

## ROUND ONE

**Question 1.** Write the body of the following method.

```
// Return true iff the values in A are in
// sorted order, low to high (left to right).

public static boolean isSorted( int[] A ) {
    MISSING CODE
}
```

**Question 2.** Complete the following method.

```
// Return true iff in the given column of
// array A, the largest value in that column
// is in the last row.
//
// Assume that A is a rectangular array with
// at least one row and one column, and that col
// is valid (i.e., is between 0 and A[0].length)

public static boolean arrayOK(int[][] A, int col) {
    MISSING CODE
}
```

**Question 3.** Consider the following code:

```
Person p1 = new Person("Sandy", 12);
Person p2 = new Person("Amelia");
if (p1.getAge() != p2.getAge()+12) {
    System.out.println("error!");
}
```

Write a complete definition of a `Person` class so that this code would compile and execute, and would not print “error”.

**Question 4.** Assume that a `Person` class has been defined with a `getAge` method that returns the person’s age (an `int`). Write a method that has one parameter, a non-empty 1-dimensional array of `Person`, and returns the index of the oldest person in the array. If there is more than one person with the same, oldest age, the method can return the index of any one of them.

**Question 5.** Complete the following method.

```
// Assume that parameter A is a non-empty,
// square array.
// Return true iff every element in the two
// diagonals of A contains the same value.

public static boolean sameOnDiags(int[][] A) {
    MISSING CODE
}
```

## ROUND TWO

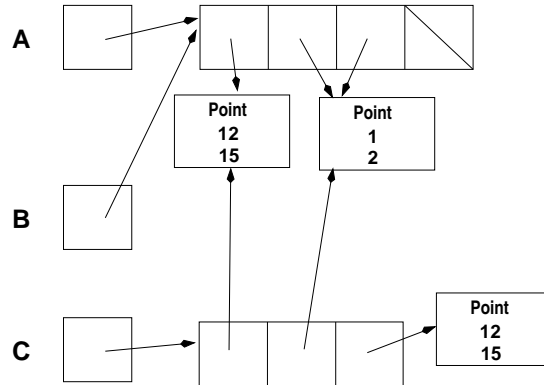
**Question 6.** Assume that a `Student` class has been defined with `public, static, final int` fields called `UGRAD`, `GRAD`, `SPECIAL`, `FULLTIME`, and `PARTTIME`. It also has public methods `getKind` (which returns `UGRAD`, `GRAD`, or `SPECIAL`), `getCredits`, and `setStatus`.

A student's status is supposed to be `FULLTIME` if their kind is `GRAD` and they have at least 6 credits, or their kind is `UGRAD` and have at least 12 credits (`SPECIAL` students are always `PARTTIME`).

Assume you have a `Student` object `s` whose `kind` and `credits` fields have been initialized but not its `status` field. Write code that is *not* part of the `Student` class that sets `s`'s status appropriately.

**Question 7.** Assume that a `Point` class has been defined. Each `Point` stores two `int` values, and the `Point` constructor takes two `int` values as arguments.

Write code that, when executed, creates this memory diagram:



**Question 8.** Assume that every element of array A (a non-empty, square array of char) contains a space. Write code to change it to have a pattern like the ones shown below for an 8-by-8 and a 5-by-5 array), but don't assume a particular size for A.

X							
X	X						
		X					
		X	X				
				X			
				X	X		
						X	
						X	X

X				
X	X			
		X		
		X	X	
				X

**Question 9.** Complete the following method.

```
// return a string that is made up of the first,
// third, fifth, etc characters from s.
// For example, if s is "ABCDEFGH"
// then return "ACEG".
// Assume that s is not null.
```

```
public static String oddChars( String s ) {
    MISSING CODE
}
```

**Question 10.** Assume that a Person class has been defined, including an equals method. Write the body of the following method.

```
// Return true iff array p contains some person
// more than once.
```

```
public static boolean hasDuplicate(Person[] p) {
    MISSING CODE
}
```

### ROUND THREE

**Question 11.** Assume that a `Person` class has been defined, with a `getEyeColor` method that returns a lower-case `String`.

Complete the following method.

```
// Return a 1-dimensional array of boolean
// whose length is the same as the number of
// rows in array p.
// The kth entry in the returned array is true
// iff the kth row of array p includes at least
// one person with blue eyes

public static boolean[] blueEyeRows(Person[][] p) {
    MISSING CODE
}
```

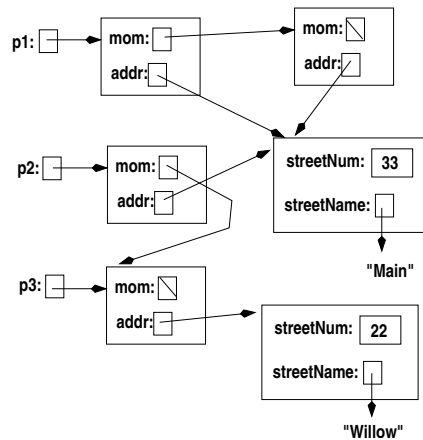
**Question 12.** Assume we have these classes:

`Address`: has two private fields: `int streetNum` and `String streetName`.

`Person`: has two private fields: `Person mom` and `Address addr`.

Assume both classes have `get` methods for both of their fields and have 2-arg constructors that initialize their fields as you would expect. Do not assume anything else about the classes.

Write code (**not** part of either the `Address` or `Person` class) that creates the memory diagram shown below. There should be no variables other than the ones shown.

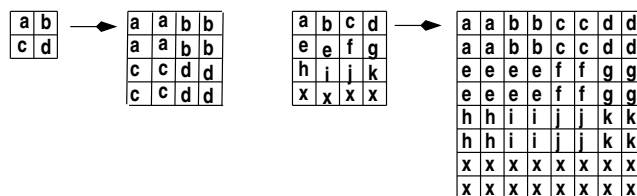


**Question 13.** Complete the following method

```
// Assume that neither A nor B contains any
// duplicate values.
// Return true iff A and B contain the same
// values (but maybe in different orders in
// the two arrays).

public static boolean same(int[] A, int[] B) {
    MISSING CODE
}
```

**Question 14.** Complete the `expandArray` method below. It should create and return a 2D array that is four times as big (has twice as many rows and columns) as its parameter A (a rectangular array). The new array should be filled in by replacing every character in A with four copies of that character as illustrated below.



```
public char[][] expandArray(char[][] A) {
}
}
```

**Question 15.** Assume that a `Point` class has been defined. Write field, constructor, and method declarations for a `Polygon` class (just declarations no code for the bodies), according to the following specification.

- A polygon should keep track of its location, color, and number of sides.
- The location should be a `Point`.
- Colors should be int values with the names `RED` or `BLUE`. Those values should be available to be used outside the class.
- It should be possible to create a polygon given its location and number of sides, or given its location, number of sides, and color.
- A polygon should be able to draw itself (at its location).
- The polygon's `draw` method should use a helper method that draws one side, given the color to use, and "from" and "to" points.
- A polygon should be able to determine if it is equal to another polygon.

## FINAL ROUND

**Question 16.** Assume a `Person` class has been defined. Fill in all missing parts of the following code to implement the `Line` class, which represents people standing in line.

You **must** use an array to represent the line of people.

```
public class Line {
    // fields
    MISSING CODE

    // create an empty line that can hold n people
    public Line( int n ) {MISSING CODE}

    // create a line that can hold n people, and that
    // already includes all of the people in array A
    // (with the person in A[0] at the front of the
    // line, the person in A[1] next, etc). If A has
    // more than n people, just include the first n
    public Line( int n, Person[] A ) {MISSING CODE}

    // add p to the end of line if there's room
    // return true if p was added, else return false
    public boolean addToEnd( Person p ) {MISSING CODE}

    // remove and return the person at the front of
    // the line and move everyone else up;
    // if the line was empty, return null
    public Person removeFromFront() {MISSING CODE}

    // return the number of people in line
    public int numInLine() {MISSING CODE}
}
```