CollegeBoard AP®

## AP® Computer Science A

# Practice Exam and Notes

For the
**Spring 2020**
**Exam**

**This exam may not be posted on school or personal websites, nor electronically redistributed for any reason.** This exam is provided by College Board for AP Exam preparation. Teachers are permitted to download the materials and make copies to use with their students in a classroom setting only. To maintain the security of this exam, teachers should collect all materials after their administration and keep them in a secure location.

**Further distribution of these materials outside of the secure College Board site disadvantages teachers who rely on uncirculated questions for classroom testing.** Any additional distribution is in violation of College Board's copyright policies and may result in the termination of Practice Exam access for your school as well as the removal of access to other online services such as the AP Teacher Community and Online Score Reports.

© 2020 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**.

# Contents

**I. Practice Exam**

Exam Content and Format . . . . . 4

Administering the Practice Exam . . . . . 4

Answer Sheet for Multiple-Choice Section . . . . . 6

AP® Computer Science A Practice Exam . . . . . 7

**II. Notes on the Practice Exam**

Multiple-Choice Section . . . . . 82

Answer Key and Question Alignment to Course Framework . . . . . 116

Free-Response Section . . . . . 119

**Contact Us** . . . . . 128

# Practice Exam

## Exam Content and Format

The AP Computer Science A Exam is 3 hours long. There are two sections:

- Section I is 1 hour, 30 minutes and consists of 40 multiple-choice questions, accounting for 50 percent of the final score.
- Section II is 1 hour, 30 minutes and consists of 4 free-response questions accounting for 50 percent of the final score.

## Administering the Practice Exam

This section contains instructions for administering the AP Computer Science A Practice Exam. You may wish to use these instructions to create an exam situation that resembles an actual administration. If so, read the indented, boldface directions to the students; all other instructions are for administering the exam and need not be read aloud. Before beginning testing, have all exam materials ready for distribution. These include test booklets and answer sheets. (Reminder: Final instructions for every AP Exam are published in the AP Exam Instructions book.)

**SECTION I: Multiple Choice**

When you are ready to begin Section I, say:

> **Section I is the multiple-choice portion of the exam. Mark all of your responses on your answer sheet, one response per question. If you need to erase, do so carefully and completely. Your score on the multiple-choice section will be based solely on the number of questions answered correctly.**
>
> **You have 1 hour and 30 minutes for this part. Open your Section I booklet and begin.**

Note Start Time _____ . Note Stop Time _____ . After 1 hour and 20 minutes, say:

> **There are 10 minutes remaining.**

After 10 minutes, say:

> **Stop working. I will now collect your Section I booklet and multiple-choice answer sheet.**

There is a 10-minute break between Sections I and II.

**SECTION II: Free Response, Questions**

After the break, say:

> **Section II is the free-response portion of the exam.**
>
> **You have 1 hour and 30 minutes to complete Section II. You may use any blank space of the page the questions or documents are printed on to organize your answers and for scratch work. You must write your answers in the answer booklet for free-response questions. At the top of each page in your booklet you must fill in the circle that indicates the question number you are answering. Open your Section II booklet and begin.**

Note Start Time _____ . Note Stop Time _____ . After 1 hour and 20 minutes, say

> **There are 10 minutes remaining.**

After 10 minutes, say:

> **Stop working and close your exam booklet. Put your exam booklet on your desk, face up. Remain in your seat, without talking, while the exam materials are collected.**

Collect a Section II booklet from each student and check that each student wrote his or her answers on the pages corresponding to each question. Then say:

> **The exam is over. You are now dismissed.**

Name: _____

# AP® Computer Science A
# Answer Sheet
# for Multiple-Choice Section

| No. | Answer |
|---|---|
| 1 | |
| 2 | |
| 3 | |
| 4 | |
| 5 | |
| 6 | |
| 7 | |
| 8 | |
| 9 | |
| 10 | |
| 11 | |
| 12 | |
| 13 | |
| 14 | |
| 15 | |
| 16 | |
| 17 | |
| 18 | |
| 19 | |
| 20 | |

| No. | Answer |
|---|---|
| 21 | |
| 22 | |
| 23 | |
| 24 | |
| 25 | |
| 26 | |
| 27 | |
| 28 | |
| 29 | |
| 30 | |
| 31 | |
| 32 | |
| 33 | |
| 34 | |
| 35 | |
| 36 | |
| 37 | |
| 38 | |
| 39 | |
| 40 | |

# AP® Computer Science A Exam

## SECTION I: Multiple Choice

**DO NOT OPEN THIS BOOKLET UNTIL YOU ARE TOLD TO DO SO.**

### At a Glance

**Total Time**
1 hour and 30 minutes
**Number of Questions**
40
**Percent of Total Score**
50%
**Writing Instrument**
Pencil required
**Electronic Device**
None allowed

### Instructions

The Java Quick Reference is located inside the front cover of this booklet.

Section I of this exam contains 40 multiple-choice questions.

Indicate all of your answers to the multiple-choice questions on the answer sheet. No credit will be given for anything written in this exam booklet, but you may use the booklet for notes or scratch work.

Use your time effectively, working as quickly as you can without losing accuracy. Do not spend too much time on any one question. Go on to other questions and come back to the ones you have not answered if you have time. It is not expected that everyone will know the answers to all of the multiple-choice questions.

Your total score on the multiple-choice section is based only on the number of questions answered correctly. Points are not deducted for incorrect answers or unanswered questions.

# Java Quick Reference

*Accessible methods from the Java library that may be included in the exam*

| Class Constructors and Methods | Explanation |
|---|---|
| **String Class** | |
| `String(String str)` | Constructs a new `String` object that represents the same sequence of characters as `str` |
| `int length()` | Returns the number of characters in a `String` object |
| `String substring(int from, int to)` | Returns the substring beginning at index `from` and ending at index `to - 1` |
| `String substring(int from)` | Returns `substring(from, length())` |
| `int indexOf(String str)` | Returns the index of the first occurrence of `str`; returns `-1` if not found |
| `boolean equals(String other)` | Returns `true` if `this` is equal to `other`; returns `false` otherwise |
| `int compareTo(String other)` | Returns a value `<0` if `this` is less than `other`; returns zero if `this` is equal to `other`; returns a value `>0` if `this` is greater than `other` |
| **Integer Class** | |
| `Integer(int value)` | Constructs a new `Integer` object that represents the specified `int` value |
| `Integer.MIN_VALUE` | The minimum value represented by an `int` or `Integer` |
| `Integer.MAX_VALUE` | The maximum value represented by an `int` or `Integer` |
| `int intValue()` | Returns the value of this `Integer` as an `int` |
| **Double Class** | |
| `Double(double value)` | Constructs a new `Double` object that represents the specified `double` value |
| `double doubleValue()` | Returns the value of this `Double` as a `double` |
| **Math Class** | |
| `static int abs(int x)` | Returns the absolute value of an `int` value |
| `static double abs(double x)` | Returns the absolute value of a `double` value |
| `static double pow(double base, double exponent)` | Returns the value of the first parameter raised to the power of the second parameter |
| `static double sqrt(double x)` | Returns the positive square root of a `double` value |
| `static double random()` | Returns a `double` value greater than or equal to `0.0` and less than `1.0` |
| **ArrayList Class** | |
| `int size()` | Returns the number of elements in the list |
| `boolean add(E obj)` | Appends `obj` to end of list; returns `true` |
| `void add(int index, E obj)` | Inserts `obj` at position `index` (`0 <= index <= size`), moving elements at position `index` and higher to the right (adds 1 to their indices) and adds 1 to size |
| `E get(int index)` | Returns the element at position `index` in the list |
| `E set(int index, E obj)` | Replaces the element at position `index` with `obj`; returns the element formerly at position `index` |
| `E remove(int index)` | Removes element from position `index`, moving elements at position `index + 1` and higher to the left (subtracts 1 from their indices) and subtracts 1 from size; returns the element formerly at position `index` |
| **Object Class** | |
| `boolean equals(Object other)` | |
| `String toString()` | |

**NO TEST MATERIAL ON THIS PAGE**

**COMPUTER SCIENCE A**

**SECTION I**

**Time—1 hour and 30 minutes**

**40 Questions**

**Directions:** Determine the answer to each of the following questions or incomplete statements, using the available space for any necessary scratch work. Then decide which is the best of the choices given and then enter the letter in the corresponding space on the answer sheet. No credit will be given for anything written in the exam booklet. Do not spend too much time on any one problem.

Notes:

- Assume that the classes listed in the Java Quick Reference have been imported where appropriate.
- Assume that declarations of variables and methods appear within the context of an enclosing class.
- Assume that method calls that are not prefixed with an object or class name and are not shown within a complete class definition appear within the context of an enclosing class.
- 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.

**GO ON TO THE NEXT PAGE.**

1. Consider the following code segment.

```
int a = 3 + 2 * 3;
int b = 4 + 3 / 2;
int c = 7 % 4 + 3;
double d = a + b + c;
```

What is the value of `d` after the code segment is executed?

(A) `14.0`

(B) `18.0`

(C) `20.0`

(D) `20.5`

(E) `26.0`

---

2. Consider the following code segment. Assume `num` is a properly declared and initialized `int` variable.

```
if (num > 0)
{
   if (num % 2 == 0)
   {
      System.out.println("A");
   }
   else
   {
      System.out.println("B");
   }
}
```

Which of the following best describes the result of executing the code segment?

(A) When `num` is a negative odd integer, `"B"` is printed; otherwise, `"A"` is printed.

(B) When `num` is a negative even integer, `"B"` is printed; otherwise, nothing is printed.

(C) When `num` is a positive even integer, `"A"` is printed; otherwise, `"B"` is printed.

(D) When `num` is a positive even integer, `"A"` is printed; when `num` is a positive odd integer, `"B"` is printed; otherwise, nothing is printed.

(E) When `num` is a positive odd integer, `"A"` is printed; when `num` is a positive even integer, `"B"` is printed; otherwise, nothing is printed.

**GO ON TO THE NEXT PAGE.**

3. Consider the method `getHours`, which is intended to calculate the number of hours that a vehicle takes to travel between two *mile markers* on a highway if the vehicle travels at a constant speed of 60 miles per hour. A mile marker is a sign showing the number of miles along a road between some fixed location (for example, the beginning of a highway) and the current location.

The following table shows two examples of the intended behavior of `getHours`, based on the `int` parameters `marker1` and `marker2`.

| marker1 | marker2 | **Return Value** |
|---|---|---|
| 100 | 220 | 2.0 |
| 100 | 70 | 0.5 |

Consider the following implementation of `getHours`.

```
public static double getHours(int marker1, int marker2)
{
   /* missing statement */
   return hours;
}
```

Which of the following statements can replace `/* missing statement */` so `getHours` works as intended?

(A) `double hours = (Math.abs(marker1) - Math.abs(marker2)) / 60.0;`

(B) `double hours = Math.abs(marker1 - marker2 / 60.0);`

(C) `double hours = Math.abs(marker1 - marker2) / 60.0;`

(D) `double hours = Math.abs((marker1 - marker2) / 60);`

(E) `double hours = (double) (Math.abs(marker1 - marker2) / 60);`

**GO ON TO THE NEXT PAGE.**

4. Consider the following method.

```
public static void message(int a, int b, int c)
{
   if (a < 10)
   {
      if (b < 10)
      {
         System.out.print("X");
      }
      System.out.print("Y");
   }
   if (c < 10)
   {
      if (b > 10)
      {
         System.out.print("Y");
      }
      else
      {
         System.out.print("Z");
      }
   }
}
```

What is printed as a result of the call `message(5, 15, 5)` ?

(A) `XY`

(B) `XYZ`

(C) `Y`

(D) `YY`

(E) `Z`

**GO ON TO THE NEXT PAGE.**

5. Consider the following class definition.

```
public class Bird
{
   private String species;
   private String color;
   private boolean canFly;

   public Bird(String str, String col, boolean cf)
   {
      species = str;
      color = col;
      canFly = cf;
   }
}
```

Which of the following constructors, if added to the `Bird` class, will cause a compilation error?

(A)
```
public Bird()
{
   species = "unknown";
   color = "unknown";
   canFly = false;
}
```

(B)
```
public Bird(boolean cf)
{
   species = "unknown";
   color = "unknown";
   canFly = cf;
}
```

(C)
```
public Bird(String col, String str)
{
   species = str;
   color = col;
   canFly = false;
}
```

**GO ON TO THE NEXT PAGE.**

(D)
```
public Bird(boolean cf, String str, String col)
{
   species = str;
   color = col;
   canFly = cf;
}
```

(E)
```
public Bird(String col, String str, boolean cf)
{
   species = str;
   color = col;
   canFly = cf;
}
```

---

6. Which of the following expressions evaluate to `3.5` ?

   I. `(double) 2 / 4 + 3`

   II. `(double) (2 / 4) + 3`

   III. `(double) (2 / 4 + 3)`

(A) I only

(B) III only

(C) I and II only

(D) II and III only

(E) I, II, and III

**GO ON TO THE NEXT PAGE.**

7. Consider the following code segment.

```
int num = /* initial value not shown */;
boolean b1 = true;
if (num > 0)
{
   if (num >= 100)
   {
      b1 = false;
   }
}
else
{
   if (num >= -100)
   {
      b1 = false;
   }
}
```

Which of the following statements assigns the same value to `b2` as the code segment assigns to `b1` for all values of `num` ?

(A) `boolean b2 = (num > -100) && (num < 100);`

(B) `boolean b2 = (num > -100) || (num < 100);`

(C) `boolean b2 = (num < -100) || (num > 100);`

(D) `boolean b2 = (num < -100) && (num > 0 || num < 100);`

(E) `boolean b2 = (num < -100) || (num > 0 && num < 100);`

**GO ON TO THE NEXT PAGE.**

8. Consider the following class definition.

```
public class Points
{
   private double num1;
   private double num2;

   public Points(int n1, int n2)            // Line 6
   {
      num1 = n1;                            // Line 8
      num2 = n2;                            // Line 9
   }

   public void incrementPoints(int value)   // Line 12
   {
      n1 += value;                          // Line 14
      n2 += value;                          // Line 15
   }
}
```

The class does not compile. Which of the following identifies the error in the class definition?

(A) In line 6, the `Points` constructor must have a `void` return type.

(B) In lines 8 and 9, `int` values cannot be assigned to `double` variables.

(C) In line 12, the `incrementPoints` method must have a non-`void` return type.

(D) In lines 14 and 15, the variables `n1` and `n2` are not defined.

(E) In lines 14 and 15, the variable `value` is not defined.

**GO ON TO THE NEXT PAGE.**

9. Consider the following code segment.

```
ArrayList<Integer> numList = new ArrayList<Integer>();

numList.add(3);
numList.add(2);
numList.add(1);
numList.add(1, 0);
numList.set(0, 2);

System.out.print(numList);
```

What is printed by the code segment?

(A) `[1, 3, 0, 1]`

(B) `[2, 0, 2, 1]`

(C) `[2, 0, 2, 3]`

(D) `[2, 3, 2, 1]`

(E) `[3, 0, 0, 1]`

---

10. Consider the following method.

```
public static void printSome(int num1, int num2)
{
   for (int i = 0; i < num1; i++)
   {
      if (i % num2 == 0 && i % 2 == 0)
      {
         System.out.print(i + " ");
      }
   }
}
```

Which of the following method calls will cause `"0 10 "` to be printed?

(A) `printSome(0, 20)`

(B) `printSome(5, 10)`

(C) `printSome(10, 5)`

(D) `printSome(20, 5)`

(E) `printSome(25, 5)`

**GO ON TO THE NEXT PAGE.**

11. Which of the following code segments produces the output `"987654321"` ?

(A)
```
int num = 10;
while (num > 0)
{
   System.out.print(num);
   num--;
}
```

(B)
```
int num = 10;
while (num >= 0)
{
   System.out.print(num);
   num--;
}
```

(C)
```
int num = 10;
while (num > 1)
{
   num--;
   System.out.print(num);
}
```

(D)
```
int num = 10;
while (num >= 1)
{
   num--;
   System.out.print(num);
}
```

(E)
```
int num = 0;
while (num <= 9)
{
   System.out.print(10 - num);
   num++;
}
```

**GO ON TO THE NEXT PAGE.**

12. Consider the following class definitions.

```
public class Person
{
   private String name;

   public String getName()
   {  return name;  }
}

public class Book
{
   private String author;
   private String title;
   private Person borrower;

   public Book(String a, String t)
   {
      author = a;
      title = t;
      borrower = null;
   }

   public void printDetails()
   {
      System.out.print("Author: " + author + " Title: " + title);

      if ( /* missing condition */ )
      {
         System.out.println(" Borrower: " + borrower.getName());
      }
   }

   public void setBorrower(Person b)
   {  borrower = b;  }
}
```

**GO ON TO THE NEXT PAGE.**

Which of the following can replace `/* missing condition */` so that the `printDetails` method CANNOT cause a run-time error?

I. `!borrower.equals(null)`

II. `borrower != null`

III. `borrower.getName() != null`

(A) I only

(B) II only

(C) III only

(D) I and II

(E) II and III

---

13. Assume that `a`, `b`, and `c` are `boolean` variables that have been properly declared and initialized. Which of the following `boolean` expressions is equivalent to `!(a && b) || c` ?

(A) `a && b && c`

(B) `a || b || c`

(C) `!a && !b || c`

(D) `!a && !b && c`

(E) `!a || !b || c`

**GO ON TO THE NEXT PAGE.**

14. The following categories are used by some researchers to categorize zip codes as urban, suburban, or rural based on population density.

