Міністерство освіти і науки України

Національний технічний університет

«Харківський політехнічний інститут»

Факультет комп’ютерних наук

ЗВІТ

про виконання лабораторної роботи №4

Варіант 7

за дисципліною

«Алгоритми і структури даних»

Виконав:

студент групи КН-320Д

Кулдошин Є.І.

Перевірила:

старший викладач

Мошко Є.О.

Харків 2022

Мета: ознайомитися із основними способами написань алгоритмів сортировок та асимптотичні алгоритми, та особливостями їх програмної реалізації. Набути практичних навичок роботи з алгоритмами сортировок.

Умова задачі:

1. Створити функцію сортування масиву алгоритмами сортування, які треба обрати у таблиці 3 відповідно до свого варіанту. Варіант завдання вибирається за номером студента в списку групи.

2. Будь-яким способом заповнити елементи масиву значеннями.

3. Виконати сортування масиву першим алгоритмом і проаналізувати отримані результати.

4. Перевірити всі варіанти вихідного заповнення масиву: випадковим чином, відсортованого у порядку зростання, відсортованого у порядку спадання. Переконатися в правильності сортування у всіх випадках. Зробити висновки.

5. Повторити пункти 2-4 для другого та третьего алгоритму сортування.

6. Дослідити складність алгоритмів. Провести асимптотичний аналіз алгоритмів сортування та зробити висновки.

7. Здійснити порівняння алгоритмів сортування для n = 10, 100, 1000, 10000 і в наступному порядку вхідних елементів: − елементи вже впорядковані; − елементи в зворотному порядку; − розстановка елементів випадкова.

8. Виконання порівняння алгоритмів сортування на основі наступних кроків: − створити таблицю асимптотичних оцінок трудомісткості алгоритмів в кращому, середньому, гіршому випадках; − розставити лічильники операцій у функціях угруповань; − провести експеримент, визначити середню кількість операцій для різних угруповань, побудувати графіки; − для кожного виклику сортування генерувати новий масив; − створити таблиці і представити графіки експериментальних оцінок алгоритмів.

Необхідно враховувати, що порівняння алгоритмів проводиться на однакових вхідних даних.

Зберегти файл з тестом програми для подальших робіт.

Заборонено використовувати готові реалізації структур даних (наприклад, STL).

Опис усіх етапів розробки:

1 – Аналіз умови завдання.

2 – Повторне прочитання лекційного матеріалу, повторення методів створення алгоритмів сортировок (Вставками, Порозрядна, Пірамідальна).

3 – Створення проекту і прив'язка до сервісу GitHub.

4 – Створення файлів для роботи з проектом (Main.cpp, Main.h, Sort.cpp).

5 – Програма буде працювати з роздільною компіляцією, це дозволяє коду бути більш читабельним.

6 – У файлі Main.cpp будуть описані русифікатори, сервісні змінні (для роботи з меню), цикл while де буде описано меню програми і вхід у функції.

7 – Файл Main.h створенний для підключення потрібних бібліотек (windows.h, iostream, timer.h, conio.h, iomanip), підключення простір імен std, створення структури даних (Data) де описуємо розміри масивів (size1, size2, size3, size4, size5). Створив змінну print, від результату (1 або 0) буде / не буде виводити елементи масиву. Створив define (N1, N2, N3, N4, N5, N6) як цілочисленну константу для розміру масиву.

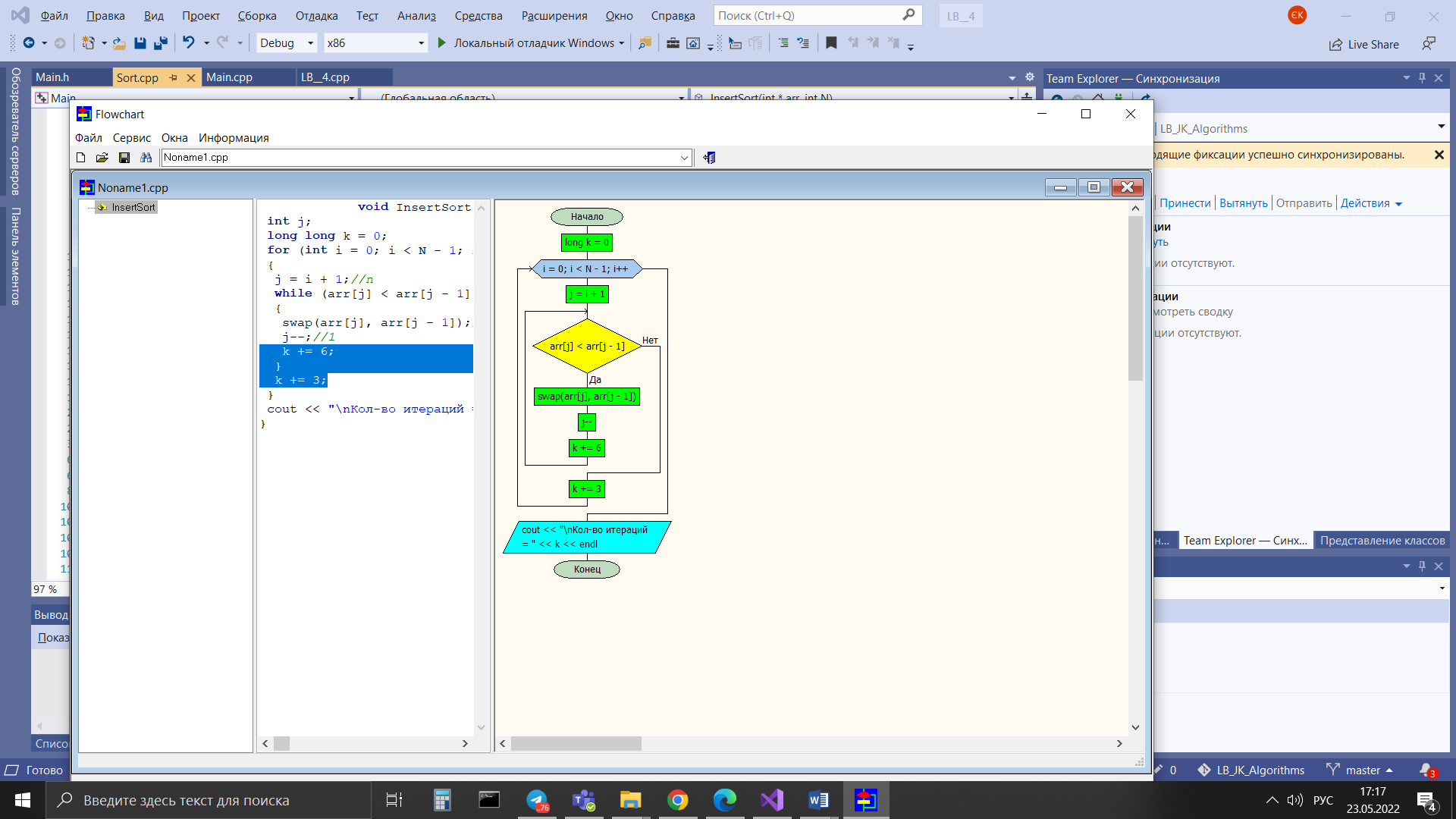
8 – У файлі Sort.cpp описана вся логіка функцій роботи з сортировками.

