CONCURSO PB

Preliminares

Este é um conjunto de quatro novos exercícios sobre cadeias de caracteres e devem ser resolvidos tratando as cadeias como arrays de caracteres (com o terminador **'\0'**), sem usar as funções de biblioteca que constam na *header file* **<string.h>.**

Devem programar todas as funções destes exercícios recorrendo a um novo estilo da função **main()**. Para poder manter todas as funções de teste operacionais, use o seguinte esquema:

```
int main(int argc, char **argv)


{
        unit_tests();
        int x = 'A';
        if (argc > 1)
                x = *argv[1];
        if (x == 'A')
                test_ints_greater_than();
        else if (x == 'B')
                test_ints_less_than();
        else if (x == 'C')
                test_decimal_weights();
        // ...
        else if (x == 'U')
                printf("All unit tests PASSED.\n");
        else
                printf("%s: Invalid option.\n", argv[1]);
        return 0;
}
```

Desta forma, quando, por exemplo, quiser testar a função **decimal_weights** deverá dar o comando **./a.out** C ou **a.exe C** (consoante o sistema operativo). Dizemos que aquele C, ou outra letra usada, conforme o caso, é a opção.

Tipicamente, à medida que acrescentamos uma função de teste, acrescentamos também um **else if** à instrução **if else** em cascata, com um novo valor para a opção. Reservamos a opção **U** para correr apenas os testes unitários. O ramo **else** final é tomado quando a opção indicada na linha de comando não for uma das previstas na função **main**.

Deverá submeter no **Concurso PI_2223_PB no Mooshak**.

Programa A – Alfa-2

Transcrito da Wikipedia:

Os códigos ISO 3166-1 alfa-2 são os códigos de países de duas-letras definido na norma ISO 3166-1, que constitui parte da norma ISO 3166 publicada pela Organi- zação Internacional para Normalização (ISO), para representar países, territórios de- pendentes e zonas especiais de interesse geográfico.

Todos usamos esses códigos diariamente, e conhecemos alguns de cor: "pt" para Portugal, "es", para Espanha, "fr" para França, etc.

Frequentemente, o código são as duas primeiras letras do nome do país, como no caso de França, ou a primeira letra e outra letra mais à frente, como no caso de Portugal. Mas, há casos em que a primeira letra do código não é o primeiro nome do país, por exemplo "lk", para o Sri Lanka.

Perante isto, queremos um programa que dado um código de duas letras e um nome de país, verifique se as duas letras do código surgem no nome do país, pela ordem respetiva.

Tarefa

Escreva um programa que leia da consola um código de duas letras e depois uma sequência de nomes de países e verifique para cada nome se as duas letras do código surgem no nome pela ordem por que surgem no código.

Input

A primeira linha do input contém uma cadeia de caracteres com duas letras, re- presentando o código do país. Seguem-se linhas em número indeterminado, cada uma com o nome de um país.

Output

Para cada nome de país o programa escreve "YES" (sem as aspas) quando sim e "NO" quando não.

Restrições

Todas as letras usadas nos casos de teste são letras minúsculas do bloco Basic La- tin do Unicode. (Não há letras com sinais diacríticos). No caso de países com no- mes formados por mais de uma palavra, as sucessivas palavras são separadas por um caractere de sublinhado, por exemplo "cabo_verde".

Exemplo:

Input

pt portugal

polonia

porto_rico

pitcairn

chipre

alemanha

paquistao

sao_tome_e_principe

beparquistao

Output

YES

NO

YES

YES

NO

NO

YES

NO

YES

Sugestão

Comece por programar a função **str_find** para cadeias de caracteres, que dada uma cadeia de caracteres S e um caractere X, calcula o índice da primeira ocor- rência de X em S ou -1 se não ocorrer.

Testes unitários

Use as seguintes funções de teste unitário, acrescentando-lhes outros casos, se quiser.

Para a função **str_find**:

```
void unit_test_str_find(void)
{
        assert(str_find("abcdedfghi", 'a') == 0);
        assert(str_find("abcdedfghi", 'd') == 3);
        assert(str_find("abcdedfghi", 'g') == 7);
        assert(str_find("abcdedfghi", 'i') == 9);
        assert(str_find("abcdedfghi", 'p') == -1);
```

```
        assert(str_find("", 'z') == -1);
}
```

Para a função que resolve o problema, admitindo que se chama alpha2:

```
void unit_test_alpha2(void)
{
        assert(alpha2("pt", "portugal"));
        assert(alpha2("fr", "franca"));
        assert(alpha2("lk", "sri lanka"));
        assert(!alpha2("de", "alemanha"));
        assert(!alpha2("zb", "brazil"));
}
```

Submeta no problema A.

