MATLAB Basics

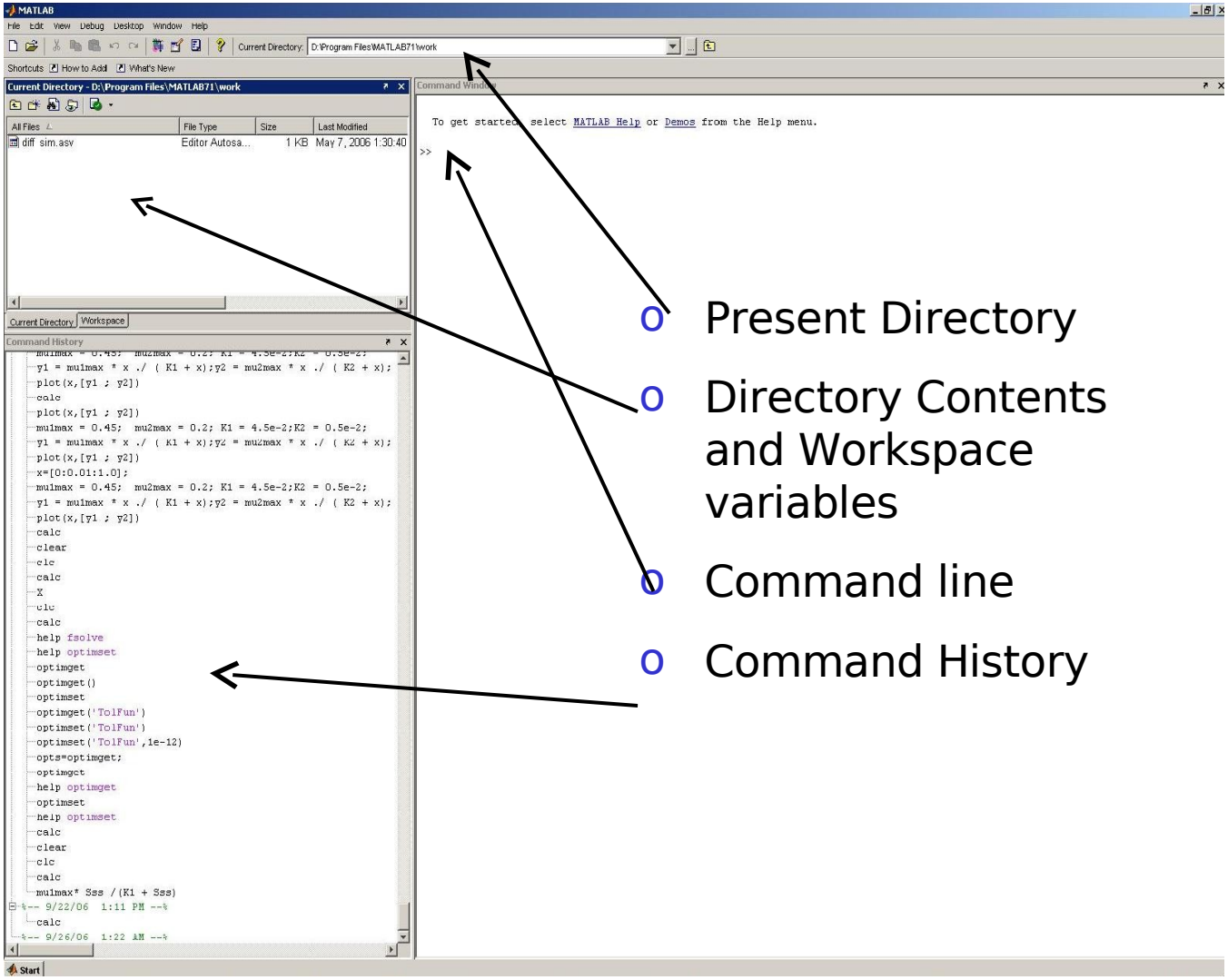
MATrix LABoratory

- www.mathworks.com

- Advantages of MATLAB
  - Ease of use
  - Platform independence
  - Predefined functions
  - Plotting
- Disadvantages of MATLAB
  - Can be slow
  - Commercial software

