

CS312 Fall 2016 Exam 2 Solution and Grading Criteria.

Grading acronyms:

AIOBE - Array Index out of Bounds Exception may occur

BOD - Benefit of the Doubt. Not certain code works, but, can't prove otherwise

Gacky or Gack - Code very hard to understand even though it works. (Solution is not elegant.)

LE - Logic error in code.

NAP - No answer provided. No answer given on test

NN - Not necessary. Code is unneeded. Generally no points off

NPE - Null Pointer Exception may occur

OBOE - Off by one error. Calculation is off by one.

RTQ - Read the question. Violated restrictions or made incorrect assumption.

1. Code Trace:

- A. `hel 7`
- B. `sea n` OR `horns n` OR `syntax error`
- C. `false`
- D. `No`
- E. `5` (The `4` when `e3` is `false` and the `1` when `e1` is `false` and `e2` is `true`.)
- F. `BDM8**`
- G. `Runtime error or Exception`
- H. `val_=_12.370` (underscores for spaces)
- I. `5 0`
- J. `[5, -3, 7, 7, 0]` (differences in braces and commas okay.)
- K. `Runtime error or Exception`
- L. `[5, 7, 3, 13]`
- M. `[]`
- N. `[0, 2, 6]`
- O. `[0, 4, 10, 18, 28]`

2. Program Logic

	<code>z == 0</code>	<code>y == 0</code>	<code>y % 4 == 0</code>
POINT A	A	N	N
POINT B	S	N	S
POINT C	N	S	A
POINT D	S	N	N
POINT E	S	A	A

Point B, `y % 4 == 0`: initially `y = 10` -> `false`, if `y` picked to be `5`, decrement to `4`, `4 != 0` at top of while loop, `4 % 4 == 0`, `true`

Point D, `y == 0`: If `y` is picked to be `0` in the loop it is decremented to `-1` in the `if` statement.

3. Scanners. 15 points. Write a complete method `ratioOfIntsToDoubles`.

```
public static double ratioOfIntsToDoubles(Scanner sc) {
    double result = -1.0;
    int numInt = 0;
    int numDouble = 0;
    while (sc.hasNext()) {
        if (sc.hasNextInt()) {
            numInt++;
        } else if (sc.hasNextDouble()) {
            numDouble++;
        }
        // consume the token, because we aren't adding the
        // actual values, can just call next method and throw
        // away the result.
        sc.next();
    }
    if (numDouble > 0) {
        result = 1.0 * numInt / numDouble;
    }
    return result;
}
```

variables for num ints and num doubles, 1 point

loop while scanner has next: 3 points

check for ints before doubles: 2 points

check for next int correctly: 2 points

check for next double correctly: 2 points

call next (or nextInt, nextDouble, next) to consume token, 3 points

calculate ratio correctly: 1 point

handle case when no doubles correctly: 1 point

4. Strings - 15 Points. Create a method `getStretchedString` that accepts two parameters, a `String` and an `int`.

```
public static String getStretchedString (String str, int num) {
    String result = "";
    for (int i = 0; i < str.length(); i++) {
        char ch = str.charAt(i);
        for (int j = 0; j < num; j++) {
            result += ch;
        }
    }
    return result;
}
```

Create result: 2 points

outer loop for length of string: 4 points (partial credit possible)

inner loop for number of characters: 4 points

access chars correctly from string, 2 point (partial credit possible)

concatenate characters correctly, 2 points

return result, 1 point

substring okay, `string.length` okay

5. Arrays 14 Points. Write a method `numLessThanPrevious`. The method has one parameter: an array of `ints`

```
public static int numLessThanPrevious (int[] data) {
    int result = 0;
    for (int i = 1; i < data.length; i++) {
        if (data[i] < data[i - 1]) {
            result++;
        }
    }
    return result;
}
```

result variable initialized correctly: 1 point

loop with correct bounds: 7 points (-3 for off by one error or AIOBE)

check current is less than previous correctly: 4 points (includes accessing array correctly)

increment result correctly: 1 point

return: 1 point

6. Arrays 15 Points. Write a method `noDuplicates` that accepts one parameters, an array of `String` variables .

```
public static boolean noDuplicates (String[] data) {
    for (int i = 0; i < data.length; i++) {
        for (int j = i + 1; j < data.length; j++) {
            if (data[j].equals(data[i])) {
                return false;
            }
        }
    }
    return true;
}
```

correctly access elements in array: 1 point

outer loop for all strings: 2 points

inner loop to check no duplicates of current String: 6 points (-2 if check all and have if for not checking self)

use equals method correctly to check if two Strings equivalent: 2 points

return false as soon as duplicate found: 2 points

return true correctly if no duplicates found: 1 point

problems:

check self, -4

no inner loop -9 (only pairwise comparison)

7. Strings 15 Points. Write a method `matchingChars`.

```
public static boolean matchingChars(String s1, String s2, char ch) {
    int minLength = s1.length();
    if (s2.length() < minLength) {
        minLength = s2.length();
    }
    // check until we run out of chars in one String
    for (int i = 0; i < minLength; i++) {
        char c1 = s1.charAt(i);
        char c2 = s2.charAt(i);
        if (c1 == ch || c2 == ch) {
            if (c1 != c2) {
                return false;
            }
        }
    }

    return charNotPresent(s1, minLength, ch)
        && charNotPresent(s2, minLength, ch);
}

public static boolean charNotPresent(String st, int start, char ch) {
    for (int i = start; i < st.length(); i++) {
        if (st.charAt(i) == ch) {
            // it is present!
            return false;
        }
    }
    return true;
}
```

check portion that exists in both strings correctly: 5 points

check portion of string that is longer correctly, 5 points

correct check for non matching chars, 4 points

return 1 point

common problems:

using arrays, not allowed - 3

padding with chars, - 4 (what if char you are padding with is the target?)

substring -2

equals method - 4

no accounting for positions of chars, -8