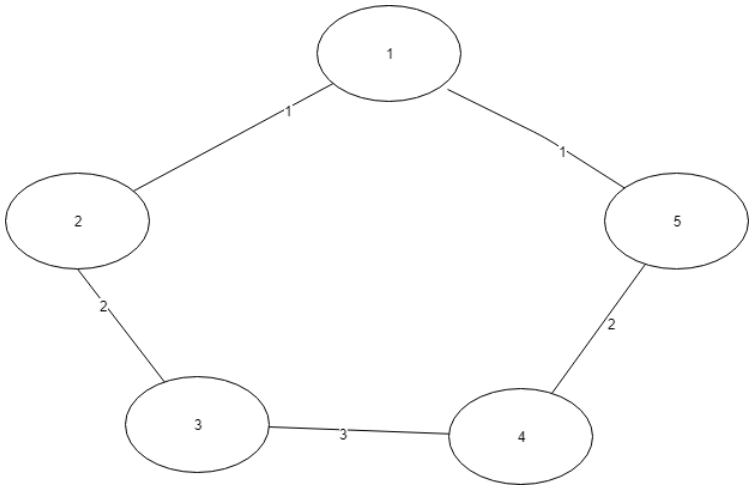
DOMINIK WISZYŃSKI

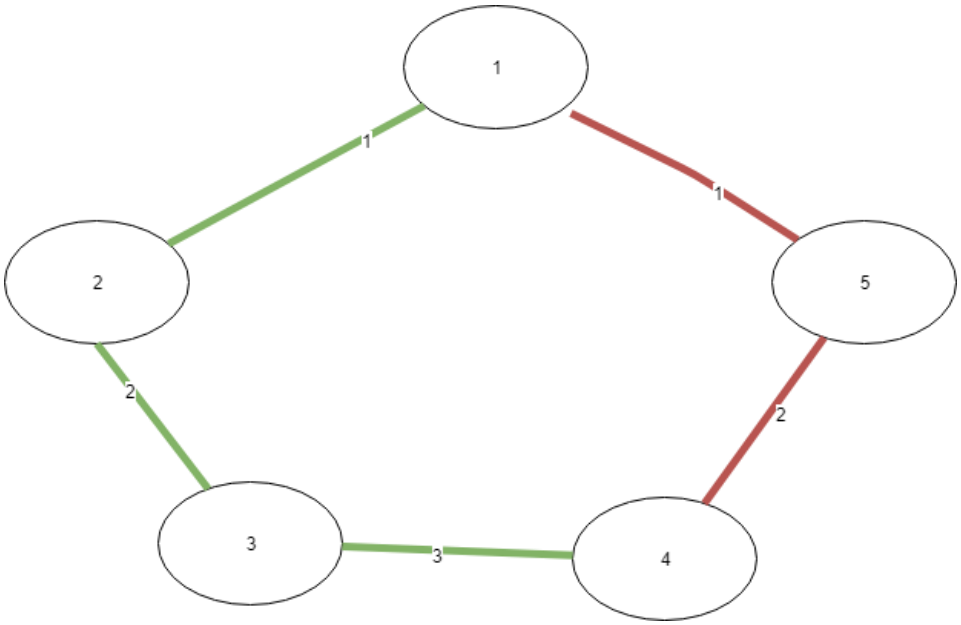
PROJEKT AAL – DOKUMENTACJA

1. Słowna reprezentacja problemu
   Wzdłuż obwodnicy miejskiej zostały ustawione maszty, na których znajdują się kamery monitoringu miejskiego. Następnie spostrzeżono, że znaczna część kierowców przekracza prędkość. Aby temu zaradzić, postanowiono zainstalować fotoradary do pomiaru prędkości i umieścić je na wspomnianych wcześniej masztach. Postanowiono, że radary będą zainstalowane na najbardziej oddalonych od siebie masztach. Mając N masztów i informacje o odległości pomiędzy sąsiadującymi masztami zaproponuj algorytm, który wybierze dwa maszty, na których powinny zostać zainstalowane radary
2. Analiza problemu
   Obwodnica miejska kształtem przypomina elipsę. Maszty znajdujące się na naszej obwodnicy są punktami znajdującymi się na torze danej elipsy. Jeżeli byśmy połączyli konkretne punkty można zauważyć, że stworzylibyśmy graf, którego wierzchołkami są maszty, a odległości między masztami to krawędzie w grafie. Graf ten jest cyklem. Przykładowy rysunek:



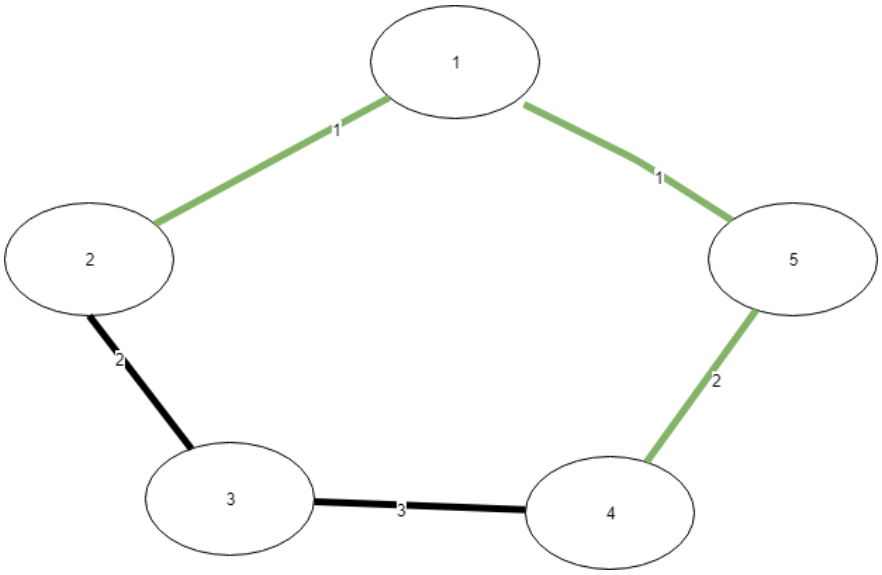
   Skoro naszą obwodnicę możemy interpretować jako cykl w którym maszty są wierzchołkami, to cały nasz problem sprowadza się do

znalezienia najbardziej oddalonych od siebie wierzchołków w cyklu. Jako, że jest to cykl, to pomiędzy dwoma dowolnymi wierzchołkami istnieją dwie ścieżki, których suma długości daje nam długość całego obwodu.