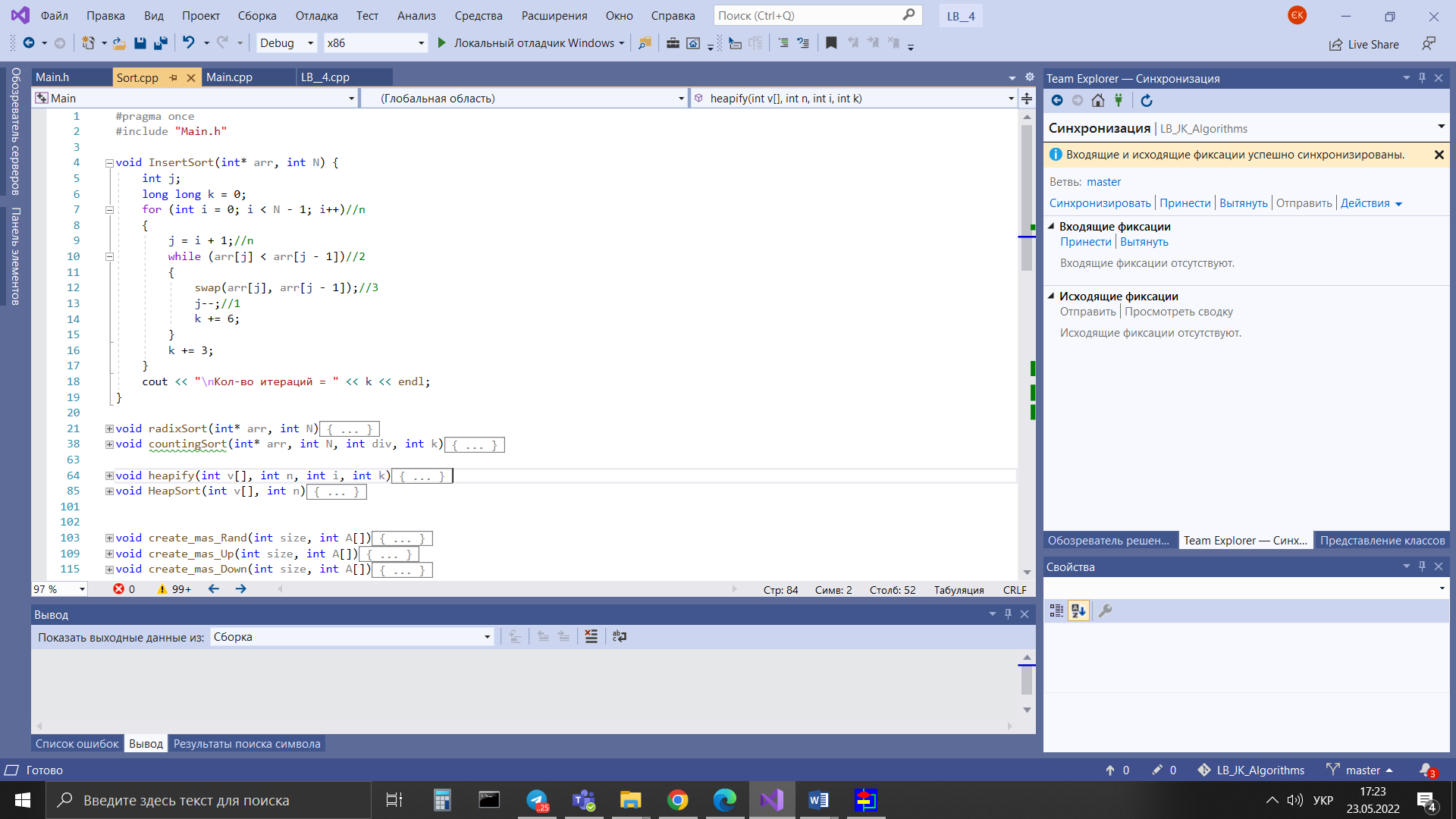
Аналіз задачі: При ознайомленні та розробці коду я навчився і застосував знання зі створенням алгоритму сортування, та особливостями їх програмної реалізації.

