Politechnika Śląska w Gliwicach
Wydział Automatyki, Elektroniki i Informatyki

Podstawy Programowania Komputerów

Dwudzielny

autor	Tomasz Sojka
prowadzący	dr inż. Krzysztof Simiński
rok akademicki	2018/2019
kierunek	informatyka
rodzaj studiów	SSI
semestr	1
termin laboratorium / ćwiczeń	poniedziałek, 08:30 – 10:00
grupa	2
sekcja	6
termin oddania sprawozdania	2018-01-25
data oddania sprawozdania	2018-01-25

1 Treść zadania

Napisac program, do sprawdzania, czy graf nieskierowany jest dwudzielny. Plik z grafem ma nastepujaca postac:
• Kazda krawedz jest podana w osobnej linii; podane sa dwa wierzchołki, które łaczy krawedz.
• W pliku moga wystapic puste linie.
• W linii moga wystapic dodatkowe (nadmiarowe) znaki białe.

Program wypisuje do pliku wyjsciowego zadany graf i komunikat, czy jest to graf dwudzielny, czy nie. Jezeli zadany graf jest dwudzielny, program wypisuje wierzchołki z obu grup. Program uruchamiany jest z linii polecen z wykorzystaniem nastepujacych przełaczników (kolejnosc przełaczników jest dowolna):

`-i` plik wejsciowy z krawedziami grafu
`-o` plik wyjsciowy z wynikami

2 Analiza zadania

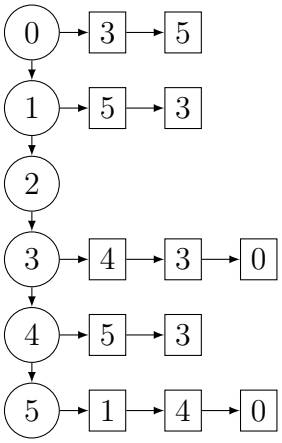
Zagadnienie przedstawia problem sprawdzenia dwudzielności grafu. Wymaga ono opracowania kodu, który odczyta krawędzi grafu z pliku wejściowego, sprawdzi czy graf ten jest dwudzielny i zapisze wynik(podany graf , wynik sprawdzania i dwie grupy wierzcholkow, jeli graf okazał się być dwudzielnym) do pliku wyjściowego.

2.1 Struktury danych

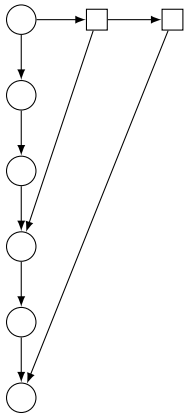
W programie wykorzystano listę jednokierunkową `element_listy`, w którą wpisano wierzcholki krawedzi sczytanych z pliku wejsciowego oraz dwie połączone struktury, `wierzcholek` i `krawedz`, dzięki którym udało się utworzyc listę sąsiedztwa (Rys 1), a także kolejkę potrzebną do jej przechodzenia. Lista jednokierunkowa pozwoliła na nieskomplikowane wczytanie danych z pliku wejciowego. Dzięki połączeniu dwóch struktur, do sprawdzania grafu nie było konieczne użycie osobnej tablicy dynamicznej z kolorami wierzchołków, czy też numerowanie tych wierzchołków(Patrz rys 2).

2.2 Algorytmy

Program sprawdza dwudzielność grafu poprzez kolorowanie jego wierzchołków na kontrastujące kolory (ustawianie wartości `kolor` ze struktury



Rysunek 1: Przykład listy sąsiedztwa przechowującej krawędzie grafu. Lista została utworzona z krawędzi sczytanych z pliku w kolejności:[(0,5), (0,3), (3,1), (4,3), (5,4), (1,5)]. Jest to jednak rysunek poglądowy i nie odzwierciedla prawdziwej budowy stworzonej w programie listy sąsiedztwa.