Matlab Windows

- Command line Interface (Main Window)
- Editor Window

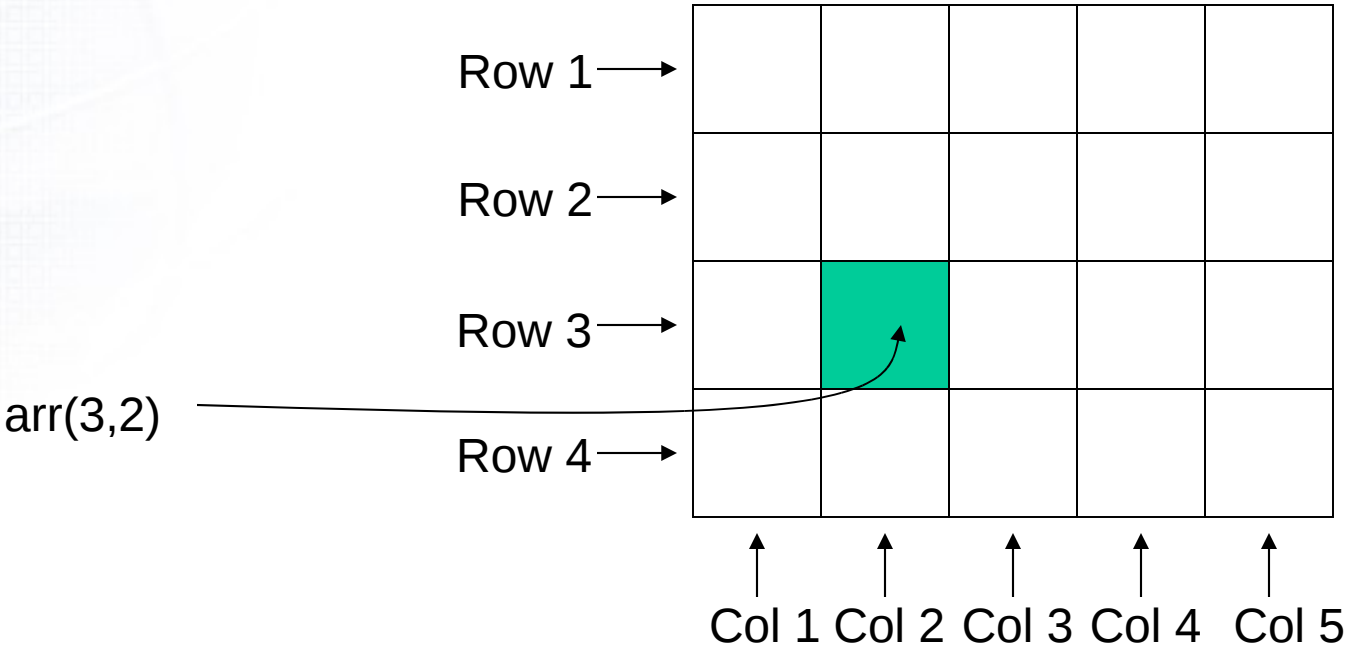


- Present Directory
- Directory Contents and Workspace variables
- Command line
- Command History

MATLAB BASICS

Variables and Arrays

- **Array:** A collection of data values organized into rows and columns, and known by a single name.



MATLAB BASICS

Arrays

- The fundamental unit of data in MATLAB
- Scalars are also treated as arrays by MATLAB (1 row and 1 column).
- Row and column indices of an array start from 1.
- Arrays can be classified as **vectors** and **matrices**.

MATLAB BASICS

- **Vector:** Array with one dimension
- **Matrix:** Array with more than one dimension
- **Size** of an array is specified by the number of rows and the number of columns, with the number of rows mentioned first (For example: n x m array).

  Total number of elements in an array is the product of the number of rows and the number of columns.

