  private int upper;

  public Range(int lower, int upper) throws IllegalArgumentException
  {
    if (lower >= upper) throw new IllegalArgumentException();
```

```
    this.lower = lower;
    this.upper = upper;
  }

  public int getUpper()
  {
    return upper;
  }

  public int getLower()
  {
    return lower;
  }

  public boolean contains(int n)
  {
    return (n >= lower) && (n <= upper);
  }

  public ArrayList<Integer> getValues()
  {
    ArrayList<Integer> values = new ArrayList<>();
    for (int i = lower; i <= upper; i++) values.add(i);
    return values;
  }
}
```

As specified the constructor throws an exception if the lower and upper values provided don't represent a valid range. Note that negative numbers are valid providing lower is less than upper. The exception is `IllegalArgumentException`, which is an exception class found in the standard class library. The JavaDoc describes this class as "Thrown to indicate that a method has been passed an illegal or inappropriate argument.", which matches the use made of it here, and there is no need to write a new exception class.

The second class specified to use `Range` objects and confirm that the `Range` class works is as follows:

```
import java.util.ArrayList;

public class Main
{
  private Input in = new Input();

  private int inputInteger(String prompt)
  {
    int n = 0;
    while (true)
    {
      System.out.print(prompt);
      if (in.hasNextInt())
      {
        n = in.nextInt();
        break;
      }
      in.nextLine();
      System.out.println("You did not type an integer, try again.");
    }
```

```
  return n;
}

private Range createRange()
{
  int lower = inputInteger("Enter the lower value of the range: ");
  int upper = inputInteger("Enter the upper value of the range: ");
  return new Range(lower, upper);
}

private void displayRange(Range range)
{
  System.out.println("The lower value of the range is: " +
                     range.getLower());
  System.out.println("The upper value of the range is: " +
                     range.getUpper());
  ArrayList<Integer> values = range.getValues();
  System.out.println("The values in the range are:");
  for (int i : values)
  {
    System.out.print(i + " ");
  }
  System.out.println();
}

private void checkRangeContainsValues(Range range)
{
  boolean missing = false;
  for (int n = range.getLower(); n <= range.getUpper(); n++)
  {
    if (!range.contains(n))
    {
      System.out.println(n + " is missing!");
      missing = true;
    }
  }
  if (!missing)
  {
    System.out.println("All the expected values are in the range!");
  }
}

private void run()
{
  Range range;
  try
  {
    range = createRange();
  }
  catch (IllegalArgumentException e)
  {
    System.out.println("The range is not valid.");
    return;
  }
  displayRange(range);
  checkRangeContainsValues(range);
```

```
  }

  public static void main(String[] args)
  {
    new Main().run();
  }
}
```