2026

AP® Computer Science A

Free-Response Questions

© 2026 College Board. College Board, Advanced Placement, AP, AP Central, and the acorn logo are registered trademarks of College Board. Visit College Board on the web: collegeboard.org.

AP Central is the official online home for the AP Program: apcentral.collegeboard.org.

COMPUTER SCIENCE A
SECTION II
TIME – 1 HOUR AND 30 MINUTES

Directions:

Section II has 4 free-response questions and lasts 1 hour and 30 minutes.

All program segments must be written in Java.

Java Quick Reference information for programming questions can be used throughout the exam. A digital version is available in this application.

Assume that the classes listed in the Java Quick Reference 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 listed in classes defined in that question. Writing significant amounts of code that can be replaced by a call to one of these methods will not receive full credit.

You may use the available paper for scratch work, but credit will only be given for responses entered in this application. Text you enter as an annotation will **not** be included as part of your answer.

You can go back and forth between questions in this section until time expires. The clock will turn red when 5 minutes remain—**the proctor will not give you any time updates or warnings.**

Note: This exam was originally administered digitally. It is presented here in a format optimized for teacher and student use in the classroom.

During the AP Exam administration, students have access to reference information. To see the reference information for this course, please visit AP Central: https://apcentral.collegeboard.org/exam-administration-ordering-scores/administering-exams/subject-specific/reference-information

© 2026 College Board.
Visit College Board on the web: **collegeboard.org.**

3. The `CourseRecord` class is used to store information about a student enrolled in a course. A partial definition of the `CourseRecord` class is shown.

```
public class CourseRecord
{
   /**
    *  Returns a unique ID for the student associated with this
    *  CourseRecord
    */
   public String getStudentID()
   { /* implementation not shown */ }

   /**
    *  Returns a nonnegative integer representing the number of times the
    *  student associated with this CourseRecord has been absent in the
    *  course
    */
   public int getAbsences()
   { /* implementation not shown */ }

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

© 2026 College Board.
Visit College Board on the web: **collegeboard.org.**

The `Attendance` class maintains two `ArrayLists` of `CourseRecord` objects, named `historyList` and `mathList`. A partial definition of the `Attendance` class is shown.

```
public class Attendance
{
   /* The students enrolled in a history course */
   private ArrayList<CourseRecord> historyList;

   /* The students enrolled in a math course */
   private ArrayList<CourseRecord> mathList;

   /**
    *  Returns the number of students who are enrolled in both
    *  the history course and the math course but have more
    *  absences in the history course than the math course
    *  Preconditions:
    *        No student ID appears multiple times in historyList.
    *        No student ID appears multiple times in mathList.
    *        historyList and mathList do not contain any null elements.
    *  Postcondition:
    *        historyList and mathList are unchanged.
    */
   public int moreHistoryThanMathAbsences()
   { /* to be implemented */ }

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

© 2026 College Board.
Visit College Board on the web: **collegeboard.org.**

Write the `Attendance` method `moreHistoryThanMathAbsences`. The method should return the number of students who are enrolled in both the history course and the math course but have more absences in the history course than the math course.

For example, suppose `historyList` and `mathList` have the following contents.

`historyList`:

Student ID	"rc29"	"br98"	"dr03"	"ot32"	"sq98"	"ry00"
Number of Absences	1	1	2	2	3	1

`mathList`:

Student ID	"fr27"	"sq98"	"dr03"	"dk12"	"ot32"	"js33"	"ry00"
Number of Absences	2	1	2	1	1	0	3

In this example, there are four student IDs (`"dr03"`, `"ot32"`, `"sq98"`, and `"ry00"`) that appear in both `historyList` and `mathList`. Of these, only `"ot32"` and `"sq98"` have more absences in the history course than the math course, so the `moreHistoryThanMathAbsences` method should return `2`.

Complete method `moreHistoryThanMathAbsences`.

```
/**
 *  Returns the number of students who are enrolled in both
 *  the history course and the math course but have more
 *  absences in the history course than the math course
 *  Preconditions:
 *        No student ID appears multiple times in historyList.
 *        No student ID appears multiple times in mathList.
 *        historyList and mathList do not contain any null elements.
 *  Postcondition:
 *        historyList and mathList are unchanged.
 */
public int moreHistoryThanMathAbsences()
```

© 2026 College Board.
Visit College Board on the web: **collegeboard.org.**