Instructions:

• **Do not open this exam until told by the proctor.** You will have exactly 110 minutes to finish it.

• **Make sure your phone is turned OFF (not to vibrate!) before the exam starts.**

• Food, gum, and drink are strictly forbidden.

• **You may not use your phone or open your bag for any reason,** including to retrieve or put away pens or pencils, until you have left the exam room.

• This exam is *closed-book, closed-notes, and closed-computational devices.*

• If you get stuck on a problem, it may be to your benefit to move on to another question and come back later.

• All answers to the True/False or Multiple Choice questions must be on the bubble sheet. Answers written on the exam booklet will **not** count for these questions.

• All code must be written out in proper java format, including all curly braces and semicolons.

• Do not separate the pages. If a page becomes loose, re-attach it with the provided staplers.

• Staple all scratch paper to your exam. Do not take any sheets of paper with you.

• If you require extra paper, please use the backs of the exam pages or the extra pages provided at the end of the exam. **Clearly indicate on the question page where the graders can find the remainder of your work (e.g., “back of page” or “on extra sheet”).**

• Use a pencil, or blue or black pen to complete the exam.

• If you have any questions, raise your hand and a proctor will come to answer them.

• When you turn in your exam, you may be required to show ID. **If you forgot to bring your ID, talk to an exam proctor immediately.**

• We wish you the best of luck.
0. **(1 pt) Cover Page Information:**
   - Check that your exam has all 10 pages (excluding the cover sheet and scratch paper).
   - Write your name, recitation number, and PennKey (username) on the front of the exam.
   - Sign the certification that you comply with the Penn Academic Integrity Code.
   - Fill in your name and PennID number (and bubble in the values!) on the bubble sheet.

**MULTIPLE CHOICE QUESTIONS — All answers must be on the bubble sheet**

**SECTION 1: MISCELLANEOUS (8 pts total)**

1. **(2 points) What would be the result of the following Processing code fragment?**
   ```java
   fill(0);
   rectMode(CENTER);
   rect(width/2, height/2, width/10, width/10);
   background(255);
   ```
   A. a black square in the center of a white background
   B. a black square in the upper half of a white background
   C. a black square in the bottom half of a white background
   D. a black square that covers the entire window
   E. none of the above

2. **(2 points) What would be the result of the following Processing code fragment?**
   ```java
   stroke(0);
   fill(255);
   for (int i = 0; i < width; i += width/4) {
     ellipse(width/2, height/2, i, i);
   }
   ```
   ![Diagram of circle patterns](image1)
   A. 
   B. 
   C. 
   D. 

3. **(2 points) What would the following code print?**
   ```java
   String s = “Hello World”;
   for (int i = 10; i >= 1; i -= 3) {
     System.out.print(s.charAt(i));
   }
   ```
   A. dlrow olleH
   B. doo
   C. dooe
   D. do1H
   E. This is invalid Java code

4. **(2 points) (char)((char) (‘A’ + 10 % 8) - ‘A’ + ‘B’)**
   A. ‘A’
   B. ‘B’
   C. ‘C’
   D. ‘D’
   E. ‘E’
SECTION 2: OPERATORS AND EXPRESSIONS  (12 points total)

(1 pt each) For each expression in the left column, choose the value on the right column that \( z \) contains. For example, the answer to “boolean \( z = (10 < 20) \)” would be “A. true”. If the code would result in an error, choose “CE. ERROR”. Note that for some answers, you may need to fill in more than one bubble.

<table>
<thead>
<tr>
<th>Expressions</th>
<th>Possible Answers</th>
</tr>
</thead>
<tbody>
<tr>
<td>5. ( \text{int } z = 11/2; \noindent \text{z--;} )</td>
<td>A. true AE. 4.0</td>
</tr>
<tr>
<td>6. ( \text{String s = “hello”; } \noindent \text{double } z = s.length() / 2; )</td>
<td>B. false BC. 4.5</td>
</tr>
<tr>
<td>7. ( \text{int } z = (9++) / 2.0; )</td>
<td>C. 2 BD. 5</td>
</tr>
<tr>
<td>8. ( \text{boolean } z = 3 &lt; 17.5 % 5; )</td>
<td>D. 2.0 BE. 5.0</td>
</tr>
<tr>
<td>9. ( \text{boolean } z = “book”.equals(“boo” + “k”); )</td>
<td>E. 2.5 CD. 5.5</td>
</tr>
<tr>
<td>10. ( \text{boolean } z = \text{false; } \noindent \text{z = !(l&lt;4) &amp;&amp; !!(true &amp;&amp; \text{false})</td>
<td></td>
</tr>
<tr>
<td></td>
<td>AC. 3.0 DE. None of the above</td>
</tr>
</tbody>
</table>

(1 pt each) For each code fragment, what is the most appropriate data type to fill in the blank? If the code would result in an error, choose “CE. ERROR”. Note that for some answers, you may need to fill in more than one bubble.

