supermanyousefi@gmail.com

Warning!
we suggest using online compilers. this program is full of bugs and can drop into INFINITE LOOPS. This can over heat your system.
we DO NOT take responsiblity

Simple CommandLine for Graph Implementation in C

all parts are case sensitive

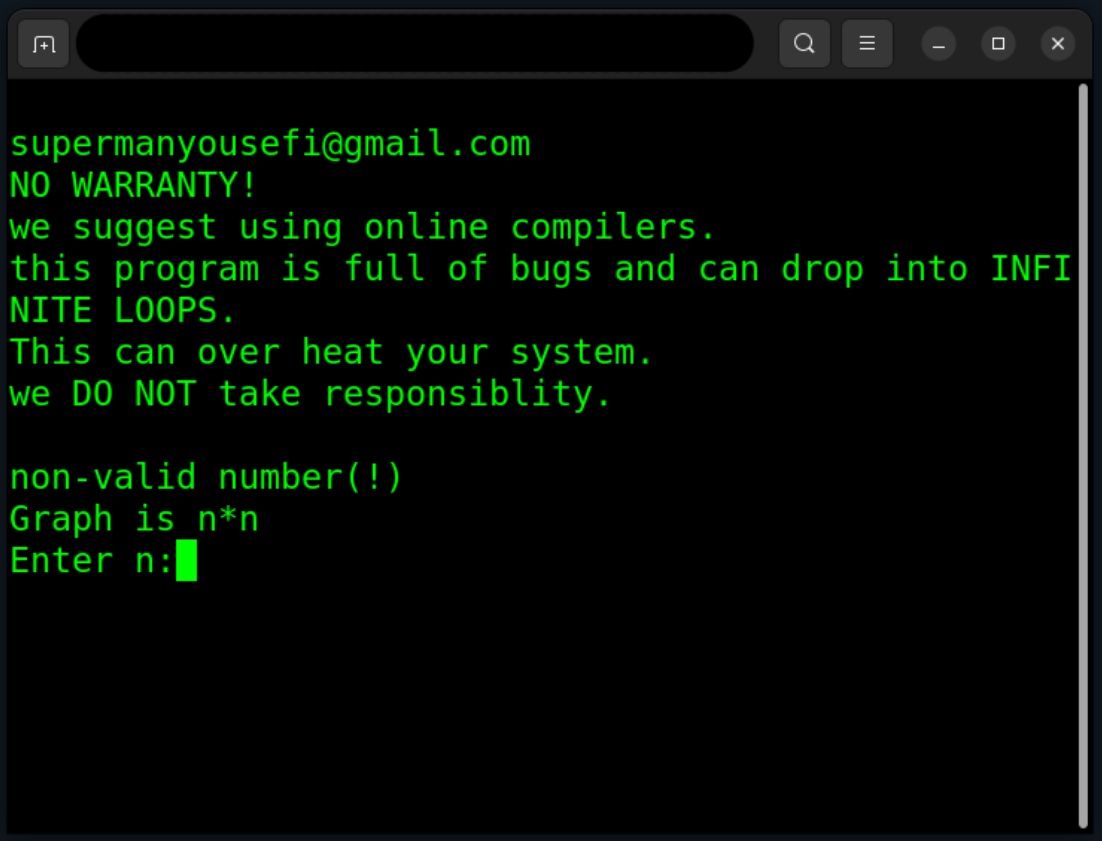
Table of Contents

3. Data Structures
    -

How to use

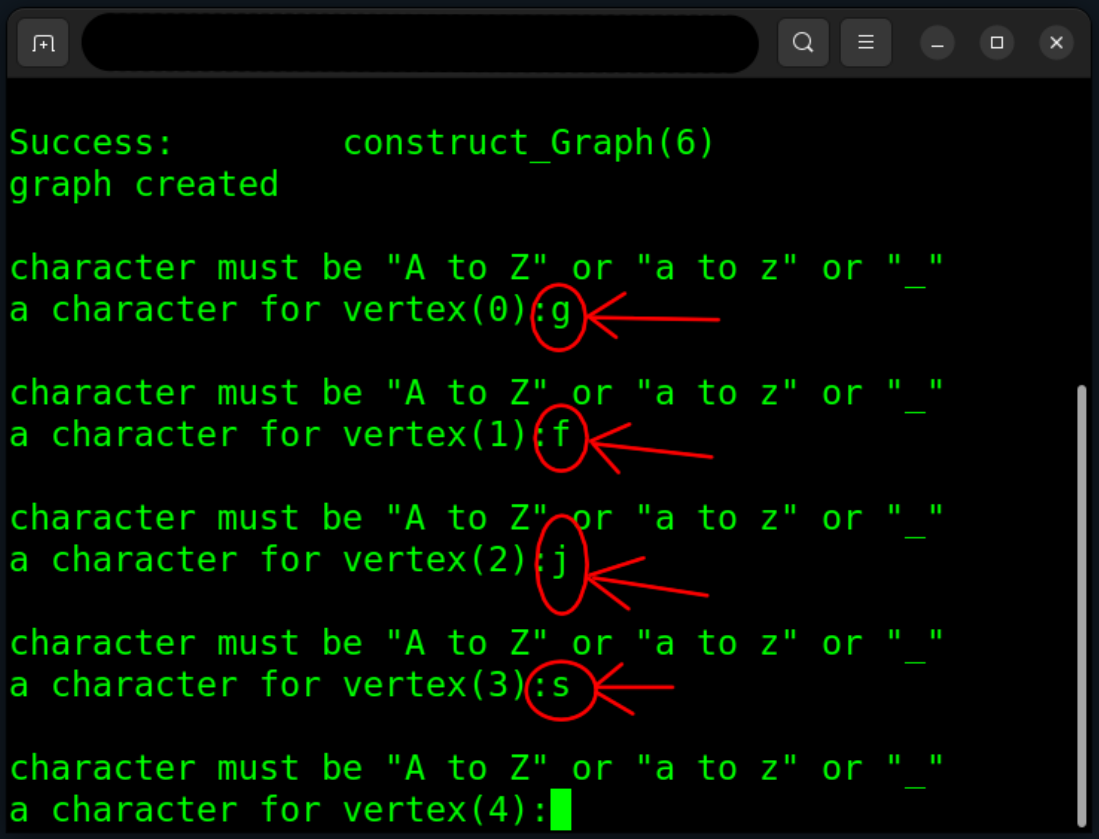
Number of Vertexes

first, you are asked to give the number of vertexes. you should give an integer number between 2 and 16. after giving correct number, memory will be allocated for an **undirected graph** and its attributes such as an array for its matrix. in case of wrong input,it will continue asking.
Be Carefull: if the input is not integer, it will go in an **INFINITE LOOP**.

