****

**Department of**

**Computer Science and Engineering**

**Lab Assignment – 01**

Course No. : CSE-354

Course Title : Compiler Design Laboratory

Name of Experiment: Report on Regular Expression

Submitted By : Kamelia Zaman Moon

ID : 299

Session : 2017-18

Date : 15-03-2021

Submitted To : Tahsina Hashem

Assistant Professor

Department of Computer Science and Engineering

Jahangirnagar University, Savar.

**Assignment Details:**

This code accepts alphabet list a-z.

Regular expressions can be for examples-

a\*bc or abc\*

a+bc or abc+

a?bc or abc?

(ab)\*c or a(bc)\*

(ab)+c or a(bc)+

(ab)?c or a(bc)?

[a-d]\*z or a[d-z]\*

[a-d]+z or a[d-z]+

[a-d]?z or a[d-z]?

**Source Code:**

#include <bits/stdc++.h>

using namespace std;

char lists[]= {'(',')','{','}','[',']','+','-','\*','^','?'};

class RegularExpr

{

private:

string re;

public:

string getOut()

{

return re;

}

void takeIn()

{

cin>>re;

}

};

class PatMatch

{

private:

string pattern;

public:

string getOut()

{

return pattern;

}

void takeIn()

{

cin>>pattern;

}

};

int countDistinct(string s)

{

unordered\_map<char, int> m;

for (int i = 0; i < s.length(); i++)

{

m[s[i]]++;

}

return m.size();

}

int countFreq(string str,char ch)

{

int count = 0;

for (int i = 0; i < str.size(); i++)

{

if (str[i] == ch)

{

++ count;

}

}

return count;

}

int op(string withMatch, string toMatch)

{

int a=0,b=0,alen=toMatch.length(),blen=withMatch.length(),flag=0;

int res=1;

while(flag==0)

{

//cout<<toMatch.at(a)<<endl;

//cout<<withMatch.at(b)<<endl;

int times=0;

if(toMatch.at(a)==lists[0])

{

string str;

while(toMatch.at(a)!=lists[1])

{

a++;

if(toMatch.at(a)==lists[1])

break;

str=str+toMatch.at(a);

}

int len=str.length();

if((str==withMatch.substr(b,len)) && (toMatch.at(a+1)==lists[6] || toMatch.at(a+1)==lists[8] || toMatch.at(a+1)==lists[10]))

{

b=b+len;

if(toMatch.at(a+1)==lists[10] && str==withMatch.substr(b,len))

{

res=2;

}

else

{

while(str==withMatch.substr(b,len))

{

if(b+len>=blen)

break;

else

{

if(str!=withMatch.substr(b,len))

break;

if(b+len<blen)

b=b+len;

}

}

}

a=a+2;

}

else if(str!=withMatch.substr(b,len))

{

if(toMatch.at(a+1)!=lists[8] && toMatch.at(a+1)!=lists[10])

{

res=2;

}

else if((toMatch.at(a+1)==lists[8] || toMatch.at(a+1)==lists[10]))

{

a=a+2;

}

}

}

/\*else if(toMatch.at(a)==lists[2])

{

flag1=1;

string str;

while(toMatch.at(a)!=lists[3])

{

a++;

if(toMatch.at(a)==lists[3])

break;

str=str+toMatch.at(a);

}

times=stoi(str);

}\*/

else if(toMatch.at(a)==lists[4])

{

a++;

if(toMatch.at(a)==lists[9])

{

string str;

while(toMatch.at(a)!=lists[5])

{

a++;

if(toMatch.at(a)==lists[5])

break;

str=str+toMatch.at(a);

}

}

else

{

char st=toMatch.at(a),en=toMatch.at(a+2);

a=a+3;

int dif=(int)en-(int)st+1,cn=0;

if(toMatch.at(a+1)==lists[6] || toMatch.at(a+1)==lists[8] || toMatch.at(a+1)==lists[10])

