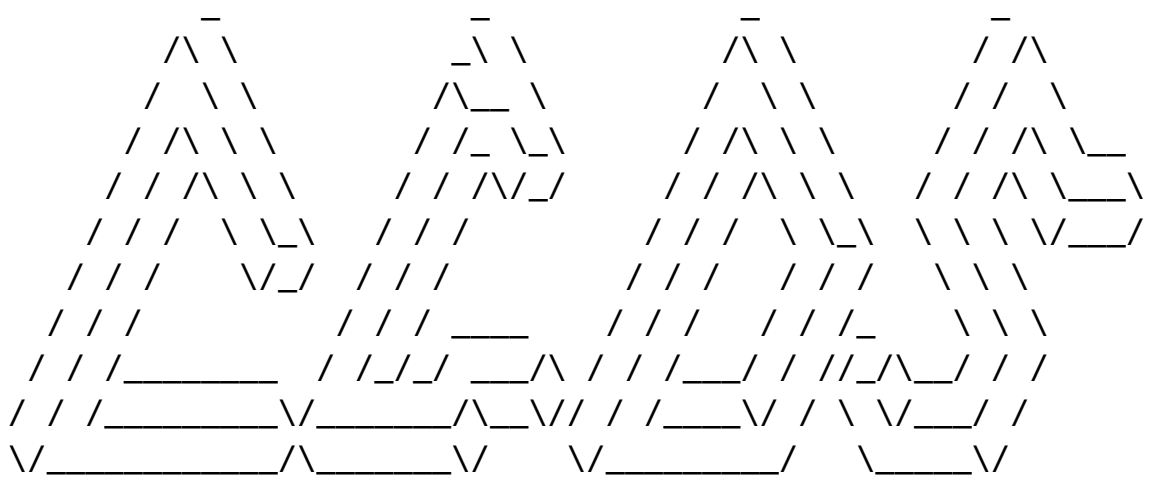


Command Line Operating System Version 0.11.1

User Manual

Contents

Running The Program

Running CLOS should be as straightforward as downloading the platorm-correct file from the releases tab in the Github here: https://github.com/happysmaran/CLOS (The latest is 0.11.1)

NOTE: If you are using the Windows version, note that *file*, *mkdir*, *rmdir*, *ls*, and *print* does not work for now in the NTFS subsystem. This bug is being explored and will be fixed eventually.

Additionally, if you are on Windows and text is monochromatic and get weird text such as \[*37m*, then please install the Visual C++ runtime from here:

ARM64: https://aka.ms/vs/17/release/vc_redist.arm64.exe

X86: https://aka.ms/vs/17/release/vc_redist.x86.exe

X64 (most common): https://aka.ms/vs/17/release/vc_redist.x64.exe

To check your system type

Settings—>System—>About—>Look at System Type.

Arm is ARM64

32bit is X86

64bit is X64

If any problems occur or you have any suggestions, please email the creator at smaran@codestraightwithsmaran.org or open an issue at the Github linked above. Thank you!

The Input System

The Input System in CLOS works differently compared to a usual calculator. Since the program is completely written in C++, simply writing 2*3 and expecting 6 as the result does not work. Instead, it relies on typing a command and then inputting all of the required values to get the desired output. This system is similar to a TI-series calculator, only if the inputs were entered on each line rather than all in one line.

As an example, to add two numbers, *add* must be types in the command line, then the two numbers. This is an example of it in action:

```
user@clos> add
Two numbers (enter as 'a b' without quotes): 2 5

7
Continue(Y/n)? n
Final result: 7
user@clos>
```

The “Continue” feature is available on all crud operations (addition, subtraction, multiplication, division, etc.) and will be discussed in the “All Comands” chapter.

Another thing to note is the *Yes-No System*. Many commands may ask a yes or no question depending on the task (such as the equation solver when it detects an invalid function), and the user will be required to answer ‘y’ for yes or ’n’ for no. Note that inputting any character apart from the two options will be automatically assumed as a no, and not entering anything will not close the application, but make a new line; the prompt will still expect a proper user input.

