Lecture 8

Sorting One-Dimensional Array

Sorting and searching operations are two basic operations very often used in computer programming. Sorting is very often used for searching what is way let's consider sorting 1-D array the first.

Sorting 1-D array represents a process of arrangement elements of array in ascending (or descending) order.

There are different algorithms of sorting for 1-D arrays (sequences). We consider three simple sorting algorithms: **1) linear selection sort, 2) selection and change sort** and **3) 'bubble' sort**. For simplification and generalization let's discuss sorting 1-D array **A** having **n** elements of integer type in ascending order. For any method of sorting are used two basic operations**:** comparison and swapping (interchanging) or moving elements of array effectuated in some order.

Algorithm of **linear selection sort** can be described as follows:

1. Consider the first element in the unsorted part of array as the current minimal value (external loop).
2. Find the minimal element in the unsorted part of array (internal loop).
3. Swap it with the first element from step 1 (external loop).
4. Repeat in the external loop the steps 1, 2, 3 until the unsorted part of array is empty.

Corresponding piece of code in C language:

```
int   A[50], i, n, min, minind, k;
for(i=0; i<n-1; i++)
   {
    min=A[i];
    minind=i;
     for( k=i+1; k<n; k++)
       {
          if (A[k]<min)
           {
             min=A[k];
             minind=k;
           }
        }
    A[minind]=A[i];
    A[i]=min;
  }
```

Algorithm of **selection and change sort** is similar with algorithm of linear selection sort and can be described as follows:

1. Consider the first element in the unsorted part of array as the current minimal value (internal loop).
2. Find the next current minimal element in the unsorted part of array and immediately swap it with the first element from step 1 (internal loop).
3. Repeat in the internal and external loop the steps 1, 2 until the unsorted part of array is empty.

Corresponding piece of code in C language:

```
int A[50], i, n, k, t;
for(i=0; i<n-1; i++)
  {
    for( k=i+1; k<n; k++)
      {
        if (A[k]<A[i])
          {
            t=A[k];
            A[k]=A[i];
            A[i]=t;
          }
      }
  }
```

Algorithm of as named 'bubble' sort can be described as follows:

1. Compare the adjacent elements in the unsorted part of array and if it is necessary immediately swap them (internal loop).
2. Repeat in the internal and external loop the step 1 until the unsorted part of array is empty.

Corresponding piece of code in C language:

```
int A[50], i, n, k, t;
for(i=0; i<n-1; i++)
  {
    for( k=0; k<n-1-i; k++)
      {
        if (A[k]>A[k+1])
          {
            t=A[k];
            A[k]=A[k+1];
            A[k+1]=t;
          }
      }
  }
```

Lecturer: Mihail Kulev, associate professor