Politechnika Śląska w Gliwicach

Wydział Automatyki, Elektroniki i Informatyki

Podstawy Programowania Komputerów

Loty

autor	Paweł Rother
prowadzący	mgr inż. Marek Kokot
rok akademicki	2015/2016
kierunek	teleinformatyka
rodzaj studiów	SSI
semestr	1
termin laboratorium	poniedziałek 12:00 – 13:30
grupa	1
data oddania sprawozdania	27.01.2016 r.

1 Treść zadania

Korzystając z dynamicznych struktur danych napisać program, który przygotowuje listy pasażerów.

Pasażerowie mogą rezerwować bilety na różne loty w różnych biurach i przez Internet. Wszystkie rezerwacje są zapisywane w biurze centralnym. Mają one następującą postać:

<symbol lotu> <lotnisko startowe> <data lotu> <nazwisko pasażera> <nr miejsca>

Przykładowy plik z rezerwacjami:

KR54R Katowice 2011-12-13 Jaworek 33
TY34O London 2012-02-03 Hastings 2
TY34O London 2012-02-03 Poirot 23
KR54R Katowice 2011-12-13 Matianek 02
TY34O London 2012-02-03 Holmes 11
KR54R Katowice 2011-12-13 Lopez 12
TY34O London 2012-02-03 Lemon 43
KR54R Katowice 2011-12-13 Chavez 43

Na podstawie pliku z rezerwacjami należy stworzyć pliki z listą pasażerów dla każdego lotu. Każda lista jest tworzona w odrębnym pliku. Nazwą pliku jest symbol lotu. W pliku umieszczona jest nazwa lotniska i data. W kolejnych liniach umieszczone są numery siedzeń i nazwiska pasażerów, posortowane wg numerów.

Dla powyższego pliku zostaną utworzone dla pliki: KR54R.txt i TY34O.txt.

Plik KR54R.txt:

symbol lotu: KR54R

lotnisko: Katowice

data lotu: 2011-12-13

lista pasażerow:
02 Matianek
12 Lopez
33 Jaworek
43 Chavez
liczba rezerwacji: 4

Program uruchamiany jest z linii poleceń z wykorzystaniem następującego przełącznika:

-i plik wejściowy z rezerwacjami

2 Cele projektu

Celem projektu jest przećwiczenie implementacji i korzystania z dynamicznych struktur danych (np. listy, drzewa) i zarządzania pamięcią w programie. Warunkiem sine qua non jest użycie w programie tych struktur. Użycie bibliotecznych kontenerów (np. vector, list) nie spełnia tego warunku.

3 Analiza zadania

Zagadnienie przedstawia problem wczytywanie z plików wejściowych, odpowiedniego sortowania i zapisu odpowiednich struktur danych we wskazany sposób do odpowiednich plików wyjściowych.

3.1 Struktury danych

Wybraną przeze mnie strukturą danych jest lista. Odpowiada ona potrzebom programu. Lista jest jednokierunkowa. Każdy element listy przechowuje informacje o danym locie tj. jego symbolu, lotnisku oraz dacie. Ponadto każdy element listy (lot) ma podwieszone drzewo binarne, w którym węzłami są poszczególne osoby, które złożyły rezerwację na ten lot. Drzewo to jest drzewem przeszukiwań binarnych w którym kluczem jest numer miejsca pasażera, także wypisywanie osób od najmniejszego numeru miejsca do największego przebiega prosto i szybko.

4 Specyfikacja zewnętrzna

Program jest uruchamiany z linii poleceń. Aby program mógł wczytać listę rezerwacji z pliku zewnętrznego należy go uruchomić z odpowiednim parametrem w następujący sposób:

program.exe *[-i]* *plik_źródłowy.txt*

gdzie:

program.exe – jest to nazwa programu z rozszerzeniem *.exe

plik_źródłowy.txt – jest to ścieżka pliku, w którym jest przechowana lista rezerwacji lotów z rozszerzeniem *.txt

4.1 Format danych wejściowych

Program wczytuje dane z jednego wejściowego pliku tekstowego *.txt, określonego parametrem [-i]. W pliku dane powinny byś zawarte w następującym formacie:

<symbol lotu> <lotnisko startowe> <data lotu> <nazwisko pasażera> <nr miejsca>

Przy czym symbol lotu jest pojedynczym wyrazem składającym się z ciągu wielkich liter i cyfr, lotnisko startowe to nazwa miejscowości, a data lotu ma postać rrrr-mm-dd.