- An urban zip code is a zip code with more than 3,000 people per square mile.
- A suburban zip code is a zip code with between 1,000 and 3,000 people, inclusive, per square mile.
- A rural zip code is a zip code with fewer than 1,000 people per square mile.

Consider the following method, which is intended to categorize a zip code as urban, suburban, or rural based on the population density of the area included in the zip code.

```
public static String getCategory(int density)
{
    /* missing code */
}
```

Which of the following code segments can replace `/* missing code */` so the `getCategory` method works as intended?

I.
```
String cat;
if (density > 3000)
{
   cat = "urban";
}
else if (density > 999)
{
   cat = "suburban";
}
else
{
   cat = "rural";
}
return cat;
```

II.
```
String cat;
if (density > 3000)
{
   cat = "urban";
}
if (density > 999)
{
   cat = "suburban";
}
cat = "rural";
return cat;
```

**GO ON TO THE NEXT PAGE.**

III.
```
if (density > 3000)
{
   return "urban";
}
if (density > 999)
{
   return "suburban";
}
return "rural";
```

(A) I only

(B) III only

(C) I and II only

(D) I and III only

(E) I, II, and III

---

15. Consider the following code segment. Assume that `a` is greater than zero.

```
int a = /* value not shown */;
int b = a + (int) (Math.random() * a);
```

Which of the following best describes the value assigned to `b` when the code segment is executed?

(A) `a`

(B) `2 * a`

(C) A random integer between `0` and `a - 1`, inclusive

(D) A random integer between `a` and `2 * a`, inclusive

(E) A random integer between `a` and `2 * a - 1`, inclusive

**GO ON TO THE NEXT PAGE.**

16. Consider the following recursive method.

```
public static void stars(int num)
{
   if (num == 1)
   {
      return;
   }

   stars(num - 1);

   for (int i = 0; i < num; i++)
   {
      System.out.print("*");
   }
   System.out.println();
}
```

What is printed as a result of the method call `stars(5)` ?

(A)
```
*****
```

(B)
```
**
***
****
*****
```

(C)
```
*
**
***
****
*****
```

(D)
```
*****
****
***
**
```

(E)
```
*****
****
***
**
*
```

**GO ON TO THE NEXT PAGE.**

17. Consider the following class definitions.

```
public class Hero
{
   private String name;
   private int power;

   public Hero(String n, int p)
   {
      name = n;
      power = p;
   }

   public void powerUp(int p)
   {
      power += p;
   }

   public int showPower()
   {   return power;   }
}

public class SuperHero extends Hero
{
   public SuperHero(String n, int p)
   {
      super(n, p);
   }

   public void powerUp(int p)
   {
      super.powerUp(p * 2);
   }
}
```

The following code segment appears in a class other than `Hero` and `SuperHero`.

```
Hero j = new SuperHero("JavaHero", 50);
j.powerUp(10);
System.out.println(j.showPower());
```

What is printed as a result of executing the code segment?

(A) `10`

(B) `20`

(C) `60`

(D) `70`

(E) `100`

**GO ON TO THE NEXT PAGE.**

18. Consider the following method, which is intended to return the number of *local maximum* values in an array. Local maximum values are array elements that are greater than both adjacent array elements. The first and last elements of an array have only a single adjacent element, so neither the first nor the last array element is counted by this method. For example, an array containing the values `{3, 9, 7, 4, 10, 12, 3, 8}` has two local maximum values: `9` and `12`.

```
public static int countPeaks(int[] data)
{
   int numPeaks = 0;

   for ( /* missing loop header */ )
   {
      if (data[p - 1] < data[p] && data[p] > data[p + 1])
      {
         numPeaks++;
      }
   }
   return numPeaks;
}
```

Which of the following can replace `/* missing loop header */` so the method `countPeaks` works as intended?

(A) `int p = data.length - 1; p > 0; p--`

(B) `int p = 0; p < data.length; p++`

(C) `int p = 0; p < data.length - 1; p++`

(D) `int p = 1; p < data.length; p++`

(E) `int p = 1; p < data.length - 1; p++`

**GO ON TO THE NEXT PAGE.**

19. Consider the following code segment.

```
int[][] values = {{1, 2, 3}, {4, 5, 6}};
int x = 0;

for (int j = 0; j < values.length; j++)
{
   for (int k = 0; k < values[0].length; k++)
   {
      if (k == 0)
      {
         values[j][k] *= 2;
      }
      x += values[j][k];
   }
}
```

What is the value of `x` after the code segment is executed?

(A) 7

(B) 17

(C) 21

(D) 26

(E) 27

**GO ON TO THE NEXT PAGE.**

20. Consider the following class definition.

```
public class Book
{
   private int pages;

   public int getPages()
   {
      return pages;
   }

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

The following code segment is intended to store in `maxPages` the greatest number of pages found in any `Book` object in the array `bookArr`.

```
Book[] bookArr = { /* initial values not shown */ };
int maxPages = bookArr[0].getPages();

for (Book b : bookArr)
{
   /* missing code */
}
```

**GO ON TO THE NEXT PAGE.**

Which of the following can replace `/* missing code */` so the code segment works as intended?

(A)
```
if (b.pages > maxPages)
{
   maxPages = b.pages;
}
```

(B)
```
if (b.getPages() > maxPages)
{
   maxPages = b.getPages();
}
```

(C)
```
if (Book[b].pages > maxPages)
{
   maxPages = Book[b].pages;
}
```

(D)
```
if (bookArr[b].pages > maxPages)
{
   maxPages = bookArr[b].pages;
}
```

(E)
```
if (bookArr[b].getPages() > maxPages)
{
   maxPages = bookArr[b].getPages();
}
```

**GO ON TO THE NEXT PAGE.**

**Questions 21 - 22 refer to the information below.**

Consider the following method.

```
public static String[] strArrMethod(String[] arr)
{
   String[] result = new String[arr.length];

   for (int j = 0; j < arr.length; j++)
   {
      String sm = arr[j];
      for (int k = j + 1; k < arr.length; k++)
      {
         if (arr[k].length() < sm.length())
         {
            sm = arr[k];    // Line 12
         }
      }
      result[j] = sm;
   }
   return result;
}
```

21. Consider the following code segment.

```
String[] testOne = {"first", "day", "of", "spring"};
String[] resultOne = strArrMethod(testOne);
```

What are the contents of `resultOne` when the code segment has been executed?

(A) `{"day", "first", "of", "spring"}`

(B) `{"of", "day", "first", "spring"}`

(C) `{"of", "day", "of", "spring"}`

(D) `{"of", "of", "of", "spring"}`

(E) `{"spring", "first", "day", "of"}`

**GO ON TO THE NEXT PAGE.**

22. Consider the following code segment.

```
String[] testTwo = {"last", "day", "of", "the", "school", "year"};
String[] resultTwo = strArrMethod(testTwo);
```

How many times is the line labeled `//` Line 12 in the `strArrMethod` executed as a result of executing the code segment?

(A) 4 times

(B) 5 times

(C) 6 times

(D) 15 times

(E) 30 times

**GO ON TO THE NEXT PAGE.**

23. Consider the following method, which is intended to print the values in its two-dimensional integer array parameter in row-major order.

```
public static void rowMajor(int[][] arr)
{
    /* missing code */
}
```

As an example, consider the following code segment.

```
int[][] theArray = {{1, 2}, {3, 4}, {5, 6}, {7, 8}};
rowMajor(theArray);
```

When executed, the code segment should produce the following output.

```
1 2 3 4 5 6 7 8
```

**GO ON TO THE NEXT PAGE.**

Which of the following code segments can replace `/* missing code */` so that the `rowMajor` method works as intended?

(A)
```
for (int j : arr)
{
   for (int k : j)
   {
      System.out.print(j + " ");
   }
}
```

(B)
```
for (int j : arr)
{
   for (int k : j)
   {
      System.out.print(k + " ");
   }
}
```

(C)
```
for (int[] j : arr)
{
   for (int k : j)
   {
      System.out.print(j + " ");
   }
}
```

(D)
```
for (int[] j : arr)
{
   for (int k : j)
   {
      System.out.print(k + " ");
   }
}
```

(E)
```
for (int[] j : arr)
{
   for (int k : j)
   {
      System.out.print(arr[k] + " ");
   }
}
```

**GO ON TO THE NEXT PAGE.**

24. Consider the following class definition.

```
public class SomeClass
{
   private int x = 0;
   private static int y = 0;

   public SomeClass(int pX)
   {
      x = pX;
      y++;
   }

   public void incrementY()
   {   y++;  }

   public void incrementY(int inc)
   {   y += inc;  }

   public int getY()
   {   return y;  }
}
```

The following code segment appears in a class other than `SomeClass`.

```
SomeClass first = new SomeClass(10);
SomeClass second = new SomeClass(20);
SomeClass third = new SomeClass(30);
first.incrementY();
second.incrementY(10);
System.out.println(third.getY());
```

What is printed as a result of executing the code segment if the code segment is the first use of a `SomeClass` object?

(A) `0`

(B) `1`

(C) `11`

(D) `14`

(E) `30`

**GO ON TO THE NEXT PAGE.**

25. Consider the following method.

```
public static String rearrange(String str)
{
   String temp = "";

   for (int i = str.length() - 1; i > 0; i--)
   {
      temp += str.substring(i - 1, i);
   }

   return temp;
}
```

What, if anything, is returned by the method call `rearrange("apple")` ?

(A) `"appl"`

(B) `"apple"`

(C) `"elppa"`

(D) `"lppa"`

(E) Nothing is returned due to a run-time error.

**GO ON TO THE NEXT PAGE.**

26. Consider the following two code segments. Assume that the `int` variables `m` and `n` have been properly declared and initialized and are both greater than `0`.

I.
```
for (int i = 0; i < m * n; i++)
{
   System.out.print("A");
}
```

II.
```
for (int j = 1; j <= m; j++)
{
   for (int k = 1; k < n; k++)
   {
      System.out.print("B");
   }
}
```

Assume that the initial values of `m` and `n` are the same in code segment I as they are in code segment II. Which of the following correctly compares the number of times that `"A"` and `"B"` are printed when each code segment is executed?

(A) `"A"` is printed `m` fewer times than `"B"`.

(B) `"A"` is printed `n` fewer times than `"B"`.

(C) `"A"` is printed `m` more times than `"B"`.

(D) `"A"` is printed `n` more times than `"B"`.

(E) `"A"` and `"B"` are printed the same number of times.

**GO ON TO THE NEXT PAGE.**

27. Consider the following statement. Assume that `a` and `b` are properly declared and initialized `boolean` variables.

```
boolean c = (a && b) || (!a && b);
```

Under which of the following conditions will `c` be assigned the value `false` ?

(A) Always

(B) Never

(C) When `a` and `b` have the same value

(D) When `a` has the value `false`

(E) When `b` has the value `false`

---

28. Consider the following method.

```
public static String abMethod(String a, String b)
{
   int x = a.indexOf(b);

   while (x >= 0)
   {
      a = a.substring(0, x) + a.substring(x + b.length());
      x = a.indexOf(b);
   }

   return a;
}
```

What, if anything, is returned by the method call `abMethod("sing the song", "ng")` ?

(A) `"si"`

(B) `"si the so"`

(C) `"si the song"`

(D) `"sig the sog"`

(E) Nothing is returned because a `StringIndexOutOfBoundsException` is thrown.

**GO ON TO THE NEXT PAGE.**

29. Consider the following method.

```
public static int calcMethod(int num)
{
   if (num == 0)
   {
      return 10;
   }
   return num + calcMethod(num / 2);
}
```

What value is returned by the method call `calcMethod(16)` ?

(A) `10`

(B) `26`

(C) `31`

(D) `38`

(E) `41`

**GO ON TO THE NEXT PAGE.**

30. Consider the following class definitions.

```
public class Rectangle
{
   private int height;
   private int width;

   public Rectangle()
   {
      height = 1;
      width = 1;
   }

   public Rectangle(int x)
   {
      height = x;
      width = x;
   }

   public Rectangle(int h, int w)
   {
      height = h;
      width = w;
   }

   // There may be methods that are not shown.
}

public class Square extends Rectangle
{
   public Square(int x)
   {
      /* missing code */
   }
}
```

Which of the following code segments can replace `/* missing code */` so that the `Square` class constructor initializes the `Rectangle` class instance variables `height` and `width` to `x` ?

(A) `super();`

(B) `super(x);`

(C) `Rectangle(x);`

(D) `Square(x, x);`

(E) `height = x;`
`width = x;`

**GO ON TO THE NEXT PAGE.**

31. Consider an integer array `nums`, which has been properly declared and initialized with one or more values. Which of the following code segments counts the number of negative values found in `nums` and stores the count in `counter` ?

I.
```
int counter = 0;
int i = -1;
while (i <= nums.length - 2)
{
   i++;
   if (nums[i] < 0)
   {
      counter++;
   }
}
```

II.
```
int counter = 0;
for (int i = 1; i < nums.length; i++)
{
   if (nums[i] < 0)
   {
      counter++;
   }
}
```

III.
```
int counter = 0;
for (int i : nums)
{
   if (nums[i] < 0)
   {
      counter++;
   }
}
```

(A) I only

(B) II only

(C) I and II only

(D) I and III only

(E) I, II, and III

**GO ON TO THE NEXT PAGE.**

32. Consider the following class definitions.

```
public class ClassA
{
   public String getValue()
   {
      return "A";
   }

   public void showValue()
   {
      System.out.print(getValue());
   }
}

public class ClassB extends ClassA
{
   public String getValue()
   {
      return "B";
   }
}
```

The following code segment appears in a class other than `ClassA` or `ClassB`.

```
ClassA obj = new ClassB();
obj.showValue();
```

What, if anything, is printed when the code segment is executed?

(A) `A`

(B) `B`

(C) `AB`

(D) `BA`

(E) Nothing is printed because the code does not compile.

**GO ON TO THE NEXT PAGE.**

33. Consider the following code segment.

```
String[][] letters = {{"A", "B", "C", "D"},
                      {"E", "F", "G", "H"},
                      {"I", "J", "K", "L"}};

for (int col = 1; col < letters[0].length; col++)
{
   for (int row = 1; row < letters.length; row++)
   {
      System.out.print(letters[row][col] + " ");
   }

   System.out.println();
}
```

What is printed as a result of executing this code segment?

(A)
```
A E I
F J
K
```

(B)
```
B F J
C G K
D H L
```

(C)
```
E I
F J
G K
H L
```

(D)
```
F G H
J K L
```

(E)
```
F J
G K
H L
```

**GO ON TO THE NEXT PAGE.**

34. The following method is intended to remove all elements of an `ArrayList` of integers that are divisible by `key` and add the removed elements to a new `ArrayList`, which the method returns.

```
public static ArrayList<Integer> match(ArrayList<Integer> numList, int key)
{
   ArrayList<Integer> returnList = new ArrayList<Integer>();

   int i = 0;
   while (i < numList.size())
   {
      int num = numList.get(i);
      if (num % key == 0)
      {
         numList.remove(i);
         returnList.add(num);
      }
      i++;
   }
   return returnList;
}
```

As an example, if the method is called with an `ArrayList` containing the values `[5, 2, 10, 20, 16]` and the parameter `key` has the value `5`, then `numList` should contain `[2, 16]` at the end of the method and an `ArrayList` containing `[5, 10, 20]` should be returned.

Which of the following best explains why the method does not always work as intended?

(A) The method attempts to add an element to `returnList` after that element has already been removed from `numList`.

(B) The method causes a `NullPointerException` to be thrown when no matches are found.

(C) The method causes an `IndexOutOfBoundsException` to be thrown.

(D) The method fails to correctly determine whether an element of `numList` is divisible by `key`.

(E) The method skips some elements of `numList` during the traversal.

**GO ON TO THE NEXT PAGE.**

35. Consider the `mode` method, which is intended to return the most frequently occurring value (mode) in its `int[]` parameter `arr`. For example, if the parameter of the `mode` method has the contents `{6, 5, 1, 5, 2, 6, 5}`, then the method is intended to return `5`.

```
/** Precondition: arr.length >= 1 */
public static int mode(int[] arr)
{
   int modeCount = 1;
   int mode = arr[0];

   for (int j = 0; j < arr.length; j++)
   {
      int valCount = 0;
      for (int k = 0; k < arr.length; k++)
      {
         if ( /* missing condition 1 */ )
         {
            valCount++;
         }
      }
      if ( /* missing condition 2 */ )
      {
         modeCount = valCount;
         mode = arr[j];
      }
   }
   return mode;
}
```

Which of the following can replace `/* missing condition 1 */` and `/* missing condition 2 */` so the code segment works as intended?

| | `/* missing condition 1 */` | `/* missing condition 2 */` |
|---|---|---|
| (A) | `arr[j] == arr[k]` | `valCount > modeCount` |
| (B) | `arr[j] == arr[k]` | `modeCount > valCount` |
| (C) | `arr[j] != arr[k]` | `valCount > modeCount` |
| (D) | `arr[j] != arr[k]` | `modeCount > valCount` |
| (E) | `arr[j] != arr[k]` | `modeCount != valCount` |

**GO ON TO THE NEXT PAGE.**

36. Consider the following methods.

