#date:09-08-19

Decided to use a struct with a union built inside it called Lexeme which can hold a token value and attribute value

#date:09-10-19

made the relop and idres machines in c and it returns a lexeme that has a token value and attribute value. The idres machine uses a match function that takes in a character and an array of characters to see if the front pointers dereferenced value is in the library of characters. Choosing to work with double pointers for the machines was really confusing at first but now it has gotten a bit easier to understand.

match(char fptr,char arr[],int arrlen)

{

for(int I = 0; I < arrlen; I++){

if(fptr == arr[I]){

return 1;

}

}

return 0;

}

this function is really useful since both the letter array and digit array are going to be character arrays.

#date: 09-15-19

integer and real were easy machines to create, but I am having an issue, if I have a string “1a1a” I would expect my compiler to return Int: 1 Id: a1a and it does, but it then traps itself in an infinite loop afterwards. If I have a string “1 a1a” there are no problems.

#date:09-16-19

the issue with the string problem seems to be that the front pointer is jumping one too many times in the case of “<Int><ID>” looking at the value \*(\*fptr--) I see a newline character was there, im just not sure where the extra jump is coming from.