Algorytmy Optymalizacji Dyskretnej

Felix Zieliński 272336

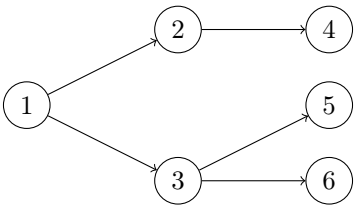
Lista 1

Wszystkie programy napisałem w języku C. Początkowo reprezentowałem grafy jako macierz sąsiedztwa ze względu na łatwość implementacji, jednakże szybko przekonałem się, że taka reprezentacja jest nieodpowiednia dla dużych danych i proces był zabijany przez brak pamięci. Dlatego też obecnie moje programy używają listy sąsiedztwa do reprezentacji grafów, która nie tylko wymaga mniej zasobów, ale jest także wydajniejsza. Programy były wywoływane poprzez: *cat {plik.txt} | {program} {argumenty}*, gdzie argumentem dla wyświetlenia drzewa przeszukań jest 'T' (tylko Zadanie 1).

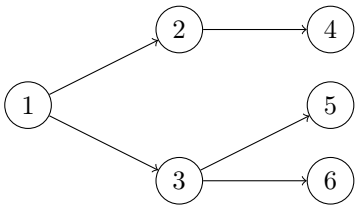
Zadanie 1.
DFS oraz BFS zaimplementowałem na podstawie ich opisów z książki Thomasa H. Cormena. Oba algorytmy tworzą listę sąsiedztwa według kolejności podanych krawędzi, i tak też przeszukują graf (więc nie kierują się numeracją wierzchołków). Kolejność odwiedziń w DFS jest zapisywana post-order. Poniżej prezentuję wyniki dla grafów z polecenia, przypadki skierowane oraz nieskierowane:

1. graf, przypadek skierowany

BFS



DFS



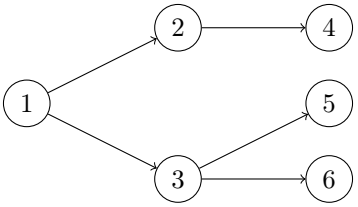
Kolejność odwiedzin:

BFS: 1 3 2 6 5 4

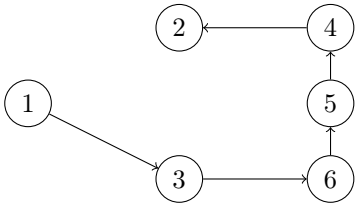
DFS: 6 3 5 4 2 1

1. graf, przypadek nieskierowany

BFS



DFS



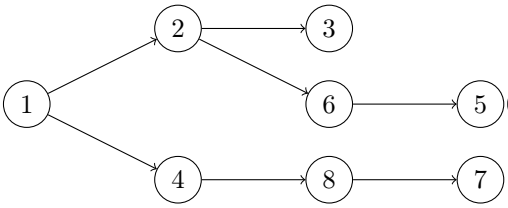
Kolejność odwiedzin:

BFS: 1 3 2 6 5 4

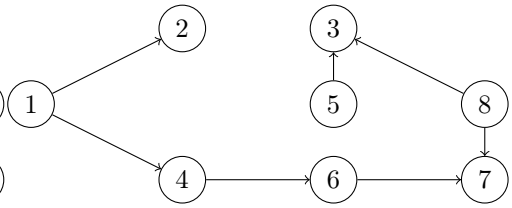
DFS: 2 4 5 6 3 1

2. graf, przypadek skierowany

BFS



DFS



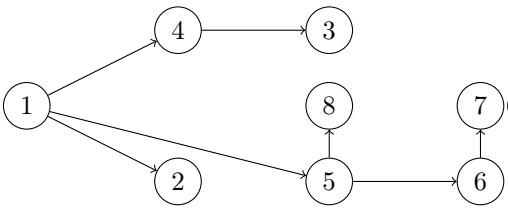
Kolejność odwiedzin:

