Zadanie 1 4-7 marca 2024 r.

kurs języka C++

liczby w notacji rzymskiej

Instytut Informatyki
Uniwersytetu Wrocławskiego Paweł Rzechonek

Prolog

Rzymski system zapisywania liczb wykorzystuje cyfry pochodzenia etruskiego, które Rzymianie przejęli i zmodyfikowali około 500 roku p.n.e. Jest to system addytywny i nadaje się do wygodnego zapisywania liczb, jest jednak niewygodny w prowadzeniu nawet prostych działań arytmetycznych.

W systemie rzymskim używa się 7 liter do zapisu wybranych wartości nominalnych:
I (1), V (5), X (10), L (50), C (100), D (500) i M (1000).
Aby utworzyć liczbę, trzeba zestawić odpowiednie symbole (zaczynając od litery oznaczającej największy nominał a kończąc na literze oznaczającej nominał najmniejszy) sumujące się do zadanej wartości. Na przykład:
187 to CLXXXVII, czyli C+L+X+X+X+V+I+I (100+50+10+10+10+5+1+1 = 187).

Jeżeli składnik liczby którą zapisujemy jest wielokrotnością wartości nominalnej, wtedy zapisywany jest z użyciem kilku następujących po sobie takich samych symboli, z zachowaniem zasady, by nie pisać czterech tych identycznych symboli po sobie. Jedną z zasad, jest umieszczanie oznaczeń I, X i C z lewej strony nominału wyższego, stąd zestawienia takie jak: IV, IX, XL, XC, CD i CM. Poniższa tabela pokazuje, jak zwykle zapisuje się liczby rzymskie:

	tysiące	setki	dziesiątki	jedności
1	M	C	X	I
2	MM	CC	XX	II
3	MMM	CCC	XXX	III
4		CD	XL	IV
5		D	L	V
6		DC	LX	VI
7		DCC	LXX	VII
8		DCCC	LXXX	VIII
9		CM	XC	IX

Liczbę zawierającą kilka cyfr dziesiętnych buduje się przez dodanie odpowiadającej jej cyfry w notacji rzymskiej, od największej do najmniejszej wartości.

Zadanie

Napisz program do przekształcania liczb całkowitych (zapisanych zwykłymi arabskimi cyframi w systemie dziesiętnym) na zapis w systemie rzymskim. Liczby w zapisie arabskim należy dostarczyć do programu poprzez argumenty wywołania. Każdy argumet wywołania programu to napis typu `const char*`, który najpierw należy przekształcić do postaci binarnej z wykorzystaniem funkcji bibliotecznej `stoi()` zadeklarowanej w pliku nagłówkowym `<string>`. Jeśli argumentu nie można poprawnie przekonwertować na liczbę całkowitą albo liczba ta jest spoza zakresu od 1 do 3999, to należy taki argument zignorować.

Liczbę typu `int` należy następnie przekształcić na odpowiadający jej zapis w notacji rzymskiej funkcją:

```
std::string toroman(int x);
```

Funkcja ta ma dla zadanej wartości typu `int` zwrócić rzymski zapis tej wartości jako łańcuch znakowy typu `std::string`. W trakcie konwersji skorzystaj operatorów konkatenacji łańcuchów oraz stablicowanych wartości liczbowych i odpowiadających im symboli rzymskich:

```
const std::vector<pair<int, std::string>> roman = {
     {1000, "M"},
     {900, "CM"}, {500, "D"}, {400, "CD"}, {100, "C"},
     {90, "XC"}, {50, "L"}, {40, "XL"}, {10, "X"},
     {9, "IX"}, {5, "V"}, {4, "IV"}, {1, "I"}
 };
```

Program powinien dla każdej prawidłowo podanej wartości wypisać na standardowym wyjściu `std:cout` jej wartość w postaci rzymskiej (każdą liczbę wypisz w osobnej linii). Wszelkie komentarze, czy informacje o błędnych argumentach posyłaj na standardowe wyjście dla błędów `std::clog`.

Istotne elementy programu

- Prawidłowe posługiwanie się standardowymi strumieniami we/wy.
- Przekształcenie łańcuchów znakowych typu `std::string` na typ `int`.
- Sprawdzanie warunków brzegowych na dane wejściowe.
- Reagowanie na wyjątki.
- Implementacja zachłannego algorytmu do przekształcania liczby binarnej na postać rzymską.
- Posługiwanie się wektorami, łańcuchami i parami elementów.
- Uruchomienie programu w wierszu poleceń.
- Napisanie skryptu do kompilacji i uruchomienia programu.