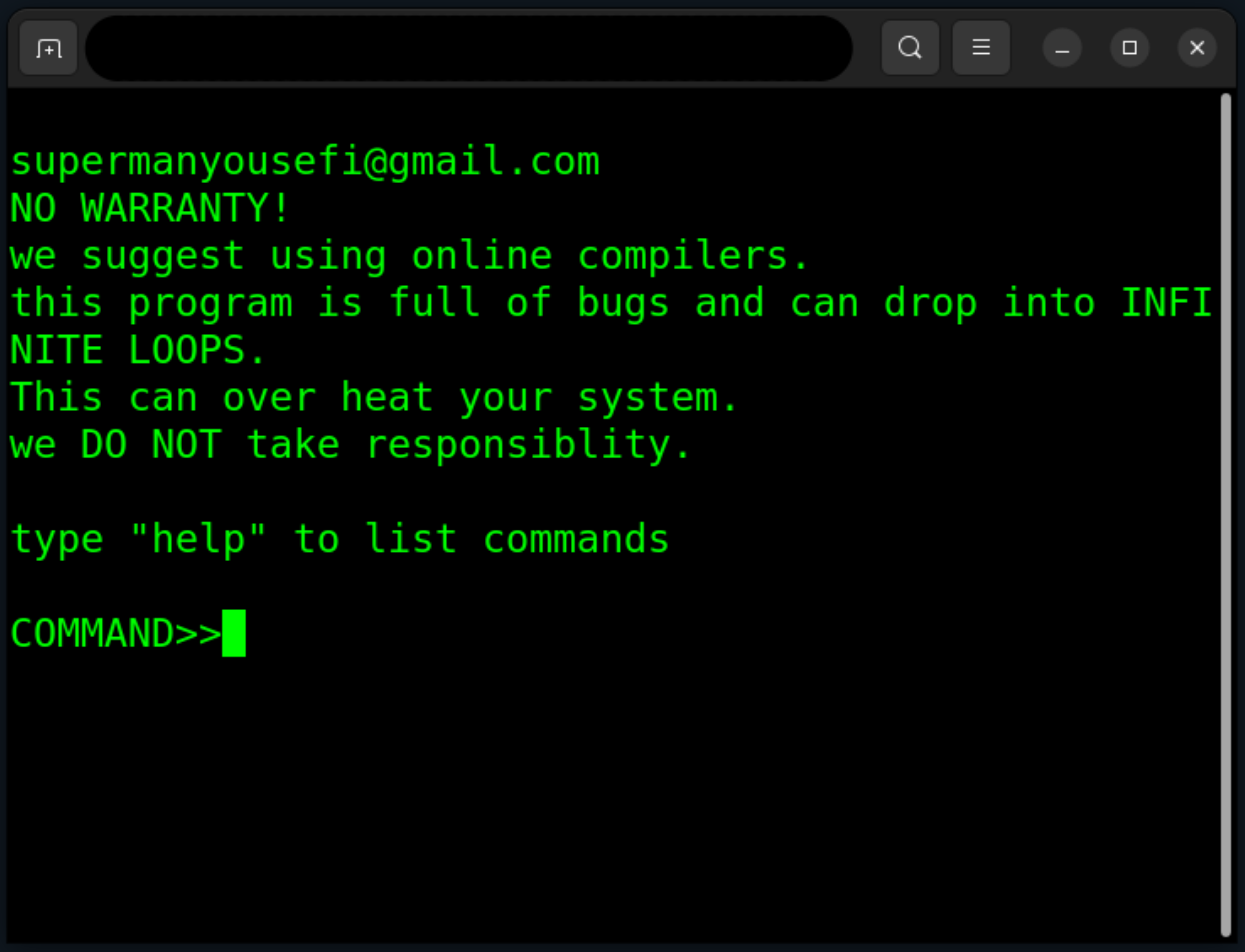


Names of vertexes

this part, you will define names for each vertexes. a vertex name is a character we use to represent a vertex. its goal is to avoid using indexes and numbers. each vertex should be represented by only one character. only alphabetical characters (A to z) or " "(underscore) is allowed. every vertex must have different name. **since its case sensitive, vertex "A" and "a" are different, so they are allowed.** if you try to give not allowed character it will continue asking. Giving more then one character cause unpredicted result.