Program jest odporny na niewielkie niezgodności formatu danych wejściowych, takie jak puste linie, dodatkowe spacje, białe znaki itp. Błędnie wprowadzone dane, tj. brak pliku, błędny format danych, pusty plik, niepoprawne parametry uruchomienia programu, nie powoduje zatrzymania programu przez system operacyjny. Program wyświetla wtedy stosowny komunikat (por. pkt 4 lit. c).

4.2 Komunikaty

- „Program służy do porządkowania rezerwacji lotów. Po skończonej pracy w katalogu źródłowym, w którym znajduje się program, zostaną utworzone pliki tekstowe z listami osób, które zarezerwowały bilety na dany lot.” – zdefiniowany jako **wprowadzenie**

- „Plik nie został otwarty poprawnie. Program się nie wykonał.” – zdefiniowany jako **blad_plik** – w wypadku gdy nie odnaleziono pliku wskazanego parametrem [-i]

- „Nieprawidłowy parametr lub nieprawidłowa ścieżka pliku źródłowego. Program się nie wykonał.” – zdefiniowany jako **blad** – w wypadku, gdy użytkownik wywołał program niepoprawnym poleceniem.

- „Plik jest pusty. Program się nie wykonał” – zdefiniowany jako **pusty_plik** – w wypadku gdy plik źródłowy nie zawiera żadnych danych.

- „Aby poprawnie uruchomić program należy w linii poleceń napisać następująca komendę:

  program.exe -i plik_zrodlowy.txt

  gdzie:
  program.exe jest to ścieżka programu z rozszerzeniem *.exe

  plik_zrodlowy.txt jest to ścieżka pliku, w którym jest przechowana lista rezerwacji lotów z rozszerzeniem *.txt” – zdefiniowany jako **pomoc**

5 Specyfikacja wewnętrzna

5.1 Zmienne

Nazwa	Typ	Opis
nazwa_pliku_wej	string	zmienna przechowująca nazwę pliku wejściowego, podaną przez parametr [-i]
wsk_plik_wej	ifstream*	wskaźnik na plik wejściowy
glowa	lot*	wskaźnik na głowę listy lotów
lot::symbol	string	przechowuje symbol lotu
lot::lotnisko	string	przechowuje lotnisko startowe
lot::data	string	przechowuje datę lotu
lot::licznik	int	służy do zliczenia liczby pasażerów danego lotu
lot::nastepny	lot*	wskaźnik na następny lot w liście
lot::korzen_drzewa	osoba*	wskaźnik na korzeń drzewa podwieszonego do lotu, przechowującego pasażerów
osoba::nazwisko	string	przechowuje nazwisko pasażera
osoba::numer_miejsca	int	przechowuje numer miejsca w samolocie
osoba::lewy	osoba*	wskaźnik na lewego syna w drzewie
osoba::prawy	osoba*	wskaźnik na prawego syna w drzewie

5.2 Funkcje

---

string czytaj_plik **(ifstream** *wsk_plik_wej**)**

Funkcja pobiera z pliku wejściowego dane do zmiennej linia i zwraca zwartość tej zmiennej.

Parametry funkcji:
---
ifstream *wsk_plik_wej - Wskaźnik na plik wejściowy.

void nowy_lot (**lot****glowa, **ifstream** *wsk_plik_wej)

Funkcja tworzy nowy element listy uzupełniając strukturę danymi pobranymi za pomocą funkcji *czytaj_plik* oraz dodaje go na koniec listy lub, gdy taki lot już jest na liście, wywołuje funkcje *nowa_osoba* w celu dodania nowej osoby do danego lotu.

