#include <stdio.h>

#include <stdlib.h>

#include <conio.h>

FILE \*input, \*output;

void enterInputFILE(void)

{

char inputDir[256];

do{

printf("Vuvedete input faila (C programa koqto SE KOMPILIRA USPE6NO!): ");

scanf("%s", &inputDir);

input = fopen(inputDir, "r");

if(input==NULL)

{

printf("Gre6ka pri otvarqne na faila!\n");

}

}while(input == NULL);

}

void enterOutput(void)

{

char outputDir[222];

printf("\nEnter output filelocation: ");

scanf("%s",&outputDir);

output=fopen(outputDir,"a+");

}

void enterFromKeyboard(void)

{

int currentChar;

input = fopen("input.txt", "w");

printf("Vuvedete programata do natiskane na CTRL + Z (bez da triete): \n");

do{

currentChar = getch();

if(currentChar != 26)

{

if(currentChar == '\r')

{

printf("\n");

fprintf(input,"\n");

}

else

{

printf("%c", currentChar);

fprintf(input,"%c",currentChar);

}

}

}while(currentChar != 26);

fclose(input);

input = fopen("input.txt", "r");

}

void calculateOperator(int \*ravnoCounter)

{

int i, currnetRowLength, rowsCounter = 0, row;

char currnetRow[1000];

int currentChar;

int check1 = 0, check2 = 0, check3 = 0;

/////////////////////////////////////

rewind(input);

do{

currentChar=fgetc(input);

if(currentChar == '\n')

{

rowsCounter++;

}

}while(currentChar != EOF);

rowsCounter++;

//////////////////////

rewind(input);

for(row = 0; row < rowsCounter; row++)

{

for(i=0;;i++)

{

currentChar=fgetc(input);

if(currentChar == EOF || currentChar == '\n')

{

currnetRow[i] = '\0';

currnetRowLength = i;

break;

}

else

{

currnetRow[i] = currentChar;

}

}

//////////////////////////

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

{

if(currnetRow[i]=='\"')

{

for(;;)

{

i++;

if(currnetRow[i]=='\\')

{

i++;

if(currnetRow[i]=='\"')

{

continue;

}

}

if(currnetRow[i]=='\"') break;

if(currnetRow[i]=='\0') break;

}

}

if( (check1 == 1) && (currnetRow[i] == '\*'))

{

for(;;)

{

i++;

if(currnetRow[i]=='\*')

{

check3=1;

continue;

}

if((check3==1) && (currnetRow[i]=='/'))

{

check1 = 0;

check2 = 0;

check3 = 0;

break;

}

else

{

check3=0;

}

if(currnetRow[i]=='\0')

{

//read new line and continue

for(i=0;;i++)

{

currentChar=fgetc(input);

if(currentChar == EOF || currentChar == '\n')

{

currnetRow[i] = '\0';

currnetRowLength = i;

break;

}

else

{

currnetRow[i] = currentChar;

}

}

i=0;

continue;

}

}

continue;

}

else

{

check1 = 0;

}

if( (check2 == 1) && (currnetRow[i] == '/'))

{

check1 = 0;

check2 = 0;

do{

i++;

}while(currnetRow[i]!='\0');

if(currnetRow[i]=='\0')

{

break;

}

else

{

continue;

}

}

else

{

check2 = 0;

}

if(currnetRow[i] == '/')

{

if(i < currnetRowLength - 1)

{

if(currnetRow[i + 1] == '/' || currnetRow[i + 1] == '\*')

{

check1 = 1;

check2 = 1;

continue;

}

}

}

if(currnetRow[i] == '=')

{

(\*ravnoCounter)++;

if(i < currnetRowLength - 1)

{

if(currnetRow[i + 1] == '=')

{

i++;

(\*ravnoCounter)--;

}

}

if(i > 0)

{

if(currnetRow[i-1] == '!')

{

(\*ravnoCounter)--;

}

else if(currnetRow[i + 1] == '<')

{

(\*ravnoCounter)--;

}

else if(currnetRow[i + 1] == '>')

{

(\*ravnoCounter)--;

}

}

}

}

}

}

void calculateFORWHILE(int \*cikulCounter)

{

int i, currnetRowLength, rowsCounter = 0, row;

char currnetRow[1000];

int currentChar, isDoFound = 0;

int check1 = 0, check2 = 0, check3 = 0;

//////////////////////////////

rewind(input);

do{

currentChar=fgetc(input);

if(currentChar == '\n')

{

rowsCounter++;

}

}while(currentChar != EOF);

rowsCounter++;

rewind(input);

for(row = 0; row < rowsCounter; row++)

{

for(i=0;;i++)

{

currentChar=fgetc(input);

if(currentChar == EOF || currentChar == '\n')

{

currnetRow[i] = '\0';

currnetRowLength = i;

break;

}

else

{

currnetRow[i] = currentChar;

}

}

//////////////////////////

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

{

if(currnetRow[i]=='\"')

{

for(;;)

{

i++;

if(currnetRow[i]=='\\')

{

i++;

if(currnetRow[i]=='\"')

{

continue;

}

}

if(currnetRow[i]=='\"') break;

if(currnetRow[i]=='\0') break;

}

}

if( (check1 == 1) && (currnetRow[i] == '\*'))

{

for(;;)

{

i++;

if(currnetRow[i]=='\*')

{

check3=1;

continue;

}

if((check3==1) && (currnetRow[i]=='/'))