MATLAB BASICS

$$a=\begin{bmatrix}1 & 2\\ 3 & 4\\ 5 & 6\end{bmatrix}$$ 3x2 matrix → 6 elements

b=[1 2 3 4] 1x4 array → 4 elements, **row vector**

$$c=\begin{bmatrix}1\\ 3\\ 5\end{bmatrix}$$ 3x1 array → 3 elements, **column vector**

a(2,1)=3 b(3)=3 c(2)=3

Row # Column #

MATLAB BASICS

Multidimensional Arrays

- A two dimensional array with m rows and n columns will occupy mxn successive locations in the computer's memory. MATLAB always allocates array elements in **column major order.**

  a= [1 2 3; 4 5 6; 7 8 9; 10 11 12];

  a(5) = a(1,2) = 2

1	2	3
4	5	6
7	8	9
10	11	12

1
4
7
10
2
5
8
11

- A 2x3x2 array of three dimensions

  c(:, :, 1) = [1 2 3; 4 5 6];

  c(:, :, 2) = [7 8 9; 10 11 12];

MATLAB BASICS

Variables

- Variable names must begin with a letter, followed by any combination of letters, numbers and the underscore (_) character.

- The MATLAB language is Case Sensitive. NAME, name and Name are all different variables.

Give meaningful (descriptive and easy-to-remember) names for the variables. Never define a variable with the same name as a MATLAB function or command.

MATLAB BASICS

Common types of MATLAB variables

- **double:** 64-bit double-precision floating-point numbers

  They can hold real, imaginary or complex numbers in the range from $\pm 10^{-308}$ to $\pm 10^{308}$ with 15 or 16 decimal digits.

  >> var = 1 + i ;

- **char:** 16-bit values, each representing a single character

  The char arrays are used to hold character strings.

  >> comment = 'This is a character string' ;

The type of data assigned to a variable determines the type of variable that is created.

MATLAB BASICS

Initializing Variables in Assignment Statements

An assignment statement has the general form

var = expression

Examples:

```
>> var = 40 * i;
>> var2 = var / 5;
>> array = [1 2 3 4];
>> x = 1;  y = 2;
>> a = [3.4];
>> b = [1.0 2.0 3.0 4.0];
>> c = [1.0; 2.0; 3.0];
>> d = [1, 2, 3; 4, 5, 6];
>> e = [1, 2, 3
        4, 5, 6];
```

```
>> a2 = [0 1+8];
>> b2 = [a2(2) 7 a];
>> c2(2,3) = 5;
>> d2 = [1 2];
>> d2(4) = 4;
```

‘**;**’ semicolon suppresses the automatic echoing of values but it slows down the execution.

MATLAB BASICS

Initializing Variables in Assignment Statements

- Arrays are constructed using brackets and semicolons. All of the elements of an array are listed in row order.
- The values in each row are listed from left to right and they are separated by blank spaces or commas.
- The rows are separated by semicolons or new lines.
- The number of elements in every row of an array must be the same.

MATLAB BASICS

Initializing with Shortcut Expressions

first: increment: last

- **Colon operator:** a shortcut notation used to initialize arrays with thousands of elements

```
>> x = 1 : 2 : 10;
>> angles = (0.01 : 0.01 : 1) * pi;
```

- **Transpose operator:** (′) swaps the rows and columns of an array

```
>> g  = [1:4];
>> h = [ g′  g′ ];
```

$$h = \begin{bmatrix} 1 & 1 \\ 2 & 2 \\ 3 & 3 \\ 4 & 4 \end{bmatrix}$$

MATLAB BASICS

Initializing with Built-in Functions

- zeros(n)
- zeros(n,m)
- zeros(size(arr))
- ones(n)
- ones(n,m)
- ones(size(arr))
- eye(n)
- eye(n,m)

- length(arr)
- size(arr)

```
>> a = zeros(2);
>> b = zeros(2, 3);
>> c = [1, 2; 3, 4];
>> d = zeros(size(c));
```

MATLAB BASICS

Initializing with Keyboard Input

- The **input** function displays a prompt string in the Command Window and then waits for the user to respond.

my_val = input(‘Enter an input value: ’);

in1 = input(‘Enter data: ’);

in2 = input(‘Enter data: ’ ,`s`);

Subarrays

- The **end** function: When used in an array subscript, it returns the highest value taken on by that subscript.

  arr3 = [1 2 3 4 5 6 7 8];

  arr3(5:end) is the array [5 6 7 8]

  arr4 = [1 2 3 4; 5 6 7 8; 9 10 11 12];

  arr4(2:end, 2:end)

