C++ - Szyfr Cezara

Ogólne

Program szyfruje zdania składające się z małych i dużych liter (w tym zdania zawierające polskie znaki) poprzez przesunięcie każdej litery o 13 znaków.

Uwaga: Nigdy nie wolno kopiować kodu z PDF-a, gdyż zawiera on niewidoczne znaki końca linii, twarde odstępy, odmienne cudzysłowy, odmienne apostrofy, odmienne myślniki. To godziny dodatkowej pracy na wykrywanie błędów i ich poprawianie. Kod należy przepisać ze zrozumieniem.

Kod - wersja podstawowa

```
#include <iostream>
using namespace std;

string tekstDoPrzetworzenia;

string zaszyfruj(string przeslanyTekst) {
   string przetworzonyTekst = "";
   for (char &kolejnaLitera : przeslanyTekst) {
      if (isalpha(kolejnaLitera)) {
         char zmienionaLitera;
         if (isupper(kolejnaLitera)) {
            zmienionaLitera = ((kolejnaLitera - 'A' + 13) % 26) + 'A';
         }
         else {
            zmienionaLitera = ((kolejnaLitera - 'a' + 13) % 26) + 'a';
         }
         przetworzonyTekst += zmienionaLitera;
      }
      else {
         przetworzonyTekst += kolejnaLitera;
      }
   }
   return przetworzonyTekst;
}

int main() {
   cout << "Podaj tekst do zaszyfrowania: ";
   getline(cin, tekstDoPrzetworzenia);
   tekstDoPrzetworzenia = zaszyfruj(tekstDoPrzetworzenia);
   cout << "Zaszyfrowany tekst: " << tekstDoPrzetworzenia << ".\n" << endl;
   cout << "\nNaciśnij ENTER, aby zakończyć..." << endl;
   cin.get();
   return 0;
}
```

Kod - wersja rozbudowana (*Windows*)

```
#include <iostream>
#include <windows.h> // Do kolorowania napisów

using namespace std;
string tekstDoPrzetworzenia, tekstDoOdszyfrowania;

string zaszyfrujOdszyfruj(string przeslanyTekst) {
    string przetworzonyTekst = "";
    // Dla każdej kolejnej litery z przesłanego do zaszyfrowania tekstu:
    for (char &kolejnaLitera : przeslanyTekst) {
        if (isalpha(kolejnaLitera)) {
            char zmienionaLitera;
            // Jeśli znak jest alfabetyczny i jest dużą literą
            // (zwróci 256; jeśli nie, zwróci 0):
            if (isupper(kolejnaLitera)) {
                // 1. Uzyskujemy WZGLĘDNĄ pozycję litery w alfabecie: kolejnaLitera-'A'.
                // 2. Następnie przesuwamy literę o 13 miejsc i sprawdzamy, czy nie wyszła poza
                // alfabet (%26).
                // 3. Jeśli wyszła poza zakres (np. o 9 pozycji: 35%26=9), dodajemy do niej kod litery
                // "A", aby uzyskać nową pozycję bezwzględną (9+'A', czyli 9+65=74, czyli 'J').
                // Jeśli nie wyszła poza zakres (np. 'C' - 'A' = 2 + 13 = 15 % 26 = 15 (nie wyszła poza
                // zakres), 15 + 'A', czyli 15 + 65 = 80, czyli litera 'P'.
                // Uwaga: Dzielenie modulo liczby mniejszej przez większą, zawsze daje w wyniku
                // liczbę, którą dzielimy.
                // Zamiana litery na numer: int ('A').
                // Zamiana liczby na literę: char (97).
                zmienionaLitera = ((kolejnaLitera - 'A' + 13) % 26) + 'A';
            }
            else {
                zmienionaLitera = ((kolejnaLitera - 'a' + 13) % 26) + 'a';
            }
            przetworzonyTekst += zmienionaLitera;
        }
        else {
            // Jeśli znak nie był alfabetyczny (np. był cyfrą lub polskim znakiem),
            // zostanie po prostu dodany bez zmian do zaszyfrowanego tekstu:
            przetworzonyTekst += kolejnaLitera;
        }
    }
    return przetworzonyTekst;
}

int main() {
    // ================ KOLOROWANIE NAPISÓW ===============
    // Tworzymy tzw. uchwyt do tego, co będzie pojawiać się na konsoli (do bufora konsoli):
    HANDLE konsola = GetStdHandle(STD_OUTPUT_HANDLE);
```

```
    // Aktywujemy virtualny terminal i to, co będzie się na nim pojawiać:
    #ifndef ENABLE_VIRTUAL_TERMINAL_PROCESSING // Jeśli nie jest zdefiniowany, to:
    #define ENABLE_VIRTUAL_TERMINAL_PROCESSING 0x0004
    #endif
    // Aby powyższe działało, musimy aktywować i to, co poniżej:
    #ifndef ENABLE_PROCESSED_OUTPUT // Jeśli nie jest zdefiniowany, to:
    #define ENABLE_PROCESSED_OUTPUT 0x0001
    #endif

    // Wartość trybu (intup lub output). Słowo "dw" to skrót od "Display Window",
    // jest to jednak nazwa zmiennej, i może być inna:
    DWORD dwMode = 0;
    dwMode |= ENABLE_PROCESSED_OUTPUT | ENABLE_VIRTUAL_TERMINAL_PROCESSING;
    SetConsoleMode(konsola, dwMode);
    // ============== KOLOROWANIE NAPISÓW - KONIEC ==============

    cout << "\n\033[1;34;40m=== SZYFR CEZARA ===\033[0m\n" << endl;
    cout << "Podaj tekst do zaszyfrowania: ";
    getline(cin, tekstDoPrzetworzenia);
    tekstDoPrzetworzenia = zaszyfrujOdszyfruj(tekstDoPrzetworzenia);
    cout << "Zaszyfrowany tekst: \033[1;33;40m" << tekstDoPrzetworzenia << "\033[0m.\n"
    << endl;
    cout << "Naciśnij ENTER, aby odszyfrować powyższy tekst\nlub wprowadź swój własny: ";
    getline(cin, tekstDoOdszyfrowania);
    if (tekstDoOdszyfrowania == "") {
        tekstDoOdszyfrowania = tekstDoPrzetworzenia;
    }

    cout << "\nOdszyfrowany tekst: \033[1;33;40m" << tekstDoOdszyfrowania << "\033[0m
    --> \033[1;33;40m" << zaszyfrujOdszyfruj(tekstDoOdszyfrowania) << "\033[0m." <<
    endl;
    cout << "\n\033[1;30;40mNaciśnij ENTER, aby zakończyć...\033[0m" << endl;
    cin.get();
    return 0;
}
```