Parametry funkcji:
---
lot**glowa – Wskaźnik na pierwszy element listy.

ifstream *wsk_plik_wej - Wskaźnik na plik wejściowy.

void nowa_osoba(**osoba****rodzic, **stringstream** &wiersz)

Funkcja wywoływana w funkcji *nowy_lot*. Tworzy nowy węzeł drzewa i wypełnia strukturę danymi zwartymi w zmiennej wiersz

Parametry funkcji:
---
osoba**rodzic – Wskaźnik na rodzica.

stringstream &wiersz – zmienna przechowująca dane do wypełnienia struktury.

void dopisz_do_drzewa (**osoba***&rodzic, **osoba*** nowa)

Funkcja wywoływana w funkcji *nowa_osoba*. Dodaje nowy węzeł drzewa jako liść w zależności od numeru miejsca. Jeżeli numer miejsca nowo dodawanego węzła jest większy niż numer miejsca rodzica to nowy węzeł jest dodawany do prawej strony, a jeżeli mniejszy – do lewej.

Parametry funkcji:
---
osoba*&rodzic – Wskaźnik na rodzica.

osoba*nowa – Wskaźnik na węzeł listy który chcemy dodać.

void wypisz_drzewo (**osoba***rodzic, **ofstream** *wsk_plik_wyj)

Funkcja, która zaczynając od najbardziej na lewo syna (od najmniejszego numeru miejsca do największego) przekazuje do pliku wyjściowego numer miejsca i nazwisko pasażera.

Parametry funkcji:
---
osoba*&rodzic – Wskaźnik na rodzica.

ofstream*wsk_plik_wyj – wskaźnik na plik wyjściowy.

void usuwanie (**osoba***rodzic)

Funkcja, która usuwa pojedynczy węzeł drzewa.

Parametry funkcji:
---
osoba*&rodzic – Wskaźnik na rodzica.

void tworzenie_plikow (**lot***glowa)

Funkcja tworzy plik tekstowy dla każdej komórki listy. Przekazuje do niego symbol , lotnisko, datę lotu i ilość pasażerów pobierając bezpośrednio z komórki listy oraz listę pasażerów poprzez wywołanie funkcji *wypisz_drzewo*. Następnie wywoływana jest funkcja *usuwanie*, w celu usunięcia całego podwieszonego drzewa do komórki listy, a następnie usuwana jest konkretna komórka listy.

Parametry funkcji:
---
lot*glowa – Wskaźnik na pierwszy element listy.

6 Testowanie

6.2 Przykładowe dane testowe

SO22G Sosnowiec 1936-05-12 Brzęczyszczykiewicz 124	poprawne dane
KR54R Katowice 2011-12-13 Jaworek 33	poprawne dane
TY340 Londyn 2012-02-03 Hastings 2	poprawne dane
SO22G Sosnowiec 1936-05-12 Lis 11	poprawne dane
	pusta linia
KR54R Katowice 2011-12-13 Gajos 12	dodatkowe spacje
TY340 Londyn 2012-02-03 Grec 36	poprawne dane
KR54R Katowice 2011-12-13 Maniek 0001	zera na początku nr miejsca

6.2 Pliki wyjściowe dla przykładowych danych

Plik *„KR54R.txt”*

symbol lotu: KR54R

lotnisko: Katowice

data lotu: 2011-12-13

lista pasazerow:

1 Maniek

12 Gajos

33 Jaworek

Liczba rezerwacji: 3

Plik *„TY340.txt”*

symbol lotu: TY340

lotnisko: Londyn

data lotu: 2012-02-03

lista pasazerow:

2 Hastings

36 Grec

Liczba rezerwacji: 2

Plik *„SO22G.txt”*

symbol lotu: SO22G

lotnisko: Sosnowiec

data lotu: 1936-05-12

lista pasazerow:

11 Lis

124 Brzęczyszczykiewicz

Liczba rezerwacji: 2

7 Wnioski

- Użycie w strukturze listy z podwieszonymi drzewami binarnymi (drzewo binarne pasażerów w liście lotów) zapobiega ciągłemu otwieraniu i zamykaniu plików wyjściowych.
- Przekazywanie zmiennych typu string w postaci referencji pozwala na zaoszczędzenie pamięci.
- Mimo, że nie jest to konieczne, zwolnienie pamięci zajmowanej przez listę jest dobrą praktyką.