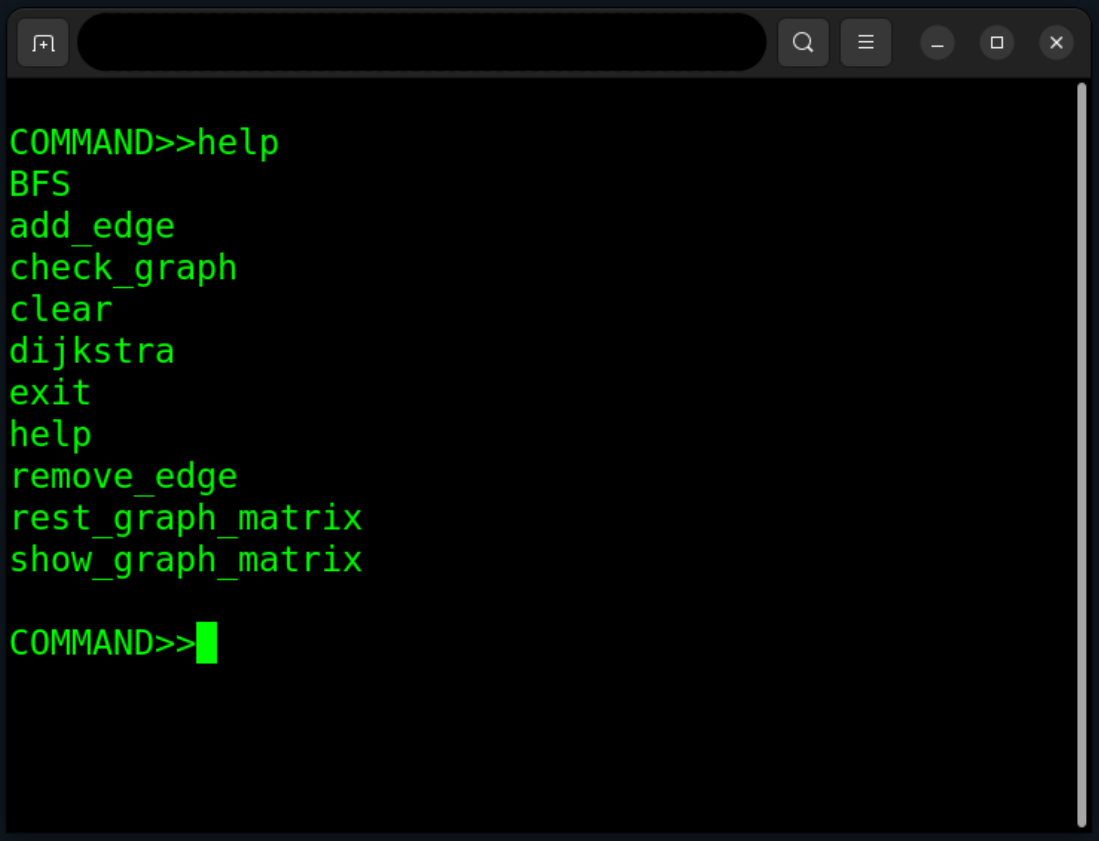
Command Line



here you have already set your graph. the shell is already yours. type help to get a list of commands.

"help"

"help" command lists all the possibl commands as it is shown below.

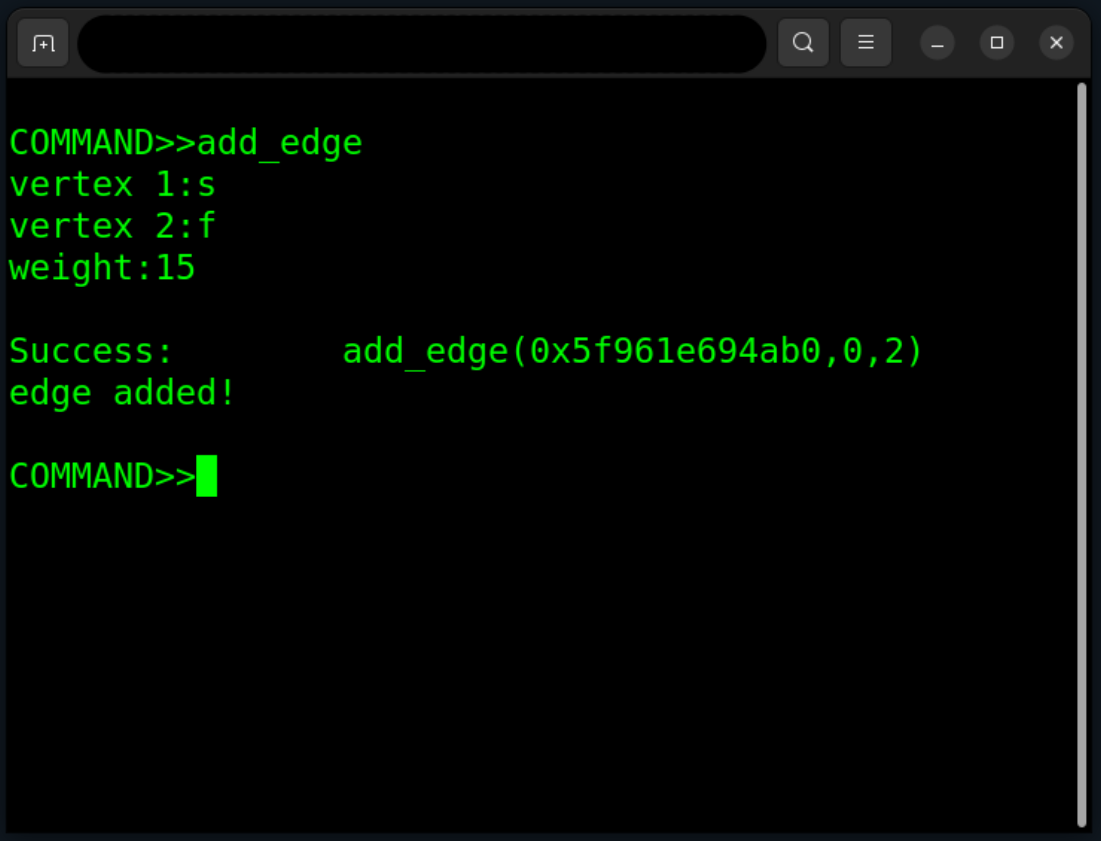


"show_graph_matrix"

"show_graph" is used to show the matrix of the graph. the table you see is representing vertex names, and each number is weight of the edges. how ever, N/E means no edges.

