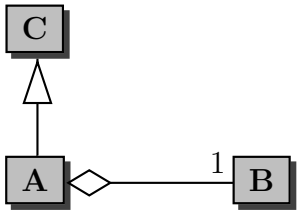
CSE 1030 Introduction to Computer Science II
Sample Test 3

1

The classes `A`, `B` and `C` are related as follows.



How are these two relations reflected in the implementation of class `A`?

2

Consider the classes `Named` and `Contact`, the code of which can be found at the end of the test. Consider the following snippet of client code.

```
String name = "Franck";
String address = "franck@cse.yorku.ca";
Contact contact = new Contact(name, address);
```

Draw the memory diagram corresponding to the execution having reached the end of the third line of the above client code.

3

Implement the method `sumOfSquares` of the class `MyMath`. For a given integer n, with $n \geq 1$, this method returns the integer

$$\sum_{i=1}^{n} i^2 = 1 + 4 + \cdots + (n-1)^2 + n^2.$$

Only **recursive** solutions will receive marks.

```
/**
 * Returns 1 + 4 + ... + (n-1)*(n-1) + n*n
 *
 * @param n an integer.
 * @pre. n >= 1
 * @return the corresponding sum of squares.
 */
public static int sumOfSquares(int n)
{
```

4

The method `areAllEven` of the class `MyList` checks if all elements of the given list are even.

```
/**
 * Tests if all the elements of the given list are even.
 *
 * @param list a list of integers.
 * @pre. list != null and list does not contain null
 * @return true if all the elements of the given list are even, false otherwise.
 */
public static boolean areAllEven(List<Integer> list)
{
   boolean areAllEven;
   if (list.size() == 0)
   {
      areAllEven = true;
   }
   else
   {
      Integer first = list.get(0);
      List<Integer> rest = new ArrayList<Integer>(list.subList(1, list.size()));
      areAllEven = (first % 2 == 0) && MyList.areAllEven(rest);
   }
   return areAllEven;
}
```

(a) Prove the recursive method correct.

(b) Prove the recursive method terminates.

```
/**
 * This class represents an object with a name.
 */
public abstract class Named
{
    private String name;

    /**
     * Creates an object with the given name.
     *
     * @param name the name of this object.
     */
    public Named(String name)
    {
        super();
        this.name = name;
    }
}


/**
 * This class represents a contact.
 * A contact has a name and an address.
 */
public class Contact extends Named
{
    private String address;

    /**
     * Creates a contact with the given name and address.
     *
     * @param name the name of this contact.
     * @param address the address of this contact.
     */
    public Contact(String name, String address)
    {
        super(name);
        this.address = address;
    }
}
```