All commands in this tool operate by asking each input one by one and not by entering things in a line. This is also due to how C++’s input system works: it ignores any space entered on a line, so how many inputs are expected is hardcoded.

All Commands

Basic Operands

- *add* - Adds two numbers

  Has the option to continue adding additional numbers afterwards. The five following commands below also have it. This is an example:

```
user@clos> add
Two numbers (enter as 'a b' without quotes): 1 2

3
Continue(Y/n)? y
Additional number (enter as 'a' without quotes): 3

6
Continue(Y/n)? n
Final result: 6
user@clos>
```

- *subtract* or *minus* - Subtracts two numbers
- *multiply* - Multiplies two numbers
- *divide* - Divides two numbers
- *power* or *pwr* - Raises the first number to power of second number

  ‘3 2’ will be interpreted as 3 to the power of 2, or 3 squared.

- *root* - Takes root of first number using second number

  ‘4 2’ will be interpreted as the 2nd root of 4, or square root of 4.

- *pi* - A parameter that can be passed in the above operands

```
user@clos> add
Two numbers (enter as 'a b' without quotes): 3 pi

6.14159
Continue(Y/n)? n
Final result: 6.14159
```

- *abs* or *absolute* - Takes the absolute value of a given number

• *round* - Rounds a given number

• *iround* - Rounds a given number by concatenation (removing decimal)

• *rand* - Outputs a random number within a given bound

• *lcm* - Finds the LCM of two given numbers

• *gcd* - Finds the GCD of two given numbers

• *sum* or *summation* - Computes the summation of a given constant, linear, or quadratic function

• *combination* - Gives the combination *n* choose *r* given the inputs

• *permutation* - Gives the permutation *n* permute *r* given the inputs

• *factorial* - Gives the factorial of a given number

Trigonometry

• *sin* or *sine* - Calculates sine of a number

• *cos* or *cosine* - Calculates cosine of a number

• *tan* or *tangent* - Calculates tangent of a number

• *csc* or *cosecant* - Calculates cosecant of a number

• *sec* or *secant* - Calculates secant of a number

• *cot* or *cotangent* - Calculates cotangent of a number

• *asin* or *arcsine* - Calculates arcsine of a number

• *acos* or *arccosine* - Calculates arccosine of a number

• *atan* or *arctangent* - Calculates arctangent of a number

• *degrad* - Converts degress to radians and vice versa

Graphing and Algebra Related Commands

• *solve* - Equation Solver v0.3. Solves a given equation

Linear and Quadratic equations. Imaginary numbers also work, but display in a different form. More info in “Equation Solving.”

• *doublesolve* - Double Equation Solver v0.1. Solves a linear system of equations

• *linegraph* - Line Grapher

Only works with linear equations and produces a 10x10 graph that can only show whole number points (if the line was y=2x, only the points (-5,-10), (-4,-8), … (0,0) … (5,10) show up). More info in “Graphing and Graph Information chapter.”

• *funcstat* - Given an equation, upper bound, and lower bound, it lists all whole number points in a table

• *minmax* - Given quadratic equation, it can find maximum or minimum

Calculus

• *integral* - Definite integral calculator for linear and quadratic functions

• *indefintegral* - Indefinite integral calculator for standard polynomials

Standard polynomials refer to polynomials in this form:

ax^n + bx^(n-1) + … + yx + z

• *deriv* or *derivative* - Definite derivative calculator for linear and quadratic functions

• *indefder* or *indefderiv* - Indefinite derivative calculator for standard polynomials

Miscellaneous

- *clear* - Clears the command line
- *help* or *h* - Opens help dialog
- *about* - About CLOS
- *exit* - Exits CLOS. A timed delay setting is also available
- *var* - Makes a temporary variable that can be used in crud operations:

```
user@clos> var
Key and Value: test 2
Done.
user@clos> add
Two numbers (enter as 'a b' without quotes): 3 var

Variable for second value: test
5
Continue(Y/n)? n
Final result: 5
user@clos>
```

- *print* - Prints value in a variable
- *restart* - Restarts CLOS. A timed delay setting is also available
- *file* - Creates a new text file or rewrites an existing one

  Note that once a text is written, it cannot be modified like a regular word processor.

