


[PI056] Well-Balanced Expressions

In this problem, you should only submit a file named **pi056.c** containing a line with `#include "stack.h"` and the implementation of a function **balanced** as described below (a complete program is not required). Check code at the end and explanation on how to compile. 

You may assume that on Mooshak you will have access to the "stack implementation code" as provided in class.

Function to Submit

- `boolean balanced(String s)`


This function receives a string composed only of round and square parentheses, and returns *true* if the expression is well-formed, and *false* otherwise.

For example, "`((()))`" and "`[()()]`" are well-formed expressions, while "`((())`", "`[()[]]`", and "`[()[]]`" are not balanced (some parentheses are not closed or are closed incorrectly).

Input/Output Examples

Call	Result
<code>balanced("((()))")</code>	<code>true</code>
<code>balanced("[()()]")</code>	<code>true</code>
<code>balanced("((())")</code>	<code>false</code>
<code>balanced("[()[]]")</code>	<code>false</code>
<code>balanced("[()[]]")</code>	<code>false</code>

[PI056] Expressões bem balanceadas

Neste problema deverá apenas submeter um ficheiro **pi056.c** contendo uma linha com `#include "stack.h"` e a implementação de uma função **balanced** como a seguir descrito (não é necessário um programa completo). Veja o código no final, bem como como copilar. 

Pode assumir que terá acesso no Mooshak ao "código qye implementa uma pilha" como dadas nas aulas.

Método a submeter

- `boolean balanced(String s)`

Recebe uma string formada apenas por parenteses rectos e curvos, e devolve *true* caso a expressão esteja bem formada e *false* caso contrário.

Por exemplo, "`((()))`" e "`[()()]`" são expressões bem formadas, ao passo que "`((())`", "`[()[]]`" ou "`[()[]]`" não estão balanceadas (faltam parenteses por fechar ou fecham os parenteses errados).

Exemplos de input/output

Chamada	Resultado
<code>balanced("((()))")</code>	<code>true</code>
<code>balanced("[()()]")</code>	<code>true</code>

balanced("(())")	false
balanced("[()[])")	false
balanced("[()[]")	false

Como compilar localmente

Para compilar localmente vai precisar de ter os ficheiros seguintes:

- stack.h e stack.c -- implementação de uma stack. Verificar se o NodeInfo está configurado para o tipo de dados a guardar na stack.
- pi056.c -- o seu código com a função pedida
- tests056.c -- ficheiro com a função main() e onde se trata da leitura e escrita.

Para compilar execute na pasta onde estão os ficheiros: gcc tests056.c stack.c pi056.c que vai gerar um executável a.out

Código para testes, ficheiro: tests056.c

```
// PI056 Testing
//

#include "stack.h"

#define MAXCHARS 1000

bool balanced(char *);

int main() {
    int n;
    scanf("%d",&n); // number of expressions
    getchar();
    const char *v[2]={"false","true"};

    for (int i=0; i<n; i++) {
        char *expr= (char *) malloc(MAXCHARS);
        fgets(expr,MAXCHARS, stdin); // read expression
        expr[strcspn(expr, "\n")] = '\0'; // replace \n by \0
        printf("%s: %s\n", expr, v[(int)balanced(expr)]);
        free(expr);
    }
}
```