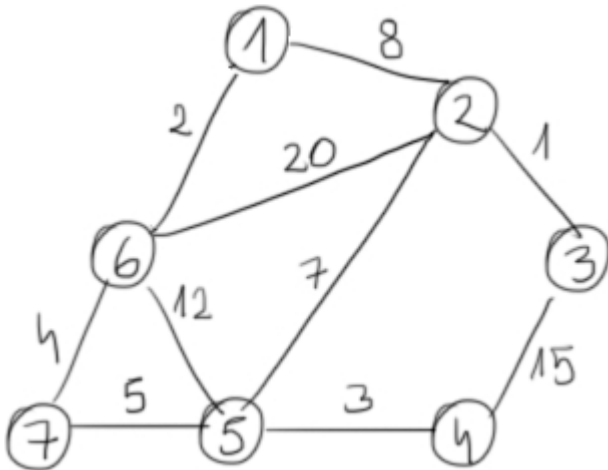


Na podstawie grafu nieskierowanego zadanego jako następująca lista krawędzi:  
 (1,2):8 (2,3):1 (3,4):15 (2,5):7 (4,5):3 (5,6):12 (1,6):2 (6,7):4 (2,6):20 (5,7):5

1. Rysunek:



2. Macierz sąsiedztwa:

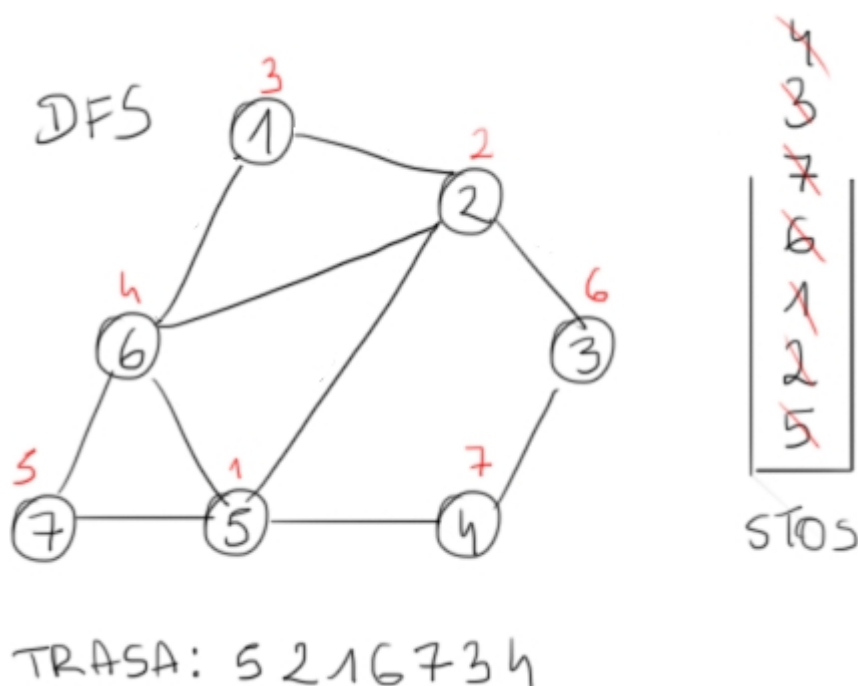
	1	2	3	4	5	6	7
1	0	8	-	-	-	2	-
2	8	0	1	-	7	20	-
3	-	1	0	15	-	-	-
4	-	-	15	0	3	-	-
5	-	7	-	3	0	12	5
6	2	20	-	-	12	0	4
7	-	-	-	-	5	4	0

### 3. Tablica list sąsiedztwa:

Wierzchołek	(sąsiad, waga)
1	(2,8) (6,2)
2	(1,8) (3,1) (5,7) (6,20)
3	(2,1) (4,15)
4	(3,15) (5,3)
5	(2,7) (4,3) (6,12) (7,5)
6	(1,2) (2,20) (5,12) (7,4)
7	(5,5) (6,4)

