**Fractions v.1.0, aka Zlomky v.1.0**

**Matvei Slavenko**

Charles University in Prague

Faculty of Mathematics and Physics

Obecná matematika, the first semester

NMIN101 Programming I

Winter semester, school year 2016/2017

1. **General Information**

This programme renders mathematical formulae using pseudo graphics. The input for the programme is a mathematical formula, and the output is the same formula presented in pseudo graphics.

The programme has a simple command-line user interface. The user can set the input and output streams at his/her convenience: the programme could both use files and standard input-output system. The list of commands and their detailed descriptions are available in the internal help module of the programme.

1. **General Ideas of Algorithm**

The main feature that allows printing of expressions on several levels with fraction lines is a printing buffer. Usually there is a sequential access to printing only. To avoid traversing the expression’s AST in the correct order, the programme uses the following scheme: before printing the expression out, the programme orders single parts of the expression in a buffer with a random writing access.

These are the main logical steps of the algorithm:

1. Parsing of the expression and building of its AST;
2. Appending of the brackets;
3. Calculation of the graphical parameters;
4. Construction of the rendered formula in a buffer;
5. Printing out the rendered formula from the buffer.

See the documentation included to the source code for detailed information about each step, actual implementation and other details. Source code is distributed freely and could be downloaded, for example, from the GitHub repository.

1. **Input Data Format**
   1. The input expression should contain all parentheses;

Example:

* (1+2) is a valid expression;
* 1+2 is not a valid expression;

1. The input expression should not contain spaces or letters of the alphabet;

Example:

* (a+2) is not a valid expression;
* (2 + 3) is not a valid expression;

1. The input expression should contain integers only;

Example:

* (1.02+59) is not a valid expression;

1. The length of the expression does not matter and is limited only by the machine’s memory.
2. **Examples of Programme’s work**

Input:

((((1/2)+(2/6))/(25/(2654+8753)))+(258/96))

Output:

1 2

(- + -)

2 6 258

(--------------- + ---)

25 96

-------------

(2654 + 8753)

Input:

((((406\*580)+33045)-((75846/237)+(3/80000)))/((3625\*80)-243))

Output:

75846 3

(((406 \* 580) + 33045) - (----- + -----))

237 80000

---------------------------------------------

((3625 \* 80) - 243)

Input:

((((57\*162)/2052)+((12768\*5)/456))-((3468\*154)/(68\*357)))

Output:

(57 \* 162) (12768 \* 5) (3468 \* 154)

((---------- + -----------) - ------------)

2052 456 (68 \* 357)

Input:

((((4561+5739)\*12)/((701-501)/5))-(((445-22)/32)/((823+177)\*30)))

Output:

(445 - 22)

----------

((4561 + 5739) \* 12) 32

(-------------------- - ------------------)

(701 - 501) ((823 + 177) \* 30)

-----------

5

1. **Ideas for Programme’s Improvement**
2. To implement the spaces tolerance;
3. To implement the support of decimal fractions and letters of the alphabet;
4. To implement the support of roots, powers, and integrals;
5. To implement the BASH-friendly working mode;
6. To deal with memory leaks.