Questions

Bookmark this page

Implementing stringValue

1/1 point (graded)

Here is the recursive implementation of `stringValue()` from the reading, with the recursive steps brought together but with the base case still missing:

```
/**
 * @param n integer to convert to string
 * @param base base for the representation. Requires 2<=base<=10.
 * @return n represented as a string of digits in the specified base, with
 *           a minus sign if n<0.  No unnecessary leading zeros are included.
 */
public static String stringValue(int n, int base) {
    if (n < 0) { return "-" + stringValue(-n, base); }
    else if (CONDITION) { BASE CASE }
    else { return stringValue(n/base, base) + "0123456789".charAt(n%base); }
}
```

Which of the following lines can be substituted `else if (CONDITIO) { BASE CASE }` to make the code correct?

- [ ] else if (n == 0) { return "0"; }
- [x] else if (n < base) { return "" + n; }
- [ ] else if (n == 0) { return ""; }
- [x] else if (n < base) { return "0123456789".substring(n,n+1); }

✔

Explanation

The first choice is wrong because it will add a leading 0 to single-digit numbers, i.e. making stringValue(3, 10) return "03" instead of just "3".
The second choice works. return "" + n is shorthand for converting the single-digit number n into a string.
The third choice is wrong because stringValue(0, 10) will return "" instead of "0".
The fourth choice works.

Submit | Show Answer

Answers are displayed within the problem

Calling stringValue

1/1 point (graded)

Assuming the code is completed with one of the base cases identified in the previous problem, what does stringValue(170, 16) do?

- [ ] returns "AA"
- [ ] returns "170"
- [ ] returns "1010"
- [x] throws StringIndexOutOfBoundsException
- [ ] doesn't compile, static error
- [ ] StackOverflowError
- [ ] infinite loop

✔

Explanation

Note first that using base=16 violates the precondition of this method, so it doesn't have to satisfy the postcondition. A valid implementation can do anything. The question is what this particular valid implementation will do.

The recursive step will be invoked, which will split the number 170 by computing 170/16=10 and 170%16=10. The charAt() call will attempt to get the 11th character of "0123456789", which is past the end of the string. A StringIndexOutOfBoundsException will result.

Submit

Show Answer

Answers are displayed within the problem

Previous Next