MATLAB BASICS

Subarrays

- Assigning a Scalar to a Subarray: A scalar value on the right-hand side of an assignment statement is copied into every element specified on the left-hand side.

```
>> arr4 = [1 2 3 4; 5 6 7 8; 9 10 11 12];
>> arr4(1:2, 1:2) = 1
arr4 =
     1    1    3    4
     1    1    7    8
     9   10   11   12
```

MATLAB BASICS

Special Values

- MATLAB includes a number of predefined special values. These values can be used at any time without initializing them.
- These predefined values are stored in ordinary variables. They can be overwritten or modified by a user.
- If a new value is assigned to one of these variables, then that new value will replace the default one in all later calculations.

```
>> circ1 = 2 * pi * 10;
>> pi = 3;
>> circ2 = 2 * pi * 10;
```

Never change the values of predefined variables.

MATLAB BASICS

Special Values

- pi: π value up to 15 significant digits
- i, j: sqrt(-1)
- Inf: infinity (such as division by 0)
- NaN: Not-a-Number (division of zero by zero)
- clock: current date and time in the form of a 6-element row vector containing the year, month, day, hour, minute, and second
- date: current date as a string such as *16-Feb-2004*
- eps: epsilon is the smallest difference between two numbers
- ans: stores the result of an expression

MATLAB BASICS

Changing the data format

```
>> value = 12.345678901234567;
```

format short	→ 12.3457
format long	→ 12.34567890123457
format short e	→ 1.2346e+001
format long e	→ 1.234567890123457e+001
format short g	→ 12.346
format long g	→ 12.3456789012346
format rat	→ 1000/81

MATLAB BASICS

The disp(*array*) function

```
>> disp( 'Hello' )
Hello
>> disp(5)
     5
>> disp( [ 'Bilkent ' 'University' ] )
Bilkent University
>> name = 'Alper';
>> disp( [ 'Hello ' name ] )
Hello Alper
```

MATLAB BASICS

The num2str() and int2str() functions

```
>> d = [ num2str(16) '-Feb-' num2str(2004) ];
>> disp(d)
16-Feb-2004
>> x = 23.11;
>> disp( [ 'answer = ' num2str(x) ] )
answer = 23.11
>> disp( [ 'answer = ' int2str(x) ] )
answer = 23
```

MATLAB BASICS

The fprintf(*format*, *data*) function

- %d integer
- %f floating point format
- %e exponential format
- %g either floating point or exponential format, whichever is shorter
- \n new line character
- \t tab character

MATLAB BASICS

```
>> fprintf( 'Result is %d', 3 )
Result is 3
>> fprintf( 'Area of a circle with radius %d is %f', 3, pi*3^2 )
Area of a circle with radius 3 is 28.274334
>> x = 5;
>> fprintf( 'x = %3d', x )
x =    5
>> x = pi;
>> fprintf( 'x = %0.2f', x )
x = 3.14
>> fprintf( 'x = %6.2f', x )
x =    3.14
>> fprintf( 'x = %d\ny = %d\n', 3, 13 )
x = 3
y = 13
```

MATLAB BASICS

Data files

- save *filename var1 var2* …

  >> save myfile.mat x y → binary

  >> save myfile.dat x –ascii → ascii

- load *filename*

  >> load myfile.mat → binary

  >> load myfile.dat –ascii → ascii

MATLAB BASICS

- *variable_name* = *expression*;
  - addition a + b → a + b
  - subtraction a - b → a - b
  - multiplication a x b → a * b
  - division a / b → a / b
  - exponent a^b → a ^ b

MATLAB BASICS

Hierarchy of operations

- x = 3 * 2 + 6 / 2
- Processing order of operations is important
  - parentheses (starting from the innermost)
  - exponentials (from left to right)
  - multiplications and divisions (from left to right)
  - additions and subtractions (from left to right)

```
>> x = 3 * 2 + 6 / 2
x =
     9
```

MATLAB BASICS

Built-in MATLAB Functions

- *result = function_name(input)*;
  - abs, sign
  - log, log10, log2
  - exp
  - sqrt
  - sin, cos, tan
  - asin, acos, atan
  - max, min
  - round, floor, ceil, fix
  - mod, rem
- help elfun →help for elementary math functions

