Wałcz dn ……

Imię i Nazwisko

SPRAWDZIAN ZALICZAJĄCY PRZEDMIOT ALGORYTY I STRUKTURY DANYCH

Kod modułu (przedmiotu) zawierający oznaczenie ISCED *Module (subject) code – ISCED included*	**kr.01.1, kr.01.2**
Kierunek studiów (nazwa studiów podyplomowych, kursu, szkolenia, itp.) *Programme*	**INFORMATYKA W BIZNESIE I ADMINISTRACJI**
Specjalność (jeśli dotyczy) *Speciality (if applicable)*	**Systemy informatyczne w zarządzaniu**
Profil studiów (jeśli dotyczy) *Programme profile(if applicable)*	**Profil praktyczny**
Poziom modułu (przedmiotu), zajęć (tu: studia pierwszego, drugiego stopnia; studia podyplomowe, kurs dokształcający, szkolenie, poziom pośredni, jeżeli ma zastosowanie)**;** *Study cycle* *(short/first/second/third), other education level/form)*	**Studia I stopnia, poziom 6 PRK**
Tryb (jeśli dotyczy) *Full time/part time (extramural)* *(if applicable)*	**stacjonarne**

Modułowy (przedmiotowy) SPRAWDZENIE EFEKTU KSZTAŁCENIA
K_W01
K_W02
K_W08
KU_04
KU_10
KU_13
K_K07

1. Zilustruj działanie procedury INSERT SORT dla podtablicy [31,41,59,26,41,58]

Pętlę główną rozpoczynamy od przedostatniej pozycji w zbiorze. Element na ostatniej pozycji jest zalążkiem listy uporządkowanej.

K01: Zaznaczamy pozycję listy uporządkowanej

31	41	59	26	41	58

K02: wybieramy 41 w miejscu 41 znajduje się puse miejsce które razem z liczbą będzie się przesówać

				41	
31	41	59	26		58

K03: 41 jest miejsze od 58 więc zostaje na miejcu przechodzimy do 26

			26		
31	41	59		41	58

K04: 26 jest miniejsze od 41 więc zostaje na miejscu przechodzimy do 59

		59			
31	41		26	41	58

K05: porównujemy 59 z liczbą 26 jest większe więc 59 zajmuje miejsce 26

			59		
31	41	26		41	58

K06: powtarzamy dla 41 i 58 poprzedni krok

				59	
31	41	26	41		58

31	41	26	41	58	59

K07: dla liczby 41 i 31 powtarzamy kroki tak jak dla 59

	41				
31		26	41	58	59

Po zamianie 41 z 26 następuje porównianie z 41 są równe więc nie zamieniają się miejscami

31	26	41	41	58	59

31					
	26	41	41	58	59

26	31	41	41	58	59

Odpowiedzią do zadania są same tabelki nie musicie robić opisów.

Dopisuję wersję drugą (którą on może wymagać) w momęcie jak ustawiamy się na pierwszym elemencie.

K01: Zaznaczamy pozycję listy uporządkowanej

31	41	59	26	41	58

K02: sprawdzamy czy 41 jest mniejsze od 31 nie jest więc zostawiamy

31	41	59	26	41	58

K03: sprawdzamy czy 59 jest mniejsze od 41 nie jest więc zostawiamy

31	41	59	26	41	58

K04: sprawdzamy czy 26 jest mniejsze od 59 jest więc zamieniamy miejscami

		26			
31	41		59	41	58

K05: Dalej sprawdzamy czy 26 jest mniejsze od 41 i 31

	26				
31		41	59	41	58

26	31	41	59	41	58

K06: sprawdzamy czy 41 jest mniejsze od 59 jest więc zamieniamy

			41		
26	31	41		59	58

K07: sprawdzamy czy 41 jest mniejsze od 41 są równe więc zostawiamy

26	31	41	41	59	58

K08: sprawdzamy czy 58 jest mniejsze od 59

				58	
26	31	41	41		59

K09: ostatnim krokiem jest sprawdzenie czy 58 jest mniejsze od 41 nie jest więc zostawiamy i zakańczamy algorytm

26	31	41	41	58	59

2. Zilustruj działanie algorytmu sortowanie przez scalanie dla podtablicy [3,41,52,26,38,59,9,49]

W tym algorytmie dzielimy wszystko na pół do puki już się nie da i porządkujemy pojedyńcze elementy aż scalimy całą tablicę.

K01: pierwszy podział

3	41	52	26	38	59	9	49

K02: drugi podział

3	41	52	26	38	59	9	49

K03: trzeci podział

3	41	52	26	38	59	9	49

Otrzymujemy pojedyńcze elementy które możemy łatwo posortować

K04: pierwsze sortowanie i scalenie

3	41	26	52	38	59	9	49

K05: kolejne scalenie

3	26	41	52	9	38	49	59

K06: ostatnie scalenie

3	9	26	38	41	49	52	59

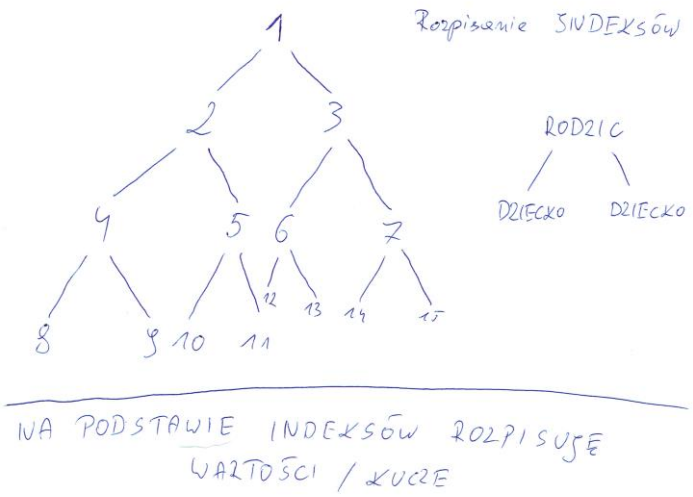
Otrzymujemy posortowaną tablicę

3. Czy ciąg (23, 17, 14, 6, 13, 10, 1, 5, 7, 12) jest kopcem typu max?

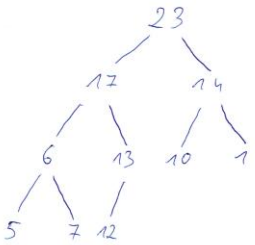
Wyróżniamy dwa rodzaje kopców binarnych: kopce binarne typu max w których wartość danego węzła niebędącego korzeniem jest zawsze mniejsza niż wartość jego rodzica oraz kopce binarne typu min w których wartość danego węzła niebędącego korzeniem jest zawsze większa niż wartość jego rodzica. To znaczy że przy max wartość dzieci musi być zawsze mniejsza od rodzica, natomiast przy min jest na odwrót.