```
/** Precondition: a > 0 and b > 0 */
public static int methodOne(int a, int b)
{
   int loopCount = 0;
   for (int i = 0; i < a / b; i++)
   {
      loopCount++;
   }
   return loopCount;
}

/** Precondition: a > 0 and b > 0 */
public static int methodTwo(int a, int b)
{
   int loopCount = 0;
   int i = 0;
   while (i < a)
   {
      loopCount++;
      i += b;
   }
   return loopCount;
}
```

Which of the following best describes the conditions under which `methodOne` and `methodTwo` return the same value?

(A) When `a` and `b` are both even

(B) When `a` and `b` are both odd

(C) When `a` is even and `b` is odd

(D) When `a % b` is equal to zero

(E) When `a % b` is equal to one

**GO ON TO THE NEXT PAGE.**

37. Consider the following code segment. Assume that `num3 > num2 > 0`.

```
int num1 = 0;
int num2 = /* initial value not shown */;
int num3 = /* initial value not shown */;

while (num2 < num3)
{
   num1 += num2;
   num2++;
}
```

Which of the following best describes the contents of `num1` as a result of executing the code segment?

(A) The product of `num2` and `num3`

(B) The product of `num2` and `num3 - 1`

(C) The sum of `num2` and `num3`

(D) The sum of all integers from `num2` to `num3`, inclusive

(E) The sum of all integers from `num2` to `num3 - 1`, inclusive

**GO ON TO THE NEXT PAGE.**

38. Consider the following class definition.

```
public class Value
{
   private int num;

   public int getNum()
   {
      return num;
   }

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

The following method appears in a class other than `Value`. It is intended to sum all the `num` instance variables of the `Value` objects in its `ArrayList` parameter.

```
/** Precondition: valueList is not null */
public static int getTotal(ArrayList<Value> valueList)
{
   int total = 0;
   /* missing code */
   return total;
}
```

Which of the following code segments can replace `/* missing code */` so the `getTotal` method works as intended?

I.
```
for (int x = 0; x < valueList.size(); x++)
{
   total += valueList.get(x).getNum();
}
```

II.
```
for (Value v : valueList)
{
   total += v.getNum();
}
```

III.
```
for (Value v : valueList)
{
   total += getNum(v);
}
```

(A) I only

(B) II only

(C) III only

(D) I and II

(E) I and III

**GO ON TO THE NEXT PAGE.**

39. Consider the following recursive method.

```
public static boolean recurMethod(String str)
{
   if (str.length() <= 1)
   {
      return true;
   }
   else if (str.substring(0, 1).compareTo(str.substring(1, 2)) > 0)
   {
      return recurMethod(str.substring(1));
   }
   else
   {
      return false;
   }
}
```

Which of the following method calls will return `true` ?

(A) `recurMethod("abcba")`

(B) `recurMethod("abcde")`

(C) `recurMethod("bcdab")`

(D) `recurMethod("edcba")`

(E) `recurMethod("edcde")`

**GO ON TO THE NEXT PAGE.**

40. Consider the following class definitions.

```
public class A
{
   public String message(int i)
   {
      return "A" + i;
   }
}

public class B extends A
{
   public String message(int i)
   {
      return "B" + i;
   }
}
```

The following code segment appears in a class other than `A` or `B`.

```
A obj1 = new B();                        // Line 1
B obj2 = new B();                        // Line 2
System.out.println(obj1.message(3));     // Line 3
System.out.println(obj2.message(2));     // Line 4
```

Which of the following best explains the difference, if any, in the behavior of the code segment that will result from removing the `message` method from class `A` ?

(A) The statement in line 3 will cause a compiler error because the `message` method for `obj1` cannot be found.

(B) The statement in line 4 will cause a compiler error because the `message` method for `obj2` cannot be found.

(C) As a result of the method call in line 3, the `message` method in class `B` will be executed instead of the `message` method in class `A`.

(D) As a result of the method call in line 4, the `message` method in class `B` will be executed instead of the `message` method in class `A`.

(E) The behavior of the code segment will remain unchanged.

**GO ON TO THE NEXT PAGE.**

**END OF SECTION I**

**IF YOU FINISH BEFORE TIME IS CALLED, YOU MAY CHECK YOUR WORK ON THIS SECTION.**

**DO NOT GO ON TO SECTION II UNTIL YOU ARE TOLD TO DO SO.**

---

**MAKE SURE YOU HAVE DONE THE FOLLOWING:**

- **PLACED YOUR AP ID LABEL ON YOUR ANSWER SHEET**
- **WRITTEN AND GRIDDED YOUR AP ID CORRECTLY ON YOUR ANSWER SHEET**
- **TAKEN THE AP EXAM LABEL FROM THE FRONT OF THIS BOOKLET AND PLACED IT ON YOUR ANSWER SHEET**

# AP® Computer Science A Exam

## SECTION II: Free Response, Questions

**DO NOT OPEN THIS BOOKLET UNTIL YOU ARE TOLD TO DO SO.**

### At a Glance

**Total Time**
1 hour and 30 minutes
**Number of Questions**
4
**Percent of Total Score**
50%
**Writing Instrument**
Pencil
**Electronic Device**
None allowed
**Weight**
The questions are weighted equally.

### Instructions

The questions for Section II are printed in this booklet. You may use the pages in this booklet to organize your answers and for scratch work, but you must write your answers in the blank space provided for each question.

The Java Quick Reference is located inside the front cover of this booklet.

Write your answer to each question in the blank space provided. Begin your response to each question at the top of a new page and completely fill in the circle at the top of each page that corresponds to the question you are answering.

All program segments must be written in Java. Show all your work. Credit for partial solutions will be given. Write clearly and legibly. Erased or crossed-out work will not be scored.

Manage your time carefully. Do not spend too much time on any one question. You may proceed freely from one question to the next. You may review your responses if you finish before the end of the exam is announced.

## Java Quick Reference

*Accessible methods from the Java library that may be included in the exam*

| Class Constructors and Methods | Explanation |
|---|---|
| **String Class** | |
| `String(String str)` | Constructs a new `String` object that represents the same sequence of characters as `str` |
| `int length()` | Returns the number of characters in a `String` object |
| `String substring(int from, int to)` | Returns the substring beginning at index `from` and ending at index `to - 1` |
| `String substring(int from)` | Returns `substring(from, length())` |
| `int indexOf(String str)` | Returns the index of the first occurrence of `str`; returns `-1` if not found |
| `boolean equals(String other)` | Returns `true` if `this` is equal to `other`; returns `false` otherwise |
| `int compareTo(String other)` | Returns a value `<0` if `this` is less than `other`; returns zero if `this` is equal to `other`; returns a value `>0` if `this` is greater than `other` |
| **Integer Class** | |
| `Integer(int value)` | Constructs a new `Integer` object that represents the specified `int` value |
| `Integer.MIN_VALUE` | The minimum value represented by an `int` or `Integer` |
| `Integer.MAX_VALUE` | The maximum value represented by an `int` or `Integer` |
| `int intValue()` | Returns the value of this `Integer` as an `int` |
| **Double Class** | |
| `Double(double value)` | Constructs a new `Double` object that represents the specified `double` value |
| `double doubleValue()` | Returns the value of this `Double` as a `double` |
| **Math Class** | |
| `static int abs(int x)` | Returns the absolute value of an `int` value |
| `static double abs(double x)` | Returns the absolute value of a `double` value |
| `static double pow(double base, double exponent)` | Returns the value of the first parameter raised to the power of the second parameter |
| `static double sqrt(double x)` | Returns the positive square root of a `double` value |
| `static double random()` | Returns a `double` value greater than or equal to `0.0` and less than `1.0` |
| **ArrayList Class** | |
| `int size()` | Returns the number of elements in the list |
| `boolean add(E obj)` | Appends `obj` to end of list; returns `true` |
| `void add(int index, E obj)` | Inserts `obj` at position `index` `(0 <= index <= size)`, moving elements at position `index` and higher to the right (adds 1 to their indices) and adds 1 to size |
| `E get(int index)` | Returns the element at position `index` in the list |
| `E set(int index, E obj)` | Replaces the element at position `index` with `obj`; returns the element formerly at position `index` |
| `E remove(int index)` | Removes element from position `index`, moving elements at position `index + 1` and higher to the left (subtracts 1 from their indices) and subtracts 1 from size; returns the element formerly at position `index` |
| **Object Class** | |
| `boolean equals(Object other)` | |
| `String toString()` | |

**COMPUTER SCIENCE A**

**SECTION II**

**Time—1 hour and 30 minutes**

**4 Questions**

**Directions: SHOW ALL YOUR WORK. REMEMBER THAT PROGRAM SEGMENTS ARE TO BE WRITTEN IN JAVA.** You may plan your answers in this Questions booklet, but no credit will be given for anything written in this booklet. You will only earn credit for what you write in the Free Response booklet.

Notes:

- 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 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 will not receive full credit.

**GO ON TO THE NEXT PAGE.**

1. A mathematical sequence is an ordered list of numbers. This question involves a sequence called a *hailstone sequence*. If $n$ is the value of a term in the sequence, then the following rules are used to find the next term, if one exists.

   - If $n$ is 1, the sequence terminates.
   - If $n$ is even, then the next term is $\frac{n}{2}$.
   - If $n$ is odd, then the next term is $3n + 1$.

   For this question, assume that when the rules are applied, the sequence will eventually terminate with the term $n = 1$.

   The following are examples of hailstone sequences.

   Example 1: 5, 16, 8, 4, 2, 1

   - The first term is 5, so the second term is $5 * 3 + 1 = 16$.
   - The second term is 16, so the third term is $\frac{16}{2} = 8$.
   - The third term is 8, so the fourth term is $\frac{8}{2} = 4$.
   - The fourth term is 4, so the fifth term is $\frac{4}{2} = 2$.
   - The fifth term is 2, so the sixth term is $\frac{2}{2} = 1$.
   - Since the sixth term is 1, the sequence terminates.

   Example 2: 8, 4, 2, 1

   - The first term is 8, so the second term is $\frac{8}{2} = 4$.
   - The second term is 4, so the third term is $\frac{4}{2} = 2$.
   - The third term is 2, so the fourth term is $\frac{2}{2} = 1$.
   - Since the fourth term is 1, the sequence terminates.

**GO ON TO THE NEXT PAGE.**

The `Hailstone` class, shown below, is used to represent a hailstone sequence. You will write three methods in the `Hailstone` class.

```
public class Hailstone
{
    /** Returns the length of a hailstone sequence that starts with n,
     *  as described in part (a).
     *  Precondition: n > 0
     */
    public static int hailstoneLength(int n)
    { /* to be implemented in part (a) */ }

    /** Returns true if the hailstone sequence that starts with n is considered long
     *  and false otherwise, as described in part (b).
     *  Precondition: n > 0
     */
    public static boolean isLongSeq(int n)
    { /* to be implemented in part (b) */ }

    /** Returns the proportion of the first n hailstone sequences that are considered long,
     *  as described in part (c).
     *  Precondition: n > 0
     */
    public static double propLong(int n)
    { /* to be implemented in part (c) */ }

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

**GO ON TO THE NEXT PAGE.**

(a) The length of a hailstone sequence is the number of terms it contains. For example, the hailstone sequence in example 1 (5, 16, 8, 4, 2, 1) has a length of 6 and the hailstone sequence in example 2 (8, 4, 2, 1) has a length of 4.

Write the method `hailstoneLength(int n)`, which returns the length of the hailstone sequence that starts with `n`.

```
/** Returns the length of a hailstone sequence that starts with n, as described in part (a).
 *  Precondition: n > 0
 */
public static int hailstoneLength(int n)
```

---

**Begin your response at the top of a new page in the Free Response booklet and fill in the appropriate circle indicating the question number. If there are multiple parts to this question, write the part letter with your response.**

Class information for this question

```
public class Hailstone

public static int hailstoneLength(int n)
public static boolean isLongSeq(int n)
public static double propLong(int n)
```

**GO ON TO THE NEXT PAGE.**

(b) A hailstone sequence is considered long if its length is greater than its starting value. For example, the hailstone sequence in example 1 (5, 16, 8, 4, 2, 1) is considered long because its length (6) is greater than its starting value (5). The hailstone sequence in example 2 (8, 4, 2, 1) is not considered long because its length (4) is less than or equal to its starting value (8).

Write the method `isLongSeq(int n)`, which returns `true` if the hailstone sequence starting with `n` is considered long and returns `false` otherwise. Assume that `hailstoneLength` works as intended, regardless of what you wrote in part (a). You must use `hailstoneLength` appropriately to receive full credit.

```
/** Returns true if the hailstone sequence that starts with n is considered long
 *  and false otherwise, as described in part (b).
 *  Precondition: n > 0
 */
public static boolean isLongSeq(int n)
```

---

**Begin your response at the top of a new page in the Free Response booklet and fill in the appropriate circle indicating the question number. If there are multiple parts to this question, write the part letter with your response.**

**GO ON TO THE NEXT PAGE.**

(c) The method `propLong(int n)` returns the proportion of long hailstone sequences with starting values between 1 and `n`, inclusive.

Consider the following table, which provides data about the hailstone sequences with starting values between 1 and 10, inclusive.

| Starting Value | Terms in the Sequence | Length of the Sequence | Long? |
|---|---|---|---|
| 1 | 1 | 1 | No |
| 2 | 2, 1 | 2 | No |
| 3 | 3, 10, 5, 16, 8, 4, 2, 1 | 8 | Yes |
| 4 | 4, 2, 1 | 3 | No |
| 5 | 5, 16, 8, 4, 2, 1 | 6 | Yes |
| 6 | 6, 3, 10, 5, 16, 8, 4, 2, 1 | 9 | Yes |
| 7 | 7, 22, 11, 34, 17, 52, 26, 13, 40, 20, 10, 5, 16, 8, 4, 2, 1 | 17 | Yes |
| 8 | 8, 4, 2, 1 | 4 | No |
| 9 | 9, 28, 14, 7, 22, 11, 34, 17, 52, 26, 13, 40, 20, 10, 5, 16, 8, 4, 2, 1 | 20 | Yes |
| 10 | 10, 5, 16, 8, 4, 2, 1 | 7 | No |

The method call `Hailstone.propLong(10)` returns `0.5`, since 5 of the 10 hailstone sequences shown in the table are considered long.

Write the `propLong` method. Assume that `hailstoneLength` and `isLongSeq` work as intended, regardless of what you wrote in parts (a) and (b). You must use `isLongSeq` appropriately to receive full credit.

```
/** Returns the proportion of the first n hailstone sequences that are considered long,
 *  as described in part (c).
 *  Precondition: n > 0
 */
public static double propLong(int n)
```

---

**Begin your response at the top of a new page in the Free Response booklet and fill in the appropriate circle indicating the question number. If there are multiple parts to this question, write the part letter with your response.**

Class information for this question

```
public class Hailstone

public static int hailstoneLength(int n)
public static boolean isLongSeq(int n)
public static double propLong(int n)
```

**GO ON TO THE NEXT PAGE.**

**NO TEST MATERIAL ON THIS PAGE**

**GO ON TO THE NEXT PAGE.**

2. This question involves the creation and use of a spinner to generate random numbers in a game. A `GameSpinner` object represents a spinner with a given number of sectors, all equal in size. The `GameSpinner` class supports the following behaviors.

- Creating a new spinner with a specified number of sectors
- Spinning a spinner and reporting the result
- Reporting the length of the *current run*, the number of consecutive spins that are the same as the most recent spin

The following table contains a sample code execution sequence and the corresponding results.

| Statements | Value Returned (blank if no value returned) | Comment |
|---|---|---|
| `GameSpinner g = new GameSpinner(4);` | | Creates a new spinner with four sectors |
| `g.currentRun();` | `0` | Returns the length of the current run. The length of the current run is initially `0` because no spins have occurred. |
| `g.spin();` | `3` | Returns a random integer between `1` and `4`, inclusive. In this case, `3` is returned. |
| `g.currentRun();` | `1` | The length of the current run is `1` because there has been one spin of `3` so far. |
| `g.spin();` | `3` | Returns a random integer between `1` and `4`, inclusive. In this case, `3` is returned. |
| `g.currentRun();` | `2` | The length of the current run is `2` because there have been two `3`s in a row. |
| `g.spin();` | `4` | Returns a random integer between `1` and `4`, inclusive. In this case, `4` is returned. |
| `g.currentRun();` | `1` | The length of the current run is `1` because the spin of `4` is different from the value of the spin in the previous run of two `3`s. |
| `g.spin();` | `3` | Returns a random integer between `1` and `4`, inclusive. In this case, `3` is returned. |
| `g.currentRun();` | `1` | The length of the current run is `1` because the spin of `3` is different from the value of the spin in the previous run of one `4`. |
| `g.spin();` | `1` | Returns a random integer between `1` and `4`, inclusive. In this case, `1` is returned. |
| `g.spin();` | `1` | Returns a random integer between `1` and `4`, inclusive. In this case, `1` is returned. |
| `g.spin();` | `1` | Returns a random integer between `1` and `4`, inclusive. In this case, `1` is returned. |
| `g.currentRun();` | `3` | The length of the current run is `3` because there have been three consecutive `1`s since the previous run of one `3`. |

**GO ON TO THE NEXT PAGE.**

Write the complete `GameSpinner` class. Your implementation must meet all specifications and conform to the example.

---

**Begin your response at the top of a new page in the Free Response booklet and fill in the appropriate circle indicating the question number.**
**If there are multiple parts to this question, write the part letter with your response.**

**GO ON TO THE NEXT PAGE.**

3. A student plans to analyze product reviews found on a Web site by looking for keywords in posted reviews. The `ProductReview` class, shown below, is used to represent a single review. A product review consists of a product name and a review of that product.