"add_edge

"add_edge" is for adding edges to the grap. first it asks for first vertex. you should give only one character here and if do more than one, unpredicted results happen. if the vertex you entered, is not in graph, then you will face an error. since loops are not allowed, then an error raises if you enter the same vertex. weight is also asked and it can not be negative or zero. (zero means no edge and negative is not allowed).



remove_edge

removes given edge

"reset_graph_matrix"

removes all edges

check_graph

it checks graph for alone vertexes

BFS

first it automatically runs check_graph then if ther are no alone vertexes, it runs BFS algorithm from the start_vertex that you will give.

dikjstra

the same as BFS, it automatically runs check_graph, in case of no alone vertexes it will run dikjstra algorithm.

clear

it clears command line

exit

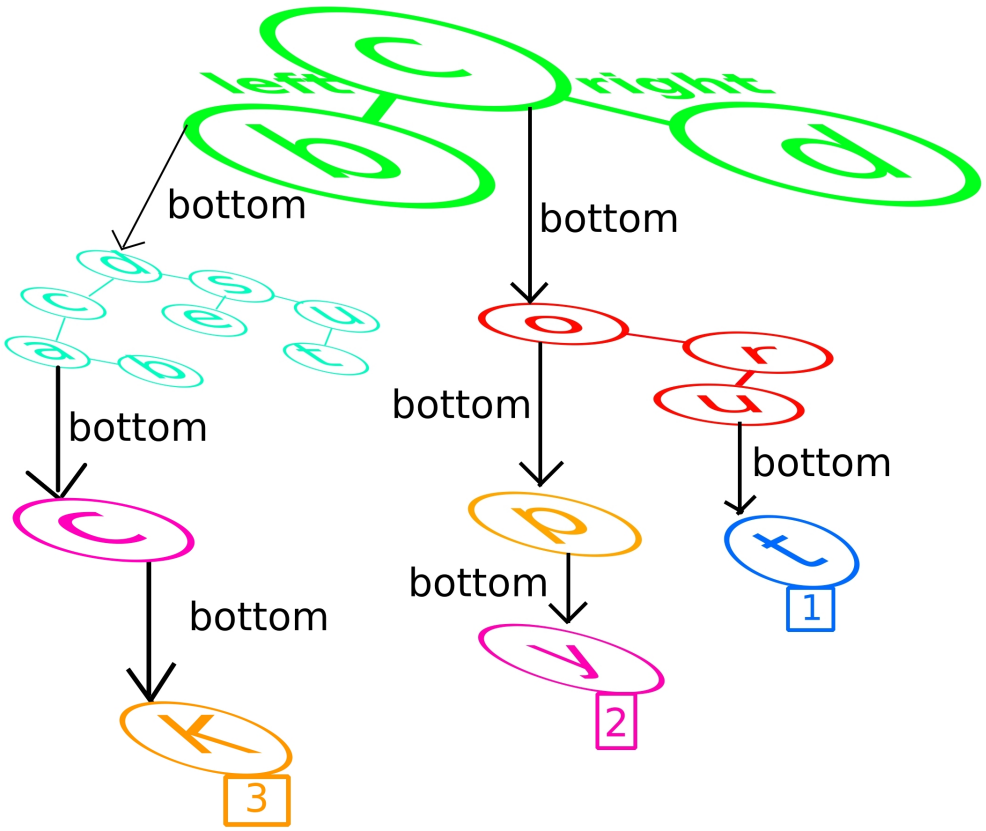
terminates the program