Indeks	1	2	3	4	5	6	7	8	9	10
Wartość/Klucz	23	17	14	6	13	10	1	5	7	12

Według mnie ten ciąg nie może być kompcem typu max, ponieważ watość 9 która posiada indeks 9 jest większa od watości na indeksie 4 czyli 6. Nie wiem czy on się pomylił, jeśli tak ma być to odpowiedź to: nie jest; a uzasadnienie bo 7 jest większe od 6.



4. Zilustruj działanie procedury MAX-HEAPIFY dla tablicy A = (27, 17, 3, 16, 13, 10, 1, 5, 7, 12,4, 8, 9,0).



Budujemy drzewo typu max czyli rodzić musi być większy od dziecka, jeśli dziecko jest większe zamieniamy z rodzicem. Ustawiamy się na 27 i lecimy po kolei.

Indeks	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Wartość/Klucz	27													

Indeks	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Wartość/Klucz	27	17												

Indeks	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Wartość/Klucz	27	17	3											

Indeks	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Wartość/Klucz	27	17	3	16										

Indeks	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Wartość/Klucz	27	17	3	16	13									

Indeks	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Wartość/Klucz	27	17	3	16	13	10								

Indeks	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Wartość/Klucz	27	17	10	16	13	3								

Indeks	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Wartość/Klucz	27	17	10	16	13	3	1							

Indeks	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Wartość/Klucz	27	17	10	16	13	3	1	5						

Indeks	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Wartość/Klucz	27	17	10	16	13	3	1	5	7					

Indeks	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Wartość/Klucz	27	17	10	16	13	3	1	5	7	12				

Indeks	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Wartość/Klucz	27	17	10	16	13	3	1	5	7	12	4			

Indeks	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Wartość/Klucz	27	17	10	16	13	3	1	5	7	12	4	8		

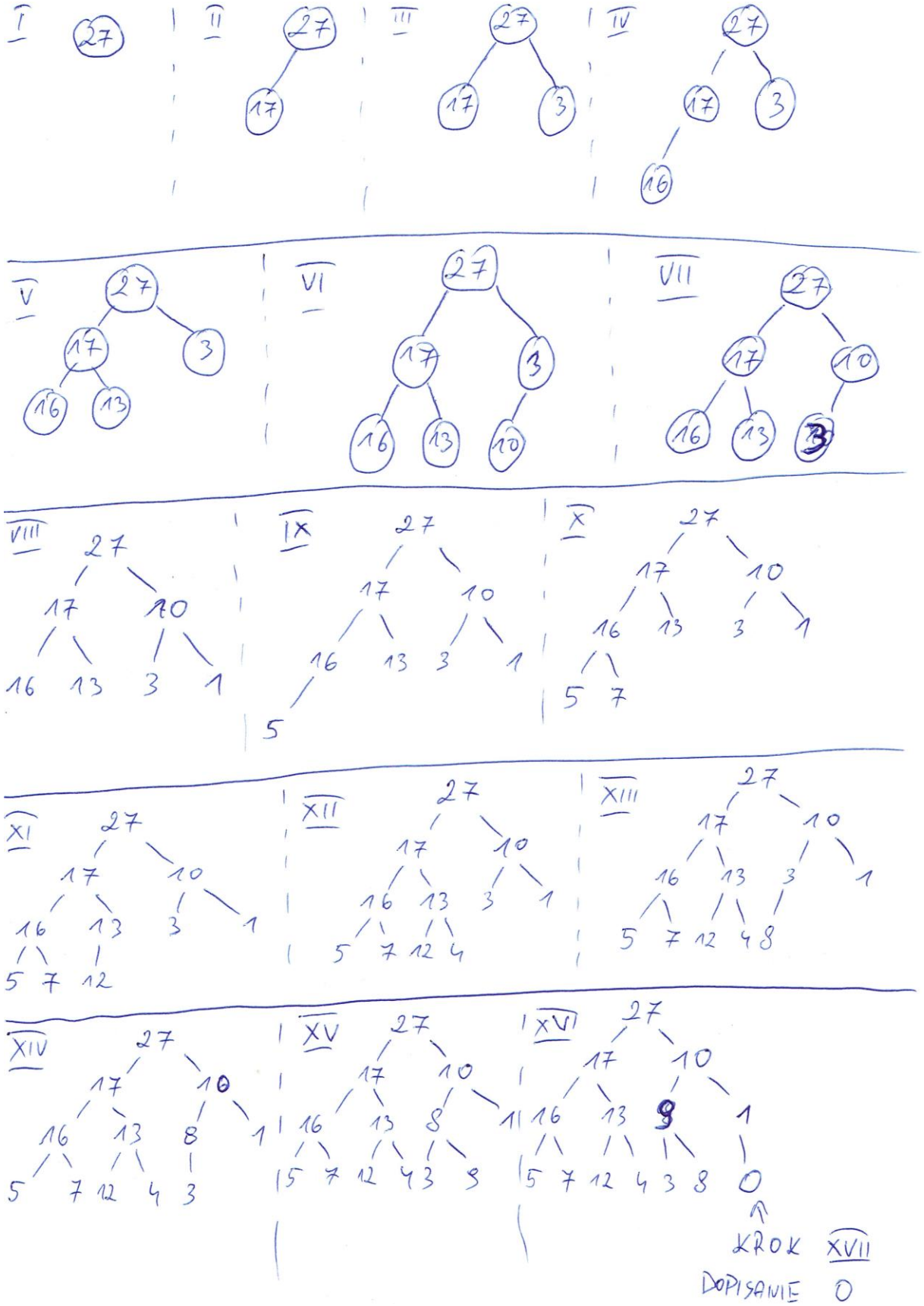
Indeks	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Wartość/Klucz	27	17	10	16	13	8	1	5	7	12	4	3		

Indeks	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Wartość/Klucz	27	17	10	16	13	8	1	5	7	12	4	3	9	

Indeks	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Wartość/Klucz	27	17	10	16	13	9	1	5	7	12	4	3	8	

Indeks	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Wartość/Klucz	27	17	10	16	13	9	1	5	7	12	4	3	8	0

Tak to wygląda na tablicach. Podspodem ilustracja na drzewach



5. Zilustruj działanie procedury COUNTING-SORT dla tablicy A = (6,0,2,0,1,3,4,6, 1,3,2).

Jest to algorytm sortowania przez zliczanie, czyli zliczamy ile razy dana wartość wystapiła, usuwany indeksy o wartościach 0 mamy posortowaną tablicę.