```
public class ProductReview
{
   private String name;
   private String review;

   /** Constructs a ProductReview object and initializes the instance variables. */
   public ProductReview(String pName, String pReview)
   {
      name = pName;
      review = pReview;
   }

   /** Returns the name of the product. */
   public String getName()
   {  return name; }

   /** Returns the review of the product. */
   public String getReview()
   {  return review; }
}
```

The `ReviewCollector` class, shown below, is used to represent a collection of reviews to be analyzed.

```
public class ReviewCollector
{
   private ArrayList<ProductReview> reviewList;
   private ArrayList<String> productList;

   /** Constructs a ReviewCollector object and initializes the instance variables. */
   public ReviewCollector()
   {
      reviewList = new ArrayList<ProductReview>();
      productList = new ArrayList<String>();
   }

   /** Adds a new review to the collection of reviews, as described in part (a). */
   public void addReview(ProductReview prodReview)
   {  /* to be implemented in part (a) */  }

   /** Returns the number of good reviews for a given product name, as described in part (b). */
   public int getNumGoodReviews(String prodName)
   {  /* to be implemented in part (b) */  }

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

**GO ON TO THE NEXT PAGE.**

(a) Write the `addReview` method, which adds a single product review, represented by a `ProductReview` object, to the `ReviewCollector` object. The `addReview` method does the following when it adds a product review.

- The `ProductReview` object is added to the `reviewList` instance variable.
- The product name from the `ProductReview` object is added to the `productList` instance variable if the product name is not already found in `productList`.

Elements may be added to `reviewList` and `productList` in any order.

Complete method `addReview`.

```
/** Adds a new review to the collection of reviews, as described in part (a). */
public void addReview(ProductReview prodReview)
```

---

**Begin your response at the top of a new page in the Free Response booklet and fill in the appropriate circle indicating the question number. If there are multiple parts to this question, write the part letter with your response.**

**GO ON TO THE NEXT PAGE.**

(b) Write the `getNumGoodReviews` method, which returns the number of *good* reviews for a given product name. A review is considered good if it contains the string `"best"` (all lowercase). If there are no reviews with a matching product name, the method returns `0`. Note that a review that contains `"BEST"` or `"Best"` is not considered a good review (since not all the letters of `"best"` are lowercase), but a review that contains `"asbestos"` is considered a good review (since all the letters of `"best"` are lowercase).

Complete method `getNumGoodReviews`.

```
/** Returns the number of good reviews for a given product name, as described in part (b). */
public int getNumGoodReviews(String prodName)
```

---

**Begin your response at the top of a new page in the Free Response booklet and fill in the appropriate circle indicating the question number. If there are multiple parts to this question, write the part letter with your response.**

Class information for this question

```
public class ProductReview

private String name
private String review

public ProductReview(String pName, String pReview)
public String getName()
public String getReview()

public class ReviewCollector

private ArrayList<ProductReview> reviewList
private ArrayList<String> productList

public ReviewCollector()
public void addReview(ProductReview prodReview)
public int getNumGoodReviews(String prodName)
```

**GO ON TO THE NEXT PAGE.**

**NO TEST MATERIAL ON THIS PAGE**

**GO ON TO THE NEXT PAGE.**

4. A theater contains rows of seats with the same number of seats in each row. Some rows contain tier 1 seats, and the remaining rows contain tier 2 seats. Tier 1 seats are closer to the stage and are more desirable. All seats in a row share the same tier.

The `Seat` class, shown below, represents seats in the theater. The `boolean` instance variable `available` is `false` if a ticket for the seat has been sold (the seat is no longer available). The `int` instance variable `tier` indicates whether the seat is a tier 1 or tier 2 seat.

```
public class Seat
{
   private boolean available;
   private int tier;

   public Seat(boolean isAvail, int tierNum)
   {
      available = isAvail;
      tier = tierNum;
   }

   public boolean isAvailable()
   {  return available;  }

   public int getTier()
   {  return tier;  }

   public void setAvailability(boolean isAvail)
   {  available = isAvail;  }
}
```

**GO ON TO THE NEXT PAGE.**

The `Theater` class represents a theater of seats. The number of seats per row and the number of tier 1 and tier 2 rows are determined by the parameters of the `Theater` constructor. Row `0` of the `theaterSeats` array represents the row closest to the stage.

```
public class Theater
{
   private Seat[][] theaterSeats;

   /** Constructs a Theater object, as described in part (a).
    *  Precondition: seatsPerRow > 0; tier1Rows > 0; tier2Rows >= 0
    */
   public Theater(int seatsPerRow, int tier1Rows, int tier2Rows)
   {  /* to be implemented in part (a) */  }

   /** Returns true if a seat holder was reassigned from the seat at fromRow, fromCol
    *  to the seat at toRow, toCol; otherwise it returns false, as described in part (b).
    *  Precondition: fromRow, fromCol, toRow, and toCol represent valid row and
    *                column positions in the theater.
    *                The seat at fromRow, fromCol is not available.
    */
   public boolean reassignSeat(int fromRow, int fromCol,
                               int toRow, int toCol)
   {  /* to be implemented in part (b) */  }
}
```

**GO ON TO THE NEXT PAGE.**

(a) Write the constructor for the `Theater` class. The constructor takes three `int` parameters, representing the number of seats per row, the number of tier 1 rows, and the number of tier 2 rows, respectively. The constructor initializes the `theaterSeats` instance variable so that it has the given number of seats per row and the given number of tier 1 and tier 2 rows and all seats are available and have the appropriate tier designation.

Row `0` of the `theaterSeats` array represents the row closest to the stage. All tier 1 seats are closer to the stage than tier 2 seats.

Complete the `Theater` constructor.

```
/** Constructs a Theater object, as described in part (a).
 *  Precondition: seatsPerRow > 0; tier1Rows > 0; tier2Rows >= 0
 */
public Theater(int seatsPerRow, int tier1Rows, int tier2Rows)
```

---

**Begin your response at the top of a new page in the Free Response booklet and fill in the appropriate circle indicating the question number. If there are multiple parts to this question, write the part letter with your response.**

Class information for this question

```
public class Seat

private boolean available
private int tier

public Seat(boolean isAvail, int tierNum)
public boolean isAvailable()
public int getTier()
public void setAvailability(boolean isAvail)

public class Theater

private Seat[][] theaterSeats

public Theater(int seatsPerRow, int tier1Rows, int tier2Rows)
public boolean reassignSeat(int fromRow, int fromCol,
                            int toRow, int toCol)
```

**GO ON TO THE NEXT PAGE.**

(b) Write the `reassignSeat` method, which attempts to move a person from a source seat to a destination seat. The reassignment can be made if the destination seat is available and has the same or greater tier than the source seat (that is, it is equally or less desirable). For example, a person in a tier 1 seat can be moved to a different tier 1 seat or to a tier 2 seat, but a person in a tier 2 seat can only be moved to a different tier 2 seat.

The `reassignSeat` method has four `int` parameters representing the row and column indexes of the source ("from") and destination ("to") seats. If the reassignment is possible, the source seat becomes available, the destination seat becomes unavailable, and the method returns `true`. If the seat reassignment is not possible, no changes are made to either seat and the method returns `false`. Assume that the source seat is occupied when the method is called.

Complete method `reassignSeat`.

```
/** Returns true if a seat holder was reassigned from the seat at fromRow, fromCol
 *  to the seat at toRow, toCol; otherwise it returns false, as described in part (b).
 *  Precondition: fromRow, fromCol, toRow, and toCol represent valid row and
 *                column positions in the theater.
 *                The seat at fromRow, fromCol is not available.
 */
public boolean reassignSeat(int fromRow, int fromCol,
                            int toRow, int toCol)
```

---

**Begin your response at the top of a new page in the Free Response booklet and fill in the appropriate circle indicating the question number. If there are multiple parts to this question, write the part letter with your response.**

**GO ON TO THE NEXT PAGE.**

**STOP**

**END OF EXAM**

**Important:** Completely fill in the circle that corresponds to the question you are answering on this page.

| Question 1 | Question 2 | Question 3 | Question 4 |
|---|---|---|---|
| ○ | ○ | ○ | ○ |

Begin your response to each question at the top of a new page.

Page 2

**Use a pencil only. Do NOT write your name. Do NOT write outside the box.**

**Important:** Completely fill in the circle that corresponds to the question you are answering on this page.

| Question 1 | Question 2 | Question 3 | Question 4 |
|---|---|---|---|
| ○ | ○ | ○ | ○ |

Begin your response to each question at the top of a new page.

Page 3

**Use a pencil only. Do NOT write your name. Do NOT write outside the box.**

**Important:** Completely fill in the circle that corresponds to the question you are answering on this page.

| Question 1 | Question 2 | Question 3 | Question 4 |
|---|---|---|---|
| ○ | ○ | ○ | ○ |

Begin your response to each question at the top of a new page.

Page 4

**Use a pencil only. Do NOT write your name. Do NOT write outside the box.**

**Important:** Completely fill in the circle that corresponds to the question you are answering on this page.

| Question 1 | Question 2 | Question 3 | Question 4 |
|---|---|---|---|
| ○ | ○ | ○ | ○ |

Begin your response to each question at the top of a new page.

Page 5

**Use a pencil only. Do NOT write your name. Do NOT write outside the box.**

**Important:** Completely fill in the circle that corresponds to the question you are answering on this page.

| Question 1 | Question 2 | Question 3 | Question 4 |
|---|---|---|---|
| ○ | ○ | ○ | ○ |

Begin your response to each question at the top of a new page.

Page 6

**Use a pencil only. Do NOT write your name. Do NOT write outside the box.**

**Important:** Completely fill in the circle that corresponds to the question you are answering on this page.

| Question 1 | Question 2 | Question 3 | Question 4 |
|---|---|---|---|
| ○ | ○ | ○ | ○ |

Begin your response to each question at the top of a new page.

Page 7

**Use a pencil only. Do NOT write your name. Do NOT write outside the box.**

**Important:** Completely fill in the circle that corresponds to the question you are answering on this page.

| Question 1 | Question 2 | Question 3 | Question 4 |
|---|---|---|---|
| ○ | ○ | ○ | ○ |

Begin your response to each question at the top of a new page.

Page 8

**Use a pencil only. Do NOT write your name. Do NOT write outside the box.**

**Important:** Completely fill in the circle that corresponds to the question you are answering on this page.

| Question 1 | Question 2 | Question 3 | Question 4 |
|---|---|---|---|
| ○ | ○ | ○ | ○ |

Begin your response to each question at the top of a new page.

Page 9

**Use a pencil only. Do NOT write your name. Do NOT write outside the box.**

**Important:** Completely fill in the circle that corresponds to the question you are answering on this page.

| Question 1 | Question 2 | Question 3 | Question 4 |
|---|---|---|---|
| ○ | ○ | ○ | ○ |

Begin your response to each question at the top of a new page.

Page 10

**Use a pencil only. Do NOT write your name. Do NOT write outside the box.**

**Important:** Completely fill in the circle that corresponds to the question you are answering on this page.

| Question 1 | Question 2 | Question 3 | Question 4 |
|---|---|---|---|
| ○ | ○ | ○ | ○ |

Begin your response to each question at the top of a new page.

Page 11

**Use a pencil only. Do NOT write your name. Do NOT write outside the box.**

**Important:** Completely fill in the circle that corresponds to the question you are answering on this page.

| Question 1 | Question 2 | Question 3 | Question 4 |
|---|---|---|---|
| ○ | ○ | ○ | ○ |

Begin your response to each question at the top of a new page.

Page 12

**Use a pencil only. Do NOT write your name. Do NOT write outside the box.**

# Notes on the AP Computer Science A Practice Exam

## Multiple-Choice Section

### Course Framework Alignment and Rationales

Question 1

| Skill | Learning Objective | Topic |
|---|---|---|
| 2.B: Determine the result or output based on statement execution order in a code segment without method calls (other than output). | CON-1.A: Evaluate arithmetic expressions in program code. | Expressions and Assignment Statements |

| | |
|---|---|
| (A) | Incorrect. This would be the result if the addition operation in the third assignment statement was evaluated before the remainder operation, as in `int c = 7 % (4 + 3)`. |
| (B) | Incorrect. This would be the result if the addition operation in the second assignment statement was evaluated before the integer division operation, as in `int b = (4 + 3) / 2`. |
| **(C)** | **Correct.** In the first assignment statement, the multiplication operation is evaluated before the addition operation and `a` is assigned the value `9`. In the second assignment statement, the integer division is evaluated first and produces a result of `1`, which is added to `4` so that the variable `b` is assigned the value `5`. In the third assignment statement, the remainder operation is evaluated before the addition operation and `c` is assigned the value `6`. The variable `d` is assigned the value `20.0`. |
| (D) | Incorrect. This would be the result if the division operator in the second assignment statement performed floating point division instead of integer division, as in `int b = 4 + 3.0 / 2`. |
| (E) | Incorrect. This would be the result if the addition operation in the first assignment statement was evaluated before the multiplication operation, as in `int a = (3 + 2) * 3`. |

Question 2

| Skill | Learning Objective | Topic |
|---|---|---|
| 5.A: Describe the behavior of a given segment of program code. | CON-2.B: Represent branching logical processes by using nested conditional statements.<br>CON-2.A: Represent branching logical processes by using conditional statements. | Compound Boolean Expressions<br>if-else Statements |

| | |
|---|---|
| (A) | Incorrect. When `num` is zero or a negative integer, whether even or odd, the body of the outer `if` statement is not executed and nothing is printed. When `num` is a positive integer, either `"A"` or `"B"` is printed. |
| (B) | Incorrect. When `num` is zero or a negative integer, whether even or odd, the body of the outer `if` statement is not executed and nothing is printed. When `num` is a positive integer, either `"A"` or `"B"` is printed. |
| (C) | Incorrect. `"B"` is only printed in the case of a positive odd integer. Nothing is printed if `num` is not positive. |
| **(D)** | **Correct.** When `num` is positive and even, `"A"` is printed. When `num` is positive and not even (odd), `"B"` is printed. When `num` is not positive, nothing is printed. |
| (E) | Incorrect. This would be the result if the condition in the second `if` statement was `num % 2 != 0`. |

Question 3

| Skill | Learning Objective | Topic |
|---|---|---|
| 1.C: Determine code that would be used to interact with completed code. | CON-1.D: Evaluate expressions that use the Math class methods.<br>CON-1.A: Evaluate arithmetic expressions in program code.<br>CON-1.C: Evaluate arithmetic expressions that use casting. | Using the Math Class<br>Expressions and Assignment Statements<br>Casting and Ranges of Variables |

| | |
|---|---|
| (A) | Incorrect. Since `marker1` and `marker2` are always positive, taking the absolute value of each one has no effect. In this statement, if `marker2` is greater than `marker1, hours` is assigned a negative value. |
| (B) | Incorrect. In this statement, parentheses are incorrectly placed, so only `marker2` is divided by `60.0`, not the absolute value of the difference between `marker1` and `marker2`. |
| **(C)** | **Correct.** The code segment takes the absolute value of the difference between `marker1` and `marker2`, always producing a positive distance, and then divides the result by the vehicle's speed. |
| (D) | Incorrect. Since `marker1` and `marker2` are both of type `int`, the expression `(marker1 - marker2) / 60` performs integer division. For example, when `marker1` has the value `100` and `marker2` has the value `70`, the expression evaluates to `0` instead of the intended `0.5`. |
| (E) | Incorrect. Since `marker1` and `marker2` are both of type `int`, the expression `(marker1 - marker2) / 60` performs integer division. The casting of the result of the division to a `double` occurs too late. |

Question 4

| Skill | Learning Objective | Topic |
|---|---|---|
| 2.B: Determine the result or output based on statement execution order in a code segment without method calls (other than output). | CON-2.B: Represent branching logical processes by using nested conditional statements.<br>CON-2.A: Represent branching logical processes by using conditional statements. | Compound Boolean Expressions<br>if Statements and Control Flow<br>if-else Statements |

| | |
|---|---|
| (A) | Incorrect. This result would be printed as a result of the call `message(5, 5, 15)`. |
| (B) | Incorrect. This result would be printed as a result of the call `message(5, 5, 5)`. |
| (C) | Incorrect. This result would be printed as a result of the call `message(15, 15, 5)`. |
| **(D)** | **Correct.** Since `a < 10` evaluates to `true`, the body of the `if` statement is executed. Since `b < 10` evaluates to `false`, `"X"` is not printed; `"Y"` is printed. Since `c < 10` evaluates to `true`, the body of the `if` statement is executed, and since `b > 10` evaluates to `true`, `"Y"` is printed. |
| (E) | Incorrect. This result would be printed as a result of the call `message(15, 5, 5)`. |

Question 5

| Skill | Learning Objective | Topic |
|---|---|---|
| 1.C: Determine code that would be used to interact with completed code. | MOD-1.C: Identify, using its signature, the correct constructor being called. | Creating and Storing Objects (Instantiation) |

| | |
|---|---|
| (A) | Incorrect. This constructor's signature differs from the signature of the existing constructor, so the new constructor can safely be added to the class definition. |
| (B) | Incorrect. This constructor's signature differs from the signature of the existing constructor, so the new constructor can safely be added to the class definition. |
| (C) | Incorrect. This constructor's signature differs from the signature of the existing constructor, so the new constructor can safely be added to the class definition. |
| (D) | Incorrect. This constructor's signature differs from the signature of the existing constructor, so the new constructor can safely be added to the class definition. |
| **(E)** | **Correct.** This constructor has the same signature as the existing constructor `(String, String, boolean)`. A compiler error will occur. |