BFS: 1 4 2 8 6 3 7 5

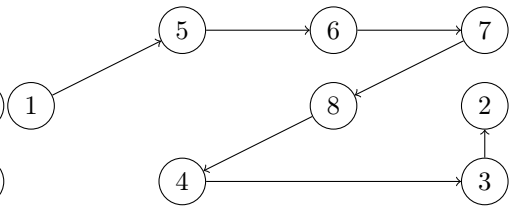
DFS: 3 7 8 4 5 6 2 1

2. graf, przypadek nieskierowany

BFS



DFS

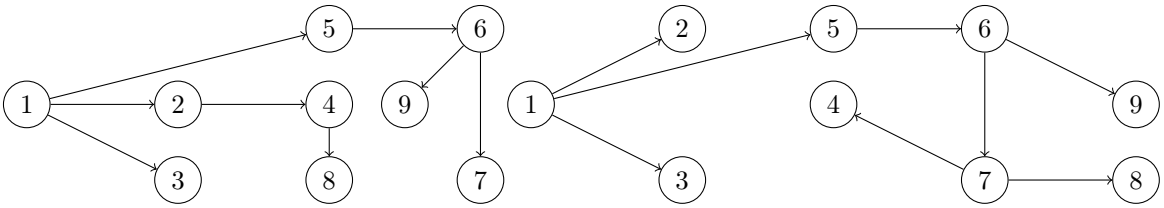


Kolejność odwiedzin:

BFS: 1 5 4 2 6 8 3 7 **DFS:** 2 3 4 8 7 6 5 1

3. graf, przypadek skierowany

BFS **DFS**

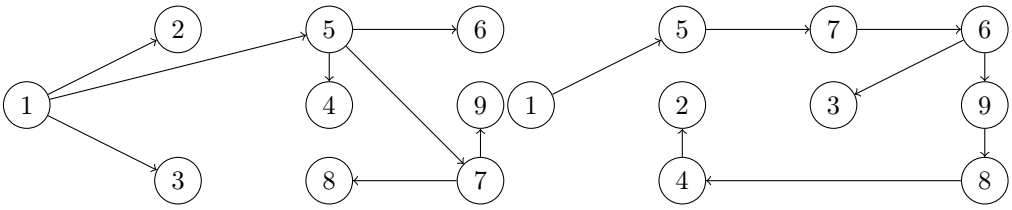


Kolejność odwiedzin:

BFS: 1 5 3 2 6 4 7 9 8 **DFS:** 8 4 7 9 6 5 3 2 1

3. graf, przypadek nieskierowany

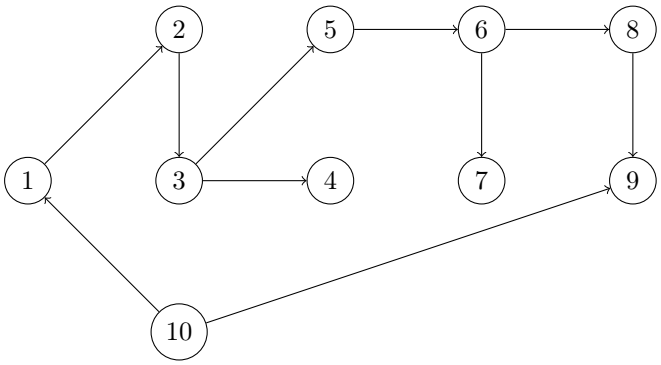
BFS **DFS**



Kolejność odwiedzin:

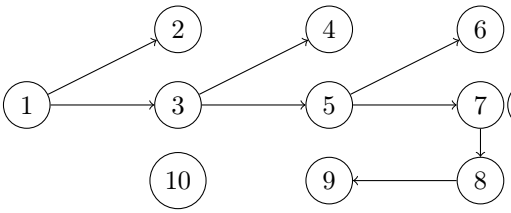
BFS: 1 5 3 2 7 4 6 9 8 **DFS:** 2 4 8 9 3 6 7 5 1

Mój graf:

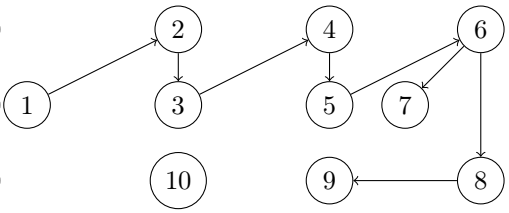


Mój graf, przypadek skierowany

BFS



DFS



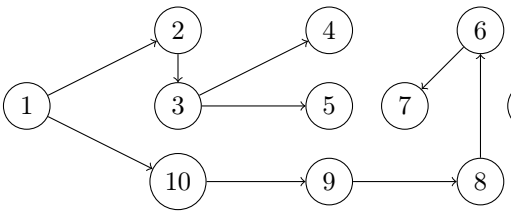
Kolejność odwiedzin:

BFS: 1 2 3 4 5 6 8 7 9 10

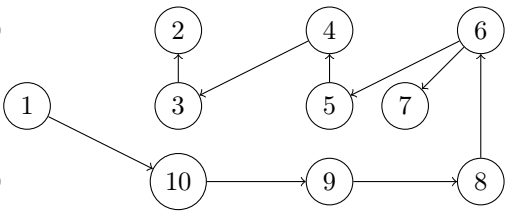
DFS: 9 8 7 6 5 4 3 2 1 10

Mój graf, przypadek nieskierowany

BFS



DFS



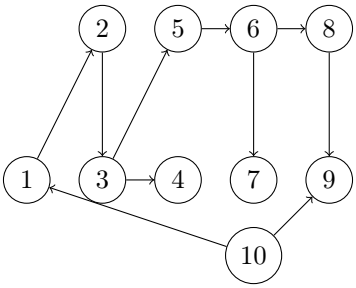
Kolejność odwiedzin:

BFS: 1 10 2 9 3 8 4 5 6 7 **DFS:** 7 2 3 4 5 6 8 9 10 1

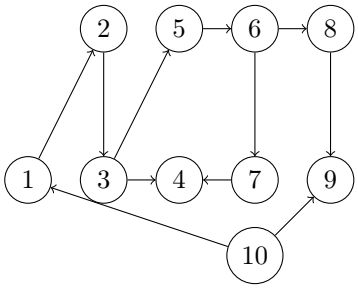
Zadanie 2.
W tym zadaniu również kierowałem się algorytmem opisanym w książce Cormena. Używam tutaj lekko zmodyfikowanego DFS-a - dodałem tablicę, aby trzymać informacje o obecnie przechodzonej trasie. Jeżeli obecnie odwiedzany wierzchołek znajduje się na niej, oznacza to, że graf zawiera skierowany cykl. Program wtedy wypisuje tę informację oraz przerywa działanie. Aby posortować wierzchołki w porządku topologicznym, program dodaje wierzchołek do odpowiedniej tablicy po odwiedzeniu wszystkich jego sąsiadów. Czasy wykonywania są podane na podstawie jednego wywołania, w mikrosekundach.

Moje grafy

acykliczny



cykliczny



Zwrócone informacje:

Kolejonść: 9 8 7 6 5 4 3 2 1 10 **Ten graf ma cykl!**

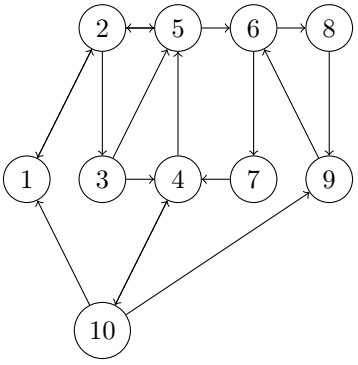
Grafy testowe

Zadanie 3.
Rozkład na silnie spójne składowe wyznaczam poprzez wykonaniu DFS-a w celu ustalenia wartości post-visit, następnie transponowania grafu oraz kolejnego jego przejścia DFS-em. DFS ma złożoność O(V + E). Dzięki zastosowaniu listy sąsiedztwa, również funkcja transponująca graf ma taką złożoność. Cały algorytm mieści się więc w wymaganej w zadaniu złożoności obliczeniowej.

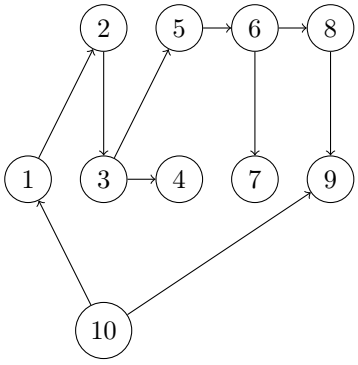
Moje grafy

Plik	Czy ma cykl?	Czas	Kolejność topologiczna
g2a-1	nie	11	16 15 12 11 14 10 8 7 6 13 9 5 4 3 2 1
g2a-2	nie	18	100 99 90 89 98 88 80 79 78 97 87 77 70 69 68 67 96 86 76 66 60 59 58 57 56 95 85 75 65 55 50 49 48 47 46 45 94 84 74 64 54 44 40 39 38 37 36 35 34 93 83 73 63 53 43 33 30 29 28 27 26 25 24 23 92 82 72 62 52 42 32 22 20 19 18 17 16 15 14 13 12 91 81 71 61 51 41 31 21 11 10 9 8 7 6 5 4 3 2 1
g2a-3	nie	28	N/A
g2a-4	nie	199	N/A
g2a-5	nie	8393	N/A
g2a-6	nie	71799	N/A
g2b-1	tak	8	N/A
g2b-2	tak	12	N/A
g2b-3	tak	25	N/A
g2b-4	tak	35	N/A
g2b-5	tak	151	N/A
g2b-6	tak	527	N/A

silnie spojny



z >1 silnie spójną składową



Outputy dla grafów testowych:

Graf 1
SCC 1: 1 5 4 3 2
Number of items in SCC: 5

SCC 2: 12 15 14 13
Number of items in SCC: 4

SCC 3: 6 8 9 7
Number of items in SCC: 4

SCC 4: 10 11
Number of items in SCC: 2

SCC 5: 16
Number of items in SCC: 1

Time: 12 microseconds

Graf 2
SCC 1: 1 6 5 4 3 2
Number of items in SCC: 6

SCC 2: 67 106 105 104 103 102 101 100 99 98 97 96 95 94 93 92 91 90 89 88 87 86 85 84 83 82 81 80 79 78 77 76 75 74 73 72 71 70 69 68
Number of items in SCC: 40

SCC 3: 7 13 19 25 31 37 38 39 40 41 42 36 35 29 28 34 22 21 27 33 15 14 20 26 32 8 9 16 10 23 17 11 30 24 18 12
Number of items in SCC: 36

SCC 4: 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66
Number of items in SCC: 24

SCC 5: 107
Number of items in SCC: 1

Time: 68 microseconds

Graf 3
SCC 1:
Number of items in SCC: 7

SCC 2:
Number of items in SCC: 400

SCC 3:
Number of items in SCC: 400

SCC 4:
Number of items in SCC: 200

SCC 5:
Number of items in SCC: 1

Time: 89 microseconds

Graf 4
SCC 1:
Number of items in SCC: 8

SCC 2:
Number of items in SCC: 4000

SCC 3:
Number of items in SCC: 3600

SCC 4:
Number of items in SCC: 2400

SCC 5:
Number of items in SCC: 1

Time: 659 microseconds

Graf 5
SCC 1:
Number of items in SCC: 9

SCC 2:
Number of items in SCC: 40000

SCC 3:
Number of items in SCC: 40000

SCC 4:
Number of items in SCC: 20000

SCC 5:
Number of items in SCC: 1

Time: 8793 microseconds

Graf 6
SCC 1:
Number of items in SCC: 10

SCC 2:
Number of items in SCC: 400000

SCC 3:
Number of items in SCC: 360000

SCC 4:
Number of items in SCC: 240000

SCC 5:
Number of items in SCC: 1

Time: 999174867 microseconds

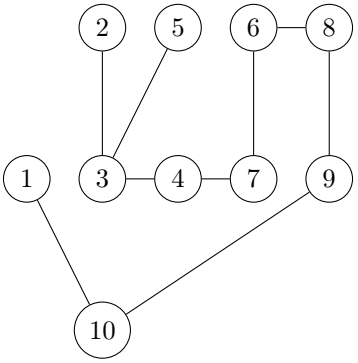
Zadanie 4.

Aby określić, czy graf jest dwudzielny, "koloruję"wierzchołki na jeden z dwóch kolorów naprzemiennie, używając BFS-a. Jeżeli podczas kolorowania zdarzy się, że odwiedzany wierzchołek sąsiaduje z wierzchołkiem o tym samym kolorze, to program przerywa działanie i zwraca informację, że graf nie jest dwudzielny. Udane pokolorowanie wskazuje na dwudzielność grafu.

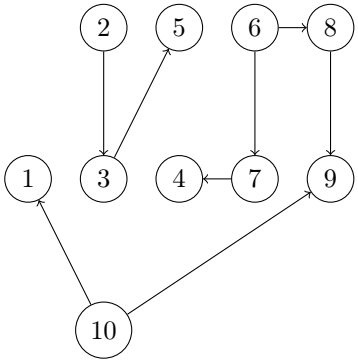
Wyniki dla grafów testowych są długie, dlatego zapisałem je w pliku res.txt

Moje grafy dwudzielne

nieskierowany

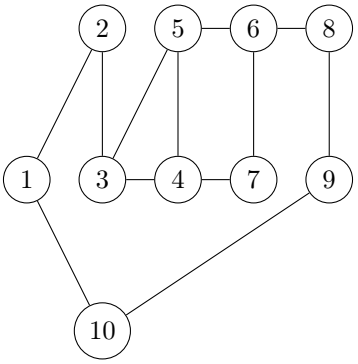


skierowany



Moje grafy niedwudzielne

nieskierowany



skierowany