# CS111 Practice Exam Problems for Midterm 2

---

These are review problems from the book **Introduction to Programming in Java: An Interdisciplinary Approach**, 2nd Edition, Robert Sedgewick and Kevin Wayne, Princeton University.

Review Q&A on chapters:

- 1.4 (p.)
- 1.5 (p.)
- 2.1 (p. )
- 2.3 (p. )

## Problems from section 1.4

1.4.3 – Learning Objectives (5.1a) (5.1g)

```
public static void main(String[] args) {
     double [] arr1 = {3.0, 5.0, 7.8, 2.3};
     double [] arr2 = {1.2, 9.3, 3.2, 4.6};
     double sum = 0;
     for(int i = 0; I < arr1.length; i++) {
          sum += Math.pow((arr2[i]-arr1[i]),2);
     }
     System.out.println(Math.sqrt(sum));
}
```

1.4.5 – Learning Objectives (5.1a) (5.1e)

The array has not been initialized

1.4.6 – Learning Objectives (5.1a) (5.1k)

```
for (int r = 0; r < array.length; r++) {
     for (int c = 0; c < array[r].length; c++) {
          if(array[r][c] == true) {
               System.out.printf("(%d,%d) *",r,c);
          } else {
               System.out.printf(("(%d,%d)  ",r,c);
          }
     }
     System.out.println();
}
```

1.4.8 – Learning Objectives (5.1a) (5.1b)

[1, 1, 2, 3, 5, 8, 13, 21, 34, 55]

1.4.9 – Learning Objectives (5.1a) (5.1c) (5.1d)

false

1.4.11 – Learning Objectives (5.1a) (5.1f) (5.1g)

```
public static void HowMany(String[] args) {
     System.out.println("There are " + args.length + "
arguments");
}
```

1.4.14 – Learning Objectives (5.1a) (5.1f) (5.1g)

```
int[][] a = { {99,85,98},{98,57,78},{92,77,76},{94,32,11}};
int[][] b = new int[a[0].length][a.length];

for(int i=0; i < b.length; i++) {
     for(int j=0; j < b[0].length; j++) {
          b[i][j] = a[j][i];
     }
}
```

1.4.33 – Learning Objectives (5.1a) (5.1f) (5.1g)

```
for (int i = 0; i < arr.length; i++) {
     for (int j = i+1; j < arr.length; j++) {
          if (arr[i] == arr[j]) {
               return true;
          }
     }
}
return false;
```

## Problems from section 1.5

1.5.3 – Learning Objectives (6.1a) (6.1b) (6.1c) (6.1d)

```
public static void main(String[] args) {

     int n = Integer.parseInt(args[0]);
     float [] arr = new float[n];

     // read value and compute mean
     float sum = 0;
     for (int i = 0; i < n; i++) {
          float value = StdIn.readFloat();
          arr[i] = value;
          sum += value;
     }
     double mean = sum/n;

     //standard deviation:
     double stdsum = 0.0;
     for (int j = 0; j<n; j++) {
          double diff = Math.pow(arr[j]-mean, 2);
          stdsum += diff;
     }
     double stdev = Math.sqrt((stdsum/(n-1)));

     System.out.println("The mean is: " + mean);
     System.out.println("The standard deviation is: " + stdev);
}
```

1.5.6 – Learning Objectives (6.1e) (6.1g) (6.1i)

```
public static void main(String[] args) {
   int prev = StdIn.readInt();
   while (!StdIn.isEmpty()) {
        int curr = StdIn.readInt();
        if (curr != prev) {
             System.out.print(prev + "");
             prev = curr;
        }
   }
   System.out.println(prev);
 }
```

## Problems from section 2.1

2.1.1 – Learning Objectives (7.1a) (7.1c) (7.1d) (7.1m)

```
public static int max3(int a, int b, int c) {
     if ( a > b ) {
          if ( a > c ) return a;
          else return c;
     } else {
          if ( b > c ) return b;
          else return c;
     }
}

public static double max3(double a, double b, double c) {
     if ( a > b ) {
          if ( a > c ) return a;
          else return c;
     } else {
          if ( b > c ) return b;
          else return c;
     }
}
```

2.1.4 – Learning Objectives (7.1a) (7.1c) (7.1d) (7.1e) (7.1g)

```
public static boolean eq(int [] a, int [] b) {
     if(a.length != b.length) return false;
     for(int i = 0; i < a.length; i++) {
          if(a[i] !=b [i]) return false;
     }
     return true;
}
```

2.1.12 – Learning Objectives (7.1a) (7.1c) (7.1d) (7.1k) (7.1l)

HelloHelloByeByeByeByeByeByeByeBye

2.1.30 – Learning Objectives (7.1a) (7.1b) (7.1c) (7.1d)

(credited to Princeton) - https://introcs.cs.princeton.edu/java/21function/Calendar.java.html

## Problems from section 2.3

2.3.1 – Learning Objectives (8.1c)

1) Negative n – return 1
2) Large values of n will result in stack overflow, meaning, JVM will run out of stack space because there are too many calls to the recursive method.

2.3.2 – Learning Objectives (8.1c) (8.1d)

```
// By using the property of log,
// i.e. take the sum of log values of n, n-1, n-2 …1.
public static double rec(int n) {
     if (n == 1) {
          return 0;
     }
     return rec(n-1) + Math.log(n);
}
```

2.3.3 – Learning Objectives (8.1a)

6 4 2 2 1 1 4 3 1 1 3 6

## 2.3.8 – Learning Objectives (8.1a) (8.1c)

1) mystery (2, 25) – returns 50
2) mystery (3, 11) – returns 33
mystery (a, b) computes the product of a and b

After replacing + with * and return 0 with return 1

1) mystery (2, 25) – returns 65795: Consist of ((((2^2)^2)^2)^2)) + 2 + 256 + 1
2) mystery (3, 11) – returns 6574: Consist of (((3^2)^2)^2)) + 3 + 9 + 1

Note: The additional + terms come from the line return mystery ((a*a), b/2) +2, whenever b%2 is not equal to 0 (an odd b), and the +1 comes from the return condition return 1 when b reaches 0.

## 2.3.19 – Learning Objectives (8.1c) (8.1d)

(credited to Princeton) - https://introcs.cs.princeton.edu/java/23recursion/Combinations.java.html