Структура основних вхідних і вихідних даних: В якості вступних даних, система роботи з алгоритмом сортировок використовує змінні цілочисленного типу, що передаються в алгоритм сортировок. А вихідні дані являють собою текстові рядки і символьну інформацію, що виводиться у вигляді списку елементів.

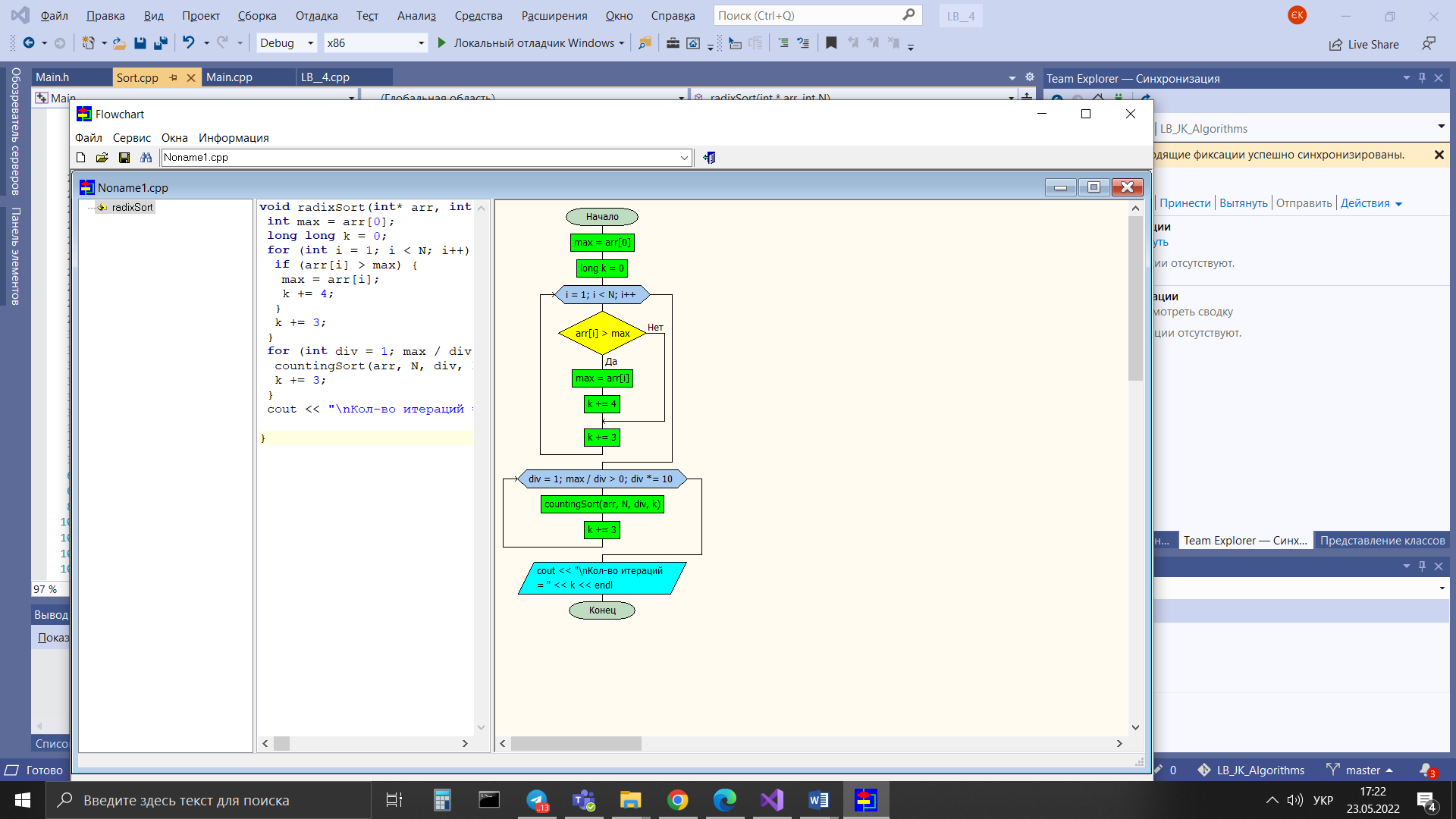
Алгоритм розв’язку завдання:

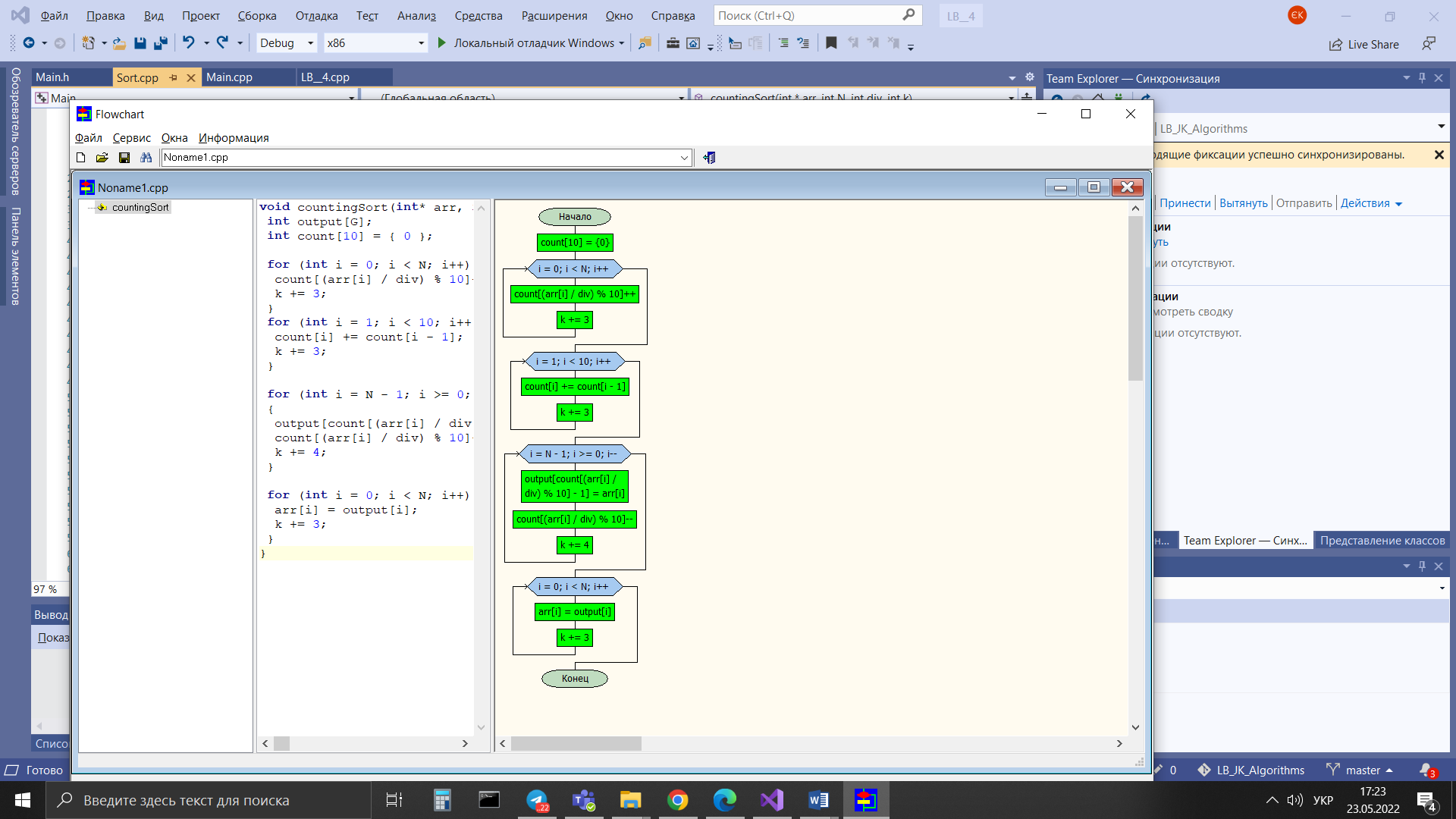
1. Алгоритм сортування вставками