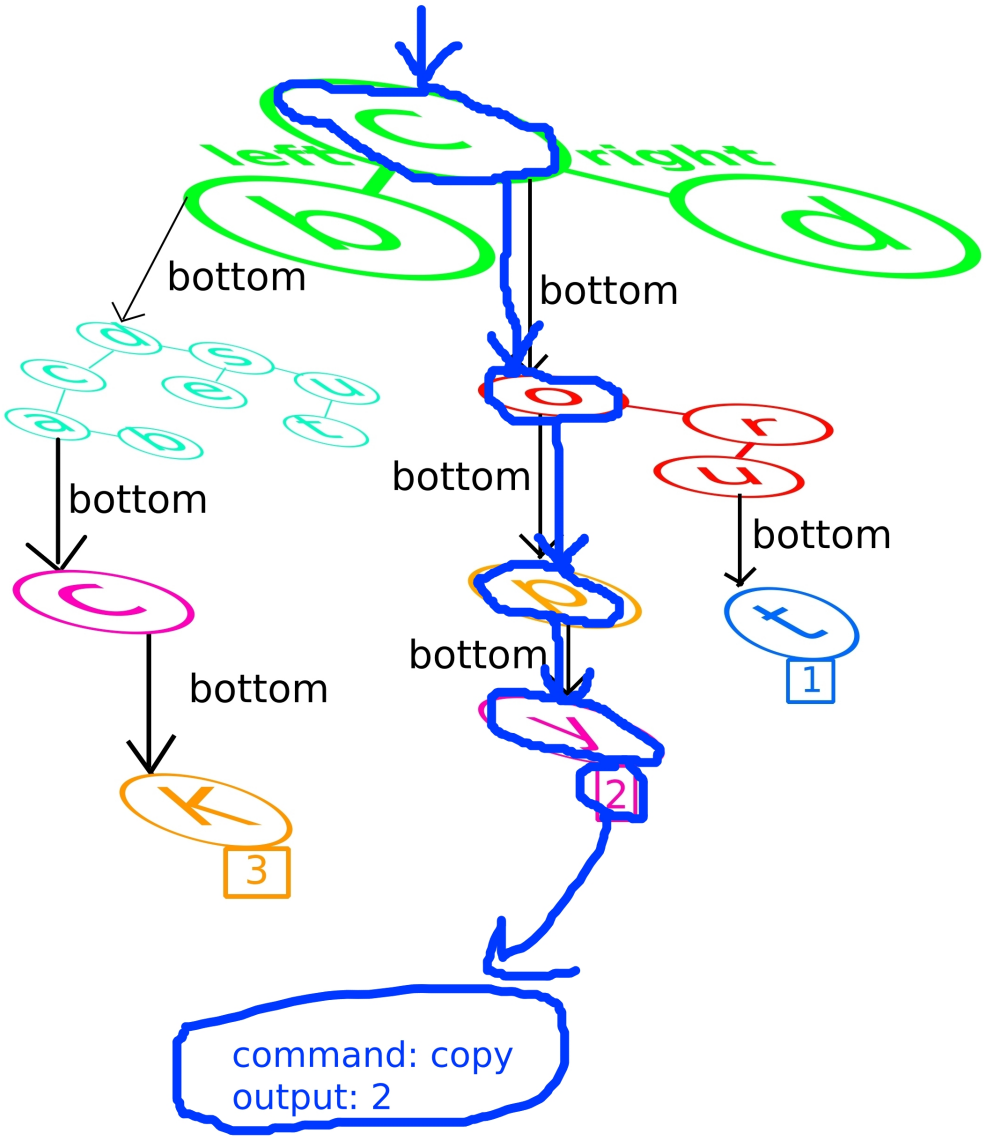
command line tree

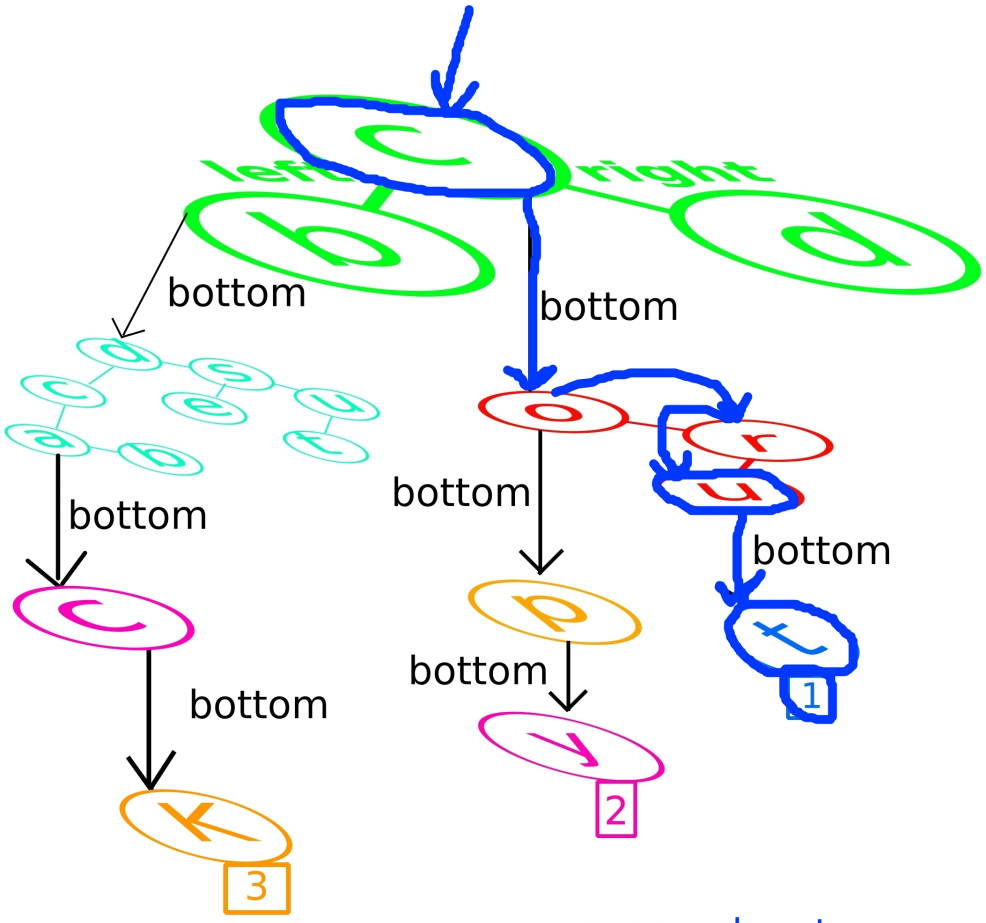
its a tree based commandline. in this tree, characters are stored in nodes, every node has neighbors left and right. for every character , characters from its right neighbors have greater ascii code and vise versa

pictures

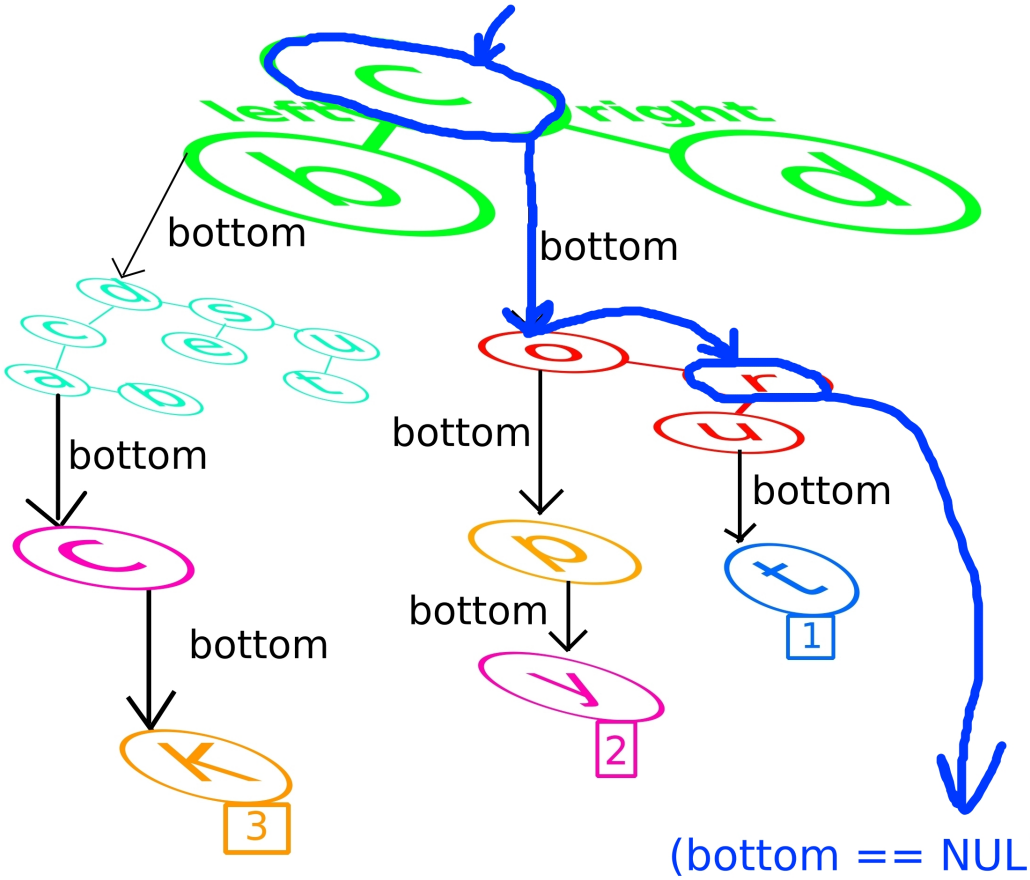
picture below explaines everything, they will provide examples of different command searching