Szukamy nawyszą liczbę w zbiorze A jest to 6 (polecam algorytm min-max do tego jak by się czepiał dlaczego tak, opisany w notatkach)

Rysujemy tablicę 7:2 ponieważ zawsze zaczynamy od 0 a mamy skończyć na 6. Krotność ustawiamy na 0 przechodzimy przez tablicę A i dodajemy do krotności +1 zgodnej z indeksem

A – główna tablica; Indeks/Krotność – tablica pomocnicza; B - tablica wyjściowa (wynik)

Indeks	0	1	2	3	4	5	6
Krotność	0	0	0	0	0	0	0

K01:

A	6	0	2	0	1	3	4	6	1	3	2

Indeks	0	1	2	3	4	5	6
Krotność	0	0	0	0	0	0	1

K02:

A	6	0	2	0	1	3	4	6	1	3	2

Indeks	0	1	2	3	4	5	6
Krotność	1	0	0	0	0	0	1

K03:

A	6	0	2	0	1	3	4	6	1	3	2

Indeks	0	1	2	3	4	5	6
Krotność	1	0	1	0	0	0	1

K04:

A	6	0	2	0	1	3	4	6	1	3	2

Indeks	0	1	2	3	4	5	6
Krotność	2	0	1	0	0	0	1

K05:

A	6	0	2	0	1	3	4	6	1	3	2

Indeks	0	1	2	3	4	5	6
Krotność	2	1	1	0	0	0	1

K06:

A	6	0	2	0	1	3	4	6	1	3	2

Indeks	0	1	2	3	4	5	6
Krotność	2	1	1	1	0	0	1

K07:

A	6	0	2	0	1	3	4	6	1	3	2

Indeks	0	1	2	3	4	5	6
Krotność	2	1	1	1	1	0	1

K08:

A	6	0	2	0	1	3	4	6	1	3	2

Indeks	0	1	2	3	4	5	6
Krotność	2	1	1	1	1	0	2

K09:

A	6	0	2	0	1	3	4	6	1	3	2

Indeks	0	1	2	3	4	5	6
Krotność	2	2	1	1	1	0	2

K10:

A	6	0	2	0	1	3	4	6	1	3	2

Indeks	0	1	2	3	4	5	6
Krotność	2	2	1	2	1	0	2

K11:

A	6	0	2	0	1	3	4	6	1	3	2

Indeks	0	1	2	3	4	5	6
Krotność	2	2	2	2	1	0	2

Teraz usuwany z tablicy kolumnę z ktornościmi 0

Indeks	0	1	2	3	4	5	6
Krotność	2	2	2	2	1	0	2

Indeks	0	1	2	3	4	6
Krotność	2	2	2	2	1	2

Wracamy na początek tablicy i idąc po indeksach wypisujemy krotność tej liczby np. 0 ma krotność 2 czyli będzie 0, 0

Indeks	0	1	2	3	4	6
Krotność	2	2	2	2	1	2

B	0	0									

Indeks	0	1	2	3	4	6
Krotność	2	1	2	2	1	2

B	0	0	1	1							

Indeks	0	1	2	3	4	6					
Krotność	2	1	2	2	1	2					
B	0	0	1	1	2	2					

Indeks	0	1	2	3	4	6					
Krotność	2	1	2	2	1	2					
B	0	0	1	1	2	2	3	3			

Indeks	0	1	2	3	4	6					
Krotność	2	1	2	2	1	2					
B	0	0	1	1	2	2	3	3	4		

Indeks	0	1	2	3	4	6					
Krotność	2	1	2	2	1	2					
B	0	0	1	1	2	2	3	3	4	6	6

Otrzymujemy posortowaną tablicę

6. Zilustruj wynik wykonania ciągu operacji PUSH(S, 4), PUSH(S, 1), PUSH(S, 3), POP(S), PUsH(S, 8) oraz PoP(S) na początkowo pustym stosie S znajdującym się w tablicy S[l . .6].

Indeks 1 jest dołem stosu. Zawsze dokładamy na górę oraz zabieramy z góry. Operacja PUSH jest dołożenie elementu do stosu, natomiast operacja POP jest to usunięcie elementu ze stosu.

PUSH(S,4)

Indeks	1	2	3	4	5	6
S	4					

PUSH(S,1)

Indeks	1	2	3	4	5	6
S	4	1				

PUSH(S,3)

Indeks	1	2	3	4	5	6
S	4	1	3			

POP(S)

Indeks	1	2	3	4	5	6
S	4	1				

PUSH(S,8)

Indeks	1	2	3	4	5	6
S	4	1	8			

POP(S)

Indeks	1	2	3	4	5	6
S	4	1				

7. Zilustruj wynik wykonania ciągu operacji ENQUEUE(Q,4), ENQUEUE(Q, 1), ENQUEUE(Q,3), DEQUEUE(Q), ENQUEUE(Q, 8) oraz DEQUEUE(Q) na początkowo pustej kolejce Q znajdującej się w tablicy Q[l . . 61.

Kolejka idzie w stronę indeksu 1 (tłumacząc na jego język głowa jest na 1 a bliżej 6) ENQUEUE jest to operacja dodania elementu do kolejki, DEQUEUE jest to operacja usunięcia elementu z kolejni. Głowa Ogon

Według mnie

ENQUEUE(Q,4)

Indeks	1	2	3	4	5	6
Q	4					

ENQUEUE(Q,1)

Indeks	1	2	3	4	5	6
Q	4	1				

ENQUEUE(Q,3)

Indeks	1	2	3	4	5	6
Q	4	1	3			

DENQUEUE(Q)

Indeks	1	2	3	4	5	6
Q	1	3				

ENQUEUE(Q,8)

Indeks	1	2	3	4	5	6
Q	1	3	8			

DENQUEUE(Q)

Indeks	1	2	3	4	5	6
Q	3	8				

Według niego

ENQUEUE(Q,4)

Indeks	1	2	3	4	5	6
Q	4					

ENQUEUE(Q,1)

Indeks	1	2	3	4	5	6
Q	4	1				

ENQUEUE(Q,3)

Indeks	1	2	3	4	5	6
Q	4	1	3			

DENQUEUE(Q)

Indeks	1	2	3	4	5	6
Q		1	3			

ENQUEUE(Q,8)

Indeks	1	2	3	4	5	6
Q		1	3	8		

DENQUEUE(Q)

Indeks	1	2	3	4	5	6
Q			3	8		

8. Zilustruj ciąg wstawień elementów o kluczach 5, 28, 19, 15, 20, 33, 12,17, 10 do tablicy z haszowaniem, wykorzystując do rozwiązywania kolizji metodę łańcuchową. Przyjmijmy, że

tablica zawiera 9 pozycji, a h(k) = k mod 9 jest funkcją haszującą.

