READ ME – ASSIGNMENT 1

157003077
jv432
Josh Vilson

Synopsis:

My program follows the finite state machine where it follows each case of the machine step by step. It first starts in the main method where it creates the object and string, does any necessary error checking for the token string and calls TKGetNextToken. From there TKGetNextToken looks for spaces to separate each token and does any necessary error checking as needed. It then calls findingEscapeCharacters to be able to find escape characters within the token. The findingEscapeCharacters function is separated into two cases the first being the position at 0 for the token is not a digit and otherwise it's a digit. The checkState function gets called by findingEscapeCharacters where it checks whether to call state0 or state19. This function also checks for errors as needed. The state0 function looks for tokens whose first position starts with 0 and limits the posibilites on the finite state machine. The state19 functions looks for tokens whose first position start with1 and may range up to 9 only for the first character. The floatMethod generally looks for floats and prints them out accordingly if it is a valid float. The hexMethod generally looks for hexadecimal values and prints. The states are printed out accordingly through these steps and errors are printed if there are escape characters as follows:

Example "123p12" is outputted as

decimal 123 mal formed [0x70] decimal 12

Example "0x02g" is outputted as

hexadecimal 0x02 mal formed [0x67]

More detailed information for each method below:

Struct

Contains variables: tokens – which contains the entire list of tokens that are separated by spaces, token- which contains a single token of the entire list of tokens and moves to the next token when fully traversed and currPos – which is the current position of the tokens variable character pointer, to be able to traverse it. Arr1 is used so I would be able to check for escape characters separately rather than with the white spaces.

Main Function

Sets up the program by calling the first method and initializes a variable to be used in a loop.

TKGetNextToken

This method first checks to see if the length of the tk->tokens is not null, if null it returns 0. If the current position is a space and has a length of 1, 0 is returned. It then traverses the string to find white spaces. When a white space is encountered this function either increments the position at the entire string by 1 and checks the length of the arr1. If the length of arr1

findingEscapeCharacters

this function searches for escape characters in every possible case. It calls check state if arr1 is populated with data.

checkState

Checks the first position of the token to see it beings with a 0 or 1 – 9 and calls state0 and state19 accordingly to simple out the cases of the finite state machine.

state0

Since the first position is a 0, we know that it could be either a octal/hexadecimal/float. This function checks to see which cases satisfy the 3 different states and calls or prints accordingly.

state19

Since the first position is a 1-9, we know that it could be either a float/decimal which makes the finite machine a bit more simple to traverse. This function checks to see which cases satisfy the 2 different states and calls or prints accordingly.

floatingMethod

prints out the floating point state, looking through each position of the individual token

hexMethod

prints out the hexadecimal state, looking through each position of the individual token.