Question 6

| Skill | Learning Objective | Topic |
|---|---|---|
| 2.A: Apply the meaning of specific operators. | CON-1.A: Evaluate arithmetic expressions in program code.<br>CON-1.C: Evaluate arithmetic expressions that use casting. | Expressions and Assignment Statements<br>Casting and Ranges of Variables |

| | |
|---|---|
| **(A)** | **Correct.** In option I, the cast applies to the value `2`, so floating-point division is performed and the expression evaluates to `0.5 + 3`, or `3.5`. In option II, the cast applies to the result of the integer division `2 / 4`, so the expression evaluates to `0.0 + 3`, or `3.0`. In option III, the cast applies to the sum of `3` and the result of the integer division `2 / 4`, so the expression evaluates to `(double) (0 + 3)`, or `3.0`. |
| (B) | Incorrect. In option I, the cast applies to the value `2`, so floating-point division is performed and the expression evaluates to `0.5 + 3`, or `3.5`. In option III, the cast applies to the sum of `3` and the result of the integer division `2 / 4`, so the expression evaluates to `(double) (0 + 3)`, or `3.0`. |
| (C) | Incorrect. In option II, the cast applies to the result of the integer division `2 / 4`, so the expression evaluates to `0.0 + 3`, or `3.0`. |
| (D) | Incorrect. In option I, the cast applies to the value `2`, so floating-point division is performed and the expression evaluates to `0.5 + 3`, or `3.5`. In option II, the cast applies to the result of the integer division `2 / 4`, so the expression evaluates to `0.0 + 3`, or `3.0`. In option III, the cast applies to the sum of `3` and the result of the integer division `2 / 4`, so the expression evaluates to `(double) (0 + 3)`, or `3.0`. |
| (E) | Incorrect. In option II, the cast applies to the result of the integer division `2 / 4`, so the expression evaluates to `0.0 + 3`, or `3.0`. In option III, the cast applies to the sum of `3` and the result of the integer division `2 / 4`, so the expression evaluates to `(double) (0 + 3)`, or `3.0`. |

Question 7

| Skill | Learning Objective | Topic |
|---|---|---|
| 4.C: Determine if two or more code segments yield equivalent results. | CON-2.B: Represent branching logical processes by using nested conditional statements.<br>CON-2.A: Represent branching logical processes by using conditional statements.<br>CON-1.F: Evaluate compound Boolean expressions in program code. | Compound Boolean Expressions<br>if-else Statements |

| | |
|---|---|
| (A) | Incorrect. The statement assigns a different value to `b2` than the code segment assigns to `b1` when `num` is between `-100`, exclusive, and `0`, inclusive, or when `num` is less than `-100`. |
| (B) | Incorrect. The statement assigns `true` to `b2` for all values of `num`. |
| (C) | Incorrect. The statement assigns a different value to `b2` than the code segment assigns to `b1` when `num` is between `0` and `100`, exclusive, or when `num` is greater than `100`. |
| (D) | Incorrect. The statement assigns a different value to `b2` than the code segment assigns to `b1` when `num` is between `0` and `100`, exclusive. |
| **(E)** | **Correct.** In the body of the first `if` clause in the code segment, `b1` retains the value `true` if `num` is between `0` and `100`, exclusive. In the body of the `else` clause, `b1` retains the value `true` if `num` is less than `-100`. The statement assigns `true` to `b2` if `num` is less than `-100` or between `0` and `100`, exclusive. |

Question 8

| Skill | Learning Objective | Topic |
|---|---|---|
| 4.B: Identify errors in program code. | VAR-1.G: Explain where variables can be used in the program code.<br>MOD-1.C: Identify, using its signature, the correct constructor being called. | Scope and Access<br>Creating and Storing Objects (Instantiation) |

| | |
|---|---|
| (A) | Incorrect. A constructor signature consists of the constructor name and the parameter list. A correct constructor header does not include a return type. |
| (B) | Incorrect. Assigning `int` values to `double` variables is allowed, although assigning `double` values to `int` variables is not allowed. |
| (C) | Incorrect. The `void` return type of the `incrementPoints` method is correct because the method does not return a value. |
| **(D)** | **Correct.** The variables `n1` and `n2` are not instance variables of the `Points` class, nor are they defined in the `incrementPoints` method. The instance variables `num1` and `num2` should have been used instead of `n1` and `n2`. |
| (E) | Incorrect. The variable `value` is the parameter passed to the `incrementPoints` method. |

Question 9

| Skill | Learning Objective | Topic |
|---|---|---|
| 2.C: Determine the result or output based on the statement execution order in a code segment containing method calls. | VAR-2.D: Represent collections of related object reference data using ArrayList objects. | ArrayList Methods |

| | |
|---|---|
| (A) | Incorrect. This output would be printed if the two-parameter `add` method call were `numList.add(0, 1)` and the `set` method call were `numList.set(2, 0)`. |
| **(B)** | **Correct.** The three single-parameter `add` method calls create an `ArrayList` with the contents `[3, 2, 1]`. The two-parameter `add` method call inserts an element with the value `0` at position `1`, so the `ArrayList` contains `[3, 0, 2, 1]`. The `set` method call sets the value of the element at position `0` to `2`, and the `ArrayList` contains `[2, 0, 2, 1]` at the end of the code segment. |
| (C) | Incorrect. This output would be printed if the one-parameter `add` method calls were `numList.add(0, 3)`, `numList.add(0, 2)`, and `numList.add(0, 1)`. |
| (D) | Incorrect. This output would be printed if the two-parameter `add` method call were `numList.add(0, 1)`. |
| (E) | Incorrect. This output would be printed if the two-parameter `set` method call were `numList.set(2, 0)`. |

Question 10

| Skill | Learning Objective | Topic |
|---|---|---|
| 4.A: Use test-cases to find errors or validate results. | CON-2.E: Represent iterative processes using a for loop.<br>CON-1.A: Evaluate arithmetic expressions in program code. | for Loops<br>Expressions and Assignment Statements |

| | |
|---|---|
| (A) | Incorrect. The loop body is never executed as a result of this method call and nothing is printed. |
| (B) | Incorrect. This method call results in the output `"0 "`. |
| (C) | Incorrect. This method call results in the output `"0 "`. |
| **(D)** | **Correct.** The `for` loop iterates from `i = 0` to `i = 19`. The expression `i % num2 == 0` evaluates to `true` when `i` is divisible by `5` and the expression `i % 2 == 0` evaluates to true when `i` is even. The only values in the range `0` to `19`, inclusive, that are both divisible by `5` and even are `0` and `10`, so the statement prints `"0 10 "`. |
| (E) | Incorrect. This method call results in the output `"0 10 20 "`. |

Question 11

| Skill | Learning Objective | Topic |
|---|---|---|
| 1.B: Determine code that would be used to complete code segments. | CON-2.C: Represent iterative processes using a while loop.<br>CON-1.B: Evaluate what is stored in a variable as a result of an expression with an assignment statement. | while Loops<br>Compound Assignment Operators |

| | |
|---|---|
| (A) | Incorrect. This code segment prints `"10987654321"`. |
| (B) | Incorrect. This code segment prints `"109876543210"`. |
| **(C)** | **Correct.** During the first iteration of the `while` loop, `num` is decremented and `"9"` is printed. During the second iteration, `num` is decremented and `"8"` is printed. This continues until the last iteration of the loop, when `num` is decremented and `"1"` is printed. At this point, the Boolean expression in the `while` loop evaluates to `false` and the loop terminates. The code segment prints `"987654321"`. |
| (D) | Incorrect. This code segment prints `"9876543210"`. |
| (E) | Incorrect. This code segment prints `"10987654321"`. |

Question 12

| Skill | Learning Objective | Topic |
|---|---|---|
| 1.C: Determine code that would be used to interact with completed code. | CON-1.H: Compare object references using Boolean expressions in program code.<br>CON-2.A: Represent branching logical processes by using conditional statements. | Comparing Objects<br>if Statements and Control Flow |

| | |
|---|---|
| (A) | Incorrect. Condition I is incorrect. If no `Person` object has been assigned to `borrower`, the method call `borrower.equals(null)` throws a `NullPointerException`. Condition II is correct. This condition ensures that `borrower` contains a reference to an object when it is used in the `println` method call that follows. |
| **(B)** | **Correct.** Condition I is incorrect. If no `Person` object has been assigned to `borrower`, the method call `borrower.equals(null)` throws a `NullPointerException`. Condition II is correct. This condition ensures that `borrower` contains a reference to an object when it is used in the `println` method call that follows. Condition III is incorrect. If no `Person` object has been assigned to `borrower`, the method call `borrower.getName()` throws a `NullPointerException`. |
| (C) | Incorrect. Condition II is correct. This condition ensures that `borrower` contains a reference to an object when it is used in the `println` method call that follows. Condition III is incorrect. If no `Person` object has been assigned to `borrower`, the method call `borrower.getName()` throws a `NullPointerException`. |
| (D) | Incorrect. Condition I is incorrect. If no `Person` object has been assigned to `borrower`, the method call `borrower.equals(null)` throws a `NullPointerException`. |
| (E) | Incorrect. Condition III is incorrect. If no `Person` object has been assigned to `borrower`, the method call `borrower.getName()` throws a `NullPointerException`. |

Question 13

| Skill | Learning Objective | Topic |
|---|---|---|
| 4.C: Determine if two or more code segments yield equivalent results. | CON-1.G: Compare and contrast equivalent Boolean expressions. | Equivalent Boolean Expressions |

| | |
|---|---|
| (A) | Incorrect. The expressions are not equivalent when `a` has the value `false`, `b` has the value `true`, and `c` has the value `true`. |
| (B) | Incorrect. The expressions are not equivalent when `a` has the value `false`, `b` has the value `false`, and `c` has the value `false`. |
| (C) | Incorrect. The expressions are not equivalent when `a` has the value `true`, `b` has the value `false`, and `c` has the value `false`. |
| (D) | Incorrect. The expressions are not equivalent when `a` has the value `true`, `b` has the value `false`, and `c` has the value `false`. |
| **(E)** | **Correct.** By De Morgan's laws, `!(a && b)` is equivalent to `!a \|\| !b` and the entire expression is equivalent to `!a \|\| !b \|\| c`. |

Question 14

| Skill | Learning Objective | Topic |
|---|---|---|
| 1.B: Determine code that would be used to complete code segments. | CON-2.A: Represent branching logical processes by using conditional statements. | if Statements and Control Flow else if Statements |

| | |
|---|---|
| (A) | Incorrect. Code segment III returns the correct category through the use of an immediate `return` within each of the one-way selection statements. |
| (B) | Incorrect. Code segment I uses multi-way selection to assign and return the correct category. |
| (C) | Incorrect. Code segment II returns `"rural"` for all values of `density` because it uses a series of one-way selection statements instead of multi-way selection. Code segment III returns the correct category through the use of an immediate `return` within each of the one-way selection statements. |
| **(D)** | **Correct.** Code segment I uses multi-way selection to assign and return the correct category. Code segment II returns `"rural"` for all values of `density` because it uses a series of one-way selection statements instead of multi-way selection. Code segment III returns the correct category through the use of an immediate `return` within each of the one-way selection statements. |
| (E) | Incorrect. Code segment II returns `"rural"` for all values of `density` because it uses a series of one-way selection statements instead of multi-way selection. |

Question 15

| Skill | Learning Objective | Topic |
|---|---|---|
| 5.A: Describe the behavior of a given segment of program code. | CON-1.D: Evaluate expressions that use the Math class methods. | Using the Math Class |

| | |
|---|---|
| (A) | Incorrect. This would describe the value assigned to `b` if the value returned by `random` was cast to an `int` before being multiplied by `a`, as in `int b = a + ((int) Math.random()) * a`. |
| (B) | Incorrect. This would describe the value assigned to `b` if the value returned by `random` was rounded up to `1` before being multiplied by `a`, as in `int b = a + (int) (Math.random() + 1) * a`. |
| (C) | Incorrect. This would describe the value assigned to `b` if the second assignment statement was `int b = (int) (Math.random() * a)`. |
| (D) | Incorrect. This would describe the value assigned to `b` if `random` returned values between `0.0` and `1.0`, inclusive. Instead, `random` returns values between `0.0`, inclusive, and `1.0`, exclusive. |
| **(E)** | **Correct.** The `random` method returns a value between `0.0`, inclusive, and `1.0`, exclusive. Multiplying that value by `a` and casting to an `int` produces a result between `0` and `a - 1`, inclusive. The sum of `a` and a value between `0` and `a - 1`, inclusive, is a value between `a` and `2 * a - 1`, inclusive. |

Question 16

| Skill | Learning Objective | Topic |
|---|---|---|
| 2.C: Determine the result or output based on the statement execution order in a code segment containing method calls. | CON-2.O: Determine the result of executing recursive methods.<br>CON-2.E: Represent iterative processes using a for loop. | Recursion<br>for Loops |

| | |
|---|---|
| (A) | Incorrect. This output would be printed if the recursive call `stars(num - 1)` were missing. |
| **(B)** | **Correct.** The recursive call of the `stars` method occurs before any output is printed, so the method call `stars(5)` results in a recursive call to `stars(4)`, then to `stars(3)`, then to `stars(2)`, and finally to `stars(1)`. The call to `stars(1)` returns immediately without printing any output, so the first call that produces output is `stars(2)`, which prints a row of two stars. Then, `stars(3)` prints a row of three stars, `stars(4)` prints a row of four stars, and finally `stars(5)` prints a row of five stars. |
| (C) | Incorrect. This output would be printed if the base case was `num == 0` instead of `num == 1`. |
| (D) | Incorrect. This output would be printed if the recursive call `stars(num - 1)` was the last line of the method instead of occurring before the statements that produce output. |
| (E) | Incorrect. This output would be printed if the recursive call `stars(num - 1)` was the last line of the method instead of occurring before the statements that produce output and if the base case was `num == 0` instead of `num == 1`. |

Question 17

| Skill | Learning Objective | Topic |
|---|---|---|
| 2.C: Determine the result or output based on the statement execution order in a code segment containing method calls. | MOD-3.B: Create an inheritance relationship from a subclass to the superclass.<br>MOD-3.D: Call methods in an inheritance relationship. | Overriding Methods<br>super Keyword<br>Polymorphism |

| | |
|---|---|
| (A) | Incorrect. This value would be printed if `j` was instantiated as a `Hero` object instead of as a `SuperHero` object, and if the `powerUp` method in the `Hero` class assigned the value `p` to the instance variable `power` rather than incrementing `power` by `p`. |
| (B) | Incorrect. This value would be printed if the `powerUp` method in the `Hero` class assigned the value `p` to the instance variable `power` rather than incrementing `power` by `p`. |
| (C) | Incorrect. This value would be printed if `j` was instantiated as a `Hero` object instead of as a `SuperHero` object. |
| **(D)** | **Correct.** Since `j` is instantiated as a `SuperHero` object, the `j.powerUp(10)` method call accesses the subclass method. The subclass method uses the `super` keyword to access the superclass method with the parameter `20`. As a result, the instance variable `power` is incremented by `20`. |
| (E) | Incorrect. This value would be printed if the `powerUp` method in the `Hero` class doubled the value of the instance variable `power` rather than incrementing `power` by `p`. |

Question 18

| Skill | Learning Objective | Topic |
|---|---|---|
| 1.B: Determine code that would be used to complete code segments. | VAR-2.B: Traverse the elements in a 1D array.<br>CON-2.E: Represent iterative processes using a for loop. | Traversing Arrays for Loops |

| | |
|---|---|
| (A) | Incorrect. The reference `data[p + 1]` in the Boolean condition in the `if` statement causes an `ArrayIndexOutOfBoundsException` to be thrown when `p` has the value `data.length - 1`. |
| (B) | Incorrect. The reference `data[p - 1]` in the Boolean condition in the `if` statement causes an `ArrayIndexOutOfBoundsException` to be thrown when `p` has the value `0`. |
| (C) | Incorrect. The reference `data[p - 1]` in the Boolean condition in the `if` statement causes an `ArrayIndexOutOfBoundsException` to be thrown when `p` has the value `0`. |
| (D) | Incorrect. The reference `data[p + 1]` in the Boolean condition in the `if` statement causes an `ArrayIndexOutOfBoundsException` to be thrown when `p` has the value `data.length - 1`. |
| **(E)** | **Correct.** By definition, the first candidate for a local maximum is the element at index `1` (the second element in the array), and the last candidate is the element at index `data.length - 2` (the next to last element in the array). |

Question 19

| Skill | Learning Objective | Topic |
|---|---|---|
| 2.B: Determine the result or output based on statement execution order in a code segment without method calls (other than output). | VAR-2.G: For 2D array objects— a. Traverse using nested for loops. b. Traverse using nested enhanced for loops.<br>VAR-2.F: Represent collections of related primitive or object reference data using two-dimensional (2D) array objects. | Traversing 2D Arrays<br>2D Arrays |

| | |
|---|---|
| (A) | Incorrect. This would be the result if the Boolean expression in the outer `for` loop was `j < values.length - 1`. |
| (B) | Incorrect. This would be the result if the Boolean expression in the inner `for` loop was `k < values[0].length - 1`. |
| (C) | Incorrect. This would be the result of adding the original elements of the `values` array, without doubling the first element of every row. |
| **(D)** | **Correct.** The nested `for` loops traverse the two-dimensional array `values`. The first element of each row is doubled and then the sum of all elements is computed as 2+2+3+8+5+6=26. |
| (E) | Incorrect. This would be the result if the condition in the `if` statement was `j == 0` instead of `k == 0`. This would have the effect of doubling the elements in the first row of `values` instead of those in the first column. |