Dane wejściowe wiemy że ciąg A ma postać

A	5	28	19	15	20	33	12	17	10

Mamy stworzyć tablicę z haszowaniem. To znaczy mamy stworzyć tablicę wielowymiarową (dodając metodę łańcuchową ma to być tablica wielowymiarowa ze wskaźnikami) do której będziemy przypisywać odpowiednie elementy do konkretnych indeksów. To gdzie będzie przypisywany dany element zalerzy od funkcji haszującej która ma w tym przypadku postać h(k) = k mod 9 gdzie h to nr indeksu a k to element który mamy ułozyć/włożyć. Mod czyli modulo zwraca nam resztę z dzielenia (najlepiej zaorkąglać w dół do jednego miejsca po przecinku). Dodatkowo mamy wykorzystać metodę łańcuchową ktura jest podobna do listy z dowiązaniami to znaczy że dany element ma wskazywać na swojego poprzednika oraz następnika. Tablica haszowana ma mieć tylko 9 pozycji:

TH	0	1	2	3	4	5	6	7	8

Warto rozpocząć od obliczenia sobie watości h z funkcji haszującek która ma nam później pokazać gdzie jakie elementy mamy włożyć.

5 mod 9 = 5 (5/9 = 0,55555 zostawiamy jedno miejsce po przecinku zaokrąglając w dół czyli 5/9 = 0,5 tak więc reszta to 5)

