Sortowanie przez wybór
Selection Sort

Idea algorytmu **sortowania przez wybór** jest bardzo prosta. Załóżmy, iż chcemy posortować zbiór liczbowy rosnąco. Zatem element najmniejszy powinien znaleźć się na pierwszej pozycji. Szukamy w zbiorze **elementu najmniejszego** i wymieniamy go z **elementem na pierwszej pozycji**. W ten sposób element najmniejszy znajdzie się na swojej docelowej pozycji.

W identyczny sposób postępujemy z resztą elementów należących do zbioru. Znów wyszukujemy element najmniejszy i zamieniamy go z elementem na drugiej pozycji. Otrzymamy dwa posortowane elementy. Procedurę kontynuujemy dla pozostałych elementów dotąd, aż wszystkie będą posortowane.

Algorytm sortowania przez wybór posiada klasę czasowej złożoności obliczeniowej równą $O(n^2)$. Sortowanie odbywa się w miejscu.

Przykład:

Dla przykładu posortujmy tą metodą zbiór {4 7 2 9 3}. Kolorem zielonym oznaczyliśmy elementy zbioru, które są już posortowane.

Zbiór	Opis operacji
4 7 2 9 3	Wyszukujemy najmniejszy element w zbiorze. Jest nim liczba 2.
2 7 4 9 3	Znaleziony element minimalny wymieniamy z pierwszym elementem zbioru - liczbą 4
2 7 4 9 3	Wśród pozostałych elementów wyszukujemy element najmniejszy. Jest nim liczba 3.
2 3 4 9 7	Znaleziony element minimalny wymieniamy z drugim elementem zbioru - liczbą 7.
2 3 4 9 7	Znajdujemy kolejny element minimalny - liczbę 4.
2 3 4 9 7	Wymieniamy go z samym sobą - element ten nie zmienia zatem swojej pozycji w zbiorze.
2 3 4 9 7	Znajdujemy kolejny element minimalny
2 3 4 7 9	Wymieniamy go z liczbą 9
2 3 4 7 9	Ostatni element jest zawsze na właściwej pozycji. Sortowanie zakończone

Podana metoda sortuje zbiór rosnąco. Jeśli chcemy posortować zbiór malejąco, to zamiast elementu minimalnego poszukujemy elementu maksymalnego. Pozostała część procedury sortującej nie ulega zmianie.

Specyfikacja problemu

Dane wejściowe

n - liczba elementów w sortowanym zbiorze, $n \in N$

d[ ] - zbiór n-elementowy, który będzie sortowany. Elementy zbioru mają indeksy od 1 do *n*.

Dane wyjściowe

d[ ] - posortowany zbiór n-elementowy. Elementy zbioru mają indeksy od 1 do *n*.

Zmienne pomocnicze

i, *j* - zmienne sterujące pętli, $i, j \in N$

p_{min} - pozycja elementu minimalnego w zbiorze d[ ], $p_{min} \in N$

Lista kroków

K01: **Dla** $j = 1, 2, ..., n - 1$: **wykonuj** K02...K04

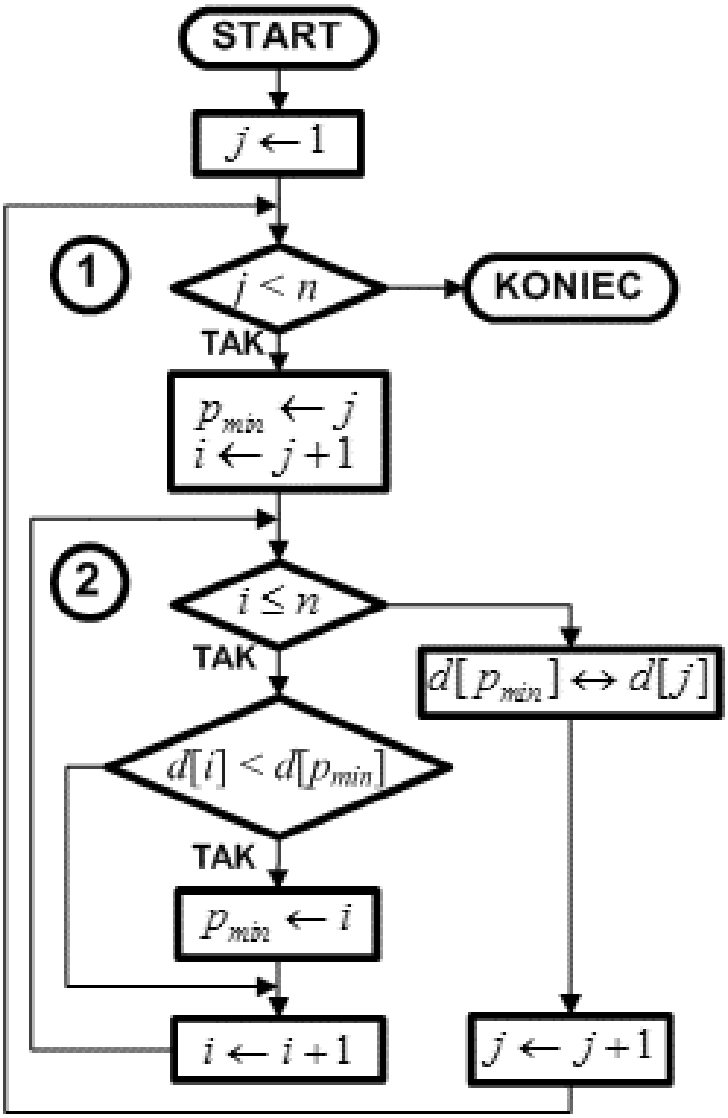
K02: $p_{min} \leftarrow j$

K03: **Dla** $i = j + 1, j + 2, ..., n$: **jeśli** $d[i] < d[p_{min}]$, **to** $p_{min} \leftarrow i$

K04: $d[j] \leftrightarrow d[p_{min}]$

K05: **Zakończ**

Schemat blokowy



Pętla zewnętrzna sterowana zmienną j wyznacza kolejne elementy zbioru o indeksach od 1 do n - 1, w których zostaną umieszczone elementy minimalne. Na początku tej pętli zakładamy, iż elementem minimalnym jest element $d[j]$ i zapamiętujemy jego indeks w zmiennej p_{min}.

W pętli numer 2 sterowanej zmienną i porównujemy pozostałe elementy zbioru z elementem $d[p_{min}]$. Jeśli element zbioru $d[i]$ jest mniejszy od elementu $d[p_{min}]$, to znaleźliśmy nowy element minimalny. W takim przypadku zapamiętujemy jego pozycję w p_{min} i kontynuujemy pętlę wewnętrzną.

Po zakończeniu pętli wewnętrznej p_{min} zawiera indeks elementu minimalnego. Zamieniamy miejscami element $d[j]$ z elementem $d[p_{min}]$. Dzięki tej operacji element minimalny znajduje się na swojej docelowej pozycji. Zwiększamy j przechodząc do kolejnego elementu zbioru i kontynuujemy pętlę zewnętrzną.