НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ  
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

ФАКУЛЬТЕТ ПРИКЛАДНОЇ МАТЕМАТИКИ

КАФЕДРА СИСТЕМНОГО ПРОГРАМУВАННЯ ТА СПЕЦІАЛІЗОВАНИХ КОМП’ЮТЕРНИХ СИСТЕМ

**КУРСОВА РОБОТА  
з дисципліни «Структури даних та алгоритми»**

Виконав: Курач В.М.

Група КВ-41

Залікова книжка: КВ-4111

Допущений до захисту

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

2 семестр 2014/2015 навч. року

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ  
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

ФАКУЛЬТЕТ ПРИКЛАДНОЇ МАТЕМАТИКИ

КАФЕДРА СИСТЕМНОГО ПРОГРАМУВАННЯ ТА СПЕЦІАЛІЗОВАНИХ КОМП’ЮТЕРНИХ СИСТЕМ

|  |  |
| --- | --- |
| *Узгоджено*  *Керівник роботи*  *\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/Марченко О.І./* | *Захищена «\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_20\_\_\_р.*  *з оцінкою \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*  *\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/Марченко О.І./* |

**Дослідження ефективності методів сортування  
на багатовимірних масивах:  
метод прямої вставки з лінійним пошуком місця вставки від елемента, що вставляється, з використанням «бар’єру»;  
алгоритм №5 методу прямого вибору;  
метод прямого обміну без модифікацій;  
швидке сортування**

Виконавець роботи:\_\_\_\_\_\_\_\_\_\_\_\_

Курач Віктор Миколайович

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_\_р.

**ПЛАН РОБОТИ**

1. Технічне завдання.
2. Теоретичні положення.
3. Схема імпорту/експорту модулів та структурна схема взаємовикликів процедур та функцій.
4. Опис призначення процедур та функцій.
5. Текст програми.
6. Тести.
7. Таблиці виміру часу.
8. Порівняльний аналіз.
9. Висновки.

**ТЕХНІЧНЕ ЗАВДАННЯ**

**<*Из методички переписать*>**

**Варіант №113**

**Задача №6.** Упорядкувати тривимірний масив ***A* [*p,m,n*]**, переставивши перерізи за незменшенням сум їх елементів.

**Методи сортування:**

№ 3. Алгоритм сортування методу прямої вставки з лінійним пошуком місця вставки від елемента, що вставляється («справа»), з використанням «бар’єру».

№ 9. Алгоритм сортування №5 методу прямого вибору.

№ 13. Алгоритм сортування методу прямого обміну без модифікацій.

№ 23. Швидке сортування.

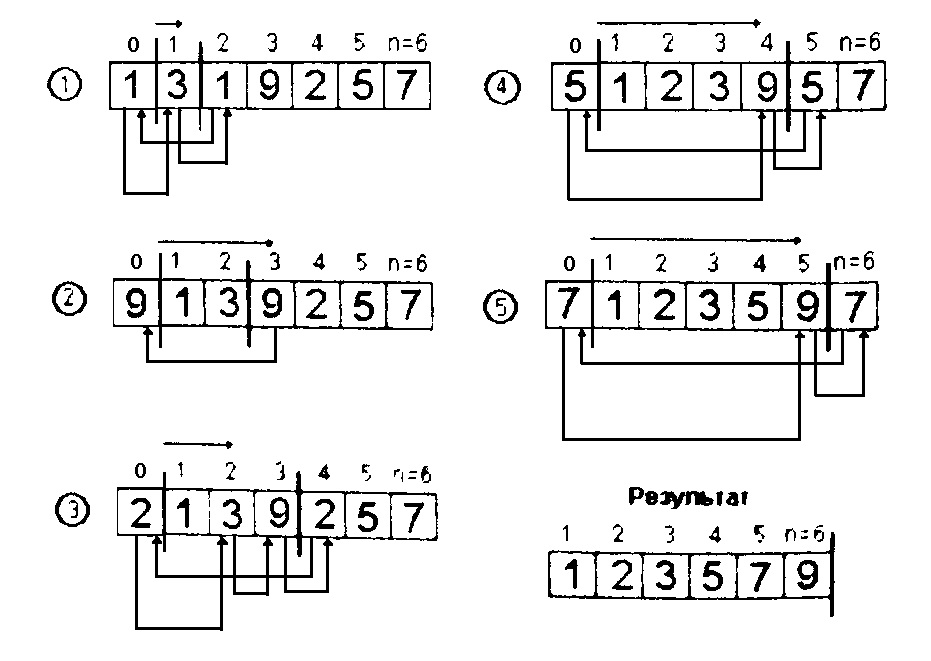
**Спосіб обходу №7.** У якості першого етапу сортування сформувати додатковий вектор Sum, довжина якого дорівнює кількості перерізів і значеннями якого є суми елементів відповідних перерізів. Використовуючи елементи вектора ***Sum*** як ключі сортування, переставляти відповідні перерізи кожен раз, коли треба переставляти ключі.