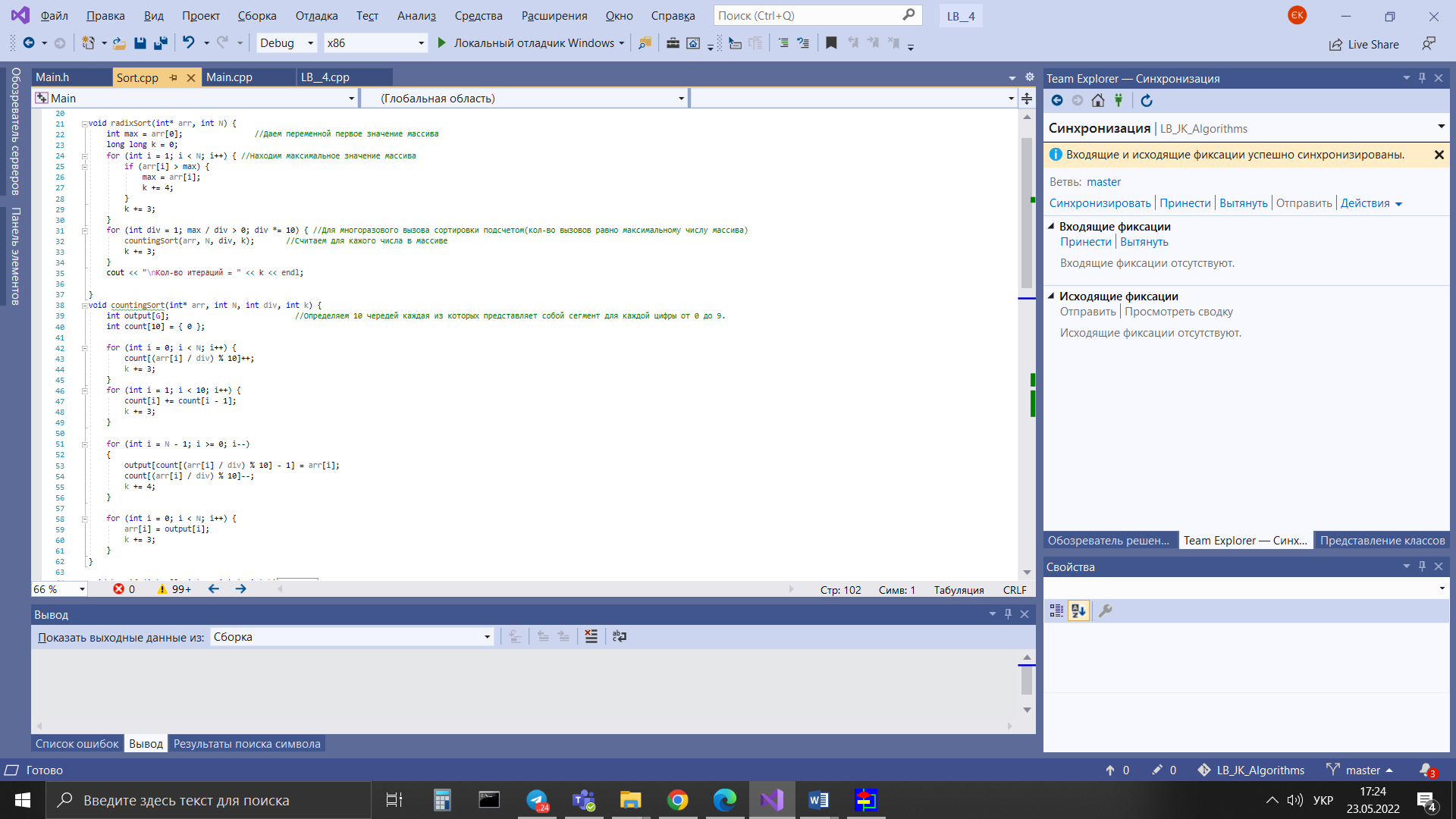




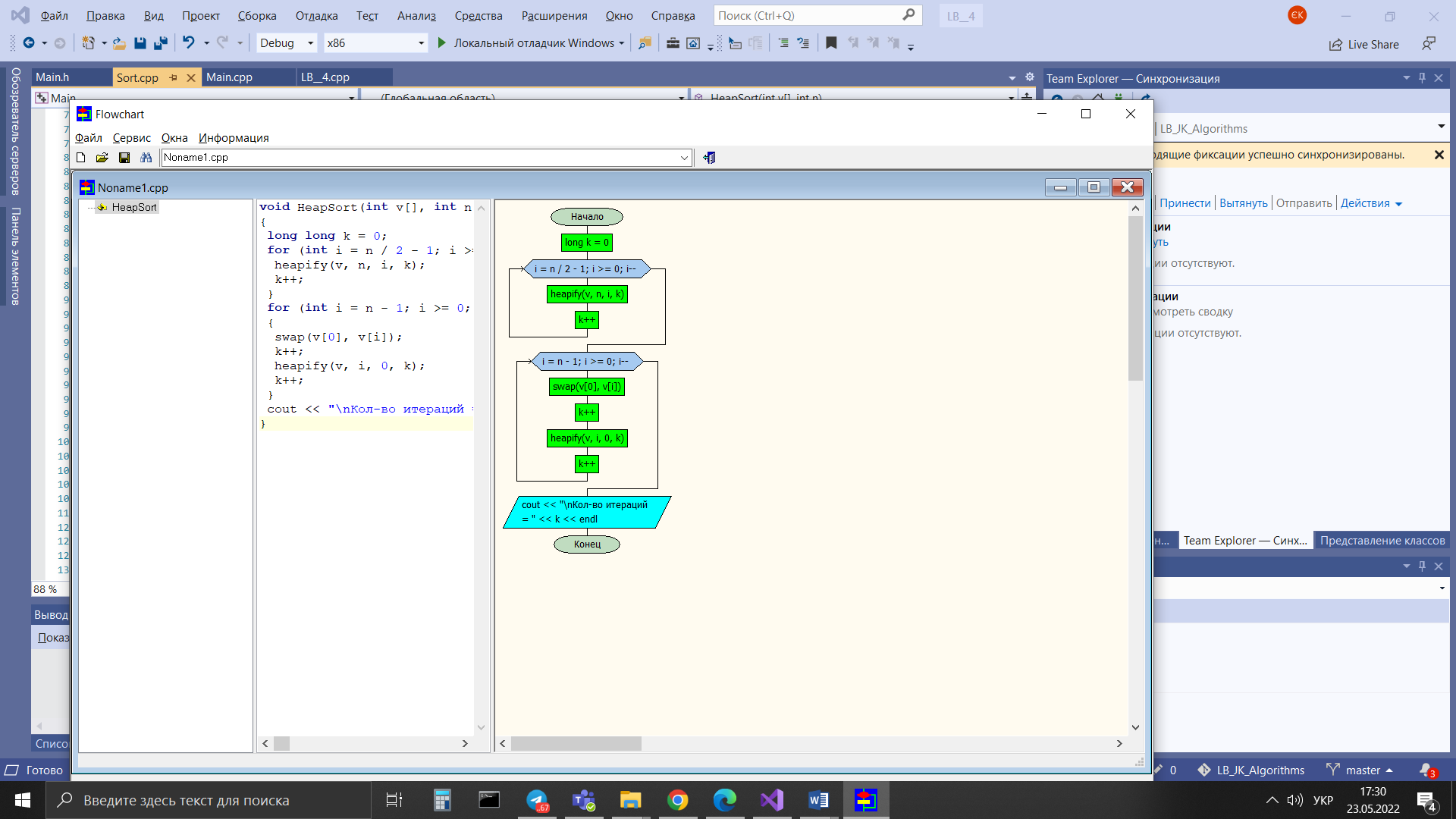