{

check1 = 0;

check2 = 0;

check3 = 0;

break;

}

else

{

check3=0;

}

if(currnetRow[i]=='\0')

{

//read new line and continue

for(i=0;;i++)

{

currentChar=fgetc(input);

if(currentChar == EOF || currentChar == '\n')

{

currnetRow[i] = '\0';

currnetRowLength = i;

break;

}

else

{

currnetRow[i] = currentChar;

}

}

i=0;

continue;

}

}

continue;

}

else

{

check1 = 0;

}

if( (check2 == 1) && (currnetRow[i] == '/'))

{

check1 = 0;

check2 = 0;

do{

i++;

}while(currnetRow[i]!='\0');

if(currnetRow[i]=='\0')

{

break;

}

else

{

continue;

}

}

else

{

check2 = 0;

}

if(currnetRow[i] == '/')

{

if(i < currnetRowLength - 1)

{

if(currnetRow[i + 1] == '/' || currnetRow[i + 1] == '\*')

{

check1 = 1;

check2 = 1;

continue;

}

}

}

if(currnetRow[i] == 'f')

{

if( i > 0)

{

if(!(currnetRow[i-1] == ' ' || currnetRow[i-1] == '\t' || currnetRow[i-1] == '\v' || currnetRow[i-1] == '\n'))

{

continue;

}

}

if(i < currnetRowLength - 1)

{

i++;

if(currnetRow[i] == 'o')

{

if(i < currnetRowLength - 1)

{

i++;

if(currnetRow[i] == 'r')

{

if(i < currnetRowLength - 1)

{

i++;

if(currnetRow[i] == '(' || currnetRow[i] == ' ' || currnetRow[i] == '\t' || currnetRow[i] == '\v' || currnetRow[i] == '\n')

{

(\*cikulCounter)++;

}

}

else

{

(\*cikulCounter)++;

}

}

}

}

}

}

if(currnetRow[i] == 'd')

{

if( i > 0)

{

if(!(currnetRow[i-1] == ' ' || currnetRow[i-1] == '\t' || currnetRow[i-1] == '\v' || currnetRow[i-1] == '\n'))

{

continue;

}

}

if(i < currnetRowLength - 1)

{

i++;

if(currnetRow[i] == 'o')

{

if(i < currnetRowLength - 1)

{

i++;

if(currnetRow[i] == '{' || currnetRow[i] == ' ' || currnetRow[i] == '\t' || currnetRow[i] == '\v' || currnetRow[i] == '\n')

{

(\*cikulCounter)++;

isDoFound = 1;

}

}

else

{

(\*cikulCounter)++;

isDoFound = 1;

}

}

}

}

if(currnetRow[i] == 'w')

{

if( i > 0)

{

if(!(currnetRow[i-1] == ' ' || currnetRow[i-1] == '\t' || currnetRow[i-1] == '\v' || currnetRow[i-1] == '\n' || currnetRow[i-1] == '}'))

{

continue;

}

}

if(i < currnetRowLength - 1)

{

i++;

if(currnetRow[i] == 'h')

{

if(i < currnetRowLength - 1)

{

i++;

if(currnetRow[i] == 'i')

{

if(i < currnetRowLength - 1)

{

i++;

if(currnetRow[i] == 'l')

{

if(i < currnetRowLength - 1)

{

i++;

if(currnetRow[i] == 'e')

{

if(i < currnetRowLength - 1)

{

i++;

if(currnetRow[i] == '(' || currnetRow[i] == ' ' || currnetRow[i] == '\t' || currnetRow[i] == '\v' || currnetRow[i] == '\n')

{

if(isDoFound == 1)

{

isDoFound = 0;

}

else

{

(\*cikulCounter)++;

}

}

}

else

{

(\*cikulCounter)++;

}

}

}

}

}

}

}

}

}

}

}

}

}

void function(int choice)

{

int cikulCounter = 0;

int ravnoCounter = 0;

//////////////////////////////////////////////////

if (choice == 1 || choice == 2)

{

enterInputFILE();

}

else

{

enterFromKeyboard();

}

calculateOperator(&ravnoCounter);

calculateFORWHILE(&cikulCounter);

if(choice == 1 || choice == 3)

{

enterOutput();

fprintf(output, "Broqt na operatorite za prisvoqvane e %d.\n", ravnoCounter);

}

else

{

printf("\nBroqt na operatorite za prisvoqvane e %d.\n", ravnoCounter);

}

/////////////////////////////////////////////////////////////////////////////////////////////////////

if(choice == 1 || choice == 3)

{

fprintf(output, "Broqt na operatorite za cikul e %d.\n", cikulCounter);

fclose(output);

}

else

{

printf("\nBroqt na operatorite za cikul e %d.\n", cikulCounter);

}

fclose(input);

system("Pause");

}

int main(void)

{

int choice;

do{

system("cls");

printf("1. Ot fail vuv fail.\n");

printf("2. Ot fail na ekrana.\n");

printf("3. Ot klaviaturata vuv fail.\n");

printf("4. Of klaviaturata na ekrana.\n");

printf("0. Izxod.\n");

printf("Izberete opciq ot slednoto menu: ");

scanf("%d", &choice);

if(choice == 1 || choice == 2 || choice == 3 || choice == 4)

{

function(choice);

}

else if(choice != 0)

{

printf("Nevalidna opciq\n");

system("Pause");

}

}while(choice != 0);

system("Pause");

}