left
right
bottom
bottom
bottom
bottom
bottom
bottom
bottom
1
2
3


left
right
bottom
bottom
bottom
bottom
bottom
bottom
bottom
1
2
3
command: copy
output: 2

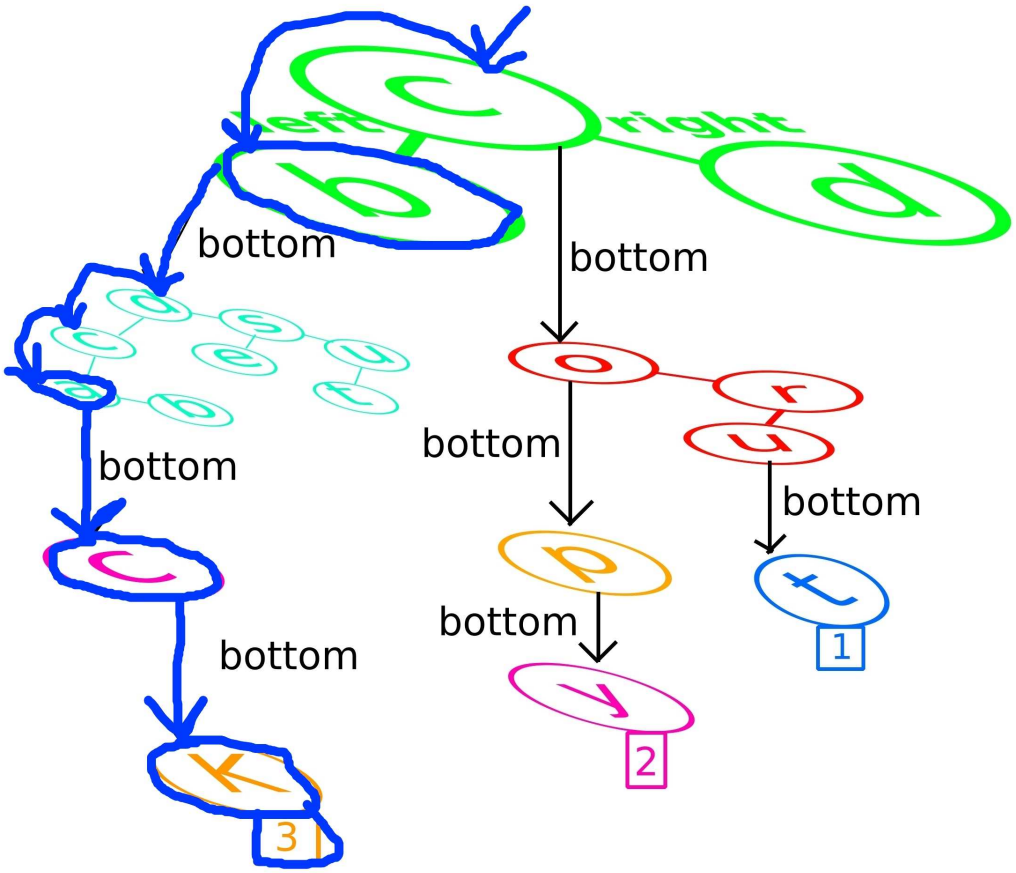


command: cut
'c' is found at green tree
'u' is found at red tree
't' is found at blue tree
so the output is found.
the output is 2



(bottom == NULL)

COMMAND : create
'c' is found.
'r' is found.
bottom == NULL
'e' not found so
the output is -1



command: back
'b' is found at green tree
'a' is found at light blue tree
'c' is found at purple tree
'k' is found
the output is 3