28 mod 9 = 1

19 mod 9 = 1

15 mod 9 = 6

20 mod 9 = 2

33 mod 9 = 6

12 mod 9 = 3

17 mod 9 = 8

10 mod 9 = 1

K01: Wstawiamy liczbę 5 pod indeks 5

Indeks/TH	
0	
1	
2	
3	
4	
5	
6	
7	
8	

→ | / | 5 | /

K02: Wstawiamy liczbę 28 pod indeks 1

Indeks/TH	
0	
1	
2	
3	
4	
5	
6	
7	
8	

→ | / | 28 | /

→ | / | 5 | /

K03: Wstawiamy liczbę 19 pod indeks 1

Indeks/TH	
0	
1	
2	
3	
4	
5	
6	
7	
8	

→ | / | 28 | | ↔ | | 19 | /

→ | / | 5 | /

K04: Wstawiamy liczbę 15 pod indeks 6

Indeks/TH	
0	
1	
2	
3	
4	
5	
6	
7	
8	

→ | / | 28 | | ↔ | | 19 | /

→ | / | 5 | /

→ | / | 15 | /

K05: Wstawiamy liczbę 20 pod indeks 2

Indeks/TH	
0	
1	
2	
3	
4	
5	
6	
7	
8	

→ | / | 28 | | ↔ | | 19 | /

→ | / | 20 | /

→ | / | 5 | /

→ | / | 15 | /

K06: Wstawiamy liczbę 33 pod indeks 6

Indeks/TH	
0	
1	
2	
3	
4	
5	
6	
7	
8	

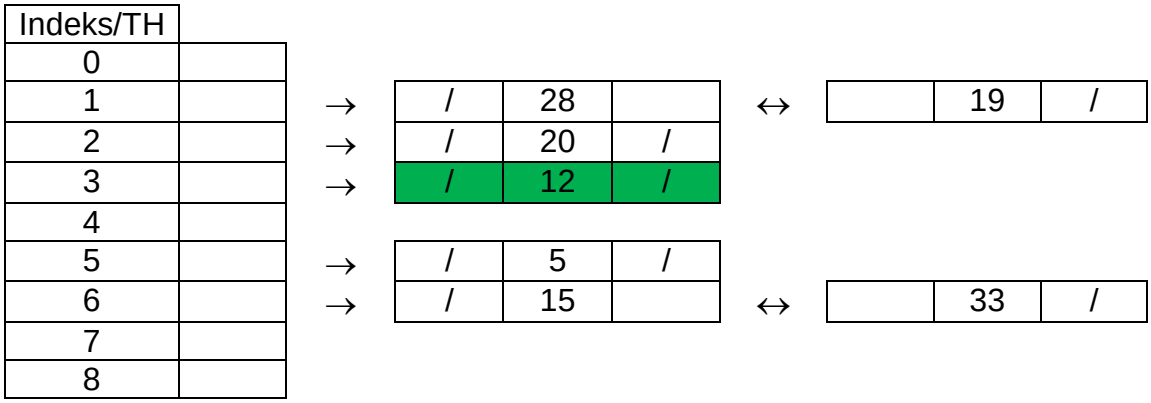
→ | / | 28 | | ↔ | | 19 | /

→ | / | 20 | /

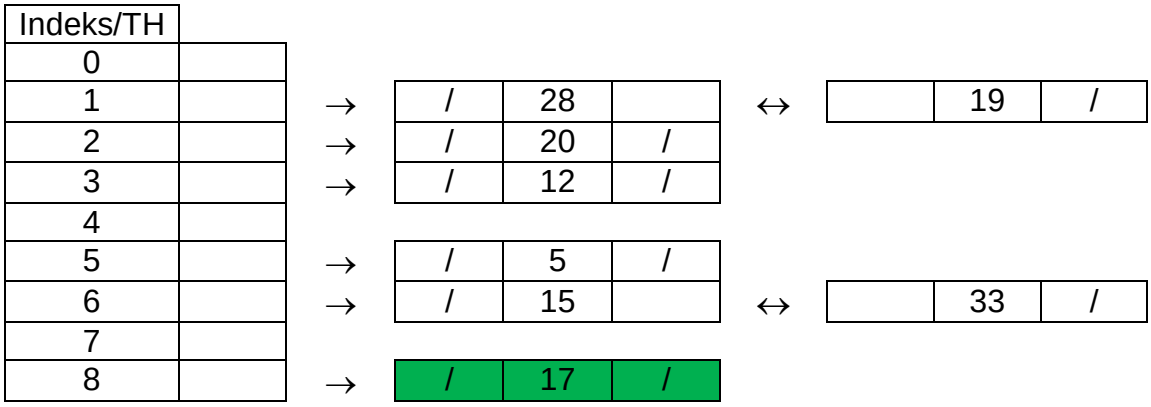
→ | / | 5 | /

→ | / | 15 | | ↔ | | 33 | /

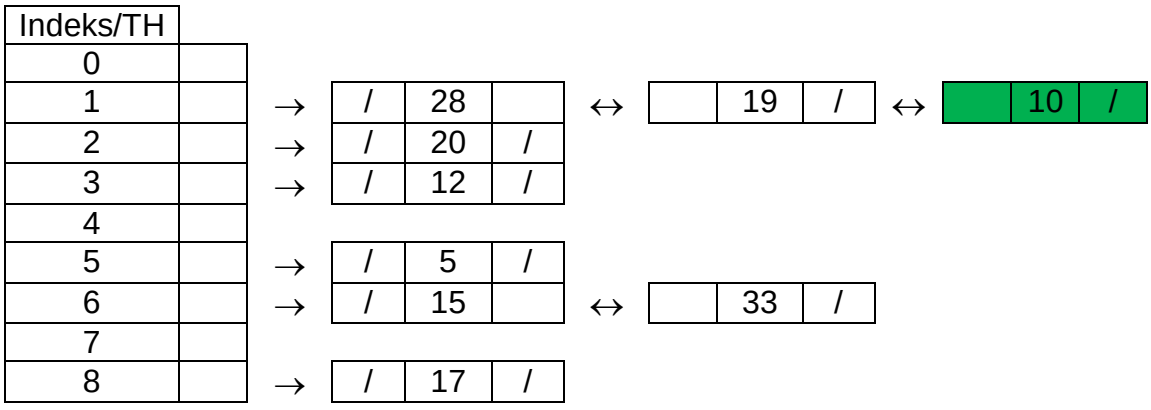
K07: Wstawiamy liczbę 12 pod indeks 3



K08: Wstawiamy liczbę 17 pod indeks 8

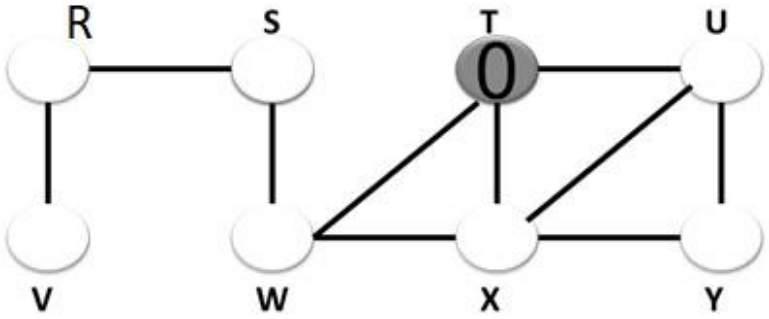


K09: Wstawiamy liczbę 10 pod indeks 1

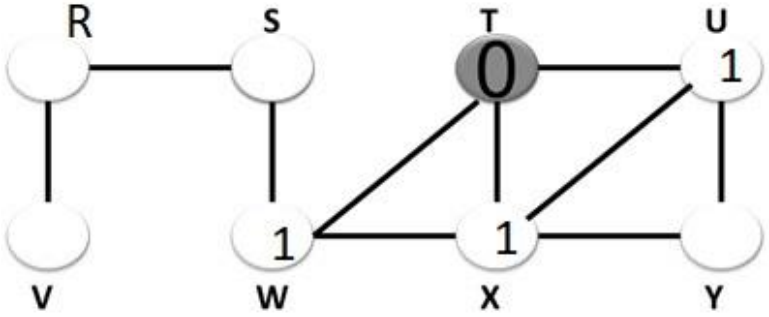


9. Zbadaj Graf algorytmem przechodzenia WSZERZ :

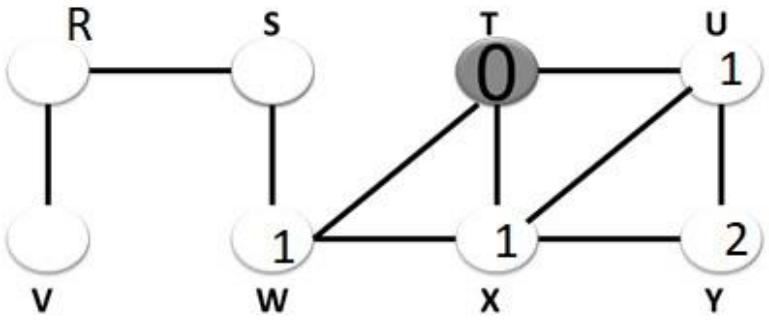
Pamiętajcie że kolejność literek wykonywanych jest taka sama jak włoży się do kolejki (piorytetowej) czyli pierwsze wchodzi pierwsze wychodzi założyłem że u mnie kolejka ma postać R, S, T, U, W, V, X, Y, Z