1. Алгоритм порозрядного сортування

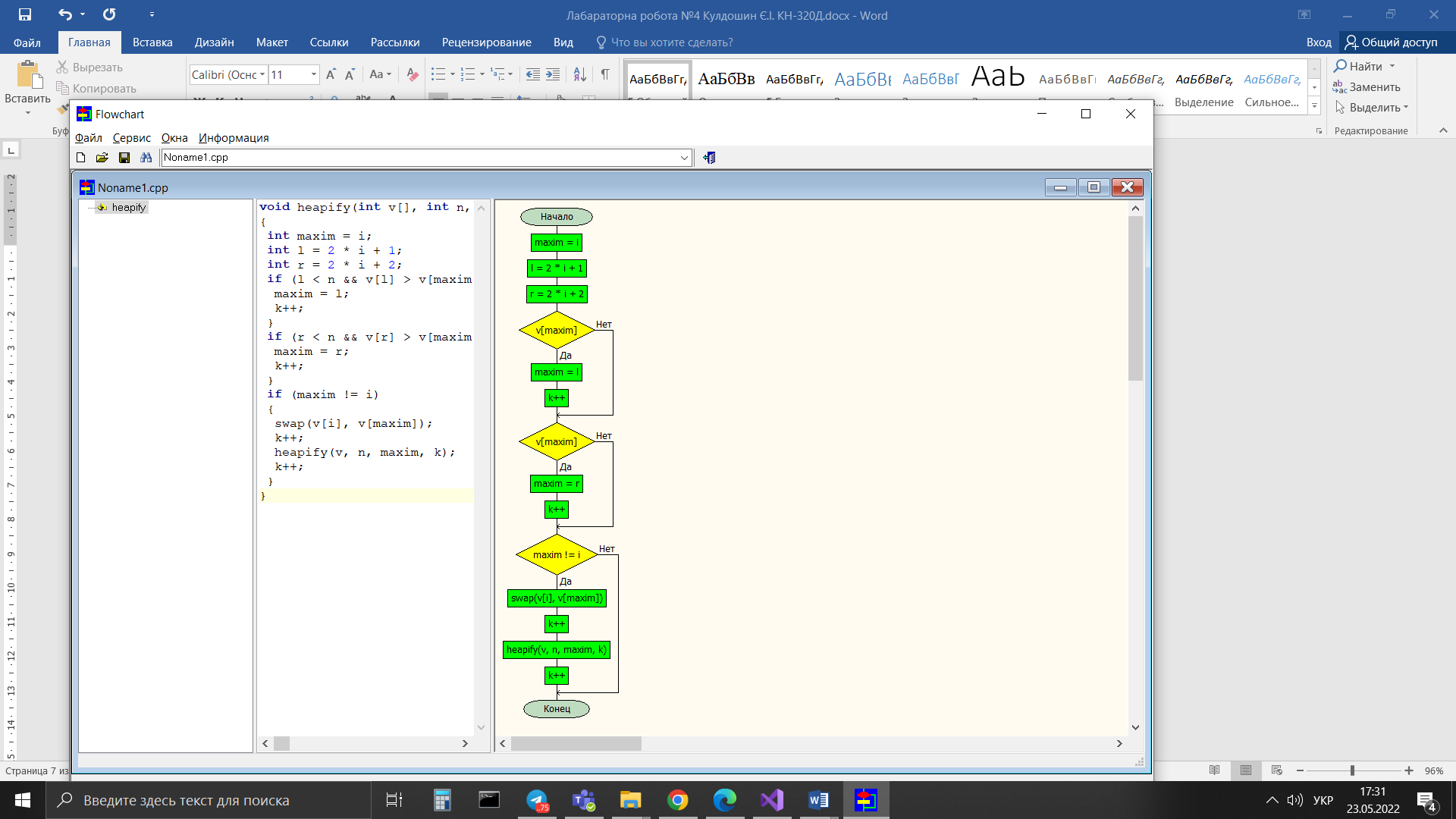


