UWAGI do drzew (dotyczy wszystkich projektów):
1) W strukturach drzewiastych w trakcie usuwania elementu który ma więcej niż jednego syna zastępujemy go jego następnikiem.
2) W trakcie usuwania z drzewa 2-3 najpierw próbujemy pożyczyć od prawego (o ile istnieje), a potem od lewego brata. Podobnie próbujemy najpierw scalić z prawym bratem, a gdy nie istnieje to z lewym.
3) W drzewie nie może być duplikatów – ponowne wstawienie tego samego klucza ma nie zadziałać

Projekt 1 – Sieć drogowa

Zaimplementować projekt do obsługi sieci połączeń drogowych. Projekt powinien składać się z następujących funkcjonalności:

1) Struktura drzewiasta umożliwiająca szybkie (log n) wykonywanie operacji (20 pkt):
- a) wyszukiwanie miasta (po nazwie – unikalna)
- b) dodawanie nowego miasta
- c) usuwanie istniejącego miasta
- d) wyszukanie liczby miast o danym prefiksie nazwy

Wariant A: implementacja drzewa AVL (20 pkt), lub drzewa BST (10 pkt)
Wariant B: implementacja 2-3 drzewa (20 pkt), lub drzewa BST (10 pkt)

2) Struktura grafu wraz z następującymi funkcjonalnościami (20 pkt):
- a) dodawanie/usuwanie miasta
- b) dodawanie/usuwanie drogi pomiędzy daną parą miast (parametr: długość w km)
- c) znalezienie najkrótszej drogi pomiędzy zadaną parą miast (postać drogi i długość)
- d) obliczenie do ilu miast skróci się długość przejazdu z miasta A, jeżeli wybudowana zostałaby droga pomiędzy miastami B i C (o zadanej długości), gdy jest ich więcej niż 100 wystarczy wypisać 100+

UWAGA: operacje dodawania/usuwania miasta są globalne: implikują zmiany zarówno w strukturze drzewiastej jak i w strukturze grafu (łącznie z usuwaniem dróg z nimi połączonych)

Algorytmy grafowe będą oceniane pod względem poprawności i szybkości działania operacji

W przypadku nie istnienia połączenia pomiędzy miastami (lub nie istnienia miast) należy wypisać NIE.

Za grafy bez ostatniej funkcjonalności jest 14 pkt, a 20 pkt można dostać tylko, gdy będzie ona zrealizowana w sposób optymalny. **UWAGA: dla uproszczenia zakładamy, że w testach zapytania dotyczące ostatniej funkcjonalności zawsze pojawiają się na końcu pliku testowego (gdy graf już się nie zmienia) i dla danego testu miasto A jest zawsze takie samo we wszystkich zapytaniach.**

3) Możliwość wczytania, modyfikacji danych i testowania na podstawie pliku z instrukcjami:

DM nazwa_miasta - dodanie miasta
UM nazwa_miasta – usunięcie miasta
WM nazwa_miasta – wyszukanie miasta (wypisanie odpowiedzi TAK/NIE)
LM prefiks – obliczenie liczby miast o danym prefiksie (i wypisanie tej liczby)
WY – wypisanie struktury drzewa na ekran
DD miasto1 miasto2 długość – dodanie drogi dwukierunkowej pomiędzy miastami
UD miasto1 miasto2 – usunięcie drogi pomiędzy miastami
ND miasto1 miasto2– obliczenie najkrótszej drogi z jednego miasta do drugiego (wypisanie jej długości)
IS miasto1 miasto2 miasto3 długość – obliczenie do ilu miast skróci się najkrótsza droga z miasta1 po potencjalnym dodaniu drogi pomiędzy miastem2 i miastem3 o zadanej długości

Projekt 2 – Sieć kolejowa

Zaimplementować projekt do obsługi sieci połączeń kolejowych. Projekt powinien składać się z następujących funkcjonalności:

1) Struktura drzewiasta umożliwiająca szybkie (log n) wykonywanie operacji (20 pkt):

a) wyszukiwanie połączenia (po dacie utworzenia + id_stacji1 + id_stacji2)

b) dodawanie nowego połączenia pomiędzy stacjami (id_stacji1, id_stacji2, długość w km, prędkość na odcinku w km/h, data utworzenia danego połączenia, zakładamy, że id_stacji są z przedziału od 1 do 10000)

c) usuwanie istniejącego połączenia (po dacie jego utworzenia + id_stacji1 + id_stacji2)

d) wyszukanie liczby połączeń utworzonych pomiędzy zadanymi datami

Wariant A: implementacja drzewa AVL (20 pkt), lub drzewa BST (10 pkt)
Wariant B: implementacja 2-3 drzewa (20 pkt), lub drzewa BST (10 pkt)

2) Struktura grafu wraz z następującymi funkcjonalnościami (20 pkt):

a) dodawanie/usuwanie połączenia kolejowego

b) znalezienie najszybszego połączenia (w minutach) pomiędzy zadaną parą stacji (postać połączenia i czas trwania). Uwaga: czas przesiadki na stacji wynosi 5 minut.

c) obliczenie, kiedy po raz pierwszy (data) udało się pomiędzy zadaną parą stacji uzyskać czas przejazdu nie większy niż zadany limit (na przesiadkę na stacji potrzeba 5 minut)

UWAGA: operacje dodawania/usuwania połączeń są globalne: implikują zmiany zarówno w strukturze drzewiastej jak i w strukturze grafu

Algorytmy grafowe będą oceniane pod względem poprawności i szybkości działania operacji

3) Możliwość wczytania, modyfikacji danych i testowania na podstawie pliku z instrukcjami:

DP RRRR-MM-DD id1 id2 predkosc dlugosc - dodanie połączenia
UP RRRR-MM-DD id1 id2 – usuniecie polaczenia
WP RRRR-MM-DD id1 id2 – wyszukanie połączenia (wypisanie odpowiedzi TAK/NIE)
LP RRRR-MM-DD RRRR-MM-DD – obliczenie liczby połączeń utworzonych pomiędzy zadanymi datami
WY – wypisanie struktury drzewa na ekran (wystarczą daty)
NP id1 id2 – obliczenie najszybszego połączenia ze stacji id1 do stacji id2 (parametry połączeń dobrane tak, że czas w minutach jest zawsze liczbą całkowitą), wypisać wynik w minutach
ND id1 id2 M – wypisanie najwcześniejszej daty, przy której istnieje połączenie z id1 do id2 o czasie trwania nie większym niż M

UWAGA: pomiędzy zadaną parą miast może powstać więcej niż jedno połączenie, a prędkości tych połączeń zawsze rosną wraz z datą utworzenia.

W przypadku nie istnienia połączenia pomiędzy stacjami (lub nie istnienia stacji), lub braku możliwości skonstruowania połączenia o czasie trwania <=M należy wypisać NIE.

Algorytmy grafowe będą oceniane pod względem poprawności i szybkości działania operacji.

Za grafy bez ostatniej funkcjonalności jest max. 14 pkt, a 20 pkt można dostać tylko, gdy będzie ona zrealizowana w sposób optymalny.

Projekt 3 – Sieć komunikacyjna

Zaimplementować projekt wspomagający obsługę sieci internetowej. Projekt powinien składać się z następujących funkcjonalności:

1) Struktura drzewiasta umożliwiająca szybkie (log n) wykonywanie operacji (20 pkt):

a) wyszukiwanie urządzenia (po adresie IP4 X.X.X.X)
b) dodawanie nowego urządzenia (adres IP4 X.X.X.X)
c) usuwanie istniejącego urządzenia (po adresie)
d) wyszukanie liczby komputerów w danej podsieci X.X.X.* (maska 24-bitowa)

Wariant A: implementacja drzewa AVL (20 pkt), lub drzewa BST (10 pkt)
Wariant B: implementacja 2-3 drzewa (20 pkt), lub drzewa BST (10 pkt)

2) Struktura grafu wraz z następującymi funkcjonalnościami (20 pkt):

a) dodawanie/usuwanie połączenia pomiędzy routerami (podsieciami X.X.X.*)
b) znalezienie najszybszego połączenia pomiędzy zadaną parą komputerow

UWAGA: koszt danego połączenia liczony jest jako iloraz przepustowości maksymalnej (100 Gbs) i jego przepustowości np. dla łącza 100 Mbs jego koszt wynosi 1000.

Algorytmy grafowe będą oceniane pod względem poprawności i szybkości działania operacji

3) Możliwość wczytania, modyfikacji danych i testowania na podstawie pliku z instrukcjami:

DK ip4 - dodanie komputera o zadanym adresie
UK ip4 – usunięcie komputera o zadanym adresie
WK ip4 – wyszukanie komputera o zadanym adresie (wypisz TAK/NIE)
LK podsiec – wypisanie liczby komputerow w danej podsieci (bez routera, który zawsze istnieje w podsieci)
WY – wypisanie struktury drzewa
DP podsiec1 podsiec2 szybkosc (np. 253.25.18 253.22.17 100M) – dodanie połączenia pomiędzy routerami (podsieciami) o danej przepustowości (100 M, może być też np. 10G).
UP podsiec1 podsiec2 – usunięcie połączenia
NP ip4 ip4 – obliczenie najszybszego połączenia pomiędzy zadaną parą komputerów (liczone jako suma kosztów), przy założeniu że uwzględniamy jedynie koszt przesyłu pomiędzy routerami (podsieciami)

W przypadku nie istnienia w drzewie komputerów, między którymi chcemy wyszukać najszybsze połączenie lub w przypadku nie istnienia samego połączenia, należy wypisać NIE.

Algorytmy grafowe będą oceniane pod względem poprawności i szybkości działania operacji