Kod - wersja rozbudowana (*Linux*)

```
#include <iostream>
using namespace std;
string tekstDoPrzetworzenia, tekstDoOdszyfrowania;

string zaszyfrujOdszyfruj(string przeslanyTekst) {
    string przetworzonyTekst = "";
    // Dla każdej kolejnej litery z przesłanego do zaszyfrowania tekstu:
    for (char &kolejnaLitera : przeslanyTekst) {
        if (isalpha(kolejnaLitera)) {
            char zmienionaLitera;
```

```
            // Jeśli znak jest alfabetyczny i jest dużą literą
            // (zwróci 256; jeśli nie, zwróci 0):
            if (isupper(kolejnaLitera)) {
                // 1. Uzyskujemy WZGLĘDNĄ pozycję litery w alfabecie: kolejnaLitera-'A'.
                // 2. Następnie przesuwamy literę o 13 miejsc i sprawdzamy, czy nie wyszła poza
                // alfabet (%26).
                // 3. Jeśli wyszła poza zakres (np. o 9 pozycji: 35%26=9), dodajemy do niej kod litery
                // "A", aby uzyskać nową pozycję bezwzględną (9+'A', czyli 9+65=74, czyli 'J').
                // Jeśli nie wyszła poza zakres (np. 'C' - 'A' = 2 + 13 = 15 % 26 = 15 (nie wyszła poza
                // zakres), 15 + 'A', czyli 15 + 65 = 80, czyli litera 'P'.
                // Uwaga: Dzielenie modulo liczby mniejszej przez większą, zawsze daje w wyniku
                // liczbę, którą dzielimy.
                // Zamiana litery na numer: int ('A')
                // Zamiana liczby na literę: char (97)
                zmienionaLitera = ((kolejnaLitera - 'A' + 13) % 26) + 'A';
            }
            else {
                zmienionaLitera = ((kolejnaLitera - 'a' + 13) % 26) + 'a';
            }
            przetworzonyTekst += zmienionaLitera;
        }
        else {
            // Jeśli znak nie był alfabetyczny (np. był cyfrą lub polskim znakiem),
            // zostanie po prostu dodany bez zmian do zaszyfrowanego tekstu:
            przetworzonyTekst += kolejnaLitera;
        }
    }
    return przetworzonyTekst;
}



int main() {
    cout << "\n\033[1;34;40m=== SZYFR CEZARA ===\033[0m\n" << endl;
    cout << "Podaj tekst do zaszyfrowania: ";
    getline(cin, tekstDoPrzetworzenia);
    tekstDoPrzetworzenia = zaszyfrujOdszyfruj(tekstDoPrzetworzenia);
    cout << "Zaszyfrowany tekst: \033[1;33;40m" << tekstDoPrzetworzenia << "\033[0m.\n"
    << endl;
    cout << "Naciśnij ENTER, aby odszyfrować powyższy tekst\nlub wprowadź swój własny: ";
    getline(cin, tekstDoOdszyfrowania);
    if (tekstDoOdszyfrowania == "") {
        tekstDoOdszyfrowania = tekstDoPrzetworzenia;
    }

    cout << "\nOdszyfrowany tekst: \033[1;33;40m" << tekstDoOdszyfrowania << "\033[0m
    --> \033[1;33;40m" << zaszyfrujOdszyfruj(tekstDoOdszyfrowania) << "\033[0m." <<
    endl;
```

```
    cout << "\n\033[1;30;40mNaciśnij ENTER, aby zakończyć...\033[0m" << endl;
    cin.get();
    return 0;
}
```

Ostatnia aktualizacja: 3 kwietnia 2024.