<table>
<thead>
<tr>
<th>Expressions</th>
<th>Possible Answers</th>
</tr>
</thead>
<tbody>
<tr>
<td>11. ( \text{String } s = “CIS 110”; } \noindent \text{z = s.charAt(0) - s.charAt(1);}</td>
<td>A. boolean AE. char</td>
</tr>
<tr>
<td>12. ( \text{String } s = “CIS 110”; } \noindent \text{z = (s == “CIS 110”);}</td>
<td>B. boolean[] BC. char[]</td>
</tr>
<tr>
<td>13. ( \text{int } \text{x }= 1083; \noindent \text{z = x / 23.0;} )</td>
<td>C. int BD. String</td>
</tr>
<tr>
<td>14. ( \text{z }= 23.0f; )</td>
<td>D. int[] BE. String[]</td>
</tr>
<tr>
<td>15. ( \text{public static } \text{minimum(int[] a)} { )</td>
<td>E. double CD. void</td>
</tr>
<tr>
<td>16. ( \text{public static } \text{int binarySearch(String[] a, } \text{??} \text{b) } { )</td>
<td>AB. double[] CE. ERROR</td>
</tr>
<tr>
<td></td>
<td>AC. float DE. None of the above</td>
</tr>
<tr>
<td></td>
<td>AD. float[]</td>
</tr>
</tbody>
</table>
SECTION 3: CONDITIONALS AND LOOPS  (9 points total)

(5 points) Consider the following code, which behaves differently depending on the value of an integer variable x. What are the possible outputs of running this code?

```java
if (x > 0) {
    if (x / 2 == x % 2) {
        System.out.println("A");
    } else if (x >= Math.sqrt(x)) {
        System.out.println("B");
    } else {
        System.out.println("C");
    }
} else {
    if (Math.pow(x, 3) > x) {
        System.out.println("D");
    } else {
        System.out.println("E");
    }
}
```

17. Will the code ever print “A” for some value of the integer x?  
   A. Yes  
   B. No

18. Will the code ever print “B” for some value of the integer x?  
   A. Yes  
   B. No

19. Will the code ever print “C” for some value of the integer x?  
   A. Yes  
   B. No

20. Will the code ever print “D” for some value of the integer x?  
   A. Yes  
   B. No

21. Will the code ever print “E” for some value of the integer x?  
   A. Yes  
   B. No

(2 pts each) Fill in the blanks in the following pairs of code segments so that both are equivalent.

22. 
```java
for (int i = 0; i < 50; i++) {
    System.out.println(??);
}
``` 

A. arr[i]  
B. arr[i+1]  
C. arr[i-1]  
D. None of the above  
E. It is not possible to make these code segments equivalent by filling in only this blank

23. 
```java
char[] c = {'H', 'e', 'l', 'l', 'o'};
for (int i = 0; i < c.length; i++) {
    System.out.println(c[i]);
}
``` 

A. i  
B. i+1  
C. i++  
D. None of the above  
E. It is not possible to make these code segments equivalent by filling in only this blank
SECTION 4: TRACERY  (12 points total)

Trace through the following code. Assume that the program is executed using the command:

```
java Tracery 4 3 2
```

The program prints the following table; fill in the blanks as you trace through the program. Then, copy the answers into the corresponding number on your bubble sheet. (1 pt each)

| 24.) ___ | 25.) ___ | 26.) ___ |
| 27.) ___ | 28.) ___ | 29.) ___ |
| 30.) ___ | 31.) ___ | 32.) ___ |
| 33.) ___ | 34.) ___ | 35.) ___ |

```java
public class Tracery {
    static char[] ans = {'A', 'B', 'C', 'D', 'E'};
    static int a = 0, b = 0, c = 0;

    public static int f1(int a, boolean b) {
        if (b) {
            c = 0;
            for (int i = 1; i < 2 * a; i *= 2) {
                c++;
            }
            c = c % ans.length;
        } else {
            a = f2(a, c);
        }
        return a;
    }

    public static int f2(int b, int c) {
        int a = (b - c) / b;
        return a;
    }

    public static void main(String[] args) {
        a = Integer.parseInt(args[0]);
        int b = Integer.parseInt(args[1]);
        System.out.println("24.) " + ans[a] + " 25.) " + ans[b] + " 26.) " + ans[c]);
        a = f1(a / 2, false);
        System.out.println("27.) " + ans[a] + " 28.) " + ans[b] + " 29.) " + ans[c]);
        b = f1(a, true);
        System.out.println("30.) " + ans[a] + " 31.) " + ans[b] + " 32.) " + ans[c]);
        a = f1(a / 2, true);
        System.out.println("33.) " + ans[a] + " 34.) " + ans[b] + " 35.) " + ans[c]);
    }
}
```
SECTION 5: READY, SET, DRAW!  
(10 points total)