Розглянути **випадки**, коли елементи масиву впорядковані відповідно до заданої ознаки, протилежно їй, та коли елементи масиву невпорядковані.**ТЕОРЕТИЧНІ ПОЛОЖЕННЯ**

**Метод прямої вставки з лінійним пошуком місця вставки від елемента, що вставляється, з використанням «бар’єру».** У кожен момент часу масив вважається поділеним на дві частини: відсортовану й невідсортовану. На початку сортування єдиний найлівіший елемент приймається за відсортовану частину. Власне алгоритм:

1. Найлівіший елемент із невідсортованої частини розміщується в комірку масиву №0, яка є одночасно і бар’єром, і буферною змінною для елемента, для якого шукається місце вставки.
2. Запам’ятовується ліва межа невідсортованості (індекс узятого елемента) як змінна ***j***.
3. Проводиться циклічний зсув елементів масиву на 1 елемент вправо з одночасним зменшенням змінної ***j*** на 1. Це триває доти, поки не буде знайдено елемент, менший узятого, або вираз ***j-1*** не вкаже на бар’єр.
4. Узятий елемент вставляється у знайдену позицію, на яку вказує ***j***.
5. Межа невідсортованості зсувається на один індекс вправо.
6. Дії 1.-5. повторюються, поки невідсортована частина не зменшиться до нуля елементів.

Схема алгоритму:

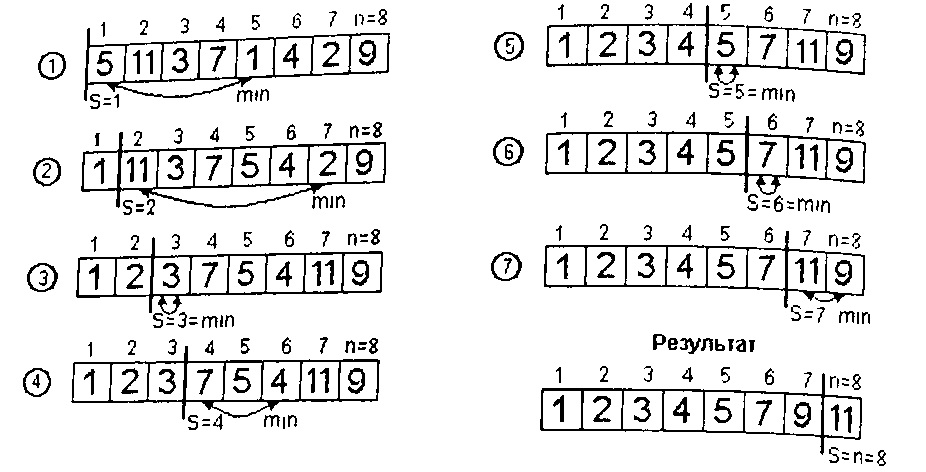


**Алгоритм сортування №5 методу прямого вибору.** На початку сортування весь масив вважається невідсортованим.

1. Найлівіший елемент невідсортованої частини умовно приймається за найменший у ній. Запам’ятовується він сам та його індекс ***s***.
2. У невідсортованій частині відшукується найменший елемент.
3. Знайдений елемент міняється місцем із елементом з індексом ***s***.
4. Межа невідсортованості ***s*** зсувається на один індекс вправо.
5. Дії 1.-4. повторюються, поки невідсортована частина не зменшиться до єдиного крайнього правого елемента (він є найбільшим у масиві).

Особливість модифікації №5 полягає в тому, що елементи міняються місцями лише якщо не рівні їх індекси, що теоретично може дещо пришвидшити роботу алгоритму.

