21BCE1882

JEFFERSON DAVID KINGSTON

COMPILER DESIGN LAB-3 ASSIGNMENT:

CODE:

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <ctype.h>

int keywordchk(char buffer[])

{

char keywords[35][10] = {"auto", "break", "case", "char", "const", "continue", "default",

"do", "double", "else", "enum", "extern", "float", "for", "goto",

"if", "int", "long", "main", "printf", "register", "return", "scanf","short", "signed",

"sizeof", "static", "struct", "switch", "typedef", "union",

"unsigned", "void", "volatile", "while"};

int i, flag = 0;

for (i = 0; i < 35; ++i)

{

if (strcmp(keywords[i], buffer) == 0)

{

flag = 1;

break;

}

}

return flag;

}

int headerfilechk(char buffer[])

{

char headerFiles[5][20] = {"stdio.h", "stdlib.h",

"string.h", "ctype.h", "math.h"};

int i, flag = 0;

for (i = 0; i < 5; ++i)

{

if (strcmp(headerFiles[i], buffer) == 0)

{

flag = 1;

break;

}

}

return flag;

}

int fileinc\_chk(char buffer[])

{

char fid[2][10] = {"include", "define"};

int i, flag = 0;

for (i = 0; i < 5; ++i)

{

if (strcmp(fid[i], buffer) == 0)

{

flag = 1;

break;

}

}

return flag;

}

int mathcst(char buffer[])

{

char mc[7][10] = {"PI", "E", "LN2", "LN10", "LOG2E", "LOG10E", "3.14"};

int i, flag = 0;

for (i = 0; i < 5; ++i)

{

if (strcmp(mc[i], buffer) == 0)

{

flag = 1;

break;

}

}

return flag;

}

int main()

{

char ch, buffer[15], operators[] = "+-\*/%=";

int insideMain = 0;

FILE \*fp;

int i, j = 0;

fp = fopen("calculate.c", "r");

if (fp == NULL)

{

printf("Error while opening file\n");

exit(0);

}

while ((ch = fgetc(fp)) != EOF)

{

if (ch == '"')

{

while((ch = fgetc(fp)) != '"')

{

continue;

}

}

if((ch == '>' || ch == '<') && insideMain == 1)

{

printf("%c\t\t--\t operator\n", ch);

}

for (i = 0; i < 6; ++i)

{

if (ch == operators[i])

printf("%c\t\t--\t operator\n", ch);

}

if(isalpha(ch) || isdigit(ch) || ch == '\_' || ch == '.')

{

buffer[j++] = ch;

}

else if ((ch == ' ' || ch == '\n' || ch == ',' || ch == ';' || ch == '(' || ch == ')' || ch == '<' || ch == '>' || ch == '=' || ch == '\*' || ch == '+' || ch == '-') && (j != 0))

{

buffer[j] = '\0';

j = 0;

if(buffer == "main") {insideMain = 1;}

if (keywordchk(buffer) == 1)

printf("%s\t--\t keyword\n", buffer);

else if(headerfilechk(buffer) == 1)

printf("%s\t--\t header file\n", buffer);

else if(fileinc\_chk(buffer) == 1)

printf("%s\t--\t File Inclusion Directive\n", buffer);

else if(mathcst(buffer) == 1)

printf("%s\t--\t Math Constant\n", buffer);

else if (isdigit(\*buffer) == 1)

printf("%s\t--\t Numeral\n", buffer);

else

printf("%s\t--\t indentifier\n", buffer);

}

}

fclose(fp);

return 0;

}

Input:

#include <stdio.h>

#include <math.h>

#define PI 3.14

void main()

{

int radius\_1;

float area\_cir, perimeter;

int i,code;

for(i=1;i<=10;i++)

{

scanf("%d",&code);

scanf("%d",&radius\_1);

if (code==1)

{

area\_cir = PI\*pow(radius\_1,2);

printf("area of the circle %f",area\_cir);

}

else

{

perimeter = 2\*PI\*radius\_1;

printf("perimeter of the circle %f",perimeter);

}

}

}

OUTPUT-

Text

Description automatically generated

Text

Description automatically generated