Smart Calculator

Dmitriy

June 2, 2023

Contents

1 Introduction

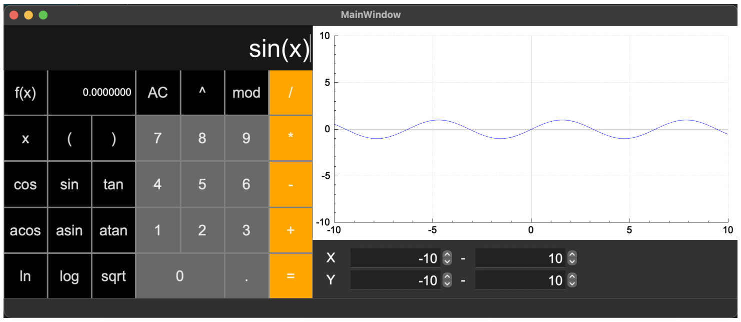
1. The program is developed in standard C11 using the gcc compiler and QT modules.
2. Assembly of the program is configured using Makefile with a set of goals: all, install, uninstall, clean, dvi, dist, test, gcov report.
3. The calculator has the ability to calculate arithmetic expressions, taking into account the priorities, as well as some mathematical functions (sine, cosine, logarithm, etc.).
4. In addition to calculating expressions, the calculator also supports the use of the variable x and the construction of the graph of the corresponding function.

2 Install

Follow these steps to install the calculator:

1. Unpack the archive on your computer.
2. Navigate to the directory with the extracted code.
3. Install the program with the command `make install`.
4. Run the calculator.

3 Screenshots



MainWindow

sin(x)

0.0000000

X -10 - 10

Y -10 - 10

Figure 1: The workspace of the calculator

4 Parsing code

```
void to_polish_not(const char *strin, char *strout) {
  Stack stack;
  stack.top = NULL;
  int i, j = 0;
  for (i = 0; strin[i] != '\0'; i++) {
    if (isdigit(strin[i])) {  // the input symbol refers to digits
      while (isdigit(strin[i]) || strin[i] == '.') {
        strout[j++] = strin[i++];
      }
      strout[j++] = '␣';
      --i;
    } else if (strin[i] == '(' ||
               is_func(strin[i])) {  // the input symbol refers to functions
      push(&stack, strin[i]);
    } else if (strin[i] == ')' || strin[i] == ',') {
      while (!is_empty(&stack) && peek_operator(&stack) != '(') {
        strout[j++] = pop_operator(&stack);
        strout[j++] = '␣';
      }
      if (!is_empty(&stack) && peek_operator(&stack) == '(') {
        pop_operator(&stack);
      }
      if (!is_empty(&stack) && is_func(peek_operator(&stack))) {
        strout[j++] = pop_operator(&stack);
        strout[j++] = '␣';
      }
    } else if (is_operator(strin[i])) {  // the input symbol refers to operators
      while (!is_empty(&stack) && peek_operator(&stack) != '(' &&
             (priority(strin[i]) <= priority(peek_operator(&stack)))) {
        strout[j++] = pop_operator(&stack);
        strout[j++] = '␣';
      }
      push(&stack, strin[i]);
    }
  }
  while (!is_empty(&stack)) {
    strout[j++] = pop_operator(&stack);
    strout[j++] = '␣';
  }
  strout[j] = '\0';
}
```