Міністерство освіти, науки, молоді та спорту України

Львівський національний університет імені Івана Франка

Кафедра радіофізики

Звіт

про виконання лабораторної роботи №1

« Методи сортування цілих чисел:

«бульбашкою», вибором, вставкою »

Студентки

Групи ФеІ – 22

Литвин Віри

Перевірив

доц. Вельгош С.Р.

Львів 2012

**Тема :** Методи сортування чисел. Сортування «бульбашкою», вставкою та вибором.

**Мета:** вивчити методи сортування чисел «бульбашкою», вставкою та вибором і якнайраціональніше їх практично реалізувати.

**Теоретичний матеріал:**

**Метод «бульбашки».**

Сортування «бульбашкою» вважається простим [алгоритмом сортування](http://uk.wikipedia.org/wiki/%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC_%D1%81%D0%BE%D1%80%D1%82%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F). Він працює таким чином — у поданому наборі даних (списку чи масиві) порівнюються два сусідні елементи.

Якщо один з елементів, не відповідає критерію сортування (є більшим, або ж, навпаки, меншим за свого сусіда), то ці два елементи [міняються місцями](http://uk.wikipedia.org/wiki/%D0%9E%D0%B1%D0%BC%D1%96%D0%BD_(%D1%96%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0)).

Прохід по списку продовжується до тих пір, доки дані не будуть відсортованими.

Алгоритм отримав свою назву від того, що процес сортування за ним нагадує поведінку бульбашок повітря у резервуарі з водою.

Оскільки для роботи з елементами масиву він використовує лише порівняння, це [сортування на основі порівнянь](http://uk.wikipedia.org/w/index.php?title=%D0%A1%D0%BE%D1%80%D1%82%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D0%BD%D0%B0_%D0%BE%D1%81%D0%BD%D0%BE%D0%B2%D1%96_%D0%BF%D0%BE%D1%80%D1%96%D0%B2%D0%BD%D1%8F%D0%BD%D1%8C&action=edit&redlink=1).

Швидкодія рівна *О*(*n*²), де *n* — кількість елементів для сортування.

**Метод вставки.**

Він також вважається простим .

Суть алгоритму полягає в тому, що на кожному кроці ми вибираємо один з елементів вхідних даних і вставляємо його на потрібну позицію у вже відсортований масив доти, доки набір вхідних даних не буде вичерпано.

Метод вибору чергового елемента із вихідного масиву довільний, тобто можна використовувати практично будь-який алгоритм вибору.

Зазвичай, з метою отримати стійкий алгоритм сортування, елементи вставляються у порядку їх появи у вхідному масиві.

Швидкодія, як і в першому випадку, рівна *О*(*n*²), де *n* — кількість елементів для сортування.

**Метод вибору.**

**Сортування вибором** — простий алгоритм сортування лінійного [масиву](http://uk.wikipedia.org/wiki/%D0%9C%D0%B0%D1%81%D0%B8%D0%B2), на [основі вставок](http://uk.wikipedia.org/w/index.php?title=%D0%A1%D0%BE%D1%80%D1%82%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D0%BD%D0%B0_%D0%BE%D1%81%D0%BD%D0%BE%D0%B2%D1%96_%D0%BF%D0%BE%D1%80%D1%96%D0%B2%D0%BD%D1%8F%D0%BD%D1%8C&action=edit&redlink=1).