Question 20

| Skill | Learning Objective | Topic |
|---|---|---|
| 1.C: Determine code that would be used to interact with completed code. | VAR-2.C: Traverse the elements in a 1D array object using an enhanced for loop.<br>CON-2.I: For algorithms in the context of a particular specification that requires the use of array traversals—a. Identify standard algorithms. b. Modify standard algorithms. c. Develop an algorithm. | Enhanced for Loop for Arrays<br>Developing Algorithms Using Arrays |

| | |
|---|---|
| (A) | Incorrect. The instance variable `pages` is declared `private` in the `Book` class, so it must be accessed from outside the `Book` class using the accessor method `getPages`. |
| **(B)** | **Correct.** The enhanced `for` loop traverses `bookArr` and the loop control variable `b` is assigned `Book` objects. The instance variable `pages` is declared `private` in the `Book` class, so it must be accessed from outside the `Book` class using the accessor method `getPages`. The `if` statement compares the value returned by the call `b.getPages()` to the current value of `maxPages`. If the returned value is greater than `maxPages`, `maxPages` is updated with the new maximum value. |
| (C) | Incorrect. In the enhanced `for` loop, `b` is a `Book` object and cannot be used as an index. In addition, `Book` is a class name, not a variable, and cannot be indexed. Finally, the instance variable `pages` is declared `private` in the `Book` class, so it must be accessed from outside the `Book` class using the accessor method `getPages`. |
| (D) | Incorrect. In the enhanced `for` loop, `b` is a `Book` object and cannot be used as an index. The instance variable `pages` is declared `private` in the `Book` class, so it must be accessed from outside the `Book` class using the accessor method `getPages`. |
| (E) | Incorrect. In the enhanced `for` loop, `b` is a `Book` object and cannot be used as an index. |

Question 21

| Skill | Learning Objective | Topic |
|---|---|---|
| 2.C: Determine the result or output based on the statement execution order in a code segment containing method calls. | CON-2.I: For algorithms in the context of a particular specification that requires the use of array traversals—a. Identify standard algorithms. b. Modify standard algorithms. c. Develop an algorithm.<br>VAR-1.E: For String class—a. Create String objects. b. Call String methods. | Developing Algorithms Using Arrays String Methods |

| | |
|---|---|
| (A) | Incorrect. This would represent the contents of `resultOne` if the method assigned values to elements of `resultOne` in ascending, alphabetical order. |
| (B) | Incorrect. This would represent the contents of `resultOne` if the method assigned values to elements of `resultOne` in increasing order of string length. |
| (C) | Incorrect. This would represent the contents of `resultOne` if the inner `for` loop initialization were `k = j + 2` instead of `k = j + 1`. |
| **(D)** | **Correct.** The method assigns the shortest string that occurs in any element of `arr` between `arr[n]` and `arr[arr.length - 1]`, inclusive, to `result[n]`. The shortest string found between `arr[0]` and `arr[3]` is `"of"`, so `result[0]` is assigned the value `"of"`. The shortest string found between `arr[1]` and `arr[3]` is also `"of"`, so `result[1]` is also assigned the value `"of"`. The same is true for the part of the array that begins at index `2` and ends at index `3`, so `result[2]` is also assigned the value `"of"`. In the last iteration of the outer `for` loop, there are no values to consider after `arr[3]`, so `result[3]` is assigned the value `"spring"`. |
| (E) | Incorrect. This would represent the contents of `resultOne` if the method assigned values to elements of `resultOne` in decreasing order of string length. |

Question 22

| Skill | Learning Objective | Topic |
|---|---|---|
| 2.D: Determine the number of times a code segment will execute. | CON-2.I: For algorithms in the context of a particular specification that requires the use of array traversals—a. Identify standard algorithms. b. Modify standard algorithms. c. Develop an algorithm.<br>VAR-1.E: For String class—a. Create String objects. b. Call String methods. | Developing Algorithms<br>Using Arrays<br>String Methods |

| | |
|---|---|
| **(A)** | **Correct.** Line 12 is executed each time the variable `sm` is updated because a new smallest value is found. When `j` has the value `0`, `sm` is updated for `"day"` and `"of"`. When `j` has the value `1`, `sm` is updated for `"of"`. When `j` has the value `4`, `sm` is updated for `"year"`. When `j` has any of the values `2`, `3`, or `5`, `sm` is not updated. Line 12 is executed four times. |
| (B) | Incorrect. This would be the result if `sm` was updated once each time a string shorter than `arr[j]` was found instead of once each time a new smallest value was identified. |
| (C) | Incorrect. This would be the result if line 12 were executed once for each element of `arr`. |
| (D) | Incorrect. This would be the result if the method had no `if` statement and `sm` was updated once for each pair `arr[j]` and `arr[k]` encountered in the nested `for` loops. |
| (E) | Incorrect. This would be the result if the method had no `if` statement and the initialization in the inner `for` loop was `k = 1` instead of `k = j + 1`. |

Question 23

| Skill | Learning Objective | Topic |
|---|---|---|
| 1.B: Determine code that would be used to complete code segments. | VAR-2.G: For 2D array objects— a. Traverse using nested for loops. b. Traverse using nested enhanced for loops. | Traversing 2D Arrays |

| | |
|---|---|
| (A) | Incorrect. The outer `for` loop in this code segment declares `j`, a row of `arr`, as an `int` rather than an `int[]`. This code segment also attempts to print `j`, a row of `arr`, instead of `k`, an element of `j`. |
| (B) | Incorrect. The outer `for` loop in this code segment declares `j`, a row of `arr`, as an `int` rather than an `int[]`. |
| (C) | Incorrect. This code segment attempts to print `j`, a row of `arr`, instead of `k`, an element of `j`. |
| **(D)** | **Correct.** The outer `for` loop stores each row of the two-dimensional array in `j`, a one-dimensional array. The inner `for` loop stores each element of `j` in `k` and prints `k`. |
| (E) | Incorrect. This code segment uses `k`, an element of the two-dimensional array, as an index in a one-dimensional array. |

Question 24

| Skill | Learning Objective | Topic |
|---|---|---|
| 2.C: Determine the result or output based on the statement execution order in a code segment containing method calls. | MOD-2.H: Define the static variables that belong to the class. | Static Variables and Methods |

| | |
|---|---|
| (A) | Incorrect. This would be the result if `y` was not declared as a `static` variable and the `SomeClass` constructor did not increment `y`. |
| (B) | Incorrect. This would be the result if `y` was not declared as a `static` variable. |
| (C) | Incorrect. This would be the result if the `SomeClass` constructor did not increment `y`. |
| **(D)** | **Correct.** Since `y` is declared as a `static` variable, it is associated with the class and all objects of the class share the single variable `y`. Each time a new `SomeClass` object is instantiated, the value of `y` is incremented by `1`. After the `third` object is instantiated, the value of `y` is `3`. The call to `incrementY` with no parameter increments the value of `y` by `1`, and the call to `incrementY` with a parameter value of `10` adds `10` to the value of `y`, resulting in `14`. |
| (E) | Incorrect. This would be the result if the `getY` method returned `x` instead of `y`. |

Question 25

| Skill | Learning Objective | Topic |
|---|---|---|
| 2.C: Determine the result or output based on the statement execution order in a code segment containing method calls. | CON-2.F: For algorithms in the context of a particular specification that involves String objects— a. Identify standard algorithms. b. Modify standard algorithms. c. Develop an algorithm.<br>VAR-1.E: For String class— a. Create String objects. b. Call String methods.<br>CON-2.E: Represent iterative processes using a for loop. | Developing Algorithms Using Strings<br>String Methods<br>for Loops |

| | |
|---|---|
| (A) | Incorrect. This would be the result if the statement assigning a value to `temp` were `temp = str.substring(i - 1, i) + temp`. |
| (B) | Incorrect. This would be the result if the statement assigning a value to `temp` were `temp = str.substring(i - 1, i) + temp` and if the loop control variable in the `for` loop were initialized to `str.length()` instead of `str.length() - 1`. |
| (C) | Incorrect. This would be the result if the loop control variable in the `for` loop were initialized to `str.length()` instead of `str.length() - 1`. |
| **(D)** | **Correct.** When `i` has the value `4`, `temp` is assigned the value `"l"`. When `i` has the value `3`, `"p"` is appended to `temp`, resulting in `"lp"`. When `i` has the value `2`, `"p"` is appended to `temp` again, resulting in `"lpp"`. In the last iteration of the `for` loop, `i` has the value `1` and `"a"` is appended to temp, resulting in `"lppa"`. |
| (E) | Incorrect. This would be the result if the condition in the `for` loop were `i >= 0` instead of `i > 0`. |

Question 26

| Skill | Learning Objective | Topic |
|---|---|---|
| 2.D: Determine the number of times a code segment will execute. | CON-2.H: Compute statement execution counts and informal run-time comparison of iterative statements.<br>CON-2.E: Represent iterative processes using a for loop.<br>CON-2.G: Represent nested iterative processes. | Informal Code Analysis<br>for Loops<br>Nested Iteration |

| | |
|---|---|
| (A) | Incorrect. This would be the correct comparison if the initialization and Boolean condition in the inner `for` loop of code segment II were `k = 0` and `k <= n`, respectively. |
| (B) | Incorrect. This would be the correct comparison if the initializations in the outer and inner `for` loops of code segment II were `j = 0` and `k = 0`, respectively. |
| **(C)** | **Correct.** There are `m * n` iterations of the `for` loop in code segment I. In code segment II, the outer loop executes `m` times and the inner loop executes `n - 1` times for each iteration of the outer loop. There are `m * n - m` iterations of the inner loop in code segment II, so `"A"` is printed `m` more times than `"B"` is printed. |
| (D) | Incorrect. This would be the correct comparison if the Boolean conditions in the outer and inner `for` loops of code segment II were `j < m` and `k <= n`, respectively. |
| (E) | Incorrect. This would be the correct comparison if the Boolean condition in the inner `for` loop of code segment II were `k <= n`. |

Question 27

| Skill | Learning Objective | Topic |
|---|---|---|
| 5.D: Describe the initial conditions that must be met for a program segment to work as intended or described. | CON-1.G: Compare and contrast equivalent Boolean expressions. | Equivalent Boolean Expressions |

| | |
|---|---|
| (A) | Incorrect. The variable `c` will be assigned the value `true` when `a` and `b` both have the value `true` or when `a` has the value `false` and `b` has the value `true`. |
| (B) | Incorrect. The variable `c` will be assigned the value `false` when `b` has the value `false`, regardless of the value of `a`. |
| (C) | Incorrect. If `a` and `b` are both `true`, then `(a && b)` is `true`, `(!a && b)` is `false`, and the entire expression `true \|\| false` evaluates to `true`. |
| (D) | Incorrect. If a has the value `false` and `b` has the value `true`, then `(a && b)` is `false`, `(!a && b)` is `true`, and the entire expression `false \|\| true` evaluates to `true`. |
| **(E)** | **Correct.** When `b` has the value `false`, both of the expressions `(a && b)` and `(!a && b)` evaluate to `false`, regardless of the value of `a`. The entire expression evaluates to `false \|\| false`, or `false`. When `b` has the value `true`, one of the expressions `(a && b)` or `(!a && b)` evaluates to `true`. The entire expression, in this case, is either `true \|\| false` or `false \|\| true`, or `true`. A truth table can be used to summarize these results. |

Question 28

| Skill | Learning Objective | Topic |
|---|---|---|
| 2.C: Determine the result or output based on the statement execution order in a code segment containing method calls. | CON-2.F: For algorithms in the context of a particular specification that involves String objects— a. Identify standard algorithms. b. Modify standard algorithms. c. Develop an algorithm.<br>VAR-1.E: For String class— a. Create String objects. b. Call String methods.<br>CON-2.C: Represent iterative processes using a while loop. | Developing Algorithms Using Strings<br>String Methods<br>while Loops |

| | |
|---|---|
| (A) | Incorrect. This value would be returned if the first statement in the `while` loop was `a = a.substring(0, x)`. |
| **(B)** | **Correct.** The method `abMethod(String a, String b)` removes all non-overlapping occurrences of string `b` from string `a` and returns the resulting `String`. It does this by repeatedly setting `x` to the index of an occurrence of `b` in `a`, then assigning `a` the result of the concatenation of the parts of `a` before and after the occurrence of `b`. The method call `abMethod("sing the song", "ng")` removes all occurrences of `"ng"` from `"sing the song"`, returning `"si the so"`. |
| (C) | Incorrect. This value would be returned if the statements inside the `while` loop were executed only one time. |
| (D) | Incorrect. This value would be returned if the first statement in the `while` loop was `a = a.substring(0, x) + a.substring(x + b.length() - 1)`. |
| (E) | Incorrect. The method does not attempt to access a substring that begins at an index less than `0` or ends at an index greater than `length - 1`. |

Question 29

| Skill | Learning Objective | Topic |
|---|---|---|
| 2.C: Determine the result or output based on the statement execution order in a code segment containing method calls. | CON-2.O: Determine the result of executing recursive methods. | Recursion |

| | |
|---|---|
| (A) | Incorrect. This would be the result if the value returned in cases other than the base case was `calcMethod(num / 2)`. |
| (B) | Incorrect. This would be the result if the value returned in cases other than the base case was `num + calcMethod(num % 2)`. |
| (C) | Incorrect. This would be the result if the value returned in the base case was `0` instead of `10`. |
| (D) | Incorrect. This would be the result if the base case was `num == 2` instead of `num == 0`. |
| **(E)** | **Correct.** The result of the method call `calcMethod(16)` is `16 + calcMethod(8)`. The result of the method call `calcMethod(8)` is `8 + calcMethod(4)`. The recursive calls continue until the call `calcMethod(0)`, which returns the value `10`. The result of the method call `calcMethod(16)` can be calculated as 16+8+4+2+1+10=41. |

Question 30

| Skill | Learning Objective | Topic |
|---|---|---|
| 1.C: Determine code that would be used to interact with completed program code. | MOD-3.B: Create an inheritance relationship from a subclass to the superclass. | Writing Constructors for Subclasses |

| | |
|---|---|
| (A) | Incorrect. Calling the no-argument superclass constructor would initialize both `height` and `width` to `1`, which is not necessarily the intended behavior. |
| **(B)** | **Correct.** A call to the one-argument superclass constructor with the single parameter `x` will set both the `height` and the `width` instance variables to `x`. |
| (C) | Incorrect. This statement would result in a compiler error. Constructors cannot be called by name from other constructors. The `super` keyword is used to call the constructor of the superclass. |
| (D) | Incorrect. This statement would result in a compiler error. Constructors cannot be called by name from other constructors. In addition, the `Square` class does not define a constructor with a matching signature. |
| (E) | Incorrect. This code segment would result in a compiler error. The instance variables `height` and `width` are defined as `private` in the superclass and cannot be accessed directly from the subclass. |

Question 31

| Skill | Learning Objective | Topic |
|---|---|---|
| 1.B: Determine code that would be used to complete code segments. | VAR-2.B: Traverse the elements in a 1D array.<br>VAR-2.C: Traverse the elements in a 1D array object using an enhanced for loop.<br>CON-2.I: For algorithms in the context of a particular specification that requires the use of array traversals—a. Identify standard algorithms. b. Modify standard algorithms. c. Develop an algorithm. | Traversing Arrays<br>Enhanced for Loop for Arrays<br>Developing Algorithms Using Arrays |

| | |
|---|---|
| **(A)** | **Correct.** In code segment I, `i` takes on the values `-1` through `nums.length - 2`, inclusive, in the `while` loop. Since `i` is incremented before the `if` statement, the array elements `nums[0]` through `nums[nums.length - 1]` are compared to `0`. In code segment II, array element `nums[0]` is excluded since the first iteration of the `for` loop accesses `nums[1]`. In code segment III, the variable `i` represents an element of the array rather than an index. |
| (B) | Incorrect. In code segment I, `i` takes on the values `-1` through `nums.length - 2`, inclusive, in the `while` loop. Since `i` is incremented before the `if` statement, the array elements `nums[0]` through `nums[nums.length - 1]` are compared to `0`. In code segment II, array element `nums[0]` is excluded since the first iteration of the `for` loop accesses `nums[1]`. |
| (C) | Incorrect. In code segment II, array element `nums[0]` is excluded since the first iteration of the `for` loop accesses `nums[1]`. |
| (D) | Incorrect. In code segment III, the variable `i` represents an element of the array rather than an index. |
| (E) | Incorrect. In code segment II, array element `nums[0]` is excluded since the first iteration of the `for` loop accesses `nums[1]`. In code segment III, the variable `i` represents an element of the array rather than an index. |

