

### Zad 3

Idea: W każdym kroku od utamka  $\frac{a_i}{b_i}$  odejmujemy największy możliwy utamek postaci  $\frac{1}{x}$  otrzymując nowy utamek  $\frac{a_{i+1}}{b_{i+1}}$ . Powtarzamy tak dugo, jak  $a_{i+1} \neq 1$ .

$$\begin{array}{l} w = \emptyset \\ i = 1 \end{array}$$

while  $a \neq 1$ :

if  $\frac{a}{b} - \frac{1}{i} > 0$   
w.append( $\frac{1}{i}$ )  
 $a = a \cdot i - b$   
 $b = b \cdot i$

i++

w.append( $\frac{a}{b}$ )

return w

Algorytm zawsze daje rozwiązańie:

Zauważmy, że kolejne wartości jakie są w a tworzą ciąg nieskończony:

$\frac{a}{b} < 2 \cdot \frac{1}{n}$ . Gdyby tak nie było, to  $\frac{a}{b} = \frac{1}{n} \Rightarrow a = 1$ .

$$\frac{1}{n-1} > \frac{a}{b} > \frac{1}{n}$$

$$\frac{a}{b} < \frac{1}{n-1} \Rightarrow \frac{a}{b} - \frac{1}{n} < \frac{1}{n-1} - \frac{1}{n} \Rightarrow$$

$$\Rightarrow \frac{a}{b} - \frac{1}{n} = \frac{an - b}{bn} = \frac{a'}{b'}$$

$$\frac{1}{n-1} > \frac{a}{b} \Rightarrow a(n-1) < b \Rightarrow a' < an - b < a$$

W końcu dojdziemy do  $a_i = 1$ .

Algorytm nie musi dawać rozwiązania optymalnego:

$$\frac{9}{20} = \frac{1}{3} + \frac{1}{9} + \frac{1}{180} = \frac{1}{4} + \frac{1}{5}$$

## Zad 2

Idea: Posortujemy odcinki względem punktu ich końca, a następnie będziemy dodawali odcinki „od lewej”, o ile początek dodawanego odcinka zaczyna się dalej lub w tym samym miejscu, co kończy poprzedni.

$S \leftarrow$  zbiór odcinków

$S \leftarrow$  posortuj względem punktu końca

$T = \emptyset$

$k = -\infty, i = 0$

Powtaraj n razy:

Jeśli  $S[i]$  zaczyna się dalej lub

w tym samym miejscu co  $k$

Dodaj  $S[:i]$  do  $T$

$k = S[i].k$

Zwróć  $T$

$O(nlgn)$

przez sortowanie

## Zad 5

Pokażemy, że w dowolnym kroku algorytmu nie powstanie cyklu.  
Załóżmy niewprost, że w jakimś kroku powstanie cyku  $C$ , składającego się z wienchołków  $v_1, v_2, \dots, v_n$  i krawędzi  $e_1, e_2, \dots, e_n$ .

Krawędź  $e_n$  łączy wienchołek  $v_n$  z  $v_1$ . Algorytm dla kolejnych wienchołków  $v_i$  wybiera krawędzie  $e_i$ . Oznacza to, że

$$w(e_1) > w(e_2) > \dots > w(e_n)$$

Jeżeli algorytm wybrał  $e_n$  która łączy  $v_n$  z  $v_1$ , podczas gdy  $v_1$  ma też krawędź  $e_1$ . To by musiało znaczyć, że

$$w(e_n) < w(e_1)$$

A to daje spójność.

Każdy "wienchołek" powstający po scaleniu wienchołków w spojną składową w poprzednich wienchołkach, jest MST.

Podstawa indukcji: Pojedynczy wienchołek jest MST.

Krok indukcyjny: Założymy, że  $n$ -elementowa spojna składowa tworzy MST. Wtedy dodanie do niej najlżejszej krawędzi nietwórnego cyklu tworzą MST poniększone o jeden wienchołek.

Spojna składowa  $S_1$  jest MST dla swoich wienchołków. Fazując ją, z innej spojnej składowej  $S_2$  najlżejszym wienchołkiem wychodzącym poza  $S_1$  otrzymujemy nową spojną składową  $T$  składającą się z  $n_1+n_2$  wienchołków oraz  $n_1-1$  najlżejszych krawędzi łączących wienchołki z  $S_1$ ,  $n_2-1$  najlżejszych krawędzi z  $S_2$  i jednej najlżejszej krawędzi łączącej  $S_1$  i  $S_2$ , czyli razem  $n_1+n_2-1$  krawędzi.

Skoro są to najlżejsze krawędzie, to  $T$  jest MST.