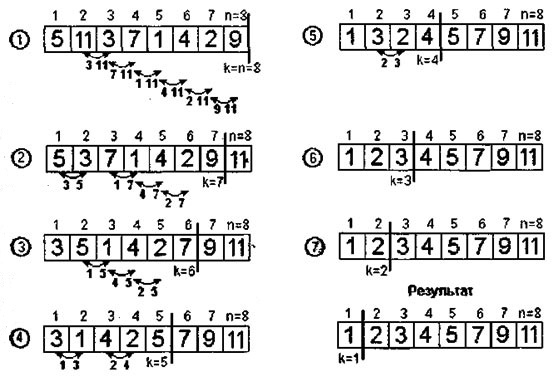
Схема алгоритму:



**Алгоритм сортування методу прямого обміну без модифікацій.** На початку сортування весь масив вважається невідсортованим.

1. Зліва направо порівнюються поруч розташовані елементи. Якщо їх взаєморозташування не відповідає умові сортування, вони міняються місцями.
2. Таким чином, найбільший елемент невідсортованої частини стане на найправішу позицію відсортованої частини.
3. Межа невідсортованості ***k*** зміщується на один індекс вліво.
4. Дії 1.-3. повторюються, поки невідсортована частина не зменшиться до єдиного крайнього лівого елемента (він є найменшим у масиві).

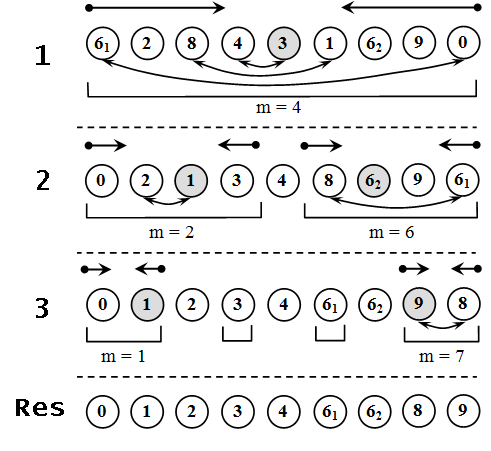
Схема алгоритму:



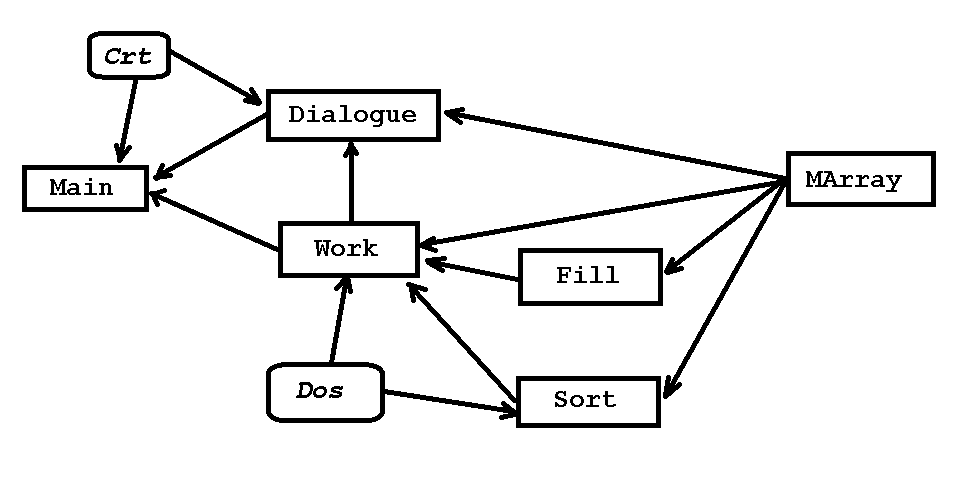
**Швидке сортування.**

1. Обирається центральний елемент масиву.
2. Зліва направо шукається елемент, що більший або дорівнює обраному центральному; на нього вказує індекс ***i***.
3. Справа наліво шукається елемент, що менший або дорівнює обраному центральному; на нього вказує індекс ***j***.
4. Елементи з індексами ***i*** та ***j*** міняються місцями. Індекс ***i*** збільшується на одиницю, ***j*** зменшується на одиницю.
5. Дії 2.-4. виконуються, поки індекси ***i*** та ***j*** не перетнуться.
6. У результаті виконання дій 1.-5. масив стає поділеним на дві частини відносно обраного центрального елемента. Для кожної з частин окремо виконуються дії 1.-5., поки всі такі частини не стануть рівними одному елементу.