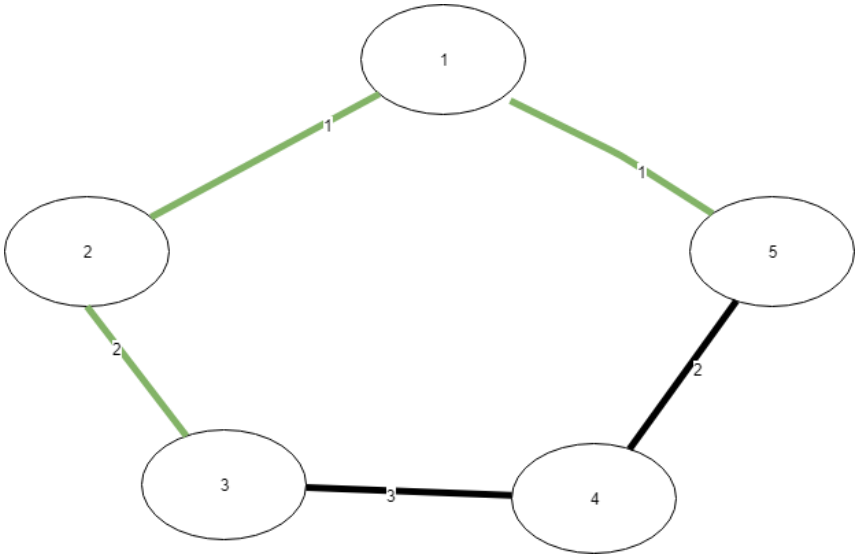


Na rysunku mamy narysowane 2 ścieżki między wierzchołkami 1 i 4. Zaznaczona na zielono przechodzi przez wierzchołki 2 i 3 i ma długość 6, zaś zaznaczona na czerwono przechodzi tylko przez wierzchołek 5 i ma długość 3. Odległością między tymi wierzchołkami jest najkrótsza możliwa ścieżka łącząca te dwa wierzchołki, czyli minimum z tych dwóch długości. W tym wypadku ta odległość będzie równa 3. Łatwo zauważyć, że skoro suma długości 2 ścieżek pomiędzy dowolnymi dwoma wierzchołkami jest stała i równa obwodowi, a my szukamy minimum z tych długości to oznacza to, że odległość między dowolnymi dwoma wierzchołkami nigdy nie przekroczy połowy obwodu naszego cyklu. Oznacza to, że każda odległość przekraczająca połowę obwodu nie może być naszą największą odległością, ponieważ istnieje druga krótsza ścieżka o długości równej: długość obwodu – długość ścieżki 1. Skoro znamy już maksymalną możliwą wartość naszej odległości to widzimy, że możemy dodawać długości kolejnych krawędzi do naszej ścieżki tak długo aż nie przekroczymy połowy obwodu. Wtedy należy usunąć pierwszą krawędź w naszej ścieżce i po raz kolejny sprawdzić czy długość tej ścieżki nie przekracza połowy obwodu. Kolejnym problemem jest fakt, że nasza najdłuższa ścieżka może przechodzić przez wierzchołek początkowy. Przykładowo dla grafu wspomnianego powyżej istnieją dwie takie ścieżki:

- 4->5->1->2(rysunek 1)



- 5->1->2->3(rysunek 2)