For each code segment below, choose the figure that was generated by that code. If none of the figures could have been generated by the code, answer DE (None of the above).

36. size(100, 100);
   background(255);
   fill(0);
   rectMode(CORNER);
   for (int x = 25; x <= width - 25; x += 25)
     for (int y = x; y <= height - 25; y += 25) {
       if (x < y)
         rect(x, y, 25, 25);
     }

37. size(100, 100);
   background(255);
   fill(0);
   rectMode(CORNER);
   for (int x = 25; x <= width - 25; x += 25)
     for (int y = x; y <= height - 25; y += 25) {
       if (x == y)
         rect(x, y, x, x);
     }

38. size(100, 100);
   background(255);
   fill(0);
   rectMode(CORNER);
   for (int x = 25; x <= width - 25; x += 25)
     for (int y = x; y <= height - 25; y += 25) {
       rect(x, y, 25, 25);
     }

39. size(100, 100);
   background(255);
   fill(0);
   rectMode(CORNER);
   for (int x = 25; x <= width - 25; x += 25)
     for (int y = 0; y <= height - 25; y += x) {
       rect(x, y, 25, 25);
     }

40. size(100, 100);
   background(255);
   fill(0);
   rectMode(CORNER);
   for (int x = 0; x <= width - 25; x += 25)
     for (int y = 25; y <= height - 25; y += y) {
       rect(x, y, 25, 25);
     }
**SECTION 6: SORTING AND SEARCHING  (10 pts total)**

41. **(2 pts)** What is the fastest way to find an element in an unsorted array?
   A. linear search
   B. binary search
   C. sort the array, then use linear search
   D. sort the array, then use binary search

42. **(2 pts)** In the worse case scenario, when using binary search on a sorted array of length 32, what is the maximum number of steps that you have to take to find the element?
   A. 4      B. 5      C. 6      D. 32      E. None of the above

43. **(2 pts)** Given a sorted array of length 1000, how many array elements do we need to access to determine the maximum value?
   A. 1      B. $\log_2 1000$      C. 1000      D. $\log_2 1000 + 1000$      E. None of the above

Recall that both insertion sort and selection sort separate the array into two portions: the left portion is always in ascending sorted order, and the right portion is not. Each step of the algorithm shifts the boundary one array position to the right, sorting as it goes.

44. **(2 pts)** Assume that we’re using insertion sort on the array {4, 1, 8, 6, 3, 0}.
   What would be the state of the array after two steps of insertion sort?
   A. {1, 4, 6, 8, 3, 0}
   B. {0, 4, 1, 8, 6, 3}
   C. {0, 1, 8, 6, 3, 4}
   D. {1, 4, 8, 6, 3, 0}
   E. None of the above

45. **(2 pts)** Assume that we’re using selection sort on the array {7, 2, 0, 5, 1, 3}.
   What would be the state of the array after one step of selection sort?
   A. {2, 5, 0, 7, 1, 3}
   B. {0, 2, 7, 5, 1, 3}
   C. {0, 1, 7, 2, 5, 3}
   D. {0, 1, 7, 5, 2, 3}
   E. None of the above

**SECTION 7: DEBUGGING  (14 points total)**

46. **(2 pts)** Which of the following code fragments would not cause a compilation error?
   A. int[] x = new int['a'];
   B. double[] x = new int[5];
   C. double[5] x = new double x[5];
   D. String[] x = "Hello how are you";
   E. They all cause compilation errors
   AB. None of the above cause compilation errors
WRITE YOUR ANSWERS DIRECTLY ON THIS PAGE, NOT THE BUBBLE SHEET

In a standard poker deck, there are 52 cards; each card has a rank (2, 3, 4, ..., 10, Jack, Queen, King, Ace) and a suit (spades, hearts, diamonds, clubs). If you draw 5 cards from a shuffled deck, what is the probability of drawing a flush (all cards with the same suit)? The program on the next page tries to answer this question, but it contains several bugs. The first bug (a missing semicolon) has been identified for you. Find and correct the six remaining bugs, filling in the table below. (In case you’re interested, the probability of a flush is approximately 0.00198079.)

