Programming Exercise – RPN Calculator

Some of the best calculators in the world have an 'RPN' (reverse polish notation) mode.

We would like you to write a command-line based RPN calculator.

Requirements

- The calculator has a stack that can contain real numbers.
- The calculator waits for user input and expects to receive strings containing whitespace separated lists of numbers and operators.
- Numbers are pushed on to the stack. Operators operate on numbers that are on the stack.
- Available operators are +, -, *, /, sqrt, undo, clear
- Operators pop their parameters off the stack, and push their results back onto the stack.
- The 'clear' operator removes all items from the stack.
- The 'undo' operator undoes the previous operation. "undo undo" will undo the previous two operations.
- sqrt performs a square root on the top item from the stack
- The '+', '-', '*', '/' operators perform addition, subtraction, multiplication and division respectively on the top two items from the stack.
- After processing an input string, the calculator displays the current contents of the stack as a space-separated list.
- Numbers should be stored on the stack to at least 15 decimal places of precision, but displayed to 10 decimal places (or less if it causes no loss of precision).
- All numbers should be formatted as plain decimal strings (ie. no engineering formatting).
- If an operator cannot find a sufficient number of parameters on the stack, a warning is displayed:

  ```
  operator <operator> (position: <pos>): insufficient parameters
  ```

- After displaying the warning, all further processing of the string terminates and the current state of the stack is displayed.

Deliverables

- The solution submitted should include structure, source code, configuration and any tests or test code you deem necessary - no need to package class files.
- Solve the problem in Java, C# or in a specific language that you may have been directed to use.
- Solve the problem as though it were "production level" code.
- It is not required to provide any graphical interface.

In order to get around firewall issues we recommend the solution be packaged as a password protected zip file.

Examples

Example 1

```
5 2
stack: 5 2
```

Example 2

```
2 sqrt
stack: 1.4142135623
clear 9 sqrt
stack: 3
```

Example 3

```
5 2 -
stack: 3
3 -
stack: 0
clear
stack:
```

Example 4

```
5 4 3 2
stack: 5 4 3 2
undo undo *
stack: 20
5 *
stack: 100
undo
stack: 20 5
```

Example 5

```
7 12 2 /
stack: 7 6
*
stack: 42
4 /
stack: 10.5
```

Example 6

```
1 2 3 4 5
stack: 1 2 3 4 5
*
stack: 1 2 3 20
clear 3 4 -
stack: -1
```

Example 7

```
1 2 3 4 5
stack: 1 2 3 4 5
* * * *
stack: 120
```

Example 8

```
1 2 3 * 5 + * * 6 5
operator * (position: 15): insufficient parameters
stack: 11
```

(the 6 and 5 were not pushed on to the stack due to the previous error)