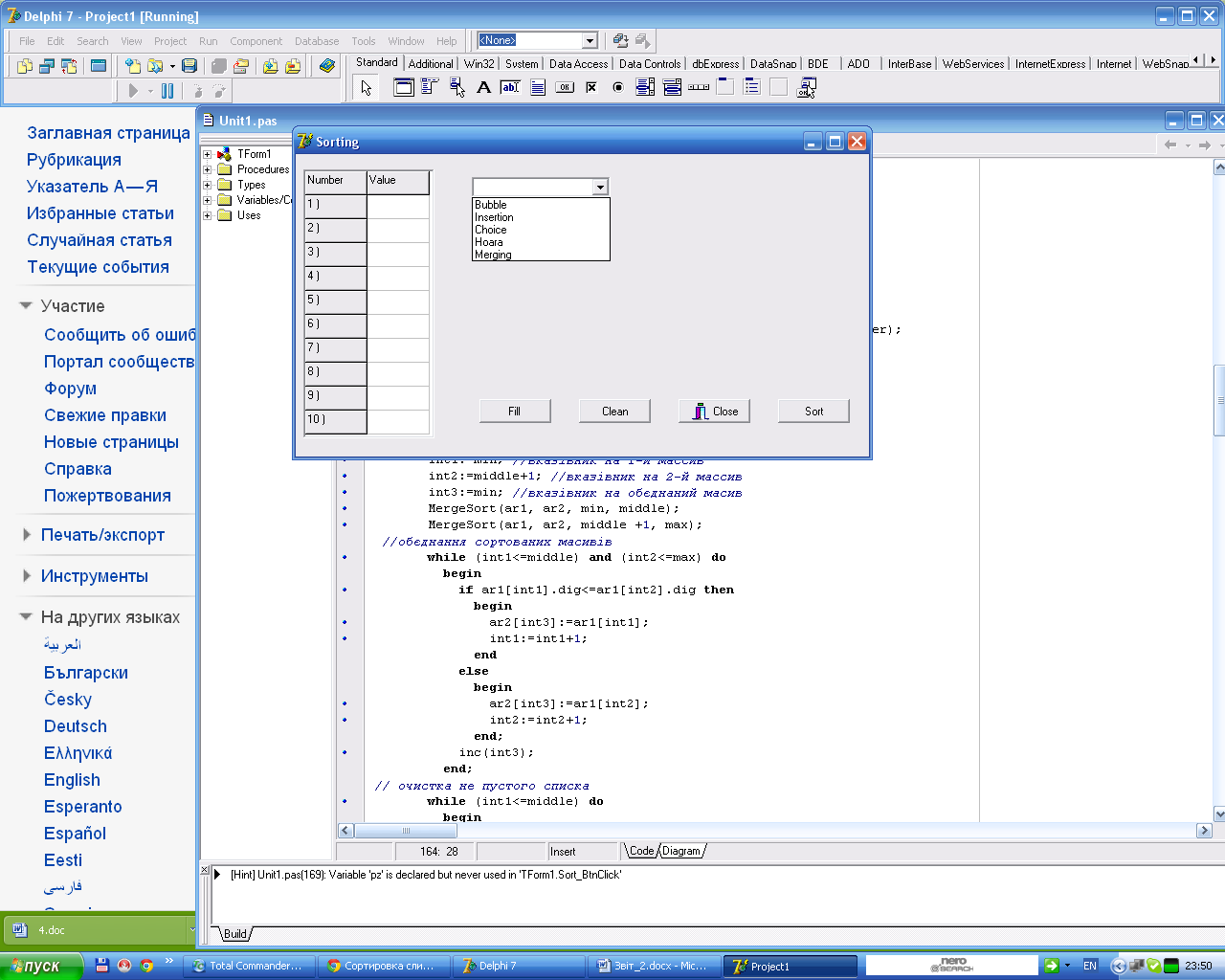
Алгоритм працює таким чином:

1. Знаходить у списку найменше значення
2. Міняє його місцями із першим значенням у списку
3. Повторює два попередніх кроки, доки список не завершиться (починаючи з другої позиції)

Фактично, таким чином ми поділили список на дві частини: перша (ліва) — повністю відсортована, а друга (права) — ні.

Швидкодія, як і в перших двох випадках, рівна *О*(*n*²), де *n* — кількість елементів для сортування.

**Практична реалізація:**

 Для програмної реалізації я використала середовище Deiphi 7. На формі створила наступний інтерфейс:

Тут дані не потрібно вводити, вони заповнюються автоматично та нумеруються, за допомогою оператора Random, при натисканні кнопки **Fill.**

Кнопка **Clean** витирає дані з колонки Value значення і впорядковує нумерацію Number.

При натисканні Close програма закриється.

А Sort відсортовує числа, переставляючи їх разом з номером.

Проте, аби сортуватидані, спершу потрібно обрати один із методів сортування: Bubble(бульбашкою), Insertion(Вставкою) чи Choice(Вибором).

**Висновок**

Хоча, алгоритм бульбашки є одним із найпростіших алгоритмів сортування, його ефективність є досить низькою, і він погано підходить для сортування великих списків. Більшість інших алгоритмів з такою ж швидкодією *O(n2)* є ефективнішими за алгоритм сортуванням бульбашками, наприклад, [сортування включенням](http://uk.wikipedia.org/wiki/%D0%A1%D0%BE%D1%80%D1%82%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D0%B2%D0%BA%D0%BB%D1%8E%D1%87%D0%B5%D0%BD%D0%BD%D1%8F%D0%BC).