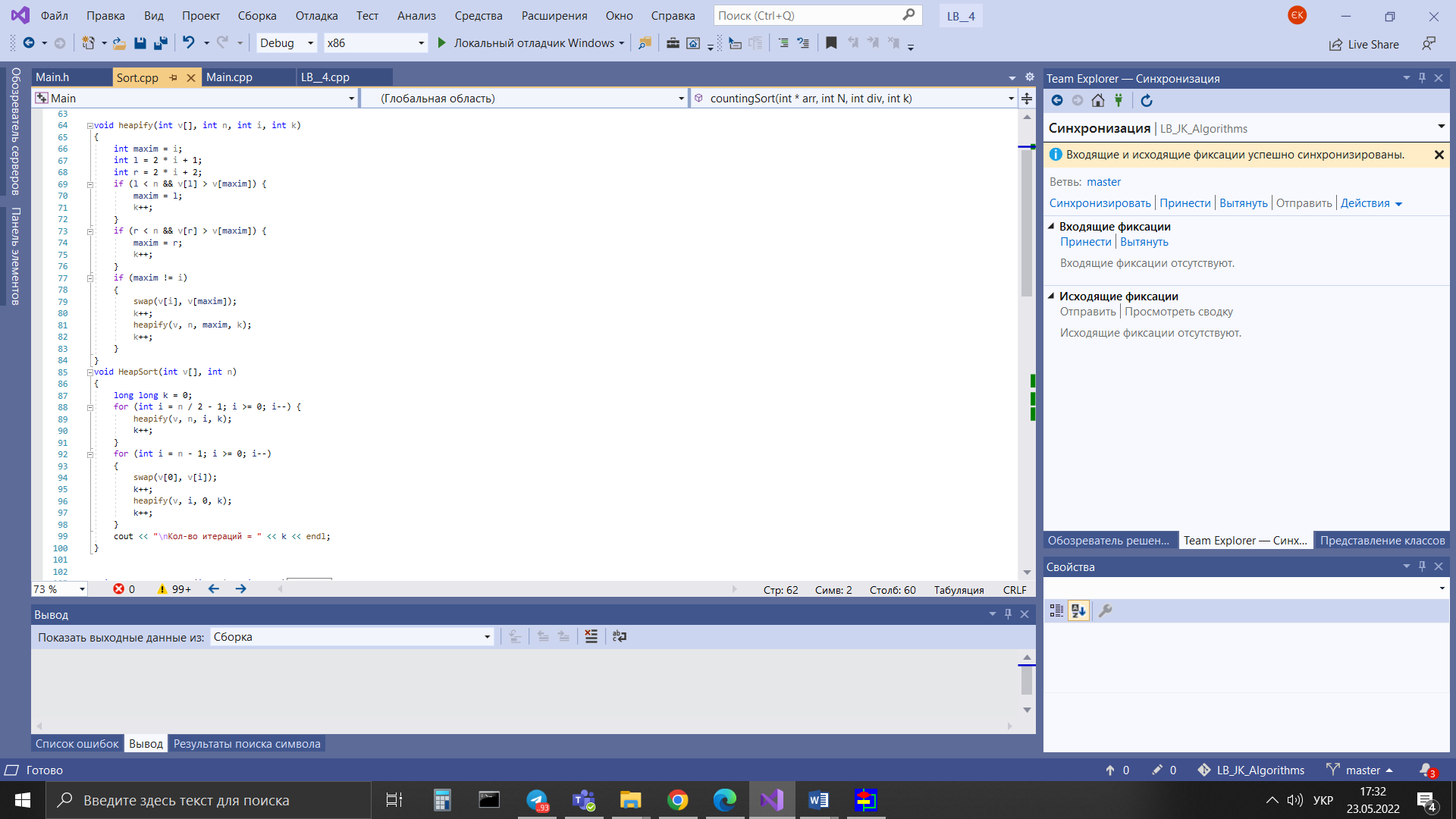




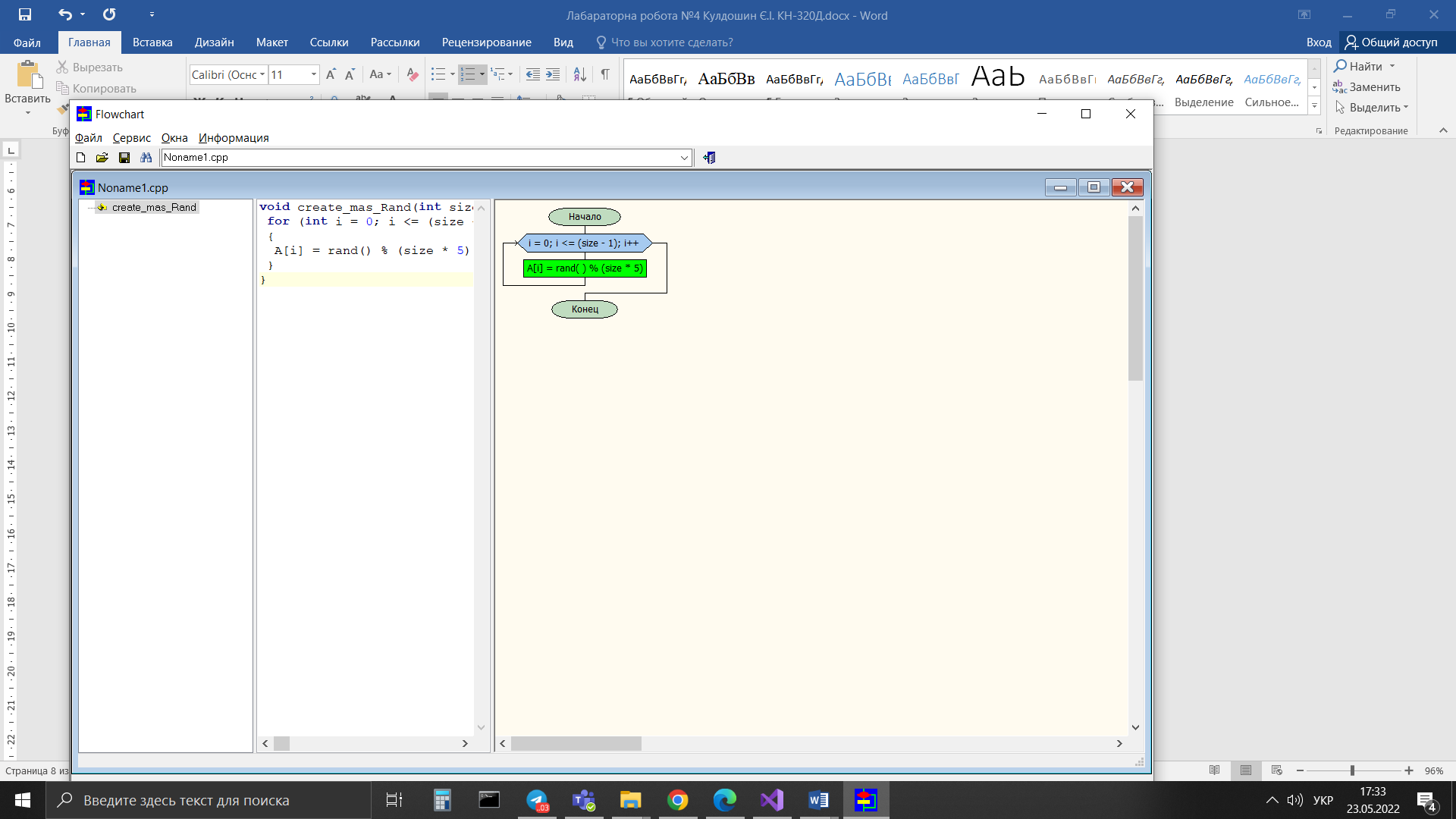
