Submit Graded Assignment 1: Java SRPN Calculator

This week you will submit your first graded assignment. This assignment is worth 40% of your total grade. Please note below the format to submit your response in.

Your Task

The Saturated Reverse Polish Notation (SRPN) Calculator

Whilst performing some maintenance on a legacy system you find that it makes use of a program called SRPN (srpn). SRPN is not documented and no one seems to know who wrote it, so your boss tells you to rewrite it in Java. SRPN is a reverse Polish notation calculator with the extra feature that all arithmetic is saturated, i.e. when it reaches the maximum value that can be stored in a variable, it stays at the maximum rather than wrapping around.

1. Your task is to write a program which matches the functionality of SRPN as closely as possible. Note that this includes not adding or enhancing existing features.

2. The legacy program and the example code you need to edit are available below.

3. The SRPN code is available below in the repl.it window. This allows you to run and interact with the srpn program.

4. You should start by typing in the 4 tests below, observe the output and them implement your code to replicate this functionality.

5. This is the second weeks in which to complete this coursework. I suggest that:

Last week you should have completed Test 1 and Test 2.

This week complete Test 3 and Test 4, as well as exploring if there are other features of the SRPN calculator that are not covered in the tests provided. The challenge this week is to understand some of the interesting features of the SRPN calculator. For example, what does 'r' do? What does 'd' do? Does the input "1+1" produce the same output as "1 + 1" (note the spaces)?

6. Your program will be tested on the same 4 tests below AND others that are similar. Successfully completing each step will give you up to 15 marks each. The remaining marks 40 marks are for good programming practice include, commenting, program structure etc. for a total of 100.

7. Submit your completed solution on Monday.

Test1

The program must be able to input at least two numbers and perform one operation correctly and output.

Input :
10
2
+
=
Input :
11
3
-
=
Input :
9
4
*
=
Input :
11
3
/
=
Input :
11
3
%
=

Test 2

The program must be able to handle multiple numbers and multiple operations.

Input :

3

3

*

4

4

*

+

=

Input :

1234

2345

3456

d

+

d

+

d

=

Test 3

The program must be able to correctly handle saturation.

Input :

2147483647

1

+

=

Input :

-2147483647

1

-

=

20

-

=

Input :

100000

0

-

d

*

=

Test 4

The program includes the less obvious features of SRPN. These include but are not limited to...

Input :
1
+
Input :
10
5
-5
+
/
Input :
11+1+1+d
Input :
This i s a comment
1 2 + # And so i s t h i s #
d
Input :
3 3 ^ 3 ^ 3 ^=
Input :
r r r r r r r r r r r r r r r r r r r r d r r d

Here's the legacy SRPN program (remember to click Run):
https://repl.it/@Ernestma3/sprn-to-emulate#main.sh

```
# Click run above to run the legacy SRPN code

chmod +x ./srpn/srpn
echo "You can now start interacting with the SRPN calculator"
./srpn/srpn

# Once you have clicked run try typing in the example input
# and observe the output
#
# For example, try
#
# 10
# 2
# +
# =
```

And here's the example code you need: Main.java and SRPN.java (click on the files icon on the top left of the repl.it window below to see both files). You should edit SRPN.java to write your solution:
https://repl.it/@Ernestma3/Coursework1-SRPN-1

```
import java.io.*;

class Main {

  // main method
  // reads in input from the command line
  // and passes this input to the processCommand method in SRPN

  public static void main(String[] args) {
    // Code to take input from the command line
    // This input is passed to the processCommand
    // method in SRPN.java
    SRPN sprn = new SRPN();
    BufferedReader reader = new BufferedReader(new
InputStreamReader(System.in));
    try {
      //Keep on accepting input from the command-line
      while(true) {
        String command = reader.readLine();
        //Close on an End-of-file (EOF) (Ctrl-D on the terminal)
        if(command == null){
          //Exit code 0 for a graceful exit
          System.exit(0);
        }
        //Otherwise, (attempt to) process the character
        sprn.processCommand(command);
      }
    }
    catch(IOException e) {
      System.err.println(e.getMessage());
      System.exit(1);
    }
  }
}
```

Guidelines

- Submit your assignment by **Wednesday** before 12:00 midday (UK time). Remember: it may take several minutes to upload your file so leave yourself plenty of time. Late assignments will be penalised.
- Submit your assignment following the **format** below.
- **Do not include your name** anywhere on your assignment.
- The coursework will be conducted individually. Attention is drawn to the University rules on **plagiarism** which are set out in your Student Handbook. While software reuse (with referencing the source) is permitted, we will only be able to assess your contribution.
- Your feedback and grade will be made available in the Grades page 15 working days after the assignment due date. Your assignment grade is provisional, subject to approval by the Faculty Board of Studies following the Board of Examiners meeting, where your overall unit result will be confirmed.

Submission format:

Your program solutions should be a zip file that can be extracted to a directory with the name SRPN-PoP. The directory should contain the files necessary to compile and run your code SRPN.java with any additional files needed to allow your program to compile and run. Before you upload your solution, make sure that the zip file contains all necessary files and creates the correct directory.

Grading

- This assignment is worth 40% of your total grade.
- Marks are available in the following areas:

1. Test 1: Max 15 marks.
2. Test 2: Max 15 marks.
3. Test 3: Max 15 marks.
4. Test 4: Max 15 marks.
5. Max 40 marks allocated for good coding practice, including comments, program structure, variable and method names, etc.