Сортування бульбашкою також погано використовує можливості сучасних [мікропроцесорів](http://uk.wikipedia.org/wiki/%D0%9C%D1%96%D0%BA%D1%80%D0%BE%D0%BF%D1%80%D0%BE%D1%86%D0%B5%D1%81%D0%BE%D1%80). Він вимагає щонайменше удвічі більше операцій, ніж [сортування включенням](http://uk.wikipedia.org/wiki/%D0%A1%D0%BE%D1%80%D1%82%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D0%B2%D0%BA%D0%BB%D1%8E%D1%87%D0%B5%D0%BD%D0%BD%D1%8F%D0%BC), удвічі більше [кеш](http://uk.wikipedia.org/wiki/%D0%9A%D0%B5%D1%88) пам'яті, [асимптоматично](http://uk.wikipedia.org/wiki/%D0%9D%D0%BE%D1%82%D0%B0%D1%86%D1%96%D1%8F_%D0%9B%D0%B0%D0%BD%D0%B4%D0%B0%D1%83) більше [передбачень переходів](http://uk.wikipedia.org/w/index.php?title=%D0%9F%D0%B5%D1%80%D0%B5%D0%B4%D0%B1%D0%B0%D1%87%D1%83%D0%B2%D0%B0%D1%87_%D0%BF%D0%B5%D1%80%D0%B5%D1%85%D0%BE%D0%B4%D1%96%D0%B2&action=edit&redlink=1).

Результати експериментів проведених [Астраханом](http://uk.wikipedia.org/w/index.php?title=%D0%9E%D1%83%D0%B2%D0%B5%D0%BD_%D0%90%D1%81%D1%82%D1%80%D0%B0%D1%85%D0%B0%D0%BD&action=edit&redlink=1) показали, що сортування рядка за алгоритмом сортування бульбашками у п'ять разів повільніше за сортування того ж рядка за алгоритмом [сортування включенням](http://uk.wikipedia.org/wiki/%D0%A1%D0%BE%D1%80%D1%82%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D0%B2%D0%BA%D0%BB%D1%8E%D1%87%D0%B5%D0%BD%D0%BD%D1%8F%D0%BC) і на 40% повільніше за сортування за алгоритмом [сортування вибором](http://uk.wikipedia.org/wiki/%D0%A1%D0%BE%D1%80%D1%82%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D0%B2%D0%B8%D0%B1%D0%BE%D1%80%D0%BE%D0%BC).

**Додаток. Код програми:**

**Сортування методом «бульбашки»:**

begin

Sort := True;

while(Sort=True) do

begin

Sort:=False;

for i:=1 to n-1 do

if(Ryad[i+1].dig < Ryad[i].dig) then

begin

Change:=Ryad[i];

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

Ryad[i+1]:=Change;

Sort:=True;

end;

end;

for i:=1 to n do

begin

StringGrid1.Cells[0, i]:=IntToStr(Ryad[i].n)+' )';

StringGrid1.Cells[1, i]:=IntToStr(Ryad[i].dig);

end;

end;

**Сортування вставками:**

begin

for i:= 2 to n do

begin

Change:=Ryad[i];

j:=i-1;

while ((j>=1) and (Ryad[j].dig > Change.dig)) do

begin

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

j:=j-1;

end;

Ryad[j+1]:=Change;

end;

for i:=1 to n do

begin

StringGrid1.Cells[0, i]:=IntToStr(Ryad[i].n)+' )';

StringGrid1.Cells[1, i]:=IntToStr(Ryad[i].dig);

end;

end;

**Сортування вибором:**

begin

for i:=1 to n-1 do

begin

min := i;

for j:= i+1 to n do

if (Ryad[j].dig < Ryad[min].dig) then

min := j;

Change := Ryad[min];

Ryad[min] :=Ryad[i];

Ryad[i] := Change;

end;

for i:=1 to n do

begin

StringGrid1.Cells[0, i]:=IntToStr(Ryad[i].n)+' )';

StringGrid1.Cells[1, i]:=IntToStr(Ryad[i].dig);

end;

end;