Flow Control Constructs

- Logic Control:
  - IF / ELSEIF / ELSE
  - SWITCH / CASE / OTHERWISE
- Iterative Loops:
  - FOR
  - WHILE

The if, elseif and else statements

- Works on Conditional statements
- Short-circuited in MATLAB - once a condition is true, the sequence terminates.

```
if I == J
   A(I,J) = 2;
elseif abs(I-J) == 1
   A(I,J) = -1;
else
   A(I,J) = 0;
end
```

Switch, Case, and Otherwise

- More efficient than elseif statements
- Only the first matching case is executed

```
switch input_num
case -1
    input_str = 'minus one';
case 0
    input_str = 'zero';
case 1
    input_str = 'plus one';
case {-10,10}
  input_str = '+/- ten';
otherwise
    input_str = 'other value';
end
```

The for loop

- Similar to other programming languages
- Repeats loop a set number of times (based on index)
- Can be nested

```
N=10;
for I = 1:N
   for J = 1:N
    A(I,J) = 1/(I+J-1);
   end
end
```

The while loop

- Similar to other programming languages
- Repeats loop until logical condition returns FALSE.
- Can be nested.

```
I=1; N=10;
while I<=N
   J=1;
   while J<=N
      A(I,J)=1/(I+J-1);
      J=J+1;
   end
   I=I+1;
end
```

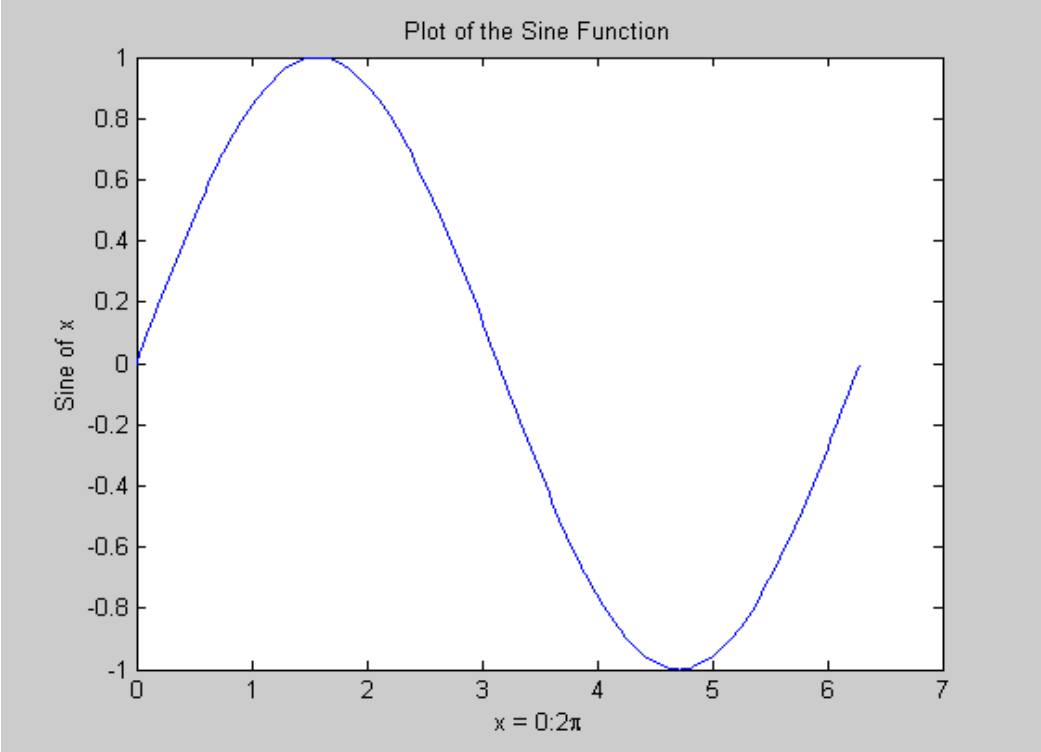
MATLAB BASICS

Types of errors in MATLAB programs

- Syntax errors
  - Check spelling and punctuation
- Run-time errors
  - Check input data
  - Can remove “;” or add “disp” statements
- Logical errors
  - Use shorter statements
  - Check typos
  - Check units
  - Ask assistants, instructor, …

