Documentation

> In reverse Polish notation, the operators follow their operands; for instance, to add 3 and 4, one would write 3 4 + rather than 3 + 4. If there are multiple operations, operators are given immediately after their second operands; so the expression written 3 − 4 + 5 in conventional notation would be written 3 4 − 5 + in reverse Polish notation: 4 is first subtracted from 3, then 5 is added to it. An advantage of reverse Polish notation is that it removes the need for parentheses that are required by infix notation. While 3 − 4 × 5 can also be written 3 − (4 × 5), that means something quite different from (3 − 4) × 5. In reverse Polish notation, the former could be written 3 4 5 × −, which unambiguously means 3 (4 5 ×) − which reduces to 3 20 − (which can further be reduced to -17); the latter could be written 3 4 − 5 × (or 5 3 4 − ×, if keeping similar formatting), which unambiguously means (3 4 −) 5 ×.

Link to Wikipedia (source)

Compilation:

Just type 'make' in terminal while you're in folder containing program.

Using the program:

In terminal after compilation type './rpnCalculator' then you will see
entry menu, after pressing enter you can use program. This program in usage is similar to dc gnome program.
Example:

```
3 2 1 +
```

this will push 3, 2 and 1 to the stack and make addition on 1 and 2, result is remaining 3 and 3 in stack.

Warning : You can type multiple operations at once in line only if you use space button to seperate them.
(Only exception is when you are inserting an negative number with '_', then you have to type it together, for example '_3' will push -3 to the stack).

List of options:

- Pushing number to the stack: type any integer number, '1' will add 1
- Pushing negative number to the stack: '_24' will add -24
- Displaying the entire stack: 'f'
- Displaying last number on the stack: 'p'
- Clearing all stack: 'c'
- Deleting last number on the stack: 'P'
- Adding numbers: You have to push two numbers to the stack and the use '+'
- Subtracting numbers: After pushing two numbers to the stack use '-'
- Multiplying numbers: After pushing two numbers to the stack use '*'
- Dividing numbers: After pushing two numbers to the stack use '/'
- Operator modulo: After pushing two numbers to the stack use '%'
- Power operator: After pushing two numbers to the stack use '^'
- Square root: Type 's' and this will square last number on the stack and push result
- Swapping last numbers: 'r' this will swap places of two last numbers in the stack
- Adding number to memory: type 'M+' (this will push last number to memory)
- Removing number from memory: type 'M-'
- Clearing memory: type 'MC' (this will clear memory stack)
- Receiving number from memory: 'MR' (This will pop number from memory and push it to the stack)
- Quiting program: type 'q'