Міністерство освіти і науки, молоді та спорту України

Національний технічний університет України

«Київський політехнічний інститут»

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

**Лабораторна робота №5**

З дисципліні «Програмування»

**Тема: «Сортування вектора»**

Виконав: Перевірив:

Студент групи ІО-33 Викладач

Заколенко Р. К. Саверченко В. Г.

Київ 2013

**Теоретичні відомості:**

1. **Сортування простими обмінами, сортування бульбашкою.**

Алгоритм складається з повторюваних проходів по сортованого масиву . За кожен прохід елементи послідовно порівнюються попарно і , якщо порядок у парі невірний , виконується обмін елементів . Проходи по масиву повторюються N- 1 раз або до тих пір , поки на черговому проході не опиниться , що обміни більше не потрібні, що означає - масив відсортований . При кожному проході алгоритму по внутрішньому циклу , черговий найбільший елемент масиву ставиться на своє місце в кінці масиву поруч з попереднім «найбільшим елементом» , а найменший елемент переміщається на одну позицію до початку масиву ( «спливає» до потрібної позиції як бульбашка у воді , звідси і назва алгоритму ) .

1. [**Сортування простими включеннями**](http://lubbook.net/book_318_glava_14_Sortuvannjaprostimi_vkljuche.html)**.**

Сортування цим методом проводиться послідовно крок за кроком. На K-му кроці вважається, що частина масиву, що містить перші K-1 елемент вже впорядкована. Далі необхідно взяти K-й елемент і підібрати для нього місце в відсортованій частині масиву таке, щоб після його вставки впорядкованість чи не порушилась. Потім треба вставити елемент [K] на знайдене місце.

**Код програми:**

Program MyArray;

Type

Tindex=1..10;

Tarray=array[Tindex] of real;

var

i,N,k:Tindex;

M:Tarray;

Procedure InputArray(var N1:Tindex;var M1:Tarray);

begin

write('Input the size of array, N=');

read(N1);

writeln('Input the array');

for i:=1 to N1 do

begin

write('M[',i,']=');

read(M1[i]);

end;

end;

Procedure OutputArray(N1:Tindex;M1:Tarray);

begin

writeln('The elements of array:');

for i:=1 to N1 do

write('M[',i,']=',M[i],' ');

writeln

end;

Procedure Sort1(N1:Tindex;var M1:Tarray);

var

i,j,x:Tindex;

a:real;

begin

x:=N1-1;

for j:=1 to x do

for i:=1 to N1-j do

if M1[i]>M1[i+1] then

begin

a:=M1[i+1];

M1[i+1]:=M1[i];

M1[i]:=a;

end;

end;

Procedure Sort2(N1:Tindex;var M1:Tarray);

var

i:Tindex;

j:integer;

x:real;

begin

for i:=2 to n do

begin

x:=M1[i];

j:=i-1;

while (j>0) and (x<M1[j]) do

begin

M1[j+1]:=M1[j];

j:=j-1;

end;

M1[j+1]:=x;

end;

end;

Procedure SearchQ(N1:Tindex;M1:Tarray);

var

i:Tindex;

j:boolean;

q:real;

begin

write('Input the elements, q=');

read(q);

j:=false;

for i:=1 to N1 do

if q=M1[i] then

begin

writeln('The position of ',q,' is i=',i);

j:=true;

end;

if j=false then writeln('not found');

end;

begin

InputArray(N,M);

writeln('1-Bubble sort');

writeln('2-Sorting by direct inclusion');

read(k);

case(k) of

1: Sort1(N,M);

2: Sort2(N,M);

end;

OutputArray(N,M);

SearchQ(N,M);

end.

**Висновки**:

1. Метод «бульбашки» є найпростішим у виконанні методом сортування елементів масиву, але він є й найповільнішим.
2. Метод «бульбашки» є ефективним тільки для невеликих масивів.
3. Для методу «бульбашки» у циклі (**for i:=1 to N-j** **do)** введене обмеження для змінної **і – (** N-j**)** дозволяє не робити зайві дії і перевіряти вже відсортовану частину масива.