Prova-03-sub

Prof. Msc. Elias Batista Ferreira
Prof. Dr. Gustavo Teodoro Laureano
Profa. Dra. Luciana Berretta
Prof. Dr. Thierson Rosa Couto

Sumário

1 Elementos únicos da Matriz (+++)

 (+++)

Dada uma matriz $nl \times nc$, encontre todos os elementos que não se repetem. Considere a dimensão máxima da matriz igual a 10×10.

Entrada

O programa deve ler a quantidade de linhas (nl) e colunas (nc) da matriz e, em seguida, os $nl * nc$ elementos da matriz.

Saída

Caso os números nl ou nc estejam fora do intervalo, o programa imprime a mensagem "dimensao invalida"e encerra. Caso não exista nenhum elemento único, o programa deve imprimir a mensagem "sem elementos unicos". Caso contrário, o programa imprime uma linha contendo os elementos que não se repetem na matriz separados por vírgula e obedecendo a sequencia em que aparecem na matriz (da esquerda para direita e de cima para baixo).

Exemplo

Entrada	Saída
3 3 1 9 3 2 5 4 3 2 1	9,5,4

Entrada	Saída
3 2 2 3 1 2 3 1	sem elementos unicos

Entrada	Saída
0 8	dimensao invalida

2 Separador de palavras (+++)

 (+++)

Uma das etapas mais frequentes de um algoritmo de processamento de texto é a separação do texto em palavras. Essa tarefa pode ser uma tafera difícil uma vez que a variabilidade com que os textos são escritos é muito grande. Por exemplo, o texto "Nossa!!! Que chuva.", possui somente 3 palavras mas pode acontecer com diversas configurações de pontuação. O objetivo desse exercício é que você desenvolva a primeira etapa de processamento de texto que é a separação de um texto, no formato de *string*, em palavras dado um conjunto de caracteres que são considerados como separadores. As palavras serão armazenadas em uma matriz de caracteres, de modo que cada linha seja uma string. Como restrições do problema, considera que cada texto tem no máximo 200 palavras e cada palavra no máximo 64 caracteres. Neste problema também não há a presença de caracteres acentuados, no entanto, os caracteres de pontuação são livres para ocorrer.

Para dar mais flexibilidade à solução desse problema, você deverá implementar uma função que receba a string original, uma matriz de caracteres e uma string contendo a lista de caracteres separadores. Considere as macros `MAX_WORDS` e `MAX_WORD_LEN` as definições dos limites máximos para a declaração da memória do programa. A função deve seguir o seguinte protótipo:

```
1
2  #define MAX_WORDS 200
3  #define MAX_WORD_LEN 64+1
4
5  /**
6   * @brief Função de separação de palavras de acordo com a uma lista de separadores.
7   *      Exemplo de chamada da função:
8   * str_split("Ola mundo! 1,23", m, ", .!?");
9   *
10  * O resultado é a separação das strings "Ola", "mundo", "1", "23", cada uma
11  * ocupando uma linha da matriz m, com base nos caracteres de pontuação fornecidos.
12  *
13  * @param str ponteiro para o início da string original
14  * @param m matriz de caracteres, sendo cada linha uma palavra da string original
15  * @param sep string com a lista de caracteres separadores
16  * @return int quantidade de palavras detectadas
17  */
18 int str_split(char * str, char m[][MAX_WORD_LEN], char * sep);
```

Entrada

Seu programa deve ler duas *strings*, a primeira o texto a ser processado e a segunda, a lista de caracteres de separação.

Saída

O programa deve apresentar um conjunto de linhas, cada uma contendo uma palavra do texto original, precedida pela sua quantidade de caracteres entre parênteses. Ao final, o programa deve apresentar a quantidade de palavras que possui exatamente a mesma quantidade de caracteres que a maior palavra encontrada.

Observações

Lembre-se que, para fazer a leitura de espaços, você deve especificar qual o caractere terminador de string no scanf, exemplo: `scanf("%[^\n]",str);`. Tente decompor o problema em problemas menores e implemente funções para cada sub-problema.

Exemplo

Entrada	Saída
Fulando de Tal da Silva. , .;:?!	(7)Fulando (2)de (3)Tal (2)da (5)Silva 1

Entrada	Saída
Nossa!!! Que chuva forte. Voce tem capa de chuva? , .;:?!	(5)Nossa (3)Que (5)chuva (5)forte (4)Voce (3)tem (4)capa (2)de (5)chuva 4

3 *string to int* (+++)

 (+++)

Faça um programa que leia um número inteiro fornecido como uma *string* e o converta para um `long int`. Você deve implementar a função:

```
/**
 * Converte a string str para o valor inteiro correspondente.
 * @param str string contendo um número inteiro
 * @return o número inteiro correspondente
 */
long int string2int( const char * str );
```

Entrada

O programa deve ler uma sequência de *strings* contento um número inteiro, de no máximo 128 caracteres, usando o comando: `scanf("%s", str);`, até atingir o final do arquivo, ou seja, usando o laço: `while( scanf("%s", str) != EOF ) { ... }`.

Saída

A saída é composta por linhas contendo o número inteiro e o seu dobro impressos usando o comando `printf("%ld %ld\n", n, n*n);`, onde *n* é o número convertido.

Observações

Para interromper o programa no Terminal use o comando Ctrl+D.

Exemplo

Entrada	Saída
1 -2 3 -4	1 2 -2 -4 3 6 -4 -8

Entrada	Saída
15	15 30

Entrada	Saída
-1234	-1234 -2468