{

string str;

while(st<=withMatch.at(b) && withMatch.at(b)<=en)

{

if(b==blen)

break;

if(st>withMatch.at(b) || withMatch.at(b)>en)

break;

else

{

str=str+withMatch.at(b);

b++;

}

cn++;

if(b==blen)

break;

}

cout<<str<<endl;

int dis=countDistinct(str);

int maxi=0;

for(int i=0; i<str.length(); i++)

{

int gg=countFreq(str,str[i]);

if(maxi<gg)

maxi=gg;

}

if(toMatch.at(a+1)==lists[6] && (dif!=dis || cn==0))

res=2;

else if(toMatch.at(a+1)==lists[10] && (maxi>1 || (dif!=dis && str.length()!=0)))

res=2;

}

}

a=a+2;

}

else if((toMatch.at(a)==withMatch.at(b)) && (toMatch.at(a+1)==lists[6] || toMatch.at(a+1)==lists[8] || toMatch.at(a+1)==lists[10]))

{

b++;

if(toMatch.at(a+1)==lists[10] && b==blen)

{

res=1;

}

else if(toMatch.at(a+1)==lists[10] && toMatch.at(a)==withMatch.at(b))

{

res=2;

}

else

{

if(b<blen)

{

while(toMatch.at(a)==withMatch.at(b))

{

b++;

if(b==blen)

{

break;

}

if(toMatch.at(a)!=withMatch.at(b))

break;

}

}

}

a=a+2;

}

else if(toMatch.at(a)!=withMatch.at(b))

{

if(toMatch.at(a+1)!=lists[8] && toMatch.at(a+1)!=lists[10])

{

res=2;

}

else if((toMatch.at(a+1)==lists[8] || toMatch.at(a+1)==lists[10]))

{

a=a+2;

}

}

else if(toMatch.at(a)==withMatch.at(b))

{

a++;

b++;

}

if(res==2)

{

res=0;

break;

}

if(a>=alen-1 || b>=blen)

{

if(withMatch.at(blen-1)!=toMatch.at(alen-2) && toMatch.at(alen-1)==lists[6] && toMatch.at(alen-2)!=lists[1])

res=0;

else if(withMatch.at(blen-1)!=toMatch.at(alen-1) && toMatch.at(alen-1)!=lists[1] && toMatch.at(alen-1)!=lists[3] && toMatch.at(alen-1)!=lists[5] && toMatch.at(alen-1)!=lists[6] && toMatch.at(alen-1)!=lists[8] && toMatch.at(alen-1)!=lists[10])

res=0;

else if(withMatch.at(blen-1)!=toMatch.at(alen-3) && toMatch.at(alen-1)==lists[6] && toMatch.at(alen-2)==lists[1])

res=0;

else if(toMatch.at(alen-2)!=lists[1] && (toMatch.at(alen-1)==lists[8] || toMatch.at(alen-1)==lists[10]) && toMatch.at(alen-3)>='a' && toMatch.at(alen-3)<='z' && toMatch.at(alen-3)!=withMatch.at(blen-1))

res=0;

break;

}

//cout<<res<<endl;

}

return res;

}

int main()

{

int n;

cin>>n;

RegularExpr ReExS[n+1];

for(int i=1; i<=n; i++)

{

ReExS[i].takeIn();

}

int m;

cin>>m;

PatMatch PatMS[m+1];

for(int i=1; i<=m; i++)

{

PatMS[i].takeIn();

}

for(int i=1; i<=m; i++)

{

string withMatch=PatMS[i].getOut();

int flag=0,k;

for(int j=1; j<=n; j++)

{

string toMatch=ReExS[j].getOut();

flag=op(withMatch,toMatch);

if(flag==1)

{

k=j;

break;

}

}

if(flag==1)

cout<<"YES, "<<k<<endl;

else

cout<<"NO, 0"<<endl;

}

return 0;

}

**Output:**

****