SCANNER ASSIGNMENT

Fady Fares Abdel Ahad Sobhy

SEC 2 Systems Software CSE422

1 USER GUIDE

1. Put your tiny language code in “tiny_sample_code.txt” and run the executable file.
2. You should find the accepted tokens printed on the screen and in an output File “scanner_output.txt”.

2 CODE EXPLANATION

1. There’s 3 files “scanner.h”, “scanner.c” and “main.c”
2. Scanner.h contains the functions prototypes and bunch of macros defined to be more readable one of the most important macros are:
    a. State machine states :
       #define State_Start 0
       #define State_InComment 1
       #define State_InNum 2
       #define State_InID 3
       #define State_InAssign 4
    b. Checking if it was a letter :
       #define LetterLimitStart 65
       #define LetterLimitEnd 122
       #define IsLetter(x) x>=LetterLimitStart&& x<=LetterLimitEnd
    c. Checking if it was a digit:
       #define DigitLimitStart 48
       #define DigitLimitEnd 57
       #define IsDigit(x) x>=DigitLimitStart&& x<=DigitLimitEnd
    d. Input and Output files name
       #define InputFile "tiny_sample_code.txt"
       #define OutputFile "scanner_output.txt"
3. There are 4 functions :
    a. void scan(void); it’s used as an API for the scanner where it’s called in the main to scan the input file. It starts by taking each character from the input file and sending it to the UpdateState() method.
    b. void printInfile(); it’s used to check for the token and compare it with reserved words and symbols in order to identify the tokens string value. Then finally print it on the screen and in a file.
    c. void UpdateState(unsigned char x); used to check for the current state and determine the next state.
    d. void Done(); used when updateState() reaches an accepting state so it call this method to do an action with the current token. Note that this function actually call printInfile() to do the checking and the printing task.