Rysunek 2: Przykład pokazuje prawdziwą budowę listy na podstawie wierzchołka nr 0 powyższego grafu i jego sąsiadów. Jak widać element listy głównej (pionowej, z wierzchołkami) zawiera: wskaźnik na następny element, na listę krawędzi(pozioma), , zaś element listy krawędzi zwieraj wskaźnik na następny element i na wierzchołek. (Wierzchołki zawierają również informacje do sprawdzania dwudzielność: `kolor` i `odwiedzony`, lecz nie są one istotne w budowie struktury.

`wierzcholek` na 1 lub -1). Po wybraniu początkowego nieodwiedzonego elementu , koloruje go na biało (wartość 1) i wszystkie wierzchołki sąsiadujące (końce krawędzi) na kolor odwrotny(tutaj wartość -1), teraz wierzchołki sąsiadujące zostają wierzchołkami głównymi i algorytm koloruje ich nieodwiedzonych sąsiadów na odwrotny kolor(tutaj wartość 1). Operacja się powtarza do momentu , aż wszystkie wierzchołki grafu zostaną sprawdzone lub do momentu gdy któryś odwiedzony sąsiad ma ten sam kolor co wierzchołek główny(aktualnie rozpartywany). Pierwsza opcja oznacza, że rozpatrywany graf posiada cechę dwudzielności, druga, że nie posiada tej cechy.

3 Specyfikacja zewnętrzna

Program jest uruchamiany z linii poleceń. Należy przekazać do programu nazwy plików: wejściowego i wyjściowego po odpowiednich przełącznikach (odpowiednio: `-i` dla pliku wejściowego i `-o` dla pliku wyjściowego), np.

```
program -i graf.txt -o graf_sprawdzony
program -o graf_sprawdzony -i graf.txt
```

Pliki są plikami tekstowymi, ale mogą mieć dowolne rozszerzenie (lub go nie mieć.) Przełączniki mogą być podane w dowolnej kolejności. Uruchomienie programu bez żadnego parametru lub z błędnymi parametrami

```
program
program -k graf.txt -m graf_sprawdzony
```

powoduje wyświetlenie komunikatu bledu krótkiej pomocy.

Podanie nieprawidłowej nazwy pliku, jego rozszerzenia, lub podanie nazwy pustego pliku powoduje wyświetlenie odpowiedniego komunikatu:

```
Brak pliku, niepoprawne rozszerzenie lub plik pusty !
```

4 Specyfikacja wewnętrzna

Program został zrealizowany zgodnie z paradygmatem strukturalnym. Program rozdzielono na cztery pliki źródłowe (`.cpp`):
`main.cpp`, `pliki_i_przelaczniki.cpp`,
`lista_i_kolejka.cpp`, `sprawdzanie_dwudzielnosci.cpp`.
Nagłówki umieszczono w plikach nagłówkowych (`.h`):
`struktury_i_listy.h`, `pliki_i_przelacznikiy.h`.

Specyfikacja wewnętrzna znajduje się na końcu pliku (Patrz str. 5))

5 Testowanie

Program został przetestowany na krawedziach o różnych wartosciach . Wierzcholki mogą być numerowane dowolnymi liczbami całkowitymi i nie muszą być numerowane w kolejność (od zera, rosnąco co 1). Znaki białe w pliku z grafem są pomijane, a puste linie lub zawierające tylko spacje, czy tabulacje nie powodują żadnych błędów w działaniu programu. Nie podanie parametru lub podanie błędnego powoduje zgłoszenie błędu, wypisanie krótkiej instrukcji i zatrzymanie wykonywania programu. Odwołanie się do nieistniejącego pliku wejściowego, lub pliku pustego powoduje wyświelenie stosownego komunikatu i zakończenie programu.

6 Wnioski

Program Dwudzielny choć z początku wydawał się być programem nie do przejścia, po dogłębnym przeanalizowaniu okazuje się być w zasięgu nawet początkującego programisty. Samodzielne tworzenie kodu pod presją czasu pozwoliło na rozszerzenie wiedzy na temat języka c++ i przede wszystkim na przećwiczenie dynamicznych struktur danych. Największym problemem podczas wykonywania zadania było utworzenie struktury listy sąsiedztwa, funkcji sprawdzającej dwudzielność grafu, oraz ograniczenie czasowe na wykonanie programu.