CALCULATOR IMPLEMENTATION

Create a calculator application with textual input. The calculator has 10 memory slots with identifiers from 0 to 9. You can read integer numbers to the slots and you can perform arithmetic operations on them. All these operations are controlled textually. The initial value of the slots is 0. Let's take the following example:

```
read 10 1
read 20 3
read 30 7
add 1 2 9
mul 3 1 6
print 9
print 6
```

The meaning of these lines:

- Read value 10 to slot #1.
- Read value 20 to slot #3.
- Read value 30 to slot #7.
- Add the values of slots #1 and #2, and store the result in slot #9. This way the value of slot #9 will be 10.
- Multiply the values of slots #3 and #1, and store the result in slot #6. This way the value of #6 will be 200.
- Print the value of slot #9 which is 10.
- Print the value of slot #6 which is 200.

You have to support the following operations as functions:

- read: reading a value to a slot Parameters: value and the number of slot
- print: printing a value of a slot. Parameter: the number of slot
- add, sub, mul, div: addition, subtraction, multiplication, integer division. Parameters: slots of two operands and slot of the result.

These functions are given the array of slots as parameter and that many further arguments as indicated above.

Create functions for computing the sum and average of numbers which get the array of slots as a parameter and return the sum and average of the contained values respectively.

Write a test code in the main program which performs some operations on some slots. Also test the sum and average functions.

If a filename is given as command line argument at program start then the calculator is controlled by the content of that file.

Place the array of slots to the heap memory. Make sure that there is no memory leak in the program. The size of the array of slots is given by a macro token (2 points).

Place functions to a separate translation unit. Use the "include guard" idiom.