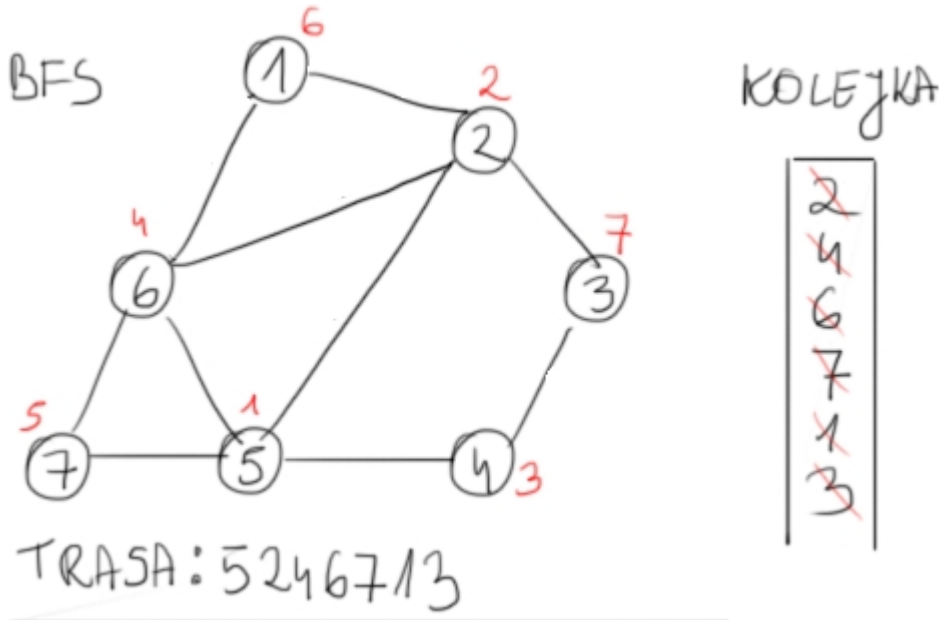
### 4. Zapisz kolejność odwiedzania wierzchołków przez algorytm DFS startując z wierzchołka 5.

- Bierzemy 5, oznaczamy ją jako odwiedzoną, wrzucamy ją na stos i do trasy.
- Przechodzimy do najmniejszego sąsiada, oznaczamy go jako odwiedzonego, wrzucamy go na stos i do trasy.
- Powtarzamy kroki. W razie wystąpienia sytuacji, w której wszyscy sąsiedzi danego elementu zostali odwiedzeni, ściągamy dany element ze stosu i rozpatrujemy jego sąsiadów pod względem wystąpienia nieodwiedzonych sąsiada. Na przykładzie:  
Jesteśmy na 7, nie mamy nieodwiedzonych sąsiadów, zdejmujemy więc ze stosu 7 i sprawdzamy czy jej sąsiedzi mają nieodwiedzonych sąsiadów, w tym przypadku dochodzimy do 3. Po odwiedzeniu 4, tj. wypełnieniu całej trasy (o czym póki co algorytm wie), sprawdzamy każdy element, tj. 3, 7, 6 itd. opróżniając stos. Po opróżnieniu stosu algorytm kończy działanie



5. Zapisz kolejność odwiedzania wierzchołków w algorytmie BFS startując z 5.

- Zaczynając od 5 (nie dodajemy jej do kolejki), wpisujemy do kolejki i trasy wszystkich nieodwiedzonych sąsiadów (od najmniejszego) tj. 2, 4, 6, 7
- Bierzymy pierwszy element z kolejki (FIFO) tj. 2 i powtarzamy kolejny krok, dodając do kolejki i do trasy kolejne elementy tj. 1, 3. Usuwamy element z kolejki.
- Przechodzimy do następnego elementu i powtarzamy poprzedni krok.
- Całość powtarzamy aż opróżnimy kolejkę.

