Lekcja 28 (kl. II. PP)

Temat: Przez wstawianie, czyli jeszcze o porządkowaniu liczb.

Cele lekcji:

– poznanie istoty sortowania przez wstawianie,
– ułożenie algorytmu realizującego sortowanie przez wstawianie

Uczeń:

– układa algorytm
w dowolnej postaci realizujący sortowanie przez wstawianie;
– weryfikuje działanie algorytmu na przykładach.

Podręcznik str. 193 – Informatyka. 2. PP, Operon

Przebieg lekcji:

1. Zapoznanie się z celami lekcji.

2. Dodajemy karty, czyli istota metody sortowania przez wstawianie.

3. Sortowanie przez wstawianie – analiza algorytmu – str. 194, rys. 28.2 i 28.3

4. Symulacja działania algorytmu – str. 195

5. Ćwiczenia praktyczne – stosowanie sortowania przez wstawianie w programowaniu w C++.

Zadania do wykonania - str. 196

Sortowanie przez wstawianie

powrót

Ten rodzaj sortowania możemy porównać do układania kart pokerzysty. Pierwszą kartę wstawiamy w odpowiednie miejsce przesuwając pozostałe, następną także wstawiamy między odpowiednie karty i tak układamy zestaw kart.

Prześledźmy przykład na następującym ciągu liczb:

$$2,\ 5,\ 3,\ 0,\ 7,\ 1$$

Strategia algorytmu:

- Rozpoczynamy od drugiego elementu, czyli

  $$5$$

  i porównujemy go z elementami poprzedzającymi - w tym przykładzie poprzedza tylko liczba

  $$2$$

  .
- Jeśli napotkamy liczbę większą, to musimy przesunąć ją o jeden w prawo.
- Czynność tą powtarzamy do momentu napotkania liczby niemniejszej lub gdy skończą nam się liczby (nie będzie spełniony warunek

  $$j >= 0$$

).
- W następnym kroku wstawiamy naszą liczbę w odpowiednie miejsce i otrzymujemy podzbiór uporządkowany.
- Powyższe czynności powtarzamy dla reszty wyrazów.

Dla przykładowego ciągu wygląda to następująco:

- **2 5 3 0 7 1** - liczba

  $$5$$

  pozostaje na swoim miejscu (dwa pierwsze elementy są posortowane)
- **2 3 5 0 7 1** - liczba

  $$3$$

  zostaje wstawiona między liczby

  $$2$$

  i

  $$5$$

  (trzy pierwsze elementy są posortowane)
- **0 2 3 5 7 1** - liczba

  $$0$$

  zostaje wstawiona na początek (cztery pierwsze elementy są posortowane)
- **0 2 3 5 7 1** - liczba

  $$7$$

  pozostaje na swoim miejscu (pięć pierwszych elementów jest posortowanych)
- **0 1 2 3 5 7** - liczba

1

zostaje wstawiona między liczby

0

i

2

. (otrzymujemy zbiór uporządkowany)

Sortowanie przez wstawianie można zaliczyć do atrakcyjniejszych algorytmów. Chociaż jego złożoność jest rzędu

$$O(n^2)$$

, a więc nie należy do najszybszych, to algorytm ma swoje zalety:

- jest stabilny
- bardzo dobrze zachowuje się w przypadku zbioru posortowanego lub częściowo posortowanego
- prosty w implementacji
- dobrze radzi sobie z niedużymi zbiorami

Algorytm najwięcej porównań elementów musi wykonać w przypadku pesymistycznej wersji zbioru danych (elementy posortowane są malejąco):

$$0+1+2+3+\ldots+n-1=\frac{n\cdot(n-1)}{2}$$

.

Program w C++ (materiał z algorytm.edu.pl**)**

```
#include<iostream>
using namespace std;

void sortowanie_przez_wstawianie(int n, int *tab)
{
     int pom, j;
     for(int i=1; i<n; i++)
     {
             //wstawienie elementu w odpowiednie miejsce
             pom = tab[i]; //ten element będzie wstawiony w odpowiednie miejsce
             j = i-1;

             //przesuwanie elementów większych od pom
             while(j>=0 && tab[j]>pom)
             {
                        tab[j+1] = tab[j]; //przesuwanie elementów
                        --j;
             }
             tab[j+1] = pom; //wstawienie pom w odpowiednie miejsce
     }
}

int main()
{
    int n, *tab;
    cout<<"Podaj wielkość zbioru: ";
    cin>>n;

    tab = new int [n];

    for(int i=0; i<n; i++)
    {
            cout<<"Podaj "<<i+1<<" element: ";
            cin>>tab[i];
    }

    cout<<"Elementy przed sortowaniem:\n";
    for(int i=0; i<n; i++)
            cout<<tab[i]<<" ";

    sortowanie_przez_wstawianie(n, tab);

    cout<<"\nElementy posortowaniem:\n";
    for(int i=0; i<n; i++)
            cout<<tab[i]<<" ";

    cin.ignore();
    cin.get();
    return 0;
}
```