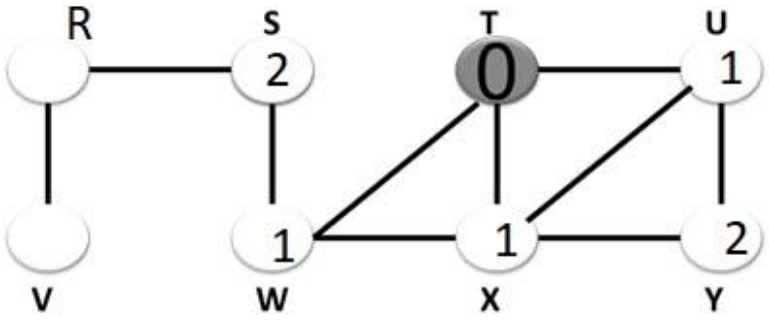
T						
0						



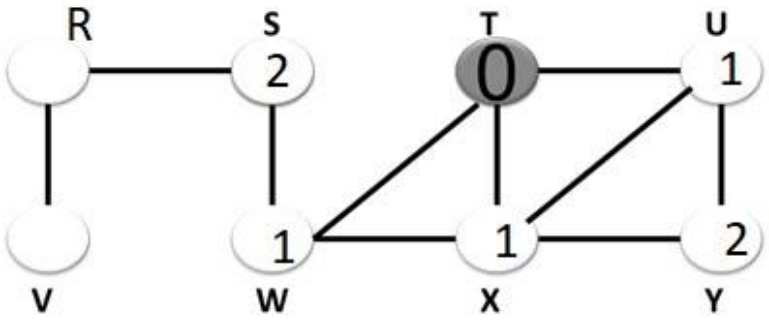
U	W	X				
1	1	1				



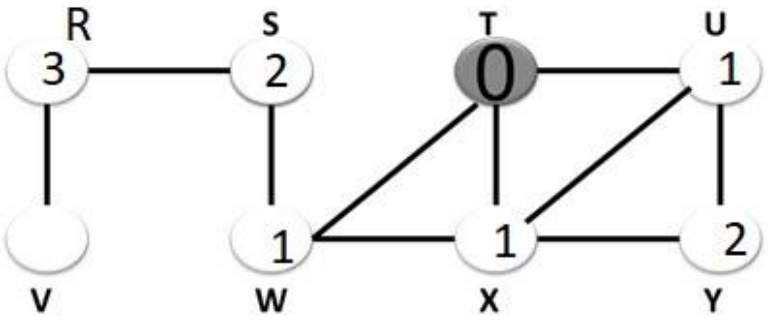
Y	W	X				
2	1	1				



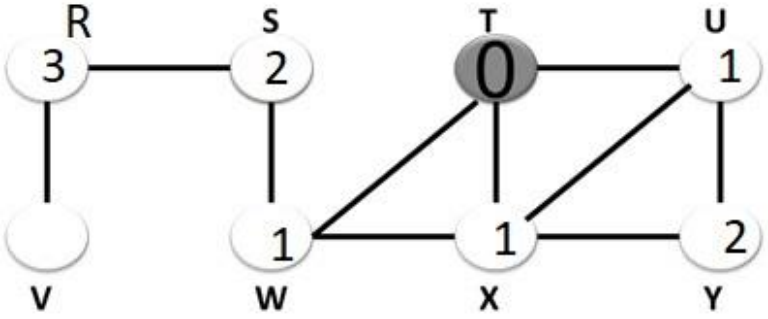
Y	S	X				
2	2	1				



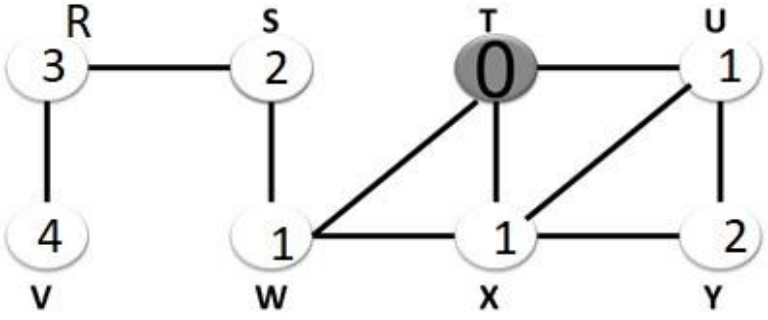
Y	S					
2	2					



Y	R					
2	3					



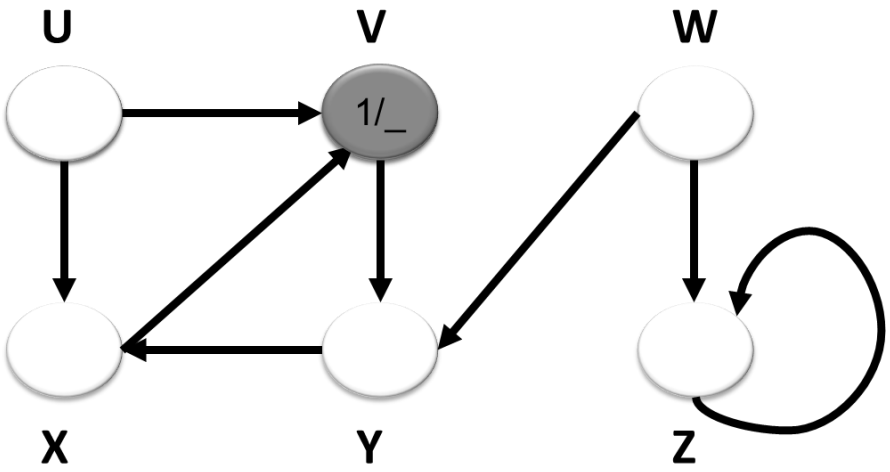
R						
3						

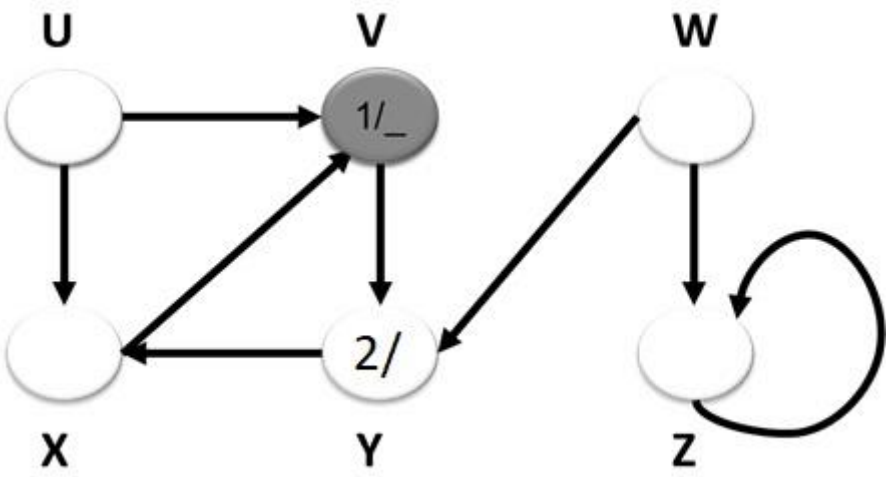


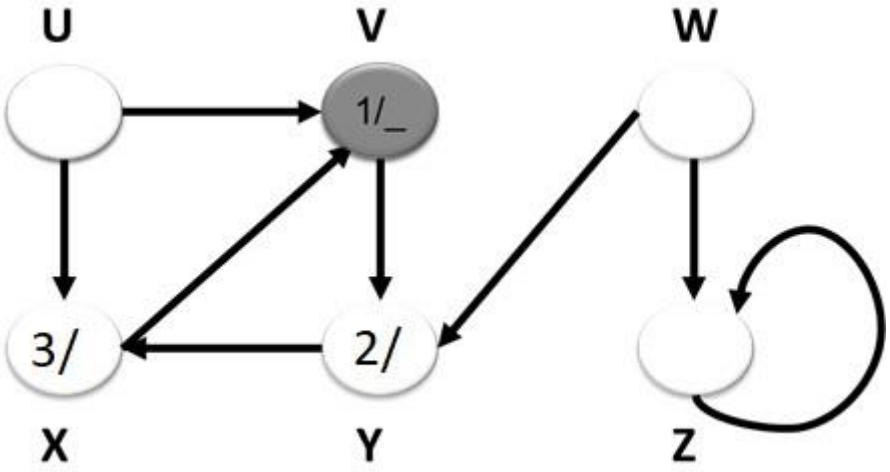
V						
4						

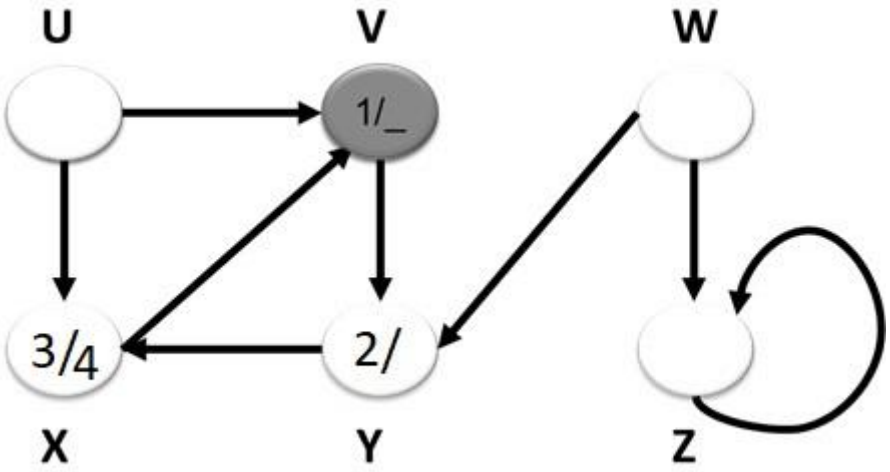
10. Zbadaj Graf algorytmem przechodzenia wgląb:

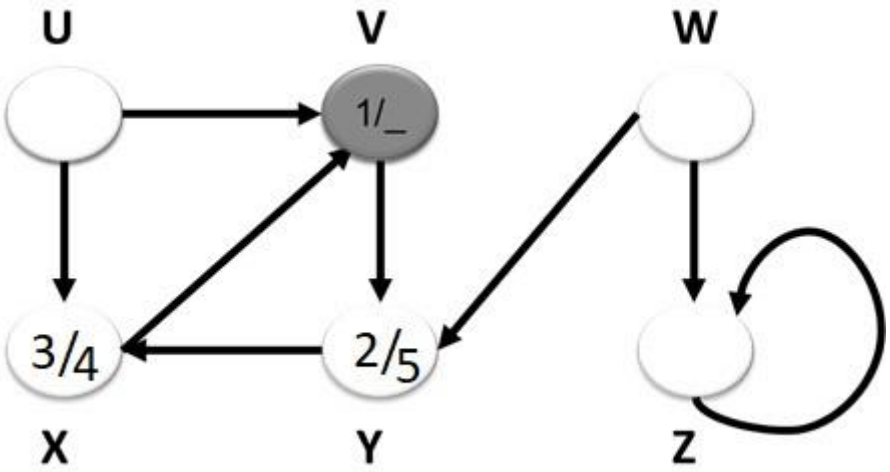
To samo co w poprzednim zadaniu potrzebna jest kolejka. Polega na tym aby przeszukiwać w głąb grafu jedną ścieżkę jeśli dotrzemy do odwiedzonego wierzchołka lub końca grafu to wracamy po śladach aby sprawdzić inne ścieżki. W momencie kiedy dochodzimy że dana ścieżka nie ma już więcej dróg wybieramy kolejną ścieżkę o najdłuższej drodze i tak aż odwiedzimy wszystkie wierzchołki.