Question 32

| Skill | Learning Objective | Topic |
|---|---|---|
| 2.C: Determine the result or output based on the statement execution order in a code segment containing method calls. | MOD-3.D: Call methods in an inheritance relationship.<br>MOD-2.B: Define instance variables for the attributes to be initialized through the constructors of a class.<br>MOD-3.B: Create an inheritance relationship from a subclass to the superclass. | Polymorphism<br>Constructors<br>Creating Superclasses and Subclasses |

| | |
|---|---|
| (A) | Incorrect. This output would be printed if `obj` was instantiated by calling the `ClassA` constructor instead of the `ClassB` constructor. |
| **(B)** | **Correct.** Since `obj` is instantiated as a `ClassB` object but the `showValue` method is not defined in `ClassB`, the `showValue` method call accesses the `showValue` method in the superclass, `ClassA`. Since the `getValue` method is defined in `ClassB`, the `getValue` method call accesses the `getValue` method in the subclass, `ClassB`, and `"B"` is printed. |
| (C) | Incorrect. This output would be printed if the `getValue` method in `ClassB` returned `super.getValue() + "B"` instead of `"B"`. The value returned by the `getValue` method of `ClassA` `("A")` would be concatenated with `"B"` and the `String "AB"` would be printed by the `showValue` method. |
| (D) | Incorrect. This output would be printed if the `getValue` method in `ClassB` returned `"B" + super.getValue()` instead of `"B"`. The String `"B"` would be concatenated with the value returned by the `getValue` method of `ClassA` and the `String "BA"` would be printed by the `showValue` method. |
| (E) | Incorrect. The code compiles without error since `obj` is declared as an object of type `ClassA` and the `showValue` method is defined in `ClassA`. |

Question 33

| Skill | Learning Objective | Topic |
|---|---|---|
| 2.B: Determine the result or output based on statement execution order in a code segment without method calls (other than output). | VAR-2.G: For 2D array objects— a. Traverse using nested for loops. b. Traverse using nested enhanced for loops. | Traversing 2D Arrays |

| | |
|---|---|
| (A) | Incorrect. This output would be generated if `col` and `row` were initialized to `0` and `col`, respectively, in the `for` loops. |
| (B) | Incorrect. This output would be generated if `col` and `row` were initialized to `1` and `0`, respectively, in the `for` loops. |
| (C) | Incorrect. This output would be generated if `col` and `row` were initialized to `0` and `1`, respectively, in the `for` loops. |
| (D) | Incorrect. This output would be generated if the positions of the two `for` loop headers were reversed, with `row` as the loop control variable in the outer `for` loop and `col` as the loop control variable in the inner `for` loop. |
| **(E)** | **Correct.** The code segment performs a column-major traversal of the array, beginning with the second column and the second row. For each column, all values in that column after the value in the first row are printed on a single line. The `println` method call causes data from subsequent columns to appear on new lines in the output. |

Question 34

| Skill | Learning Objective | Topic |
|---|---|---|
| 5.B: Explain why a code segment will not compile or work as intended. | VAR-2.E: For ArrayList objects— a. Traverse using a for or while loop. B. Traverse using an enhanced for loop. | Traversing ArrayLists |

| | |
|---|---|
| (A) | Incorrect. The order of addition and removal does not matter, since the value to be added to `returnList` is stored in a separate variable `num` prior to its removal from `numList`. |
| (B) | Incorrect. When there are no matches, the method returns an empty `ArrayList`. |
| (C) | Incorrect. The `get` and `remove` methods are always called with a valid index, since the body of the `while` loop is executed for values of `i` between `0` and the current value of `numList.size() - 1`. |
| (D) | Incorrect. The expression `num % key == 0` is correctly used to identify values of `num` that are divisible by `key`. |
| **(E)** | **Correct.** When the element at position `i` is removed from `numList`, subsequent elements are shifted left. After the removal, the element that used to be at position `i + 1` is now at position `i`. Because the method increments `i` regardless of whether the element at position `i` was removed, the method does not always work as intended. For example, if two adjacent elements are both divisible by `key`, only the first element is removed. The method could be corrected by incrementing `i` only when the element at position `i` is not removed or by decrementing `i` when an element is removed. |

Question 35

| Skill | Learning Objective | Topic |
|---|---|---|
| 1.B: Determine code that would be used to complete code segments. | CON-2.I: For algorithms in the context of a particular specification that requires the use of array traversals— a. Identify standard algorithms. b. Modify standard algorithms. c. Develop an algorithm. | Developing Algorithms Using Arrays |

| | |
|---|---|
| **(A)** | **Correct.** For each element `arr[j]`, the inner `for` loop counts the number of times that `arr[j]` appears in `arr` and stores the result in `valCount`. It does this by evaluating the condition `arr[j] == arr[k]` and incrementing `valCount` when the condition evaluates to `true`. After the inner loop completes, the method evaluates the condition `valCount > modeCount`. If the condition evaluates to `true`, a new mode has been found and `mode` and `modeCount` are updated. |
| (B) | Incorrect. The replacement for `/*` *missing condition 2* `*/` is incorrect. It would update the mode value to be returned only if it was less common than values considered previously. |
| (C) | Incorrect. The replacement for `/*` *missing condition 1* `*/` is incorrect. Instead of counting the number of times that `arr[j]` appears in `arr`, it would count the number of times that values different than `arr[j]` appear in `arr`. |
| (D) | Incorrect. The replacements for `/*` *missing condition 1* `*/` and `/*` *missing condition 2* `*/` are incorrect. Because of the incorrect replacement for `/*` *missing condition 1* `*/`, the method would count the number of times that values different than `arr[j]` appear in `arr` instead of the number of times that `arr[j]` appears in `arr`. Because of the incorrect replacement for `/*` *missing condition 2* `*/`, the method would update the mode value to be returned only if it was less common than values considered previously. |
| (E) | Incorrect. The replacements for `/*` *missing condition 1* `*/` and `/*` *missing condition 2* `*/` are incorrect. Because of the incorrect replacement for `/*` *missing condition 1* `*/`, the method would count the number of times that values different than `arr[j]` appear in `arr` instead of the number of times that `arr[j]` appears in `arr`. Because of the incorrect replacement for /* *missing condition 2* `*/`, the method would update the mode value to be returned only if it occurred either more or less frequently than values considered previously. |

Question 36

| Skill | Learning Objective | Topic |
|---|---|---|
| 5.D: Describe the initial conditions that must be met for a program segment to work as intended or described. | CON-2.E: Represent iterative processes using a for loop.<br>CON-2.C: Represent iterative processes using a while loop.<br>CON-1.A: Evaluate arithmetic expressions in program code. | for Loops<br>while Loops<br>Expressions and Assignment Statements |

| | |
|---|---|
| (A) | Incorrect. When `a` has the value `6` and `b` has the value `4`, `methodOne` returns `1` and `methodTwo` returns `2`. In general, if `a % b` is not equal to `0`, `methodOne` returns `a / b` and `methodTwo` returns `a / b + 1`. |
| (B) | Incorrect. When `a` has the value `21` and `b` has the value `5`, `methodOne` returns `4` and `methodTwo` returns `5`. In general, if `a % b` is not equal to `0`, `methodOne` returns `a / b` and `methodTwo` returns `a / b + 1`. |
| (C) | Incorrect. When `a` has the value `10` and `b` has the value `3`, `methodOne` returns `3` and `methodTwo` returns `4`. In general, if `a % b` is not equal to `0`, `methodOne` returns `a / b` and `methodTwo` returns `a / b + 1`. |
| **(D)** | **Correct.** The body of the `for` loop in `methodOne` is executed `a / b` times. The body of the `while` loop in `methodTwo` is executed `a / b` times only when `a % b` is equal to `0`. When `a % b` is not equal to `0`, the body of the `while` loop in `methodTwo` is executed an additional time. For example, when `a` has the value `11` and `b` has the value `5`, `a / b` evaluates to `2` and the `for` loop is executed two times but the `while` loop is executed three times. |
| (E) | Incorrect. When `a` has the value `7` and `b` has the value `3`, `methodOne` returns `2` and `methodTwo` returns `3`. In general, if `a % b` is not equal to `0`, `methodOne` returns `a / b` and `methodTwo` returns `a / b + 1`. |

Question 37

| Skill | Learning Objective | Topic |
|---|---|---|
| 5.A: Describe the behavior of a given segment of program code. | CON-2.C: Represent iterative processes using a while loop.<br>CON-2.D: For algorithms in the context of a particular specification that does not require the use of traversals— a. Identify standard algorithms. b. Modify standard algorithms. c. Develop an algorithm. | while Loops |

| | |
|---|---|
| (A) | Incorrect. The product of `num2` and `num3` could be computed by an algorithm that adds `num2` to `num1` a total of `num3` times. |
| (B) | Incorrect. The product of `num2` and `num3 - 1` could be computed by an algorithm that adds `num2` to `num1` a total of `num3 - 1` times. |
| (C) | Incorrect. The sum of `num2` and `num3` could be computed by an algorithm that adds `1` to `num2` a total of `num3` times. |
| (D) | Incorrect. This would correctly describe the behavior of the code segment if the Boolean condition in the `while` loop was `num2 <= num3`. |
| **(E)** | **Correct.** Each iteration of the `while` loop adds `num2` to `num1` and then increments `num2`. The last value assigned to `num2` and added to `num1` is `num3 - 1`. Since each value of `num2` is added to `num1`, the code segment computes the sum of the integers from `num2` to `num3 - 1`. |

Question 38

| Skill | Learning Objective | Topic |
|---|---|---|
| 1.C: Determine code that would be used to interact with completed program code. | VAR-2.D: Represent collections of related object reference data using ArrayList objects.<br>VAR-2.E: For ArrayList objects— a. Traverse using a for or while loop. b. Traverse using an enhanced for loop. | ArrayList Methods<br>Traversing ArrayLists |

| | |
|---|---|
| (A) | Incorrect. Option II is correct. The code segment uses an enhanced `for` loop to traverse the `valueList` array. The statement inside the loop calls the `getNum` method to access the `num` instance variable. |
| (B) | Incorrect. Option I is correct. The code segment uses a `for` loop to traverse the `valueList` array. The statement inside the loop calls the `get` method to access a `Value` object and then calls the `getNum` method to access the `num` instance variable. |
| (C) | Incorrect. Option I is correct. The code segment uses a `for` loop to traverse the `valueList` array. The statement inside the loop calls the `get` method to access a `Value` object and then calls the `getNum` method to access the `num` instance variable. Option II is correct. The code segment uses an enhanced `for` loop to traverse the `valueList` array. The statement inside the loop calls the `getNum` method to access the `num` instance variable. Option III is incorrect. The code segment causes a compilation error because the `getNum` method must be called using the dot operator, not by passing the object reference as an argument. |
| **(D)** | **Correct.** Option I is correct. The code segment uses a `for` loop to traverse the `valueList` array. The statement inside the loop calls the `get` method to access a `Value` object and then calls the `getNum` method to access the `num` instance variable. Option II is correct. The code segment uses an enhanced `for` loop to traverse the `valueList` array. The statement inside the loop calls the `getNum` method to access the `num` instance variable. Option III is incorrect. The code segment causes a compilation error because the `getNum` method must be called using the dot operator, not by passing the object reference as an argument. |
| (E) | Incorrect. Option II is correct. The code segment uses an enhanced `for` loop to traverse the `valueList` array. The statement inside the loop calls the `getNum` method to access the `num` instance variable. Option III is incorrect. The code segment causes a compilation error because the `getNum` method must be called using the dot operator, not by passing the object reference as an argument. |

Question 39

| Skill | Learning Objective | Topic |
|---|---|---|
| 4.A: Use test-cases to find errors or validate results. | CON-2.O: Determine the result of executing recursive methods. | Recursion |

| | |
|---|---|
| (A) | Incorrect. This method call returns `false` because the first character is lexicographically less than the second character of the string. |
| (B) | Incorrect. This method call returns `false` because the first character is lexicographically less than the second character of the string. |
| (C) | Incorrect. This method call returns `false` because the first character is lexicographically less than the second character of the string. |
| **(D)** | **Correct.** If the first character of `str` is lexicographically greater than the second character of `str`, the method returns the result of the recursive call with a parameter that contains all but the first character of `str`. If the first character of `str` is lexicographically less than or equal to the second character of `str`, the method returns `false`. If no such character pair (where the first character of `str` is lexicographically less than or equal to the second character of `str`) is found, the base case is reached and the value `true` is returned. |
| (E) | Incorrect. This method call returns `false` because the third character is lexicographically less than the fourth character of the string. |

Question 40

| Skill | Learning Objective | Topic |
|---|---|---|
| 5.C: Explain how the result of program code changes, given a change to the initial code. | MOD-3.D: Call methods in an inheritance relationship. | Polymorphism |

| | |
|---|---|
| **(A)** | **Correct.** At compile time, methods in or inherited by the declared type determine the correctness of a non-static method call. In line 1, `obj1` is declared as an object of type `A`. Therefore, at compile time, there must be a `message` method in class `A` or its superclass. If the `message` method in class `A` is removed, the statement in line 3 will no longer compile. |
| (B) | Incorrect. At compile time, methods in or inherited by the declared type determine the correctness of a non-static method call. In line 2, `obj2` is declared as an object of type `B`. Therefore, at compile time, there must be a `message` method in class `B` or its superclass. Since the `message` method in class `B` has not been removed, this statement will not cause a compiler error. |
| (C) | Incorrect. The `message` method in class `A` is not executed in the original code segment. At run-time, the method in the actual object type is executed for a non-static method call. In line 1, `obj1` is instantiated as an object of type `B`. Therefore, in line 3 at run-time, the `message` method of class `B` rather than the `message` method of class `A` is executed. |
| (D) | Incorrect. The `message` method in class `A` is not executed in the original code segment. At run-time, the method in the actual object type is executed for a non-static method call. In line 2, `obj2` is instantiated as an object of type `B`. Therefore, in line 4 at run-time, the `message` method of class `B` rather than the `message` method of class `A` is executed. |
| (E) | Incorrect. At compile time, methods in or inherited by the declared type determine the correctness of a non-static method call. In line 1, `obj1` is declared as an object of type `A`. Therefore, at compile time, there must be a `message` method in class `A` or its superclass. If the `message` method in class `A` is removed, the code will no longer compile. |

## Answer Key and Question Alignment to Course Framework

| Multiple-Choice Question | Answer | Skill | Learning Objective | Topic |
|---|---|---|---|---|
| 1 | C | 2.B | CON-1.A | Expressions and Assignment Statements |
| 2 | D | 5.A | CON-2.B | Compound Boolean Expressions |
| | | | CON-2.A | if-else Statements |
| 3 | C | 1.C | CON-1.D | Using the Math Class |
| | | | CON-1.A | Expressions and Assignment Statements |
| | | | CON-1.C | Casting and Ranges of Variables |
| 4 | D | 2.B | CON-2.B | Compound Boolean Expressions |
| | | | CON-2.A | if Statements and Control Flow |
| | | | | if-else Statements |
| 5 | E | 1.C | MOD-1.C | Creating and Storing Objects (Instantiation) |
| 6 | A | 2.A | CON-1.A | Expressions and Assignment Statements |
| | | | CON-1.C | Casting and Ranges of Variables |
| 7 | E | 4.C | CON-2.B | Compound Boolean Expressions |
| | | | CON-2.A | if-else Statements |
| | | | CON-1.F | Compound Boolean Expressions |
| 8 | D | 4.B | VAR-1.G | Scope and Access |
| | | | MOD-1.C | Creating and Storing Objects (Instantiation) |
| 9 | B | 2.C | VAR-2.D | ArrayList Methods |
| 10 | D | 4.A | CON-2.E | for Loops |
| | | | CON-1.A | Expressions and Assignment Statements |
| 11 | C | 1.B | CON-2.C | while Loops |
| | | | CON-1.B | Compound Assignment Operators |
| 12 | B | 1.C | CON-1.H | Comparing Objects |
| | | | CON-2.A | if Statements and Control Flow |
| 13 | E | 4.C | CON-1.G | Equivalent Boolean Expressions |
| 14 | D | 1.B | CON-2.A | if Statements and Control Flow |
| | | | | else if Statements |
| 15 | E | 5.A | CON-1.D | Using the Math Class |
| 16 | B | 2.C | CON-2.O | Recursion |
| | | | CON-2.E | for Loops |

| Multiple-Choice Question | Answer | Skill | Learning Objective | Topic |
|---|---|---|---|---|
| 17 | D | 2.C | MOD-3.B | Overriding Methods |
| | | | | super Keyword |
| | | | MOD-3.D | Polymorphism |
| 18 | E | 1.B | VAR-2.B | Traversing Arrays |
| | | | CON-2.E | for Loops |
| 19 | D | 2.B | VAR-2.G | Traversing 2D Arrays |
| | | | VAR-2.F | 2D Arrays |
| 20 | B | 1.C | VAR-2.C | Enhanced for Loop for Arrays |
| | | | CON-2.I | Developing Algorithms Using Arrays |
| 21 | D | 2.C | CON-2.I | Developing Algorithms Using Arrays |
| | | | VAR-1.E | String Methods |
| 22 | A | 2.D | CON-2.I | Developing Algorithms Using Arrays |
| | | | VAR-1.E | String Methods |
| 23 | D | 1.B | VAR-2.G | Traversing 2D Arrays |
| 24 | D | 2.C | MOD-2.H | Static Variables and Methods |
| 25 | D | 2.C | CON-2.F | Developing Algorithms Using Strings |
| | | | VAR-1.E | String Methods |
| | | | CON-2.E | for Loops |
| 26 | C | 2.D | CON-2.H | Informal Code Analysis |
| | | | CON-2.E | for Loops |
| | | | CON-2.G | Nested Iteration |
| 27 | E | 5.D | CON-1.G | Equivalent Boolean Expressions |
| 28 | B | 2.C | CON-2.F | Developing Algorithms Using Strings |
| | | | VAR-1.E | String Methods |
| | | | CON-2.C | while Loops |
| 29 | E | 2.C | CON-2.O | Recursion |
| 30 | B | 1.C | MOD-3.B | Writing Constructors for Subclasses |
| 31 | A | 1.B | VAR-2.B | Traversing Arrays |
| | | | VAR-2.C | Enhanced for Loop for Arrays |
| | | | CON-2.I | Developing Algorithms Using Arrays |
| 32 | B | 2.C | MOD-3.D | Polymorphism |
| | | | MOD-2.B | Constructors |
| | | | MOD-3.B | Creating Superclasses and Subclasses |
| 33 | E | 2.B | VAR-2.G | Traversing 2D Arrays |

| Multiple-Choice Question | Answer | Skill | Learning Objective | Topic |
|---|---|---|---|---|
| 34 | E | 5.B | VAR-2.E | Traversing ArrayLists |
| 35 | A | 1.B | CON-2.I | Developing Algorithms Using Arrays |
| 36 | D | 5.D | CON-2.E<br>CON-2.C<br>CON-1.A | for Loops<br>while Loops<br>Expressions and Assignment Statements |
| 37 | E | 5.A | CON-2.C<br>CON-2.D | while Loops |
| 38 | D | 1.C | VAR-2.D<br>VAR-2.E | ArrayList Methods<br>Traversing ArrayLists |
| 39 | D | 4.A | CON-2.O | Recursion |
| 40 | A | 5.C | MOD-3.D | Polymorphism |

## Applying the Scoring Criteria

Apply the question scoring criteria first, which always takes precedence. Penalty points can only be deducted in a part of the question that has earned credit via the question rubric. No part of a question (a, b, c) may have a negative point total. A given penalty can be assessed only once for a question, even if it occurs multiple times or in multiple parts of that question. A maximum of 3 penalty points may be assessed per question.