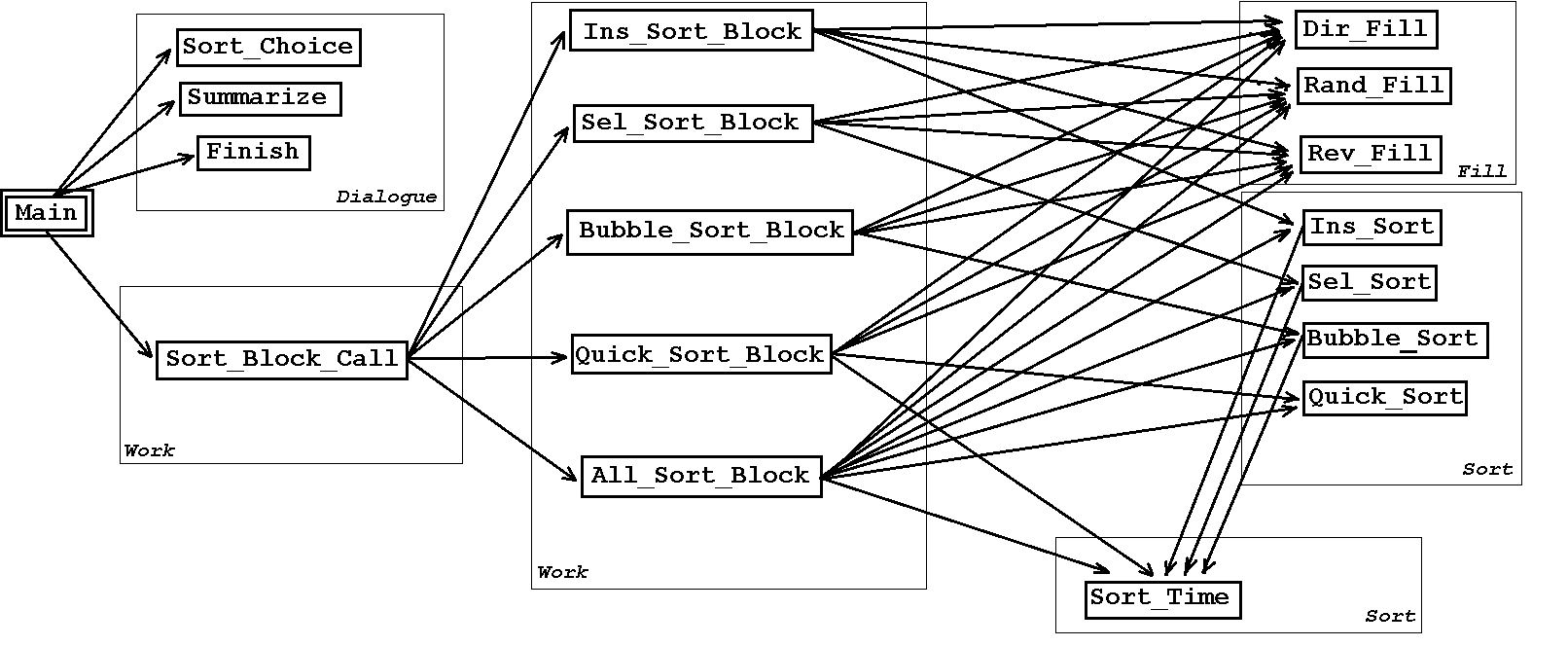
Схема швидкого сортування:



**СТРУКТУРНІ СХЕМИ ПРОГРАМИ**

****

***Схема імпорту/експорту модулів***



***Схема взаємовикликів процедур та функцій***

**ОПИС ПРИЗНАЧЕННЯ ПРОЦЕДУР ТА ФУНКЦІЙ**

**Модуль *MArray*** містить масив ***A*[*p, m, n*]**, із яким працює програма. Масив розміщено в окремому модулі для того, щоби можна було задати його в максимально можливих геометричних розмірах.

