When I wrote this Program, I found that this program have three function (not include main), there are: char *TKGetNextToken(TokenizerT * tk) {, TokenizerT *TKCreate(char * ts) { and void TKDestroy(TokenizerT * tk) {, also I have a main function.

So I was focusing on how to use these functions, first let's talk about the structure and TKCreate:

Because this function is want to get a char* type, which is a string, so I was thinking if I can separate the space and get a string which will be the string that I can input as char* ts. So I just wrote one single line for the structure of the string, it will be char* string in the Tokenizer structure. This char* will get the string which is exactly the string, a word, between two spaces, or more spaces for start at 0 and with a space.

Now, consider the main method, because main method should call TKCreate(char * ts) and then it will return a TokenizerT type pointer. Then we can use that point to plug in TKGetNextToken(TokenizerT * tk).

Before we talk about the TKGetNextToken(TokenizerT * tk) function, let's talk about the main function for a bit.

So I was thinking if I can get a main function which can already help me decide the space and separate the space by char * that will be perfect, because what I want in TKCreate(char * ts) is actually the string that with out space. so I used a for loop to find the string between two space. I created a integer a which will locate the string is the front location of the whole string, and then I will use the for loop to go through every character and until find the space. This will also include is start at zero(location). Now, before go to next string, I will use this string first (which is I already found without space).

Also, for the input that is not belongs to digit or '+' or '-' or 'X' or 'x' or it is before 'a' and after 'f'or before 'A' and after 'F', all will be error, so I will treat these character as space, when I find my string, will not include these characters and I will use my method to print out the error in [0xhh] form.

After get the tokenizer type pointer, I could use TKGetNextToken(TokenizerT *tk)now. In this function, I will get a string that is not invalid input, so I have to decide if it is hex, Octal, decimal or float, so I wrote 4 recurs function for deciding if it is one of them. After all, if all these function go through and still cannot find it it is one of these numbers, it will be mal formed because I have a condition before go through recurs, if the first char is digit, otherwise, it will be invalid input such as "e", it will be able to go in to TKGetNextToken(TokenizerT * tk), because it is suitable for

the condition in main function. and finally it will return invalid input or mal formed or hex octal decimal or float as a string.

I used a trick for this one, because I know the main is using the sting call the function, so I will be able to use the sting again(exactly same string as in TKGetNextToken), so I took this advantage for my program.

After all, for the Invalid input, I used main to decide if the single character is an invalid input or mal form. Considering all character after f and not a digit will be invalid input. Also when it start at 0, it must be a Octal, float, or a Hex. So I read the first character of a string to see if that is 0 or not, also considering that if the string leads by 0 it will only be able to print a point (“.”) at first location, and other wise is invalid input.

Also, I will use the string first, and then I will ask a char type variable remember the character, so that I will be able to call the error function after I printed out the output (a type of number or mal form), also I will print error before I print mal form.

Thank you for reading.

Have a great day.

Chijun Sha