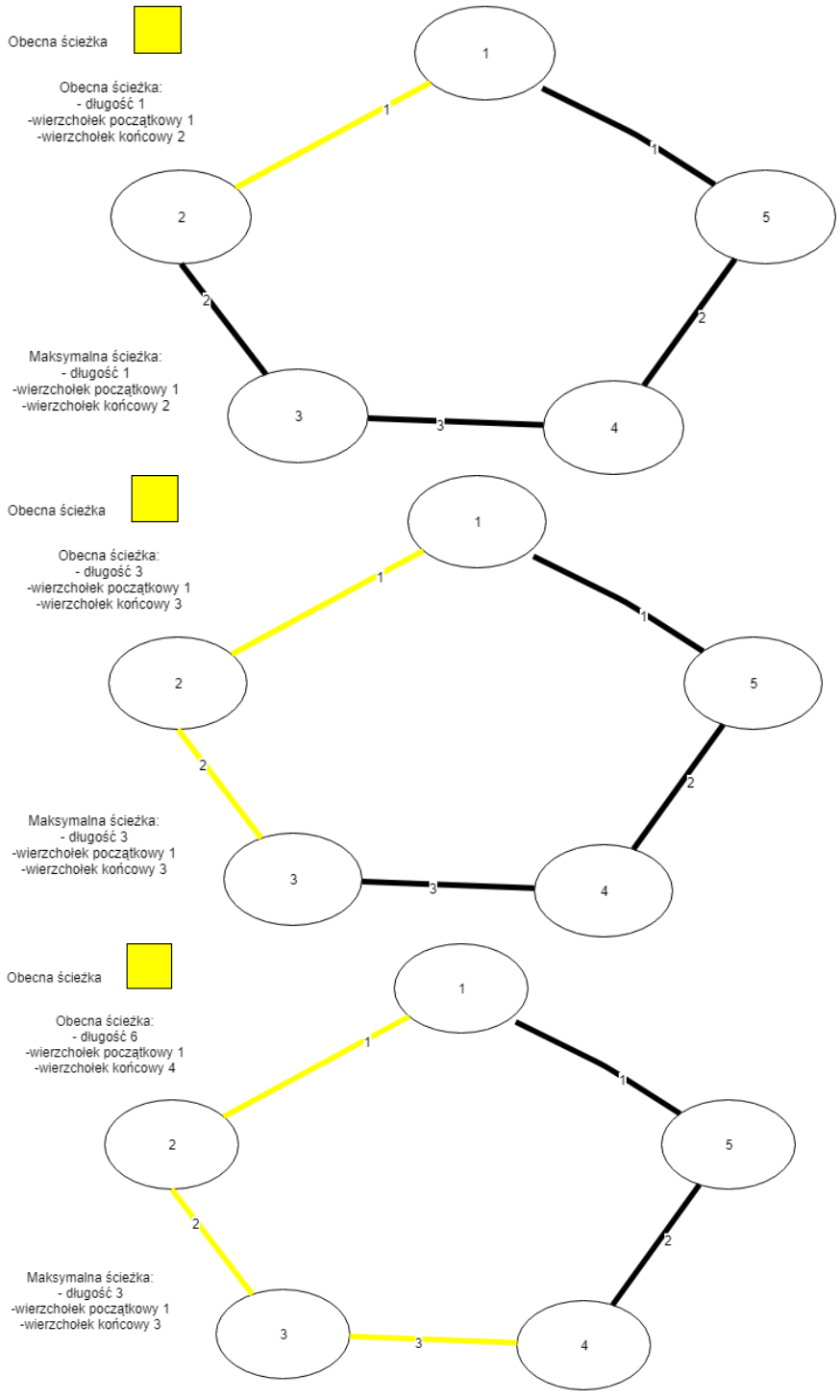


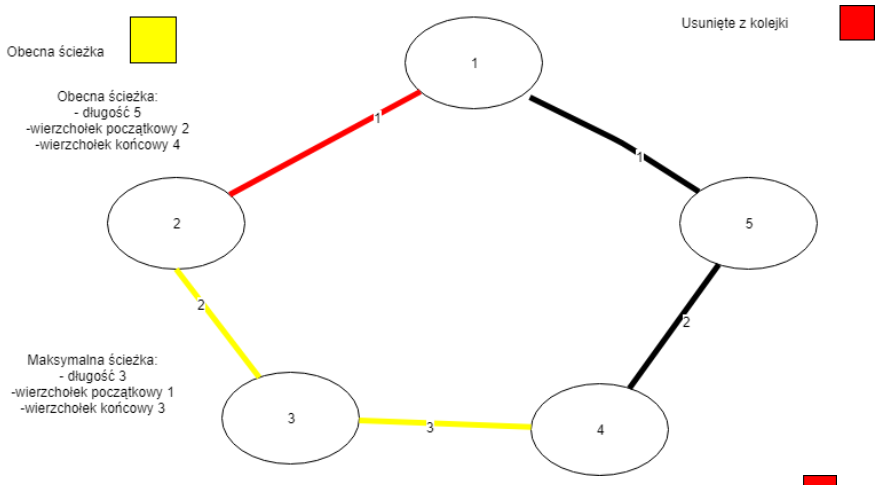
Oznacza to, że jedno przejście po grafie może nie wystarczyć, aby wyznaczyć maksymalną odległość między dwoma wierzchołkami. Dlatego, będziemy musieli przejść dwa razy, żeby uwzględnić i takie opcje.

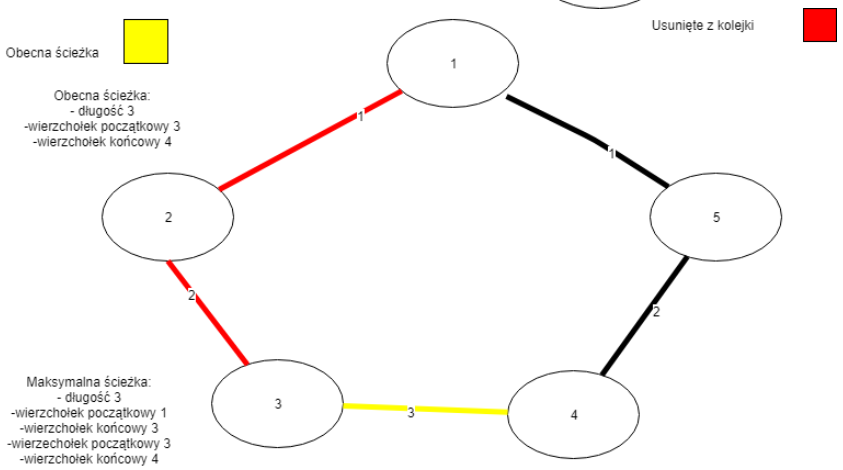
3. Proponowane rozwiązanie z pseudo kodem.
   Na wejście będą podane liczba wierzchołków oraz kolejne odległości między wierzchołkami np.
   3 //liczba wierzchołków
   4 //odległość między wierzchołkiem 1 i 2

