Zadanie 9 *23–26 maja 2017 r.*

JĘZYK PROGRAMOWANIA C++

BEZPIECZNE PLIKI TEKSTOWE

Instytut Informatyki Uniwersytetu Wrocławskiego *Paweł Rzechonek*

Dane przetwarzane przez komputery mogą być cyfrową reprezentacją dowolnych informacji: tekstowych, obrazowych, dźwięków, filmów itp. Dane są pamiętane w postaci *plików* na nośnikach elektronicznych. Plik to uporządkowana sekwencja danych o skończonej długości. Plik jest podstawową jednostką danych w każdym systemie plików.

Zadanie.

Manipulatory dla strumieni czytających. Zdefiniuj własny manipulator bezparametrowy `clearline` dla strumienia wejściowego, który będzie usuwał wszystkie znaki, aż do napotkania znaku przejścia do nowej linii (ten znak także należy usunąć ze strumienia) lub znaku końca pliku. Zdefiniuj również manipulator z parametrem `ignore (int x)`, którego zadaniem będzie pominięcie `x` znaków ze strumienia wejściowego, chyba że wcześniej zostanie wyjęty znak przejścia do nowej linii lub stumień się skończy.

Manipulatory dla strumieni piszących. Dla strumienia wyjściowego zdefiniuj bezparametrowe manipulatory `comma` wypisujący przecinek z odstępem `", "` oraz `colon` wypisujący dwukropek z odstępem `": "`. Zdefiniuj także manipulator z parametrem `index (int x, int w)`, który wypisze liczbę `x` w nawiasach kwadratowych i na liczbę tą przeznaczy co najmniej `w` pozycji (dosuń liczbę do prawego nawiasu kwadratowego).

Testowanie manipulatorów. Napisz program testujący zdefiniowane przez ciebie manipulatory — program powinien odczytać wszystkie linie danych zapamiętując je w kontenerze `vector<>`. Następnie posortuj odczytane linie leksykograficznie i wypisz je wraz z pierwotnymi numerami linii. Numer linii umieść na początku wiersza w nawiasach kwadratowych (numeracja wszystkich wierszy ma się zaczynać od 1 i ma zajmować tyle samo przestrzeni ale wartość szerokości ma być minimalna).

Wrappery na strumienie. W oparciu o technikę *zdobywania zasobów poprzez inicjalizację* zaimplementuj bezpieczne klasy opakowujące pliki: `PlikWejsciowy` dla plików tekstowych do czytania (opakowanie dla obiektu `ifstream`) oraz `PlikWyjsciowy` dla plików tekstowych do pisania (opakowanie dla obiektu `ofstream`). Plik należy otworzyć w konstruktorze (jeśli okaże się to nieżmożliwe zgłoś wyjątek) a zamknąć w destruktorze. Zadbaj o to, by ustawienie flagi

błędu `ios_base::badbit` lub `ios_base::failbit` powodowało automatyczne zgłoszenie wyjątku `ios_base::failure`.

Klasa `PlikWejsciowy` powinna umieć odczytać linię tekstu (wyciągając ze strumienia znak przejścia do nowej linii bez umieszczania go w wynikowym łańcuchu) zwracając obiekt typu `string`, pojedynczy znak `char` a także liczby całkowitą `int` i rzeczywistą `double` z pominięciem początkowych białych znaków.

Klasa `PlikWyjsciowy` powinna umieć zapisać obiekt typu `string`, pojedynczy znak `char`, liczbę całkowitą `int` i rzeczywistą `double` a także znak przejścia do nowej linii. W klasie `PlikWejsciowy` zdefiniuj zaprzyjaźnione operatory do czytania `operator>>` a w klasie `PlikWyjsciowy` operatory do pisania `operator<<`.

Testowanie wrapperów. Na koniec napisz program, który przetestuje zachowanie się obiektów obu klas (również w sytuacjach wyjątkowych) — program powinien odczytać z pliku ciąg liczb rzeczywistych i zapisać go w odwrotnej kolejności do pliku tekstowego o tej samej nazwie. Z początku nie wiadomo ile liczb jest zapisanych w pliku z danymi, dlatego posłuż się kontenerem `vector<>`.

Uzupełnienie.

Definicje klas opakowujących pliki umieść w przestrzeni nazw `strumienie`.

Uwaga.

Podziel program na pliki nagłówkowe i źródłowe.

Elementy w programie, na które należy zwrócić szczególną uwagę.

- Definicja manipulatorów bezparametrowych.
- Definicja manipulatorów z parametrami.
- Testowanie manipulatorów.
- Wrappery dla plików realizujące ideę zdobywania zasobów poprzez inicjalizację.
- Obsługa błędów w strumianiach za pomocą wyjątków.
- Testowanie wrapperów.
- Podział programu na pliki nagłówkowe i źródłowe.