Practical work nr. 7 – Strings

Subjects

- Strings

Exercises

1. Write a function that given a string prints the string backwards. For example, if given `Banana` it prints `ananaB`. Test it in a program that asks the user for the string.
2. Consider the following tasks regarding string manipulation. Each task should be implemented using a function following the provided declaration.
    A. Calculate the total number of alphabetic characters in a string
    ```
    int count_alpha(const string &s)
    ```
    B. Calculate the number of lowercase and uppercase characters of a string
    ```
    int count_lowers(const string &s)
    int count_uppers(const string &s)
    ```
    C. Indicate (print) whether the two strings are equal (or what is the greater one, i.e., "comes after the other")
    ```
    void equals_or_greater(const string &s1, const string &s2)
    ```
    D. Convert a string to the corresponding one with lowercase characters
    ```
    string to_lower(const string &s)
    ```
    E. Replace in a string all occurrences of multiple spaces followed by a single space
    ```
    string replace(const string &s)
    ```
    F. Capitalize the first letter of each word in the string
    ```
    string capitalize(const string &s)
    ```
    G. Indicate whether a string is a palindrome or not. Consider only alphabetic characters
    ```
    bool is_palindrome(const string &s)
    ```

    Implement a program to test all the functions above, starting from the user two strings to be used on the tests.

3. Write a program that reads a sentence from the keyboard (a string), and calculates some information about the characters that make it up. The program should count and print the number of lowercase characters, the number of uppercase characters, the number of numeric characters, the number of vowels, and the number of consonants. The data output must conform to the following format (replace # by the appropriate information):

```
Phrase analysis:
  phrase -> ##################################
  number of lowercase characters -> ##
  number of uppercase characters -> ##
  number of numeric characters -> ##
  number of vowels -> ##
  number of consonants -> ##
```

4. In Portugal, car license plates can have (on October 2019) only three patterns:

   AA-00-00, 00-00-AA ou 00-AA-00

   Write a function with prototype

   `int matchPattern(char *str, char *pattern);`

   that indicates if a given string matches the given pattern (a non-zero return value means yes, a zero means no). An **`A`** in the patterns matches any uppercase letter in the string, a **`0`** in the patterns matches any digit, and any other characters must match exactly. Calling this function, write a program that checks if a string entered by the user is a valid Portuguese license plate.

5. Implement a program to count words and count the occurrence of a specific keyword. You should only consider exact matches, i.e. if the keyword is "contained" inside another word should not be counted. The program should start getting from the user a keyword and then gets paragraphs (a paragraph finishes with the symbol `'$'`) in an iterative way (the user has to write a paragraph with the word `"End"` to finish the program).

   For each introduced paragraph, the program should count the number of words in the paragraph (consider the most common symbols as separators) and count the number of occurrences of the keyword.