Seria domowa nr III

28 maja 2014

Zad. 1. Odwracanie wiersza.
Wczytaj z klawiatury cztery linie tekstu (każda linia złożona z kilku słów) i zapisz je w pliku o nazwie podanej jako argument wejścia programu. [*Wskazówka:* można wykorzystać strumień `ostringstream` i komendę `getline`.]

Następnie wczytaj zapisany plik linia po linii. W każdej linii wypisz wyrazy w odwróconym szyku. [*Wskazówka:* przy wykorzystaniu strumienia `istringstream` mogą – choć nie muszą – być pomocne metody `istringstream ::str (string)` i `istringstream::clear()`.]

Zad. 2. Klasa Punkt2D
Napisz klasę Punkt2D, której polami będą współrzędne x i y dwuwymiarowego punktu. Zaprogramuj dwa konstruktory: domyślny (zerujący współrzędne) oraz dwuargumentowy (do przypisania współrzędnych podczas tworzenia obiektu). Napisz metodę, która wypisuje współrzędne na ekran. Ponadto, zaprogramuj operatory + i –, dzięki którym umożliwisz "dodawanie/odejmowanie dwóch punktów" jako dodawanie/odejmowanie ich współrzędnych. Napisz następnie metodę "`obrot`", ktora przyjmie kąt w stopniach i dokona obrotu współrzędnych o ten kąt.

Wskazówka: Zgodnie z macierzą obrotu, współrzędne transformują się następująco:

$$x_{nowe} = x_{stare} \cdot \cos\alpha - y_{stare} \cdot \sin\alpha$$

$$y_{nowe} = x_{stare} \cdot \sin\alpha + y_{stare} \cdot \cos\alpha$$

W funkcji main zadeklaruj punkty: **A** [1,0] , **B**[0,1] , **C** $[1/\sqrt{2}, 1/\sqrt{2}]$. Następnie utwórz punkt **D** będący sumą **A**+**B** oraz punkt **E** będący różnicą **A**–**B**. Następnie poproś użytkownika o podanie kąta w stopniach – i obróć każdy z punktów o ten kąt. Na koniec wypisz współrzędne obróconych punktów.

Zaawansowane: zaprogramuj `operator+=`, wykonujący obrót, tak aby użytkownik klasy Punkt2D mógł np. wykonać obrót punktu **P** o 90° w ten sposób: "`P+=90 ;`" *Wskazówka:* `operator+=` musi zwrócić rodzimy obiekt i robimy to komendą "`return *this`" .

Zad. 3. Klasa Zespolona

Napisz klasę Zespolona, której polami będą część rzeczywista Re i część urojona Im liczby zespolonej. Zaprogramuj dwa konstruktory: domyślny (zerujący skladowe) oraz dwuargumentowy (do przypisania wartości części rzeczywistej i urojonej liczby zespolonej podczas tworzenia obiektu). Zaprogramuj operator « w taki sposób, aby w wyniku jeo działania na obiekt typu Zespolona wypisywał na ekran wartość części rzeczywistej i urojonej liczby zespolonej w poniższej formie `Re + i Im`. Zaprogramuj operator +, dzięki któremu umożliwisz dodawanie liczb zespolonych, jako dodawanie ich cześci rzeczywistych i urojonych. Utwórz metodę "`modul`, która oblicza moduł liczby zespolonej.

W funkcji main zadeklaruj 2 liczby zespolone za pomocą operatora przypisania: **Z1** [2,3] , **Z1**[5,1]. Następnie utwórz liczbę zespoloną **Z** będącą sumą **Z1**+**Z2** i oblicz moduł liczby Z. Na koniec wypisz na ekran liczbę Z za pomocą operatora « oraz wartość modułu liczby Z.

Zad. 4. Klasa vector

Napisz program wykorzystujący obiekty klasy `<vector>`. Utwórz funkcję, która jako argument przyjmie `vector<vector<double> >`, wypełni go dwoma wektorami (`vector<double>`) zawierającymi po 100 liczb pseudolosowych z zakresu od 1 do 10, następnie zwróci `vector<vector<double> >`. Następnie napisz funkcję, która dopasuje prostą do wygenerowanych liczb pseudolosowych (pierwszy wektor potraktuj jako x, a drugi jako y) metodą najmniejszych kwadratów. Otrzymane współczynniki dopasowania wykorzystaj do wypełnienia jeszcze jednego wektora, który będzie zawierał wartości $y_nowe = a * x + b$ z Twojego dopasowania. Trzy kolumny danych zapisz do pliku tekstowego "`dane.txt`ź nagłówkami odpowiednich kolumn: `X`, `Y` i `Y_MNK`.

Wskazówka:

$$S_x = \sum_{x=1}^{N} x_i$$

$$S_y = \sum_{x=1}^{N} y_i$$

$$S_{xx} = \sum_{x=1}^{N} x_i^2$$

$$S_{xy} = \sum_{x=1}^{N} x_i y_i$$

$$S_{yy} = \sum_{x=1}^{N} y_i^2$$

$$\Delta = N S_{xx} - S_x^2$$

$$a = \frac{N S_{xy} - S_x S_y}{\Delta}$$

$$b = \frac{S_{xx} S_y - S_x S_{xy}}{\Delta}$$

Zad. 5. Klasa Macierz2D

Napisz klasę Macierz2D z czterema polami do reprezentowania czterech elementów w dwuwymiarowej macierzy kwadratowej (np. a, b, c, d). Zaimplementuj trzy konstruktory: domyślny nadający wszystkim polom wartość 0, dwuargumentowy, nadający wartości elementom na diagonali, oraz czteroargumentowy, wypełniający wszystkie pola w klasie. Przeładuj operatory +, - i * do dodawania, odejmowania i mnożenia macierzy. Zaimplementuj metodę 'wyznacznik()' zwracającą wartość wyznacznika macierzy. Spróbuj także przeładować operator `<<`, tak aby wykonanie `cout << macierz` (gdzie macierz jest przykładowym obiektem Macierz2D) powodowało wypisanie wartości wszystkich elementów macierzy.

Wskazówka: Aby przeładować operator `<<` powinno się skorzystać z funkcji zaprzyjaźnionych:
`http://pl.wikibooks.org/wiki/C++/Przeci%C4%85%C5%BCanie_operator%C3%B3w`

Zad. 6. Silne hasło

Utwórz plik zawierający dowolny ciąg liter. Napisz program, który zamieni podany ciąg liter na hasło do internetowego konta bankowego. Przykładowo możesz zamienić co drugą małą literę na wielką, wybrane litery zamienić na ich odpowiedniki ze slangu leet (`http://pl.wikipedia.org/wiki/Leet_speak`) a na koniec hasła dodać swoją ulubioną liczbę. Skorzystaj z klasy string i zaimplementowanych dla niej metod.