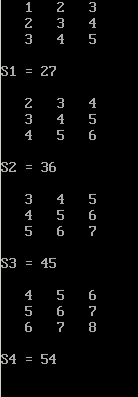
**Mодуль *Work*** містить процедури, кожна з яких забезпечує дослідження того алгоритму, який обрав користувач.

1. Процедура ***Ins\_Sort\_Block*** викликає сортування методом вставки: спочатку прямо впорядкованого, потім випадково заповненого і обернено впорядкованого масиву.
2. Процедура ***Sel\_Sort\_Block*** викликає сортування методом вибору аналогічно попередній процедурі.
3. Процедура ***Bubble\_Sort\_Block*** викликає сортування методом обміну аналогічно попередній процедурі.
4. Процедура ***Quick\_Sort\_Block*** викликає швидке сортування аналогічно попередній процедурі.
5. Процедура ***All\_Sort\_Block*** виводить на екран таблицю, в якій наведено час сортування прямо впорядкованого, випадково заповненого та обернено впорядкованого масиву всіма чотирма методами сортування.

***<І так далі…>*ТЕКСТ ПРОГРАМИ**

<***Коды, коды, коды, одни коды, и все на паскале***>

**ТЕСТИ**

1. Тестування процедури, що заповнює масив за збільшенням сум перерізів (по ознаці сортування).

program direct\_filling\_test;

uses crt;

const p=4; m=3; n=3;

type mainarray=array [1..p, 1..m, 1..n] of word;

var a: mainarray; k, i, j: word; s: longint;

begin

clrscr;

for k:=1 to p do begin

s:=0;

for i:=1 to m do begin

for j:=1 to n do begin

a[k,i,j]:=k+(i-1)+(j-1);

s:=s+a[k,i,j];

write(a[k,i,j]:4);

end;

writeln;

end;

writeln;

writeln('S', k, ' = ', s);

writeln;

end;

readln;

end.

1. Тестування процедури, що заповнює масив випадково. <…>
2. Тестування процедури, що заповнює вектор за зменшенням сум перерізів (протилежно ознаці сортування). <…>
3. Тестування процедури, що сортує масив методом вставки. <…>
4. Тестування процедури, що сортує масив методом вибору. <…>
5. Тестування процедури, що сортує масив методом обміну. <…>
6. Тестування процедури, що сортує масив швидким сортуванням. <…>
7. Тестування функції, що обчислює вимір часу сортування. <…>

**ТАБЛИЦІ ВИМІРУ ЧАСУ**

**Дослід 1.**Залежність часу сортування від довжини вектора ключів (кількості розрізів масиву) при однаковому перерізі.

|  |  |  |  |
| --- | --- | --- | --- |
| Вектор довжиною 1024 елементи | Упорядкований | Невпорядкований | Обернено  впорядкований |
| Метод вставки | 0,00 | 0,18 | 0,32 |
| Метод вибору | 0,10 | 0,12 | 0,16 |
| Метод обміну | 0,13 | 0,29 | 0,39 |
| Швидке сорт-ння | 0,00 | 0,01 | 0,00 |

|  |  |  |  |
| --- | --- | --- | --- |
| Масив  A[1024, 4, 4] | Упорядкований | Невпорядкований | Обернено  впорядкований |
| Метод вставки | 0,06 | 5,45 | 10,65 |
| Метод вибору | 0,33 | 0,29 | 0,16 |
| Метод обміну | 0,34 | 5,67 | 10,90 |
| Швидке сорт-ння | 0,03 | 0,06 | 0,03 |

<***И множетво похожих таблиц>***

**ПОРІВНЯЛЬНИЙ АНАЛІЗ**

**<*Ну без этого никак>***

**ВИСНОВКИ**

***<…>***

**ВИКОРИСТАНА ЛІТЕРАТУРА**

1. Дослідження ефективності алгоритмів сортування та двійкового пошуку на багатовимірних масивах: методичні вказівки та завдання до виконання курсової роботи з дисципліни «Структури даних та алгоритми» для студентів денної форми навчання напрямку підготовки 6.050102 «Компютерна інженерія» [Електронне видання] / О.І. Марченко. – К.: НТУУ «КПІ», 2013. – 67 с.
2. Марченко А.И., Марченко Л.А. Программирование в среде Turbo Pascal 7.0. – 6-е изд. – К.: Век+, Спб.: КОРОНА-Век, 2007. – 464 с., ил.