**АНАЛИЗ**

**НА РЕШЕНИЕ НА ЗАДАЧАТА**

**СКОБИ**

Дефиницията е за правилна последователност от скоби /ППС/ е почти както израз, в който има само малки скоби. Обаче тук има вече средни и големи.

Тогава решението ще е следното:

След като между две еднакви скоби **от един и същи вид** трябва да има ППС, това значи, че когато лявата е на върха на стека и дясната е следващата, ще се изтрият и става ППС. Тогава обратното – ако не е ППС, изводът за целия израз е НЕ.

Т.е. вървейки по израза, изтриваме от стека само тези двойки скоби за които лявата е на върха на стека и дясната е текуща. Иначе пъхаме текущата в стека.

Какво ще стане в този пример: { [ ( ) ] ) } ?

В жълто ще се изтрият всички скоби. Лявата голяма ще остане и в стека ще е само е тя. Отиваме на следващата скоба ) и тя не може да се изтрие, защото върха на стека е лява скоба, но от друг вид. Вкарваме в стека последната и той изглежда { ) }. Ако беше правилна последователност от скоби, стекът трябва да е празен по дефиниция. Следователно програмата ще изведе “NO”.

Ето основния код:

void SK(){

stack <char> st;

string s;

int d. i

char p;

cin>>s;

d=s.size(); st.push(s[0]);

for (i=1; i<d; i++){

if (!st.empty()){

p=st.top();

if (Iztr (p, s[i])) st.pop();

else st.push(s[i]);

}

else

st.push(s[i]);

}

if (st.empty()) cout<<"YES";

else cout<<"NO";

cout<<'\n';

}

В булевата функция ***Iztr*** се влиза със скобата на върха на стека и с тази, която идва от цикъла. Функцията връща *true* само когато двете скоби са от един и същи вид и първата е лява, а втората е дясна.

*Автор: д-р Цветана Димитрова*