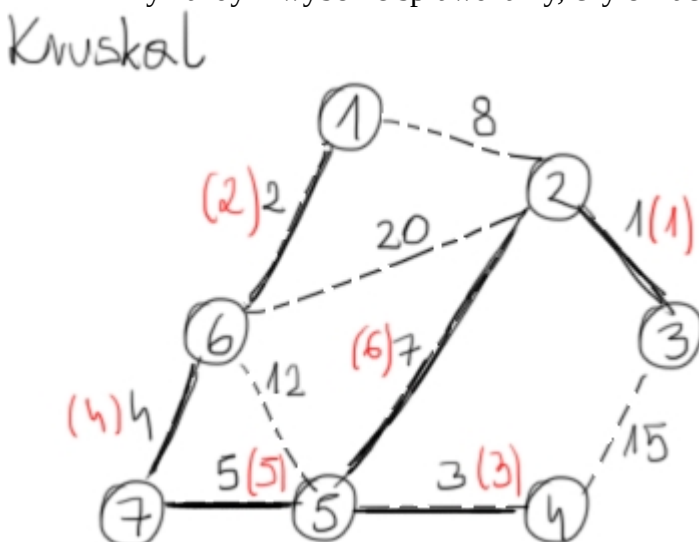


6. Zasymuluj Kruskala i zilustruj rysunkiem:

- liniami przerywanymi krawędzie nie należące do drzewa wynikowego
- liniami ciągłymi należące
- przy każdej krawędzi w nawiasie okrągłym podaj kolejność w jakiej

TIP: zacznij od stworzenia połączeń przerywanych, potem ew. zamień je na ciągłe

- Wybieramy najłżejsze połączenie, w tym przypadku (2,3):1
- Wybieramy kolejne najłżejsze połączenie
- Przy każdym wyborze sprawdzamy, czy oznaczając je jako ciągłe nie tworzy ono cyklu

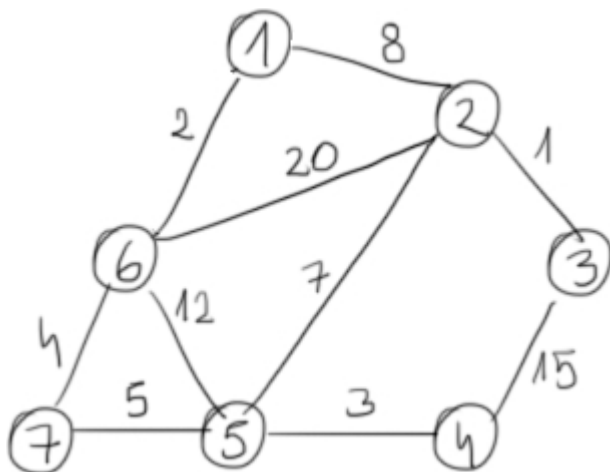


7. Zasymuluj działanie algorytmu Dijkstry z wierzchołka 3. Zapisz kroki algorytmu podając w każdym kroku:

- numer odwiedzanego wierzchołka
- wykonane w tym kroku operacje decrease\_key i odpowiednie zmiany w tablicy poprzedników (prev)
- wypisując jaka jest zawartość kolejki priorytetowej po wykonaniu kroku

Na końcu algorytmu dla każdego wierzchołka zapisz:

- odległość od wierzchołka startowego
- numer wierzchołka będącego poprzednikiem

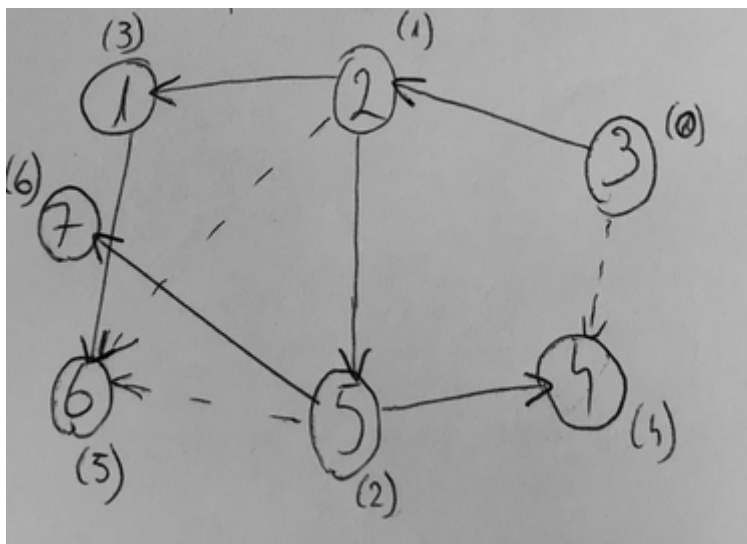


Krok	W	dec_k	prev	queue	1	2	4	5	6	7	W <sub>2</sub>	prev <sub>2</sub>	dł
0	3	2 4	-	3 2 14 5 6 7	∞	2(1)	4(15)	∞	∞	∞	1	2	0
1	2	1 5 6	3	2 1 6 7 5 4	1(9)	<del>2(1)</del>	4(15)	5(8)	6(21)	∞	2	3	1
2	5	4 6 7	2 3	5 6 7 1 4	1(9)	-	4(11)	<del>5(8)</del>	6(20)	7(13)	3	-	8
3	1	6	2 3	1 6 7 4	<del>1(9)</del>	-	4(11)	-	6(11)	7(13)	4	5	9
4	4	-	5 2 3	4 7 6	-	-	<del>4(11)</del>	-	6(11)	7(13)	5	2	11
5	6	-	1 2 3	6 7	-	-	-	-	<del>6(11)</del>	7(13)	6	1	20
6	7	-	5 2 3	7	-	-	-	-	-	<del>7(13)</del>	7	5	13

- W – wierzchołek
- dec\_k (decrease\_key) – sąsiedzi danego wierzchołka, bez poprzedzającego
- prev – najkrótsze możliwe dojście do danego wierzchołka, tj. jak do niego doszliśmy
- queue (+ elementy od 1 do 7 po prawej) – tj. kolejka priorytetowa. Bierzemy elementy sąsiadujące dla danego wierzchołka i określamy wagi, porównując je w dalszej części z kolejno otrzymanymi ich wers

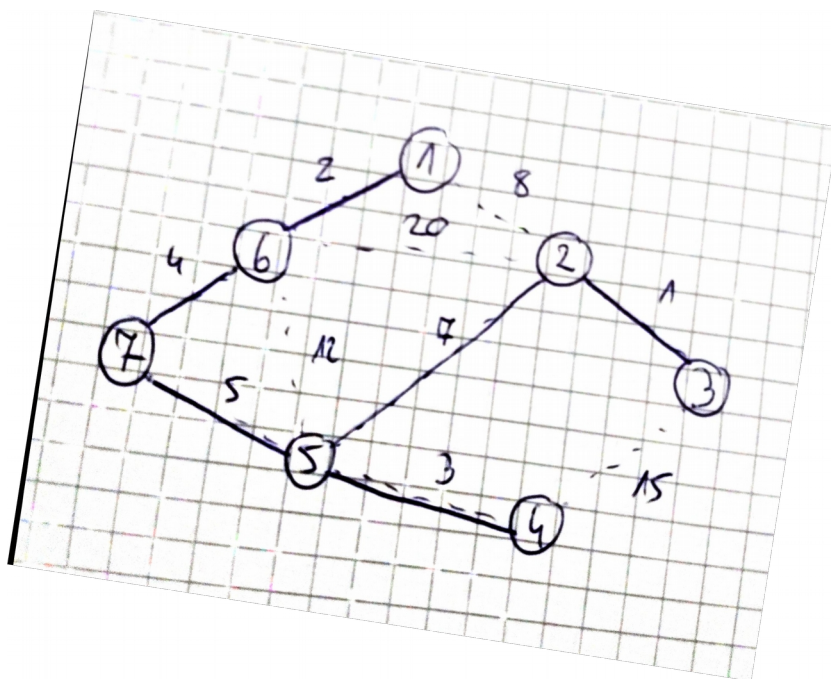
Algorytm zilustruj grafem, w którym:

- przy każdym wierzchołku będzie podany w nawiasie okrągłym numer kroku algorytmu, w którym wierzchołek został odwiedzony
- strzałkami ciągłymi oznaczone będą krawędzie należące do drzewa wynikowego
- przerywanymi - krawędzie, które w trakcie działania algorytmu wskazywały na poprzednika, jednak nie należą do drzewa wynikowego



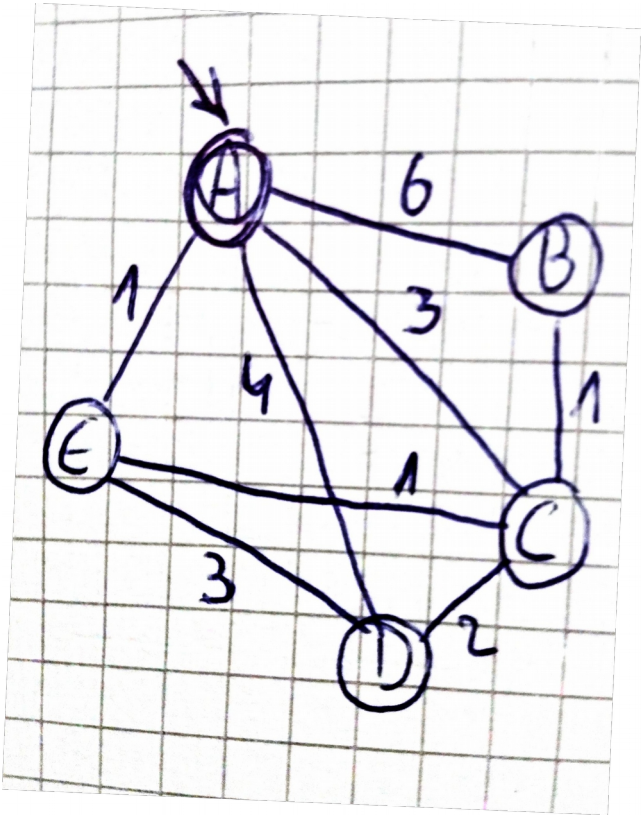
8. Zasymuluj Prima i zilustruj rysunkiem:

Zasady jak u góry



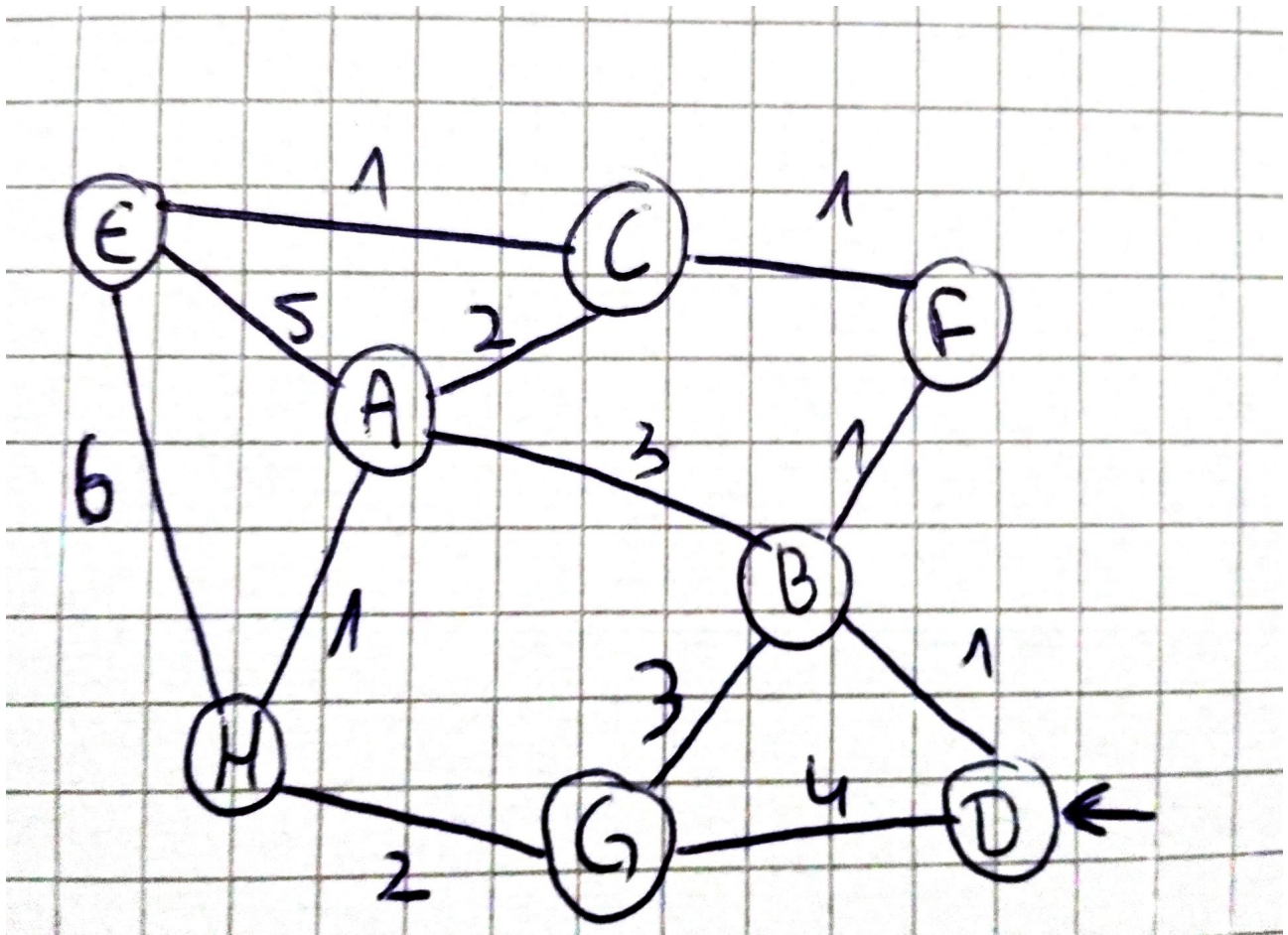
- Zaczynamy z wierzchołka 3
- Najkrótsza ścieżka prowadzi do 2
- Najkrótsza ścieżka z 2 prowadzi do 5
- Najkrótsza ścieżka z 5 prowadzi do 4
- Jakikolwiek ruch z 4 zamyka cykl, więc usuwamy ze stosu 4 i bierzemy element najniższej wagi dla 5, tj. 7
- Kontynuujemy aż do końca, unikając stworzenia cyklu.

DODATKOWE Dijkstra:



Zaczynamy z A

Krok	Kolejka	B	C	D	E
0	-	$\infty$ , -	$\infty$ , -	$\infty$ , -	$\infty$ , -
1	A B C D E	6 , A	3 , A	4 , A	1 , A
2	E C D B	6 , A	2 , E	4 , A	-
3	C D B	3 , C	-	4 , A	-
4	B D	-	-	4 , A	-
5	D	-	-	-	-
6	-	-	-	-	-



Zaczynamy D

LISTA	A	B	C	E	F	G	H
D	$\infty$	<u>1, D</u>	$\infty$	$\infty$	$\infty$	4, D	$\infty$
D B	4, B	-	$\infty$	$\infty$	<u>2, B</u>	4, D	$\infty$
D B F	4, B	-	<u>3, F</u>	$\infty$	-	4, D	$\infty$
D B F C	<u>4, B</u>	-	-	4, C	-	4, D	$\infty$
D B A	-	-	-	<u>4, C</u>	-	4, D	5, A
D B F C E	-	-	-	-	-	<u>4, D</u>	5, A
D G	-	-	-	-	-	-	<u>5, A</u>
D B A H	-	-	-	-	-	-	-