1. Алгоритм пірамідального сортування

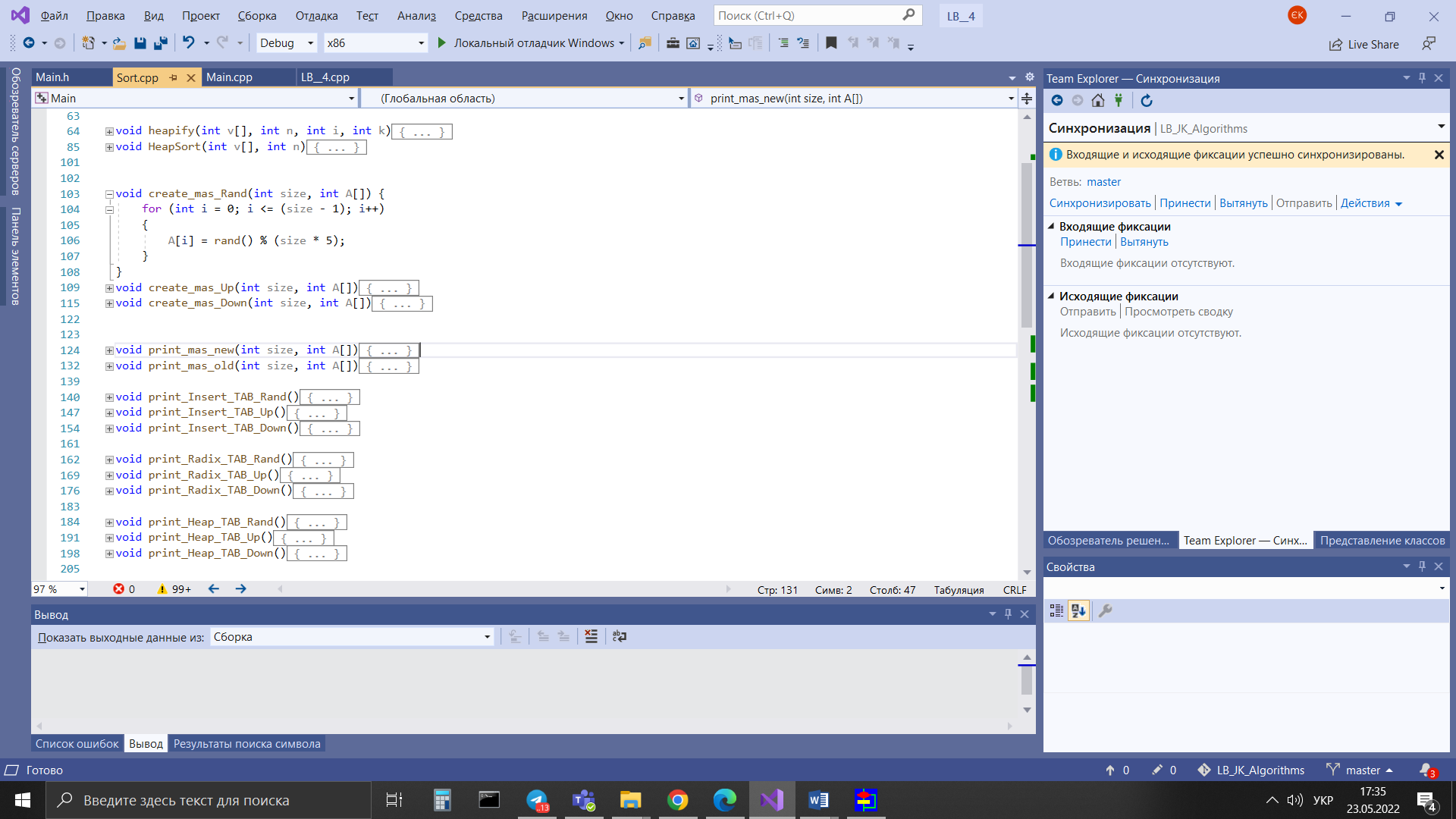




