Programowanie II R

Zadania – seria 4.

Łańcuchy tekstowe – typ `std::string`.

Zadanie 1. `palindrome` – Palindromy.

Palindromem nazywamy wyrażenie brzmiące tak samo przy czytaniu od lewej strony do prawej, jak i odwrotnie. Przy badaniu, czy dane wyrażenie jest palindromem, nie należy brać pod uwagę wielkości liter ani znaków interpunkcyjnych. Palindromami są na przykład: „Anna", „kajak", „O, ty z Katowic, Iwo? Tak, Zyto!"

Napisz funkcję

```
bool IsPalindrome(const std::string& str)
```

która przyjmuje jako argument referencję łańcucha tekstowego i zwraca wartość `true`, jeśli jest on palindromem, lub `false` w przeciwnym przypadku.

Korzystając z tej funkcji napisz program `palindrome`, który przyjmuje jako argumenty wywołania dowolnie wiele łańcuchów tekstowych i wypisuje na standardowe wyjście te z nich, które są palindromami.

Zadanie 2. `caesar` – Szyfr Cezara.

Szyfr Cezara, nazywany też *szyfrem przesuwającym*, to jedna z najstarszych i zarazem najprostszych technik szyfrowania tekstu.

Szyfrowanie łańcucha tekstowego szyfrem Cezara z przesunięciem $n \in \mathbb{Z}$ polega na zastąpieniu każdej z liter tego łańcucha literą występującą w alfabecie n pozycji za literą oryginalną, gdy $n \geqslant 0$, lub $|n|$ pozycji przed literą oryginalną, gdy $n < 0$. Przyjmujemy przy tym, że litery wielkie i małe tworzą odrębne zbiory (a więc litera wielka zawsze zostanie zastąpiona literą wielką, zaś litera mała – literą małą) oraz że zbiory te są uporządkowane cyklicznie (tzn. przed literą A występuje litera Z, zaś za literą Z – litera A; analogicznie dla liter małych). Znaki, które nie są literami (np. cyfry, znaki specjalne), nie są w żaden sposób zmieniane. Na przykład szyfrując łańcuch tekstowy *Programowanie13w@$C++* szyfrem Cezara z przesunięciem 3 dla 26-literowego alfabetu łacińskiego otrzymamy *Surjudprzdqlh13z@$F++*.

Napisz funkcję

```
std::string Caesar(int n, const std::string& str)
```

która przyjmuje jako argumenty liczbę całkowitą oraz referencję łańcucha tekstowego. Funkcja ta powinna szyfrować przekazany jej jako drugi argument łańcuch za pomocą szyfru Cezara z przesunięciem zadanym liczbą całkowitą przekazaną jako pierwszy argument oraz zwracać zaszyfrowany łańcuch.

Korzystając z tej funkcji napisz program `caesar` szyfrujący i deszyfrujący łańcuchy tekstowe omówioną metodą. Program powinien przyjmować trzy argumenty wywołania. Pierwszy z nich to `encrypt` dla szyfrowania lub `decrypt` dla deszyfrowania, drugi – liczba całkowita określająca przesunięcie szyfru, trzeci zaś – łańcuch tekstowy do zaszyfrowania lub odszyfrowania. Zaszyfrowany lub odszyfrowany łańcuch tekstowy powinien zostać wypisany na standardowe wyjście.

Przyjmij, że litery pojawiające się w łańcuchach tekstowych należą do 26-literowego alfabetu łacińskiego.

Zadanie 3. `todec` – Konwersja liczb na system dziesiętny.

Napisz funkcję

```
int ToDec(int n, const std::string& str)
```

której zadaniem jest konwersja liczby całkowitej zapisanej w dowolnym liczbowym systemie pozycyjnym o podstawie nie większej niż 20 na liczbę w systemie dziesiętnym. Funkcja ta przyjmuje dwa argumenty: `n` jest liczbą całkowitą oznaczającą podstawę systemu liczbowego, w którym zapisana jest konwertowana liczba, zaś `str` – referencją łańcucha tekstowego zawierającego zapis konwertowanej liczby w tym systemie. Wartością zwracaną przez tę funkcję powinna być wartość konwertowanej liczby zapisana w systemie dziesiętnym.

Korzystając z tej funkcji napisz program `todec`, który przyjmuje jako argumenty wywołania liczbę całkowitą oznaczającą podstawę systemu liczbowego oraz łańcuch tekstowy zawierający liczbę zapisaną w tym systemie oraz wypisuje na standardowe wyjście wynik konwersji tej liczby na system dziesiętny.

Opracowanie: Bartłomiej Zglinicki.