- *read* - Reads a file
- *mkdir* - Makes a folder
- *delete* - Deletes a file
- *rmdir* - Removes an empty folder
- *ls* - Lists the contents of a given folder
- *time* or *date* - Displays the time and date in GMT
- *version* or *ver* - Gives current tool version

Equation Solving and Graphing

As mentioned in the descriptions of the equation solver tools in the previous chapter, there are some limitations of the program for now.

- *solve*

  Solves linear and quadratic equations. Imaginary numbers also work, but display those numbers in this form:

  a+bi will become (a, b)

  Furthermore, it can only do linear and quadratic equations. General polynomial solving is not a feature *yet*.

- *doublesolve*

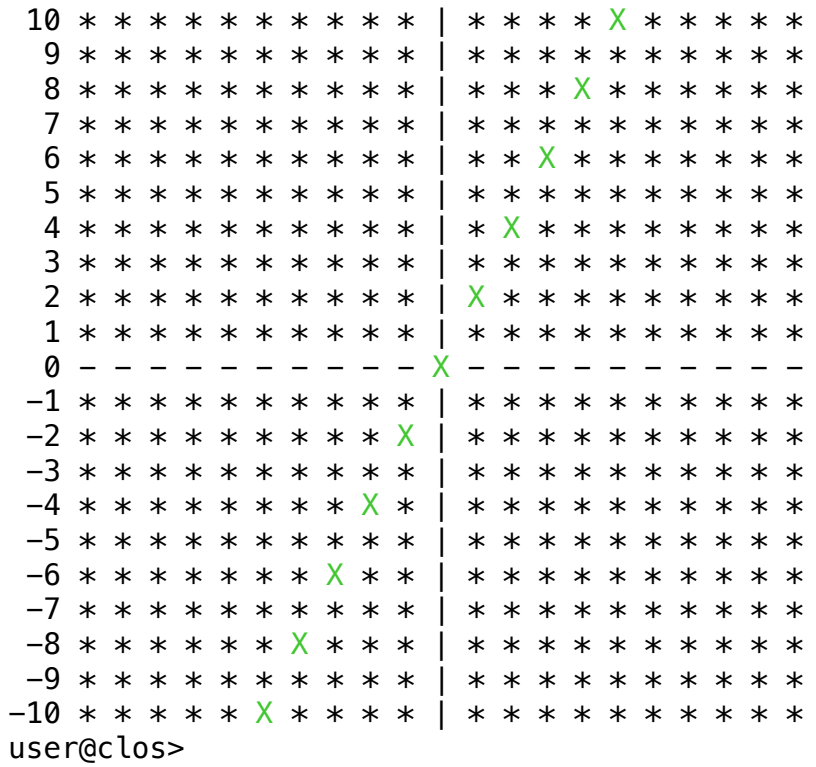
  Can only do two linear functions in the system. Also it is only linear functions.

- *linegraph*

  Only works with linear equations and produces a 10x10 graph that can only show whole number points (if the line was y=2x, only the points (-5,-10), (-4,-8), … (0,0) … (5,10) show up). That graph would look like this:

```
user@clos> linegraph
Line Grapher v0.1
This tool can be used to graph linear equations.
Enter the coefficient for x term (m. default m=1): 2
Enter the value for constant (b): 0
```

  <Graph shown on next page>



• *funcstat*

  Only does whole number x values.

• *minmax*

  Has a hypothetical limit on where it checks for the maximum or minimum. If the max or min lies outside it, it will simply give the lowest value in the range.