Solutions Manual

Data Structures and Algorithms in Java, 5th edition

M. T. Goodrich and R. Tamassia

Chapter 1

Reinforcement

Solution R-1.3

Since, after the clone, $A[4]$ and $B[4]$ are both pointing to the same GameEntry object, $B[4]$.score is now 550.

Solution R-1.7

```
for (int i=1; i<=58; i++) {
   wallet[0].chargeIt((double)i);
   wallet[1].chargeIt(2.0*i);
   wallet[2].chargeIt((double)3*i);
}
```

This change will cause credit card 2 to go over its limit.

Solution R-1.10

```
public boolean isMultiple(long n, long m) {
   if (n%m == 0)
      return true;
   else
      return false;
}
```

Solution R-1.12

```
public integer sumToN(integer N) {
   if (n == 1)
      return 1;
   else
      return (n−1 + sumToN(n−1));
}
```

Creativity

Solution C-1.3

```
public boolean allDistinct( int[] ints ) {
  for( int i = 0; i < ints.length−1; i++ )
    // we don't need to test numbers at indices already checked
    for (int j = i+1; j < ints.length; j++)
      if (ints[i]==ints[j]) return false;
  return true;
}
```

```
import java.util.Vector;

public class CharPermutation{

  private void Permute( Vector bag, Vector permutation ) {

    // When the bag is empty, a full permutation exists
    if( bag.isEmpty() ) {
      System.out.println( permutation );
    }
    else {

      // For each element left in the bag
      for( int i = 0; i < bag.size(); i++ ) {

        // Take the element out of the bag
        //      and put it at the end of the permutation
        Character c = (Character) bag.elementAt( i );
        bag.removeElementAt( i );
        permutation.addElement( c );

        // Permute the rest of the bag
        this.Permute( bag, permutation );

        // Take the element off the permutation
        //      and put it back in the bag
        permutation.removeElementAt( permutation.size() - 1 );
        bag.insertElementAt( c, i );
      }
    }
  }
```

Solution C-1.5

Here is a possible solution:

```
  public void PrintPermutations( char[] elements ) {
    Vector bag = new Vector();
    Vector permutation = new Vector();

    for( int i = 0; i < elements.length; i++ ) {
      bag.addElement( new Character( elements[i] ) );
    }

    this.Permute( bag, permutation );

  }


  public static void main( String[] args ) {
    char[] elements = { 'c', 'a', 'r', 'b', 'o', 'n' };
    new CharPermutation().PrintPermutations( elements );
  }
}
```

Solution C-1.7

```
class ArraySizeException extends Exception {
    public ArraySizeException() { super(); }
    public ArraySizeException( String s ) { super( s ); }
}

public int[] compute( int[] a, int[] b ) throws ArraySizeException {
    if( a.length != b.length ) {
      throw new ArraySizeException( "arrays must have same length" );
    }

    int[] c = new int[a.length];
    for( int i = 0; i < a.length; i++ ) {
      c[i]= a[i] * b[i];
    }

    return c;
}
```