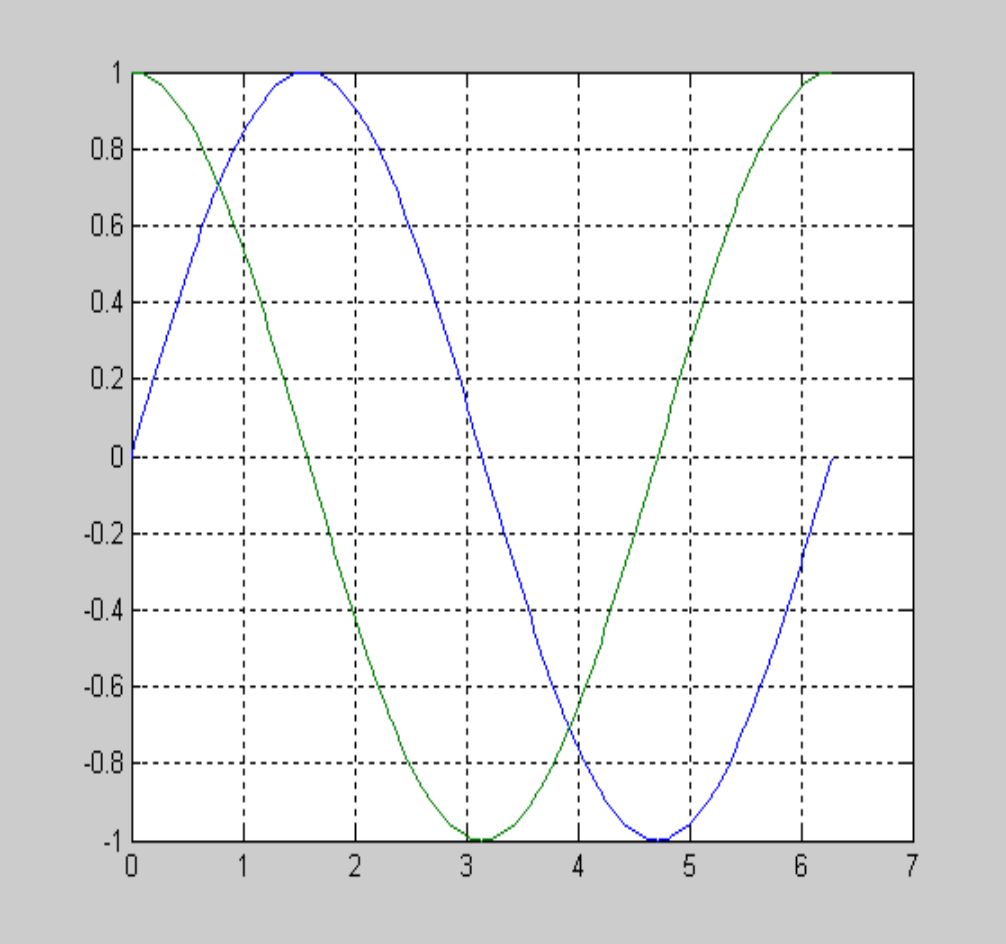
Matlab Graphs

```
x = 0:pi/100:2*pi;
y = sin(x);
plot(x,y)
xlabel('x = 0:2\pi')
ylabel('Sine of x')
title('Plot of the
   Sine Function')
```



Multiple Graphs

```
t = 0:pi/100:2*pi;
y1=sin(t);
y2=sin(t+pi/2);
plot(t,y1,t,y2)
grid on
```



MATLAB BASICS

Summary

- help *command* → Online help
- lookfor *keyword* → Lists related commands
- which → Version and location info
- clear → Clears the workspace
- clc → Clears the command window
- diary *filename* → Sends output to file
- diary on/off → Turns diary on/off
- who, whos → Lists content of the workspace
- more on/off → Enables/disables paged output
- Ctrl+c → Aborts operation
- … → Continuation
- % → Comments