laboratorium: zadanie 5 *termin: 25 marca – 5 kwietnia 2013 r.*

KURS JĘZYKA C++

KOLEJKA DAT

Instytut Informatyki Uniwersytetu Wrocławskiego *Paweł Rzechonek*

Ważne.
W tym zadaniu wykorzystaj klasę `Data`, którą zdefiniowałeś(-aś) w zadaniu 3.

Zadanie.
Zdefiniuj klasę `DataGodz` do przechowywania daty i godziny, dziedzicząc po klasie `Data` z zadania 3. W klasie tej zdefiniuj operatory relacji (`==`, `<`, itp.) do porównywania dat z uwzględnieniem godzin. Dalej, zdefiniuj typ `Para` dla pary reprezentującej datę z godziną i opis zdarzenia `pair<DataGodz,string>`.

Następnie zdefiniuj klasę `Lista` reprezentującą jednokierunkową listę z obiektami typu `Para`. Zakładamy, że elementy listy będą wirtualnie ponumerowane kolejnymi liczbami naturalnymi (zaczynając od 0). Twoja lista powinna obsługiwać takie operacje jak wstawienie nowego elementu na zadaną pozycję do listy (pozycja 0 oznacza wstawienie na początek), usunięcie wskazanego elementu z listy, pobranie referencji do wskazanego elementu za pomoca operatora indeksowania oraz podanie rozmiaru listy. Klasa `Lista` ma być tylko opakowaniem homogenicznej struktury złożonej z węzłów (klasa `Wezel` zdefiniowana wewnątrz listy). Pamiętaj, aby węzeł był tak bogato oprogramowany, aby można było na nim swobodnie wykonywać wszytkie wymienione operacje listowe.

Po liście zdefiniuj kolejkę — klasę `Kolejka` dziedziczącą niepublicznie po klasie `Lista`. W kolejce ma być stale zachowany chronologiczny porządek przechowywanych w niej par. W klasie tej zaprogramuj operację wstawiania nowego elementu do kolejki (z zachowaniem porządku chronologicznego), wyciągania elementu z początku kolejki, podawania długości kolejki, itp.

Pamiętaj, aby w liście i kolejce zdefiniować konstruktor domyślny, konstuktor kopiujący, destruktor, przypisanie kopiujące oraz zaprzyjaźnione operatory czytania z i pisania do strumienia.

Na koniec napisz program, który bardzo rzetelnie przetestuje całą funkcjonalność zaprogramowaną w klasach `Lista` i `Kolejka`. Dane do programu wczytaj ze standardowego wejścia `cin` za pomocą operatora strumieniowego `>>`. Wyniki wypisz na standardowym wyjściu `cout` za pomocą operatora strumieniowego `<<`. Ewentualne komunikaty o błędach wypisz na standardowym wyjściu dla błędów `cerr`.

Uwaga 1.
W funkcjach składowych listy i kolejki zgłaszaj błędy za pomocą instrukcji `throw`.

Uwaga 2.
Podziel program na pliki nagłówkowe i źródłowe. Definicję listy i kolejki umieść w pliku `struktury.hpp` a definicje funkcji i operatorów w pliku `struktury.cpp`. Program testujący napisz w pliku `main.cpp`.