

## Section 2: Printing Exceptional Atoi

Your section leader should have told you to pair up with a new student, someone different than you paired with last week. (Why? So that you get to meet and work with new people, just like you will after you graduate. You won't get to pick the people on your team; the team will be picking you to join them.)

Decide who will be the first driver, and let's get started.

### PART I: Printf Practice

*System.out.printf()* is a very useful output method, but one that doesn't get used as often as it should. This part will (re)introduce you to *printf()* and its formatting scheme.

1. Log into the driver's account, if you haven't already, start a Terminal window, and launch DrJava.
2. DrJava's Interactions pane (click on the tab labeled "Interactions" in the lower-left) is very useful for trying method calls, math expressions, etc., without having to write a program to do it. We're going to use it to experiment with `System.out.printf()`.
3. The basic structure of a call to `printf()` is `System.out.printf(<string>,<argument1>,<argument2>,...)`; For example, `System.out.printf("%d\n",4)`; In the Interactions pane, use these strings and arguments to form a complete call to `printf`, and write down the output in the space provided. Count the spaces shown in the output as accurately as you can! *Hint*: You can press the "up-arrow" key while in the interactions pane and edit the previous command; saves a lot of typing!

- (a) `"%d\n",4` \_\_\_\_\_
- (b) `"%d %f\n",4,2.3456` \_\_\_\_\_
- (c) `"%4d %4f\n",4,2.3456` \_\_\_\_\_
- (d) `"value = %.1f\n",2.3456` \_\_\_\_\_
- (e) `"%s = %8.3f\n","value",2.3456` \_\_\_\_\_

4. Now, based on what you observed, predict (don't type it into DrJava) the exact output of this code:

```
int i = 8192;
double d = 18.0904;
System.out.printf("i=%6d,\nd=%10.5f\n",i,d); // no spaces in the string!
```

5. Now type all three of those lines into the Interactions Pane, to see how you did. If your prediction didn't match reality, take some time and figure out your mistakes. Feel free to use DrJava to try examples of your own creation to help you understand how `printf()` works. Again, spaces matter!
6. Answer this question, based on your observations (don't consult any research materials!): What do the two values used with `%f` (for example, the 10 and the 5 in `%10.5f`) each control? When you have an answer in which you have confidence, proceed to the checkpoint.

 **CHECKPOINT 1** Raise your hand. Your SL will come over and check your answer(s).

(Continued ...)

## PART II: Introduction to Exceptions

Remember: New part, new driver; switch roles after each checkpoint!

1. Go to the class web page, find `Exceptions.java`, and load it into DrJava.
2. The method `nCharSubstrings()` is supposed to build and return an array of all of the three-consecutive-character substrings of the given `StringBuffer` object. For example, there are three such substrings of the word “array”: “arr”, “rra”, and “ray”.

Take some time to read the code of `nCharSubstrings()` carefully. Before you move on to the next step, be sure that you understand how the method is supposed to work. Soon, you’ll need to fix it; it’s hard to fix what you don’t understand.

3. Compile and run the program. You should see this error message:

```
Exception in thread "main" java.lang.StringIndexOutOfBoundsException: String index out of range: 12
    at java.lang.AbstractStringBuilder.substring(AbstractStringBuilder.java:906)
    at java.lang.StringBuffer.substring(StringBuffer.java:482)
    at Exceptions.nCharSubstrings(Exceptions.java:26)
    at Exceptions.main(Exceptions.java:9)
```

The “at” lines tell us where Java discovered the problem, but in reverse order: `main()` was called first, then it called `nCharSubstrings()`, which called `StringBuffer`’s `substring()` method, and so on. The place to start looking for the logical error is line 26 in `nCharSubstrings()`, the last line of our code that was executed.

4. Before you try to fix the problem, let’s first try to understand the error message. Java reported that the error was a `StringIndexOutOfBoundsException`. But is this a checked or an unchecked exception? Use the Java API and the information from Monday’s lecture (remember, the slides are posted on the class web page) to decide. (And write down your answer; your SL will ask!)
5. You should have discovered that this is the category of exception that is caused by a logical error within the programmer’s control; that is, there’s an error in our code that we can fix.
6. Debugging time: Find and fix the error! Your goal is to make the program produce the correct output for the string hard-coded into `main()`. You may find that fixing one error exposes others; you need to fix all of the errors in `nCharSubstrings()`.



### CHECKPOINT 2

Raise your hand. Your SL will come over to verify that your fix(es) work(s).

(Continued ...)

## PART III: Writing an `atoi()` Method


*The Computer Science Department Head, Prof. Proebsting, likes to remind the faculty that students in his junior-level class last spring had a very hard time writing a method that takes a string containing numeric characters ('0' through '9') and converting it to an `int`. For example, the string "1042" would be converted to the integer 1042. You probably remember that Java provides a method to do this: `Integer.parseInt()`. In the language C, it's called `atoi()` (ASCII to Integer). Writing such a method is a common job interview question for programmers. Today, you and your partner will write it, improving your job prospects and making our department head happy.*

*(Did you remember to switch drivers?)*

1. Go to the class web page, find `Atoi.java`, and load it into DrJava.


This program has a stub of the `atoi()` method, plus a `main()` method that tests it. We've provided a few test cases, but you should add more of your own.

2. Complete the method `atoi()` (without using `Integer.parseInt()` or any similar methods, of course!). Here are some suggestions to help you:
  - (a) Before writing any code, using pen and paper, write out an integer string and, step by step, sketch out a plan for how to convert the characters in that string into a single integer value.
  - (b) The ASCII value of a character can be determined by casting the character to be of type `int`.
  - (c) The ASCII value of a digit character differs from its value by 48. (For example, the ASCII value of '4' is 52.  $52 - 48 = 4$ .)
  - (d) `Math.pow()` can create powers of 10, but it's just as easy to keep a running product of powers of 10.
3. Thing you've got it? Be a little more sure: We've provided a few test cases in the `main()` method, but you should add more of your own.

 **CHECKPOINT 3** Raise your hand. Your SL will come over and check your work.

## PART IV: Clean Up!

1. Log out of your computer.
2. Pick up your papers, writing implements, cell phones, trash, etc.
3. Push in your chair(s).

 **CHECKPOINT 4** Raise your hand. Your SL will come over and pass judgement on your ability to clean and tidy your area.

You're free to go! But, if you have time, we recommend that you use it to work some more on the programming assignment, if you're not already done.