/// MD Abdullah Al Nasim

///15.01.04.085

/\*

assignment 2:

separate first then recognize and mark the lexemes as different types of tokens like keywords, identifiers, operators, separators, parenthesis, numbers, etc.

\*/

#include <bits/stdc++.h>

#define MAX 100

using namespace std;

int keywordRec(char lex[MAX])

{

char key[MAX][MAX] = {"char","int","float","if","else" };

for(int i=0;i<5;i++) if(strcmp(lex,key[i])==0) return 1;

return 0;

}

int separatorRec(char lex[MAX])

{

if(!strcmp(lex,",") || !strcmp(lex,";") || !strcmp(lex,"'")) return 1;

else return 0;

}

int operatorRec(char lex[MAX])

{

if(!strcmp(lex,"+") || !strcmp(lex,"-") || !strcmp(lex,"\*") || !strcmp(lex,"/") || !strcmp(lex,"<") || !strcmp(lex,">") || !strcmp(lex,"<=") || !strcmp(lex,">=") || !strcmp(lex,"=") || !strcmp(lex,"!")) return 1;

else return 0;

}

int parenthesisRec(char lex[MAX])

{

if(!strcmp(lex,"(") || !strcmp(lex,")") || !strcmp(lex,"{") || !strcmp(lex,"}") || !strcmp(lex,"[") || !strcmp(lex,"]")) return 1;

else return 0;

}

int identifierRec(char lex[MAX])

{

int s=0;

if(isalpha(lex[0]) || lex[0]=='\_') s=1;

if(s==1){

for(int i=1; i<strlen(lex); i++){

if(!isalnum(lex[i]) && lex[i]!='\_'&& !isalpha(lex[i])){

s=0;

break;

}

}

}

return s;

}

int numberRec(char lex[MAX])

{

int flag=0;

for(int i=1; i<strlen(lex); i++){

if(isdigit(lex[i])) continue;

else{

flag=1;

break;

}

}

if(flag==1) return 1;

else return 0;

}

int main()

{

FILE \*input;

FILE \*output;

char c;

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

output = fopen("output1.txt", "w");

if(!input) printf("\nFile can't be opened!");

else{

while((c=fgetc(input))!=EOF){

if(!isalnum(c) && c!=' ' && c!='\_' && c!='.') fputc(' ', output);

fputc(c, output);

if(c=='>' || c=='<'|| c=='=' || c=='!'){

char ch;

if((ch=fgetc(input))=='='){

fputc(ch, output);

if(!isalnum(ch) && ch!=' ' && ch!='\_' && ch!='.') fputc(' ', output);

}

else{

fputc(' ', output);

fputc(ch, output);

if(!isalnum(ch) && ch!=' ' && ch!='\_' && ch!='.') fputc(' ', output);

}

}

else if(!isalnum(c) && c!=' ' && c!='\_' && c!='.') fputc(' ', output);

}

}

fclose(input);

fclose(output);

///input

printf("Input: \n");

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

while((c=fgetc(input))!=EOF) printf("%c",c);

printf("\n");

///step1

input = fopen("output1.txt", "r");

output = fopen("output2.txt", "w");

while((c=fgetc(input))!=EOF){

if((c==' ') || (c=='\t')){

fputc(' ', output);

c=fgetc(input);

if(c!=EOF) fputc(c, output);

}

else fputc(c, output);

}

fclose(input);

fclose(output);

input = fopen("output2.txt", "r");

printf("\nStep1: \n");

while((c=fgetc(input))!=EOF) printf("%c",c);

///step2

input = fopen("output2.txt", "r");

output = fopen("output3.txt", "w");

char str[MAX];

while(fscanf(input, "%s", &str)!=EOF){

if(keywordRec(str)) fprintf(output, "[kw %s] ", str);

else if(separatorRec(str)) fprintf(output, "[sep %s] ", str);

else if(operatorRec(str)) fprintf(output, "[op %s] ", str);

else if(parenthesisRec(str)) fprintf(output, "[par %s] ", str);

else if(identifierRec(str)) fprintf(output, "[id %s] ", str);

else if(numberRec(str)) fprintf(output, "[num %s]", str);

else fprintf(output, "[unkn %s]", str);

}

fclose(input);

fclose(output);

printf("\nFinal Output: \n");

input = fopen("output3.txt", "r");

while((c=fgetc(input))!=EOF) printf("%c",c);

return 0;

}