COMPUTER SCIENCE A
SECTION II
Time—1 hour and 45 minutes
Number of questions—4
Percent of total score—50

Directions: SHOW ALL YOUR WORK. REMEMBER THAT PROGRAM SEGMENTS ARE TO BE WRITTEN IN JAVA.

Notes:
- Assume that the classes listed in the appendices have been imported where appropriate.
- Unless otherwise noted in the question, assume that parameters in method calls are not `null` and that methods are called only when their preconditions are satisfied.
- In writing solutions for each question, you may use any of the accessible methods that are listed in classes defined in that question. Writing significant amounts of code that can be replaced by a call to one of these methods may not receive full credit.

1. A mountain climbing club maintains a record of the climbs that its members have made. Information about a climb includes the name of the mountain peak and the amount of time it took to reach the top. The information is contained in the `ClimbInfo` class as declared below.

```
public class ClimbInfo
{
  /** Creates a ClimbInfo object with name peakName and time climbTime.
   *  @param peakName the name of the mountain peak
   *  @param climbTime the number of minutes taken to complete the climb
   */
  public ClimbInfo(String peakName, int climbTime)
  {  /* implementation not shown */  }


  /** @return the name of the mountain peak
   */
  public String getName()
  {  /* implementation not shown */  }


  /** @return the number of minutes taken to complete the climb
   */
  public int getTime()
  {  /* implementation not shown */  }

  // There may be instance variables, constructors, and methods that are not shown.
}
```

The `ClimbingClub` class maintains a list of the climbs made by members of the club. The declaration of the `ClimbingClub` class is shown below. You will write two different implementations of the `addClimb` method. You will also answer two questions about an implementation of the `distinctPeakNames` method.

```
public class ClimbingClub
{
  /** The list of climbs completed by members of the club.
   *  Guaranteed not to be null. Contains only non-null references.
   */
  private List<ClimbInfo> climbList;


  /** Creates a new ClimbingClub object. */
  public ClimbingClub()
  {  climbList = new ArrayList<ClimbInfo>();  }


  /** Adds a new climb with name peakName and time climbTime to the list of climbs.
   *  @param peakName the name of the mountain peak climbed
   *  @param climbTime the number of minutes taken to complete the climb
   */
  public void addClimb(String peakName, int climbTime)
  {  /* to be implemented in part (a) with ClimbInfo objects in the order they were added  */
     /* to be implemented in part (b) with ClimbInfo objects in alphabetical order by name */
  }


  /** @return the number of distinct names in the list of climbs */
  public int distinctPeakNames()
  {  /* implementation shown in part (c) */  }

  // There may be instance variables, constructors, and methods that are not shown.
}
```

Part (a) begins on page 4.

(a) Write an implementation of the `ClimbingClub` method `addClimb` that stores the `ClimbInfo` objects in the order they were added. This implementation of `addClimb` should create a new `ClimbInfo` object with the given name and time. It appends a reference to that object to the end of `climbList`. For example, consider the following code segment.

```
ClimbingClub hikerClub = new ClimbingClub();
hikerClub.addClimb("Monadnock", 274);
hikerClub.addClimb("Whiteface", 301);
hikerClub.addClimb("Algonquin", 225);
hikerClub.addClimb("Monadnock", 344);
```

When the code segment has completed executing, the instance variable `climbList` would contain the following entries.

Peak Name	"Monadnock"	"Whiteface"	"Algonquin"	"Monadnock"
Climb Time	274	301	225	344

Information repeated from the beginning of the question

```
public class ClimbInfo

public ClimbInfo(String peakName, int climbTime)
public String getName()
public int getTime()


public class ClimbingClub

private List<ClimbInfo> climbList
public void addClimb(String peakName, int climbTime)
public int distinctPeakNames()
```

WRITE YOUR SOLUTION ON THE NEXT PAGE.

Complete method `addClimb` below.

```
/** Adds a new climb with name peakName and time climbTime to the list of climbs.
 *  @param peakName the name of the mountain peak climbed
 *  @param climbTime the number of minutes taken to complete the climb
 *  Postcondition: The new entry is at the end of climbList;
 *                 The order of the remaining entries is unchanged.
 */
public void addClimb(String peakName, int climbTime)
```

Part (b) begins on page 6.

AP® COMPUTER SCIENCE A
2012 SCORING GUIDELINES

Question 1: Climbing Club

Part (a)	`addClimb` (append)	**2 points**

Intent: *Create new* `ClimbInfo` *using data from parameters and append to* `climbList`

+1 Creates new `ClimbInfo` object using parametric data correctly

+1 Appends the created object to `climbList`
(*no bounds error and no destruction of existing data*)
(*point not awarded if inserted more than once*)

Part (b)	`addClimb` (alphabetical)	**6 points**

Intent: *Create new* `ClimbInfo` *object using data from parameters and insert into* `climbList`, *maintaining alphabetical order*

+1 Creates new `ClimbInfo` object(s), using parametric data correctly

+1 Compares `peakName` value with value retrieved from object in list (*must use* `getName`)

+1 Inserts object into list based on a comparison (other than equality) with object in list
(*point not awarded if inserted more than once*)

+1 Compares parametric data with all appropriate entries in `climbList` (*no bounds error*)

+1 Inserts new `ClimbInfo` object into `climbList` (*no destruction of existing data*)

+1 Inserts new `ClimbInfo` object into `climbList` once and only once in maintaining alphabetical order (*no destruction of existing data*)

Part (c)	analysis	**1 point**

Intent: *Analyze behavioral differences between* **append** *and* **alphabetical** *versions of* `addClimb`

+1 (i) NO (ii) YES Both must be answered correctly

Question-Specific Penalties

-1 (z) Attempts to return a value from `addClimb`

AP® COMPUTER SCIENCE A
2012 CANONICAL SOLUTIONS

Question 1: Climbing Club

Part (a):

```
public void addClimb(String peakName, int climbTime) {
   this.climbList.add(new ClimbInfo(peakName, climbTime));
}
```

Part (b):

```
public void addClimb(String peakName, int climbTime) {
   for (int i = 0; i < this.climbList.size(); i++) {
      if (peakName.compareTo(this.climbList.get(i).getName()) <= 0) {
         this.climbList.add(i, new ClimbInfo(peakName, climbTime));
         return;
      }
   }
   this.climbList.add(new ClimbInfo(peakName, climbTime));
}
```

Part (c):

NO

YES

These canonical solutions serve an expository role, depicting general approaches to solution. Each reflects only one instance from the infinite set of valid solutions. The solutions are presented in a coding style chosen to enhance readability and facilitate understanding.