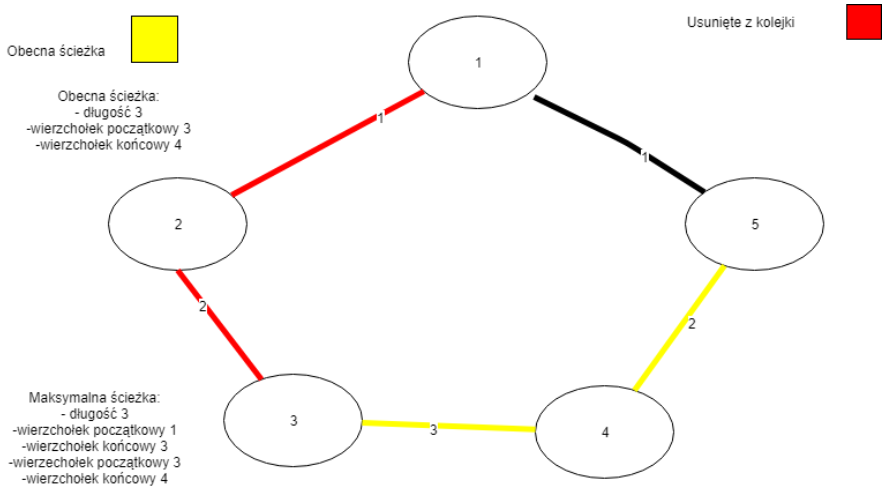
3 //odległość między wierzchołkiem 2 i 3
4 //odległość między wierzchołkiem 3 i 1
Odległości te zawsze będą podane w takiej kolejności dlatego nie będzie wymagane sortowanie ani patrzenie z jakim wierzchołkiem jest połączony jest konkretny wierzchołek. I-ty wierzchołek zawsze będzie połączony z wierzchołkiem i-1%liczba wierzchołków oraz i+1%liczba wierzchołków. Sumujemy nasze odległości aby uzyskać wartość obwodu naszego grafu. Długości krawędzi trafiają do tablicy wraz z indeksem wierzchołka z którego wychodzą np. (1,4). Następnie przechodzimy w pętli po naszym zbiorze wierzchołków 2 razy(w kolejności wierzchołków 1..5,1..5), dla pewności, że znajdziemy 2 najbardziej odległe od siebie wierzchołki. Po kolei dla każdej z krawędzi wykonujemy następujące czynności:
1.Pobierz kolejną krawędź z grafu.
2.Dodajemy długość krawędzi to długości obecnej ścieżki(wartość początkowa = 0)
3.Dodajemy do kolejki FIFO z krawędziami obecnymi w naszej ścieżce
4.Sprawdzamy czy długość obecnej ścieżki jest większa niż połowa naszego obwodu. Jeśli jest przejdź do kroku 5, jeśli nie przejdź do kroku 6.
5.Usuń pierwszą krawędź z kolejki FIFO i zmniejsz długość obecnej ścieżki o długość usuwanej krawędzi. Przejdź do kroku 3.
6.Sprawdź czy długość obecnej ścieżki jest większa lub równa maksymalnej długości ścieżki w naszym grafie. Jeśli jest równa to dodaj parę wierzchołków do zbioru rozwiązań. Jeśli jest większa usuń poprzedni zbiór rozwiązań i dodaj naszą ścieżkę do nowego zbioru rozwiązań.
7.Sprawdź czy są jeszcze jakieś krawędzie do sprawdzenia. Jeśli są wróć do kroku 1. Jeśli nie zakończ algorytm. Zbiór rozwiązań jest naszym rozwiązaniem. Należy zakończyć algorytm po usunięciu krawędzi n->1 żeby uniknąć powtórzeń.

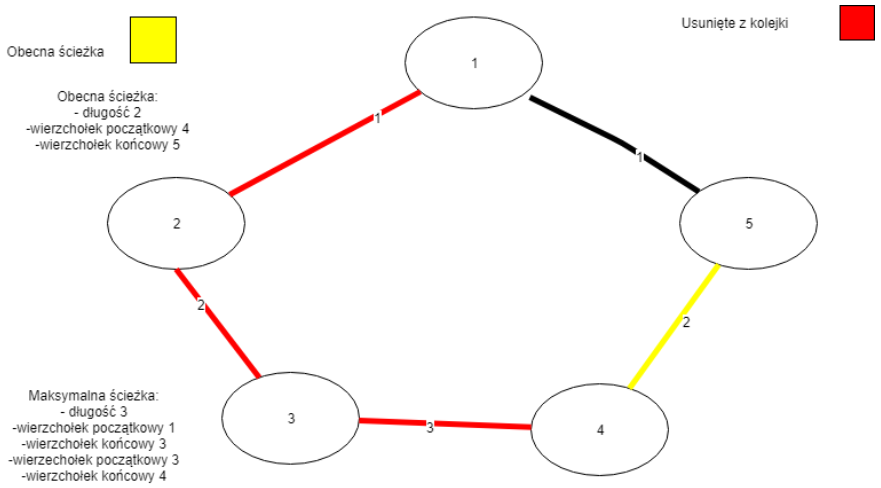
4. Graficzny przykład iteracji algorytmu