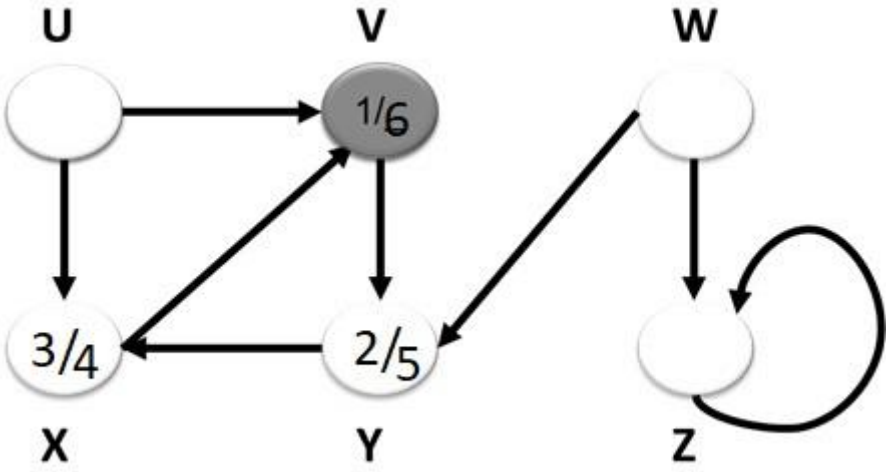


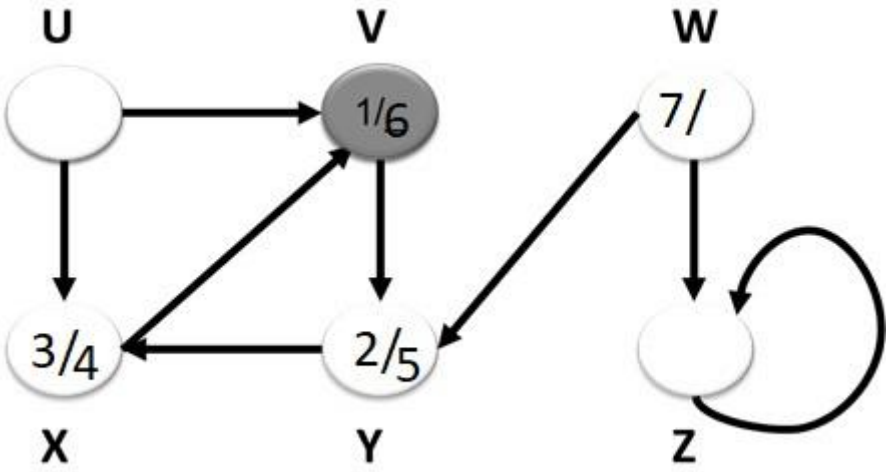


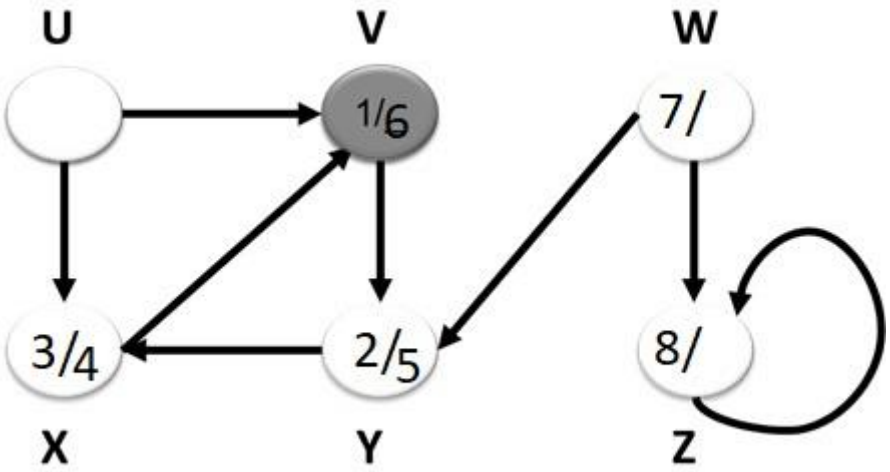

U
V
W
1/_
3/
2/
X
Y
Z

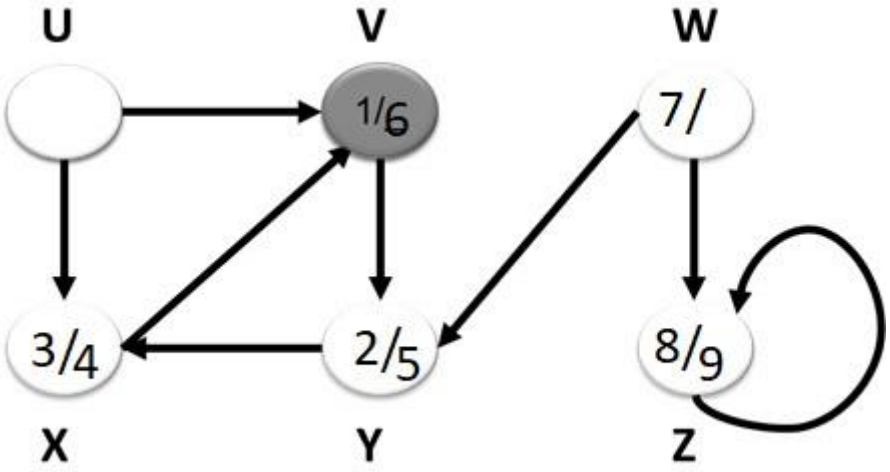

U
V
W
1/_
3/4
2/
X
Y
Z

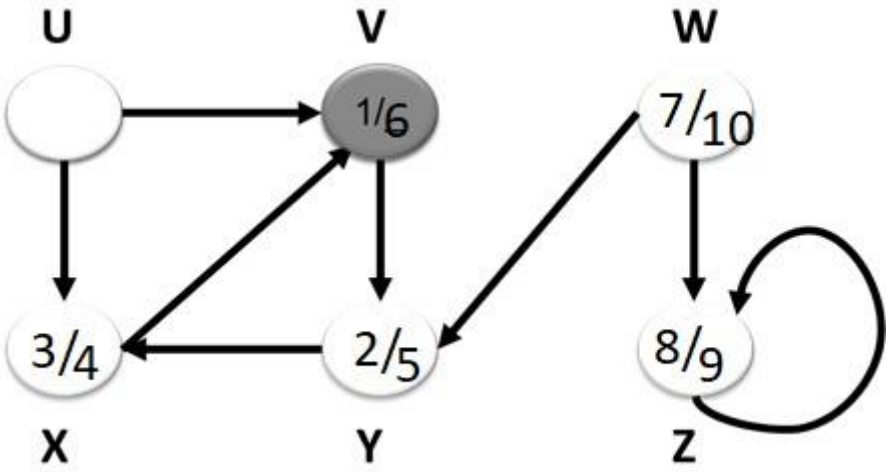

U
V
W
1/_
3/4
2/5
X
Y
Z

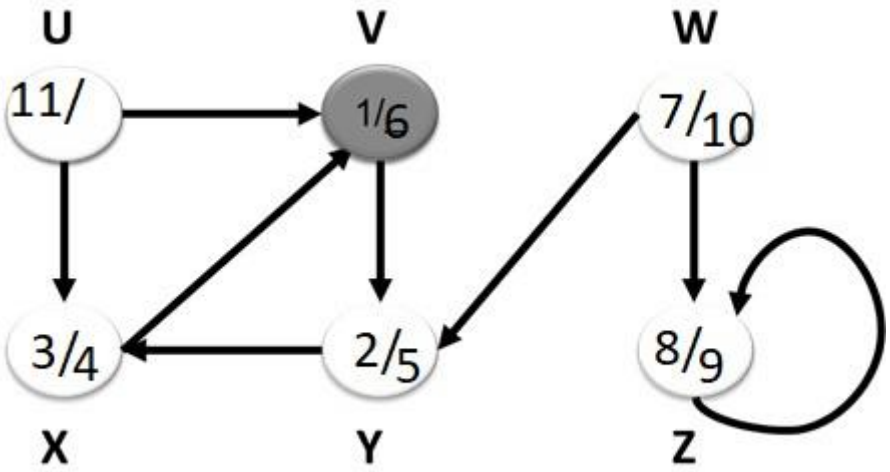

U
V
W
1/6
3/4
2/5
X
Y
Z

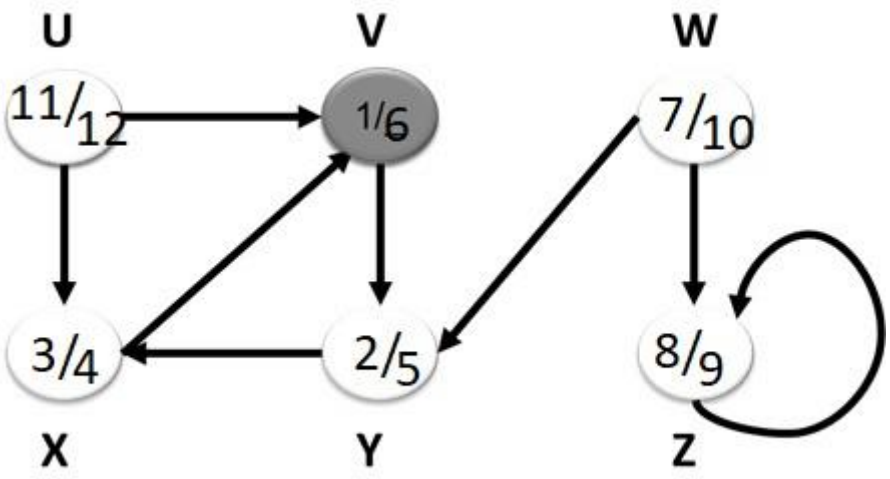

U
V
W
1/6
7/
3/4
2/5
X
Y
Z


U
V
W
1/6
7/
3/4
2/5
8/
X
Y
Z


U
V
W
1/6
7/
3/4
2/5
8/9
X
Y
Z


U
V
W
1/6
7/10
3/4
2/5
8/9
X
Y
Z

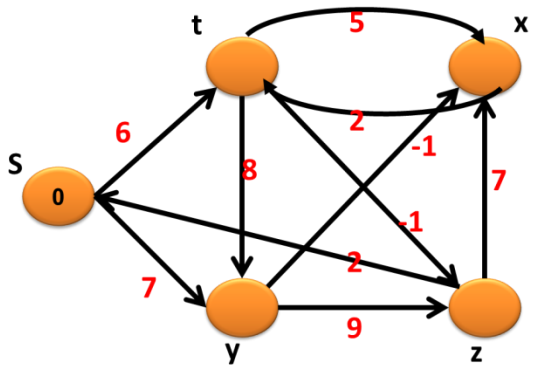

U
V
W
11/
1/6
7/10
3/4
2/5
8/9
X
Y
Z



Zadanie 11 i 12 przerabiałem wcześniej są do pobrania odpowiednio FORD.pdf i DIJKSTRA 2.pdf

11. Zbadaj Graf algorytmem BELLMANA FORDA:

Przerabialiśmy to już wcześniej jako zadanie domowe macie to w pliku FORD.pdf



12. Zbadaj Graf algorytmem DIJKSTRY:

Przerabialiśmy to już wcześniej jako zadanie domowe macie to w pliku DIJKSTRA 2.pdf