**1-Point Penalty**

v) Array/collection access confusion (`[] get`)

w) Extraneous code that causes side-effect (e.g., printing to output, incorrect precondition check)

x) Local variables used but none declared

y) Destruction of persistent data (e.g., changing value referenced by parameter)

z) Void method or constructor that returns a value

**No Penalty**

- Extraneous code with no side-effect (e.g., valid precondition check, no-op)
- Spelling/case discrepancies where there is no ambiguity*
- Local variable not declared provided other variables are declared in some part
- `private` or `public` qualifier on a local variable
- Missing `public` qualifier on class or constructor header
- Keyword used as an identifier
- Common mathematical symbols used for operators (× • ÷ ≤ ≥ <> ≠)
- `[]` vs. `()` vs. `<>`
- `=` instead of `==` and vice versa
- `length/size` confusion for array, `String`, `List`, or `ArrayList`; with or without `( )`
- Extraneous `[]` when referencing entire array
- `[i,j]` instead of `[i][j]`
- Extraneous size in array declaration, e.g., `int[size] nums = new int[size];`
- Missing `;` where structure clearly conveys intent
- Missing `{ }` where indentation clearly conveys intent
- Missing `( )` on parameter-less method or constructor invocations
- Missing `( )` around `if` or `while` conditions

**Spelling and case discrepancies for identifiers fall under the "No Penalty" category only if the correction can be **unambiguously** inferred from context, for example, "ArayList" instead of "ArrayList". As a counterexample, note that if the code declares "`int G=99, g=0;`", then uses "`while (G < 10)`" instead of "`while (g < 10)`", the context does **not** allow for the reader to assume the use of the lower case variable.*

## Question 1: Methods and Control Structures — 9 points

**Learning Objectives:** CON-1.A CON-1.C CON-1.E CON-2.A CON-2.C CON-2.E MOD-1.G MOD-2.F

**Canonical solution**

(a) — 3 points

```
public static int hailstoneLength(int n)
{
   int count = 1;
   while (n > 1)
   {
      if (n % 2 == 0)
      {
         n = n / 2;
      }
      else
      {
         n = 3 * n + 1;
      }
      count++;
   }
   return count;
}
```

(b) — 2 points

```
public static boolean isLongSeq(int n)
{
   return hailstoneLength(n) > n;
}
```

(c) — 4 points

```
public static double propLong(int n)
{
   int count = 0;
   for (int i = 1; i <= n; i++)
   {
      if (isLongSeq(i))
      {
         count++;
      }
   }
   return (double) count / n;
}
```

**(a)** `hailstoneLength`

| | Scoring Criteria | Decision Rules | |
|---|---|---|---|
| 1 | Loops from given starting value `n` until the sequence terminates, using updated values for the current term | **Responses still earn the point even if they...**<br>• update `n` incorrectly. | **1 point**<br>3.C<br>CON-2.C |
| 2 | Computes the next value | **Responses still earn the point even if they...**<br>• use a correct formula in an incorrect case. | **1 point**<br>3.C<br>CON-1.A |
| 3 | Uses correct formula for next value depending on even/odd | | **1 point**<br>3.C<br>CON-2.A |
| | | **Total for part (a)** | **3 points** |

**(b)** `isLongSeq`

| | Scoring Criteria | Decision Rules | |
|---|---|---|---|
| 4 | Calls `hailstoneLength` | | **1 point**<br>3.A<br>MOD-1.G |
| 5 | Correctly compares length and starting value to determine return value | **Responses still earn the point even if they...**<br>• call `hailstoneLength` incorrectly. | **1 point**<br>3.C<br>CON-1.E |
| | | **Total for part (b)** | **2 points** |

**(c)** `propLong`

| | Scoring Criteria | Decision Rules | |
|---|---|---|---|
| 6 | Calls `isLongSeq` in the context of a loop | | **1 point**<br>3.A<br>MOD-1.G |
| 7 | Loops `1` to `n` (*no bounds errors*) | | **1 point**<br>3.C<br>CON-2.E |
| 8 | Calculates `double` proportion | **Responses still earn the point even if they...**<br>• use incorrect values for the count of long sequences or `n`. | **1 point**<br>3.C<br>CON-1.C |
| 9 | Returns correctly calculated value | | **1 point**<br>3.B<br>MOD-2.F |
| | | **Total for part (c)** | **4 points** |
| | **Question-specific penalties** | | |
| | None | | |
| | | **Total for question 1** | **9 points** |

## Question 2: Class Design — 9 points

**Learning Objectives:** MOD-2.B MOD-2.D CON-1.B CON-1.D CON-2.A

**Canonical solution**

9 points

```
public class GameSpinner
{
   private int sectors;
   private int previousSpin = 0;
   private int currentLength = 0;

   public GameSpinner(int s)
   {
      sectors = s;
   }

   public int spin()
   {
      int newSpin = (int)(Math.random() * sectors) + 1;

      if (newSpin == previousSpin)
      {
         currentLength++;
      }
      else
      {
         previousSpin = newSpin;
         currentLength = 1;
      }
      return newSpin;

   }

   public int currentRun()
   {
      return currentLength;
   }

}
```

| | | | |
|---|---|---|---|
| | `GameSpinner` | | |

| | Scoring Criteria | Decision Rules | |
|---|---|---|---|
| 1 | Declares all appropriate `private` instance variables | | 1 point<br>3.B<br>MOD-2.B |
| 2 | Declares method headers:<br>`public int spin()` and<br>`public int currentRun()` | | 1 point<br>3.B<br>MOD-2.D |
| 3 | Declares header:<br>`GameSpinner(int __)`<br>(*must not be* `private`) | | 1 point<br>3.B<br>MOD-2.B |
| 4 | Constructor initializes instance variable for number of sectors using parameter. Instance variables for previous spin and length of current run initialized correctly when declared or in constructor with default values. | **Responses still earn the point even if they...**<br>• declare instance variables incorrectly. | 1 point<br>3.B<br>MOD-2.B |
| 5 | Computes random integer [1, number of sectors] | | 1 point<br>3.A<br>CON-1.D |
| 6 | Compares new spin and last spin to determine required updates to state | **Responses still earn the point even if they...**<br>• use an incorrectly computed random integer for new spin; or<br>• incorrectly declare the instance variable intended to store last spin. | 1 point<br>3.C<br>CON-2.A |
| 7 | Updates instance variable that represents length of current run appropriately if new spin and previous spin are the same | **Responses still earn the point even if they...**<br>• incorrectly compare new spin and last spin. | 1 point<br>3.B<br>MOD-2.D |
| 8 | Updates previous spin and length of current run appropriately when new spin differs from the previous spin | **Responses still earn the point even if they...**<br>• incorrectly compare new spin and last spin. | 1 point<br>3.C<br>CON-1.B |
| 9 | `currentRun` returns updated instance variable value | **Responses still earn the point even if they...**<br>• incorrectly update instance variables in the `spin` method. | 1 point<br>3.B<br>MOD-2.D |
| | **Question-specific penalties** | | |
| | None | | |
| | | **Total for question 2** | **9 points** |

## Question 3: Array/ArrayList — 9 points

**Learning Objectives:** VAR-1.E.b VAR-2.D VAR-2.E.a MOD-1.G CON-2.F.a CON-2.J.a CON-2.K

**Canonical solution**

**(a)** — 6 points

```
public void addReview(ProductReview prodReview)
{
   reviewList.add(prodReview);

   String prodName = prodReview.getName();
   boolean found = false;
   for (String n : productList)
   {
      if (n.equals(prodName))
      {
         found = true;
      }
   }
   if (!found)
   {
      productList.add(prodName);
   }
}
```

**(b)** — 3 points

```
public int getNumGoodReviews(String prodName)
{
   int numGoodReviews = 0;
   for (ProductReview prodReview : reviewList)
   {
      if (prodName.equals(prodReview.getName()))
      {
         String review = prodReview.getReview();
         if (review.indexOf("best") >= 0)
         {
            numGoodReviews++;
         }
      }
   }
   return numGoodReviews;
}
```

**(a)** `addReview`

| | Scoring Criteria | Decision Rules | |
|---|---|---|---|
| 1 | Adds a `ProductReview` object to `reviewList` | **Responses still earn the point even if they...**<br>• add a `ProductReview` object other than the one referenced by the parameter `prodReview`. | **1 point**<br>3.D<br>VAR-2.D |
| 2 | Gets product name of review to be added | | **1 point**<br>3.A<br>MOD-1.G |
| 3 | Traverses `productList` (*no bounds errors*) | **Responses still earn the point even if they...**<br>• use a `for`, an enhanced `for`, or a `while` loop. | **1 point**<br>3.D<br>VAR-2.E.a |
| 4 | Compares name in `productList` with name from review to be added | **Responses still earn the point even if they...**<br>• use an incorrectly accessed value for either name. | **1 point**<br>3.C<br>VAR-1.E.b |
| 5 | Adds new product name to `productList` | **Responses still earn the point even if they...**<br>• add the new product name under the wrong conditions; or<br>• add an incorrectly accessed value for the new product name | **1 point**<br>3.D<br>VAR-2.D |
| 6 | Correctly adds product name to `productList` if and only if the product name is not already in `productList` | | **1 point**<br>3.D<br>CON-2.K |
| | | **Total for part (a)** | **6 points** |

**(b)** `getNumGoodReviews`

| | Scoring Criteria | Decision Rules | |
|---|---|---|---|
| 7 | Traverses `reviewList` (*no bounds errors*) | **Responses still earn the point even if they...**<br>• use a `for`, an enhanced `for`, or a `while` loop. | **1 point**<br>3.D<br>VAR-2.E.a |
| 8 | Selects all and only reviews with matching product names that contain `"best"` | | **1 point**<br>3.C<br>CON-2.F.a |
| 9 | Returns correct count of good reviews | | **1 point**<br>3.D<br>CON-2.J.a |
| | | **Total for part (b)** | **3 points** |
| | **Question-specific penalties** | | |
| | None | | |
| | | **Total for question 3** | **9 points** |

## Question 4: 2D Array — 9 points

**Learning Objectives:** MOD-1.D.b MOD-1.G CON-1.H CON-2.A CON-2.N.c VAR-2.F VAR-2.G.a

**Canonical solution**

**(a)** — **5 points**

```
public Theater(int seatsPerRow, int tier1Rows,
               int tier2Rows)
{
   theaterSeats =
      new Seat[tier1Rows + tier2Rows][seatsPerRow];
   for (int r = 0; r < tier1Rows + tier2Rows; r++)
   {
      for (int c = 0; c < seatsPerRow; c++)
      {
         if (r < tier1Rows)
         {
            theaterSeats[r][c] = new Seat(true, 1);
         }
         else
         {
            theaterSeats[r][c] = new Seat(true, 2);
         }
      }
   }
}
```

**(b)** — **4 points**

```
public boolean reassignSeat(int fromRow, int fromCol,
                            int toRow, int toCol)
{
   Seat toS = theaterSeats[toRow][toCol];
   if (!toS.isAvailable())
   {
      return false;
   }

   Seat fromS = theaterSeats[fromRow][fromCol];
   if (toS.getTier() < fromS.getTier())
   {
      return false;
   }

   toS.setAvailability(false);
   fromS.setAvailability(true);
   return true;
}
```

**(a)** `Theater`

| | Scoring Criteria | Decision Rules | |
|---|---|---|---|
| 1 | Instantiates a new `Seat[][]` with the correct number of rows and columns, based on parameters | | **1 point** 3.E VAR-2.F |
| 2 | Traverses the `theaterSeats` array (*no bounds errors*) | | **1 point** 3.E VAR-2.G.a |
| 3 | Instantiates a new `Seat` object with a tier and availability status | **Responses still earn the point even if they...**<br>• incorrectly assign the new object to a `theaterSeats` element. | **1 point** 3.A MOD-1.D.b |
| 4 | Accesses a `theaterSeats` element and assigns it a new `Seat` object | **Responses still earn the point even if they...**<br>• incorrectly instantiate the new `Seat` object; or<br>• assign the new `Seat` object to an incorrect `theaterSeats` element. | **1 point** 3.E VAR-2.F |
| 5 | Correct tiers assigned to all array elements | | **1 point** 3.C CON-2.A |
| | | **Total for part (a)** | **5 points** |

**(b)** `reassignSeat`

| | Scoring Criteria | Decision Rules | |
|---|---|---|---|
| 6 | Accesses *from* and *to* `Seat` objects | | **1 point** 3.E VAR-2.F |
| 7 | Calls `isAvailable` and `getTier` on `Seat` objects | **Responses still earn the point even if they...**<br>• correctly call methods on `theaterSeats` elements other than the *to* and *from* seats. | **1 point** 3.A MOD-1.G |
| 8 | Checks if move can be made based on both tiers and the availability status of *to* `Seat` object | | **1 point** 3.C CON-1.H |
| 9 | Correctly updates availability of both seats and returns `true` if the move can be made; otherwise, returns `false` | | **1 point** 3.E CON-2.N.c |
| | | **Total for part (b)** | **4 points** |
| | **Question-specific penalties** | | |
| | None | | |
| | | **Total for question 4** | **9 points** |

# Scoring Worksheet for 2020 AP Computer Science A Practice Exam

## Section I: Multiple Choice

| | | | |
|---|---|---|---|
| NUMBER CORRECT | | x 1.0000 = | |
| | (out of 40) | | **WEIGHTED SECTION I SCORE** (do not round) |

## Section II: Free Response

| | | | |
|---|---|---|---|
| QUESTION 1 | | x 1.1111 = | |
| | (out of 9) | | (do not round) |
| QUESTION 2 | | x 1.1111 = | |
| | (out of 9) | | (do not round) |
| QUESTION 3 | | x 1.1111 = | |
| | (out of 9) | | (do not round) |
| QUESTION 4 | | x 1.1111 = | |
| | (out of 9) | | (do not round) |
| | | Sum = | |
| | | | **WEIGHTED SECTION II SCORE** (do not round) |

## Composite Score

| | | | | |
|---|---|---|---|---|
| | + | | = | |
| **WEIGHTED SECTION I SCORE** (do not round) | | **WEIGHTED SECTION II SCORE** (do not round) | | **COMPOSITE SCORE** (round to nearest whole number) |

© 2020 College Board

When an AP Exam is administered, psychometric analysis determines the score ranges corresponding with each AP Exam score (5, 4, 3, 2, and 1) based on a composite score scale that combines and weights the exam parts. Due to minor variations in exam difficulty, the number of points corresponding with each AP Exam score can vary on different exams. Because this practice exam was never administered, AP has developed these estimated score ranges that teachers can use to approximate AP Exam scores. We caution that these ranges, and the resulting AP Exam scores, are only estimates, and student performance on this practice exam does not necessarily predict performance on a different exam.

### AP Score Conversion Chart
## Computer Science A

| COMPOSITE SCORE RANGE | AP EXAM SCORE |
|---|---|
| 69–80 | 5 |
| 57–68 | 4 |
| 36–56 | 3 |
| 22–35 | 2 |
| 0–21 | 1 |

# Contact Us

**apcentral.collegeboard.org**

**New York Office**
250 Vesey Street
New York, NY 10281
212-713-8000
212-713-8277/55 (Fax)

**AP Services for Educators**
P.O. Box 6671
Princeton, NJ 08541-6671
877-274-6474 (toll free in the United States and Canada)
212-632-1781
610-290-8979 (fax)
Email: apexams@info.collegeboard.org

**Call Center Hours**
M–F, 8 a.m. to 8 p.m. ET
April 20 to May 29, 2020, M–F, 7 a.m. to 9 p.m. ET

**AP Canada Office**
2950 Douglas Street, Suite 550
Victoria, BC, Canada V8T 4N4
250-472-8561
800-667-4548 (toll free in Canada only)
Email: gewonus@ap.ca

**College Board International**
*Serving all countries outside the United States and Canada*
250 Vesey Street
New York, NY 10281
212-373-8738
Email: international@collegeboard.org