Programa B – Alfa-3

Os códigos alfa-3 são como os alfa-2, mas com 3 letras. Por exemplo, “prt” para Portugal, “usa” para os Estados Unidos da América, “ago” para Angola, “bra” para Brasil.

Tarefa

Análoga à anterior, para códigos de três caracteres.

Exemplo:

Input

prt portugal

polonia

porto_rico

pitcairn

chipre

alemanha

paquistao

sao_tome_e_principe

beparquistao

Output

YES

NO

YES

NO

NO

NO

NO

NO

YES

Submeta no Programa B.

Programa C – Subsequências

Os dois problemas anteriores são casos particulares do problema das subsequências, neste caso subsequências de caracteres. Dizemos que uma cadeia **S** é uma subsequência da cadeia **T** se todos os caracteres de **S** estiverem presentes em **T**, pela mesma ordem, considerando as repetições. Por exemplo, "ncsd", é uma subsequência de "necessidade" e "brgdo" é uma subsequência de "obrigado"; "aab" não é uma subsequência de "ab".

Tarefa

Escreva um programa que, dada uma sequência de pares de cadeias, indique para cada par, se a primeira cadeia é uma subsequência da segunda.

Input

Cada linha do input tem duas cadeias de caracteres. O número de linhas é indeterminado.

Output

Para cada linha de input o programa escreverá uma linha de output, com a mensagem "YES", no caso de a linha de input corresponder a uma situação em que a primeira cadeia é subsequência da segunda, e "NO", no caso contrário.

Exemplo

Input

ncsd necessidade

ptlgr portalegre

lsb lisboa

arg algarve

tlf telefone

telefonia telefone

abcd abcde

abcd abc

aa a

aa aa

aab aba

aab aaaaaabb

bb aaabaaabaaa

ababab aaabaaabaaa

Output

YES

YES

YES

NO

YES

NO

YES

NO

NO

YES

NO

YES

YES

NO

Testes unitários

Use as seguintes funções de teste unitário, acrescentando-lhes outros casos, se quiser.

```
void unit_test_is_subsequence(void)
{
        assert(is_subsequence("prt", "portugal"));
        assert(is_subsequence("fra", "franca"));
        assert(is_subsequence("nka", "sri lanka"));
        assert(!is_subsequence("deu", "alemanha"));
        assert(!is_subsequence("zbl", "brazil"));
        assert(is_subsequence("aefj", "abcdefghij"));
        assert(is_subsequence("", "abcdefghij"));
        assert(is_subsequence("a", "abcdefghij"));
        assert(is_subsequence("j", "abcdefghij"));
        assert(is_subsequence("abcdefghij","abcdefghij"));
        assert(!is_subsequence("abcdefghijk", "abcdefghij"));
```

}

Submeta no Programa C.

Programa D – Telegramas

Antigamente, quando era preciso enviar mensagens muito urgentes, usavam-se telegramas. Os telegramas eram taxados por caractere. Por isso, era habitual as pessoas escreverem os telegramas sem pontuação ou espaços. Assim, ficava mais barato.

Tarefa

Escreva um programa que dada uma cadeia representando um telegrama, onde os espaços são representados por caracteres de sublinhado, construa outra cadeira com o mesmo conteúdo, mas eliminado todos os sublinhados.

Input

Cada linha do input contem uma única cadeia de caracteres, sem espaços e com zero ou mais caracteres de sublinhado, representando o texto de um telegrama.

Output

Para cada linha do input, haverá uma linha de output, contendo o texto do tele- grama, sem os sublinhados.

Exemplo:

Input

manda_mais_dinheiro

atacamos_ao_amanhecer

chegarei_a_tempo____de_____jantar

_atencao ao mau_____tempo confinamento_obrigatorio_a_partir_de_dezasseis_de_janeiro

Output

mandamaisdinheiro

atacamosaoamanhecer

chegareiatempodejantar

atencaoaomautempo

confinamentoobrigatorioapartirdedezasseisdejaneiro

Submeta no Programa D.