47. (12 points) List the six errors in the code (excluding the missing semicolon on line 9).

<table>
<thead>
<tr>
<th>Line Number</th>
<th>Error</th>
<th>Correction</th>
</tr>
</thead>
<tbody>
<tr>
<td>09</td>
<td>Missing a semicolon</td>
<td>Add semicolon at end of line</td>
</tr>
</tbody>
</table>

Score: _______
/** Simulate multiple 5-card hands to compute the probability of a flush.
* Each experiment simulates one random 5-card hand from a 52-card deck.
* Usage: java ComputeProbabilityOfFlush n
* n - the number of experiments to run
*/
public class ComputeProbabilityOfFlush {
  // constants to represent the possible suits
  public static int SPADES = 1001, HEARTS = 1002, DIAMONDS = 1003, CLUBS = 1004;
  public static int INVALID_SUIT = -1000
  public static void main(String[] args) {
    numTrials = Integer.parseInt(args[0]);
    // fill the deck with 52 cards, numbered 0-51:
    //      0-12 will represent the 2 through ace of spades
    //     13-25 will represent the 2 through ace of hearts
    //     26-38 will represent the 2 through ace of diamonds
    //     39-51 will represent the 2 through ace of clubs
    int[] deck = new int[];
    for (int c = 0; c < 52; c++) {
      deck[c] = c;
    }
    int numFlushes = 0;
    // simulate a number of random hands of cards
    for (int trial = 0; trial < numTrials; trial++) {
      StdRandom.shuffle(deck); // randomize the order of the cards array
      // draw five cards from the top of the deck
      int[] hand = new int[5];
      for (int i = 0; i < 5; i++) {
        hand[i] = deck[i];
      }
      // track the total number of flushes
      if (isFlush(hand)) numFlushes++;
    }
    System.out.println("Probability of a flush: " + numFlushes / numTrials);
  }
  // determines whether a 5-card hand contains a flush
  public boolean isFlush(int[] hand) {
    // check if all cards in the hand have the same suit
    return (suit(hand[0]) == suit(hand[1]) && suit(hand[1]) == suit(hand[2]) &&
      suit(hand[2]) == suit(hand[3]) && suit(hand[3]) == suit(hand[4]));
  }
  // determines the suit of the given card
  // returns the suit of the card (SPADES, HEARTS, DIAMONDS, CLUBS),
  // or INVALID_SUIT if card is invalid
  public static int suit(int[] card) {
    if ( 0 <= card && card <= 12) return SPADES;
    if (13 <= card && card <= 25) return HEARTS;
    if (26 <= card && card <= 38) return DIAMONDS;
    if (39 <= card && card <= 51) return CLUBS;
  }
}

The first bug has been found for you, and is listed as an example in the table.
Hint: The program contains 5 more syntax bugs that prevent the code from compiling and 1 more logical bug that will prevent the program from giving the correct answer. Each correction should require one line of code or less.
** SECTION 8: FUNCTIONS! **  (22 points total)

48. (10 pts) Write the public static function `numPerfectSquares` based on the header below:

```java
/** Function Name: numPerfectSquares() 
 * Parameters:
 *   arr - an array of integers
 * Returns:
 *   the number of elements in arr that are perfect squares
 *   (i.e., the element is equal to the square of a whole number.
 *   For example, 9 and 16 are perfect squares but 15 is not)
 */
```
WRITE YOUR ANSWER DIRECTLY ON THIS PAGE, NOT THE BUBBLE SHEET

49. (12 points) Write the public static function `cycle()` based on the header below:

```java
/**
 * Function Name: cycle
 * Shifts all elements of an array the specified number of positions to the right; elements shifted off the end of the array are placed back on the opposite end. The shift amount can be negative to shift elements to the left.
 * Parameters:
 * arr – an array of integers of length n
 * shift – the number of positions to shift all elements of arr
 * Returns:
 * a new array of length n representing the cycled array
 * Error checking:
 * if the array is null, return null
 * Examples: cycle({0, 1, 2, 3, 4, 5}, 2) -> {4, 5, 0, 1, 2, 3}
 * cycle({0, 1, 2, 3, 4, 5}, -1) -> {1, 2, 3, 4, 5, 0}
 * cycle({0, 1, 2, 3, 4, 5}, 9) -> {3, 4, 5, 0, 1, 2}
 */
```