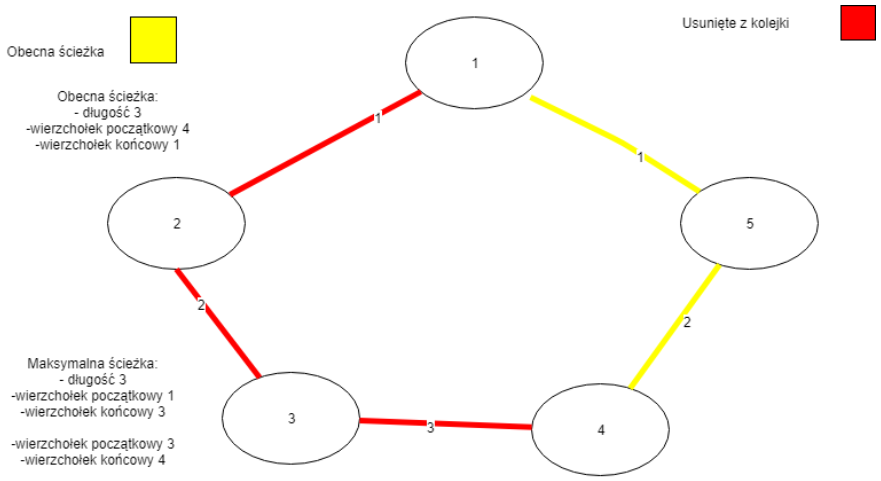


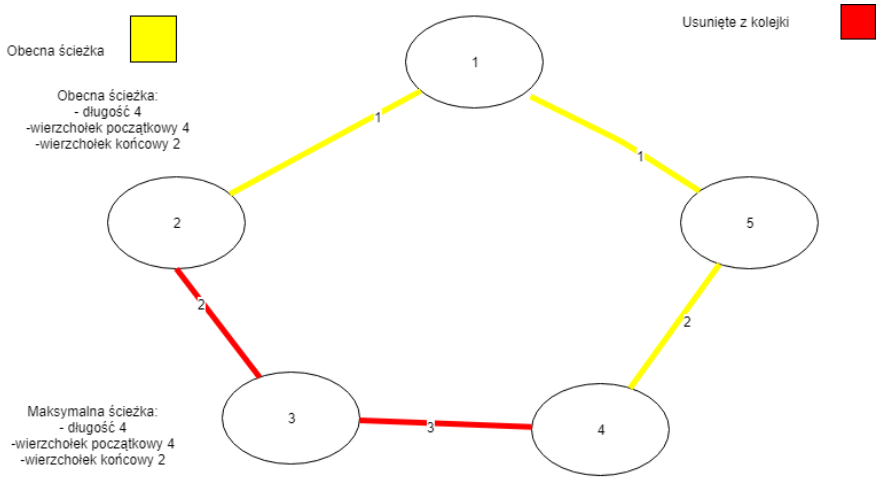

Obecna ścieżka
Usunięte z kolejki
Obecna ścieżka:
- długość 5
-wierzchołek początkowy 2
-wierzchołek końcowy 4
Maksymalna ścieżka:
- długość 3
-wierzchołek początkowy 1
-wierzchołek końcowy 3
1
2
3
4
5
1
1
2
2
3

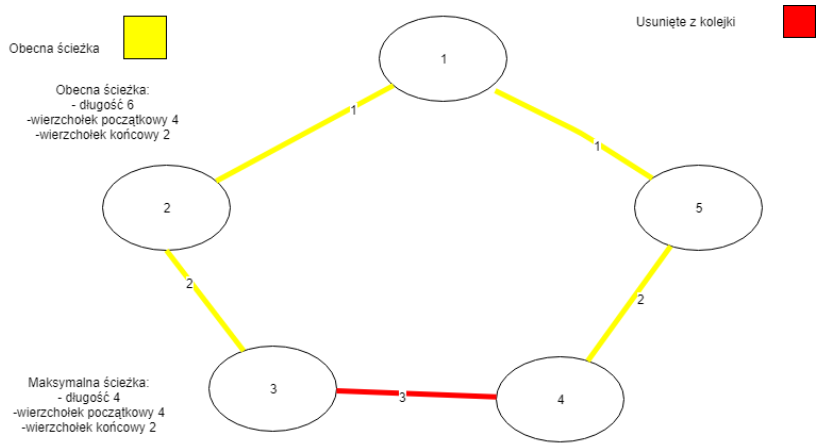

Obecna ścieżka
Usunięte z kolejki
Obecna ścieżka:
- długość 3
-wierzchołek początkowy 3
-wierzchołek końcowy 4
Maksymalna ścieżka:
- długość 3
-wierzchołek początkowy 1
-wierzchołek końcowy 3
-wierzechołek początkowy 3
-wierzchołek końcowy 4
1
2
3
4
5
1
1
2
2
3

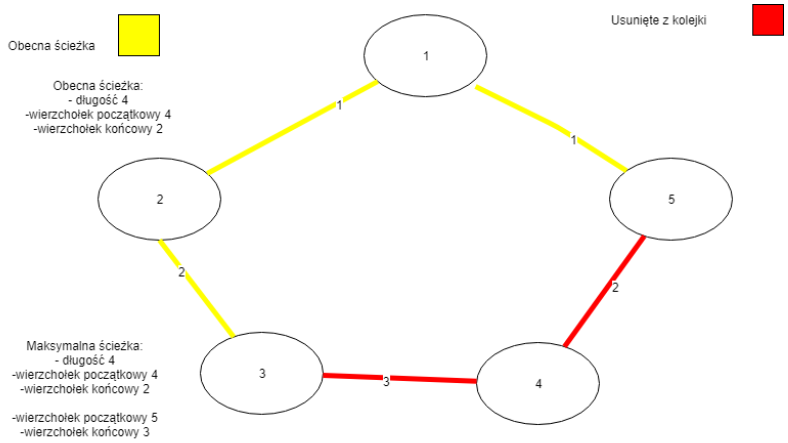

Obecna ścieżka
Usunięte z kolejki
Obecna ścieżka:
- długość 3
-wierzchołek początkowy 3
-wierzchołek końcowy 4
Maksymalna ścieżka:
- długość 3
-wierzchołek początkowy 1
-wierzchołek końcowy 3
-wierzechołek początkowy 3
-wierzchołek końcowy 4
1
2
3
4
5
1
1
2
2
3

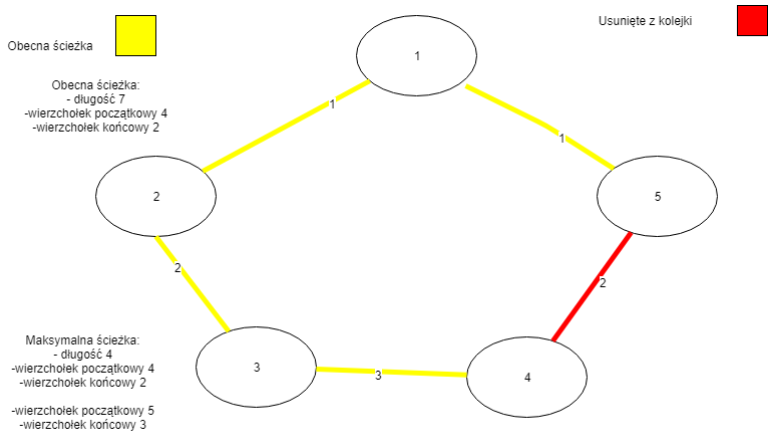

Obecna ścieżka
Usunięte z kolejki
Obecna ścieżka:
- długość 2
-wierzchołek początkowy 4
-wierzchołek końcowy 5
1
2
3
4
5
1
1
2
2
3
Maksymalna ścieżka:
- długość 3
-wierzchołek początkowy 1
-wierzchołek końcowy 3
-wierzechołek początkowy 3
-wierzchołek końcowy 4

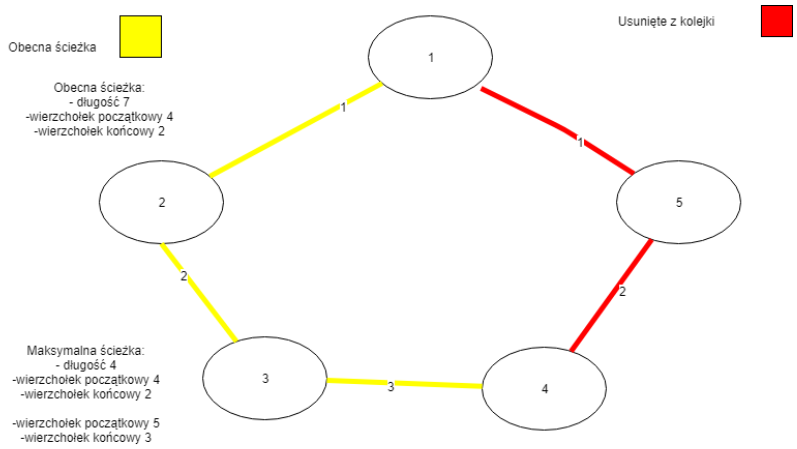

Obecna ścieżka
Usunięte z kolejki
Obecna ścieżka:
- długość 3
-wierzchołek początkowy 4
-wierzchołek końcowy 1
1
2
3
4
5
1
1
2
2
3
Maksymalna ścieżka:
- długość 3
-wierzchołek początkowy 1
-wierzchołek końcowy 3
-wierzchołek początkowy 3
-wierzchołek końcowy 4
-wierzchołek początkowy 4
-wierzchołek końcowy 1


Obecna ścieżka
Usunięte z kolejki
Obecna ścieżka:
- długość 4
-wierzchołek początkowy 4
-wierzchołek końcowy 2
1
2
3
4
5
1
1
2
2
3
Maksymalna ścieżka:
- długość 4
-wierzchołek początkowy 4
-wierzchołek końcowy 2


Obecna ścieżka
Usunięte z kolejki
Obecna ścieżka:
- długość 6
-wierzchołek początkowy 4
-wierzchołek końcowy 2
1
2
3
4
5
1
1
2
2
3
Maksymalna ścieżka:
- długość 4
-wierzchołek początkowy 4
-wierzchołek końcowy 2


Obecna ścieżka
Usunięte z kolejki
Obecna ścieżka:
- długość 4
-wierzchołek początkowy 4
-wierzchołek końcowy 2
1
2
3
4
5
1
1
2
2
3
Maksymalna ścieżka:
- długość 4
-wierzchołek początkowy 4
-wierzchołek końcowy 2
-wierzchołek początkowy 5
-wierzchołek końcowy 3


Obecna ścieżka
Usunięte z kolejki
Obecna ścieżka:
- długość 7
-wierzchołek początkowy 4
-wierzchołek końcowy 2
1
2
3
4
5
1
1
2
2
3
Maksymalna ścieżka:
- długość 4
-wierzchołek początkowy 4
-wierzchołek końcowy 2
-wierzchołek początkowy 5
-wierzchołek końcowy 3



5. Rozważania na temat złożoności algorytmu

   Przy obliczaniu obwodu wykonujemy n obliczeń, przy przejściu przez wszystkich wierzchołki wykonujemy maksymalnie 2n przejść podczas których usuniemy maksymalnie n elementów ze ścieżki na skutek maksymalnie 2n+n porównań czy przekraczamy połowę obwodu. Nasza złożoność obliczeniowa nie powinna przekroczyć 4n obliczeń, co oznacza, że złożoność naszego algorytmu jest T(n)=n+2n+2n+n=6n=O(n). Optymistycznie nasza złożoność czasowa będzie wynosić T(n)=n+n+1+n+1+n=5n+2=O(n)

6. Problemy implementacyjne
   Wybrałem język C++ do napisania tego algorytmu, ponieważ był on najczęściej przeze mnie używany na przestrzeni studiów. Jednym z problemów, których miałem przy implementacji było sprawdzenie czy program nie wypisze wielokrotnie tych samych wyników. Sprawdzenie takie mogłoby zwiększyć złożoność naszego algorytmu. Na szczęście Prowadzący projektu, poinformował mnie, że wystarczy, że wypiszę pierwsze rozwiązanie. Kolejnym problemem była przenośność danego projektu. Jako, że został on napisany w c++, istniała możliwość, że nie uda się go odpalić na innej platformie. Jak się okazało, moje przypuszczenia były słuszne. Nie udało się go odpalić na innej platformie tylko i wyłącznie przy pomoc makefile w folderze cmake-build-relase. Postanowiłem, że skoro projekt jest na tyle prosty to zrobię w moim repozytorium dodatkowy folder o nazwie „release” gdzie umieszczę prosty makefile w raz z plikiem main.cpp i 3 plikami do testów.

Przy kolejnych projektach najprawdopodobniej użyję Javy z Maven’em, ponieważ te narzędzia w znacznym stopniu ułatwiają przenoszenie kodu na inne platformy. Ewentualnie można by dostosować cmake’a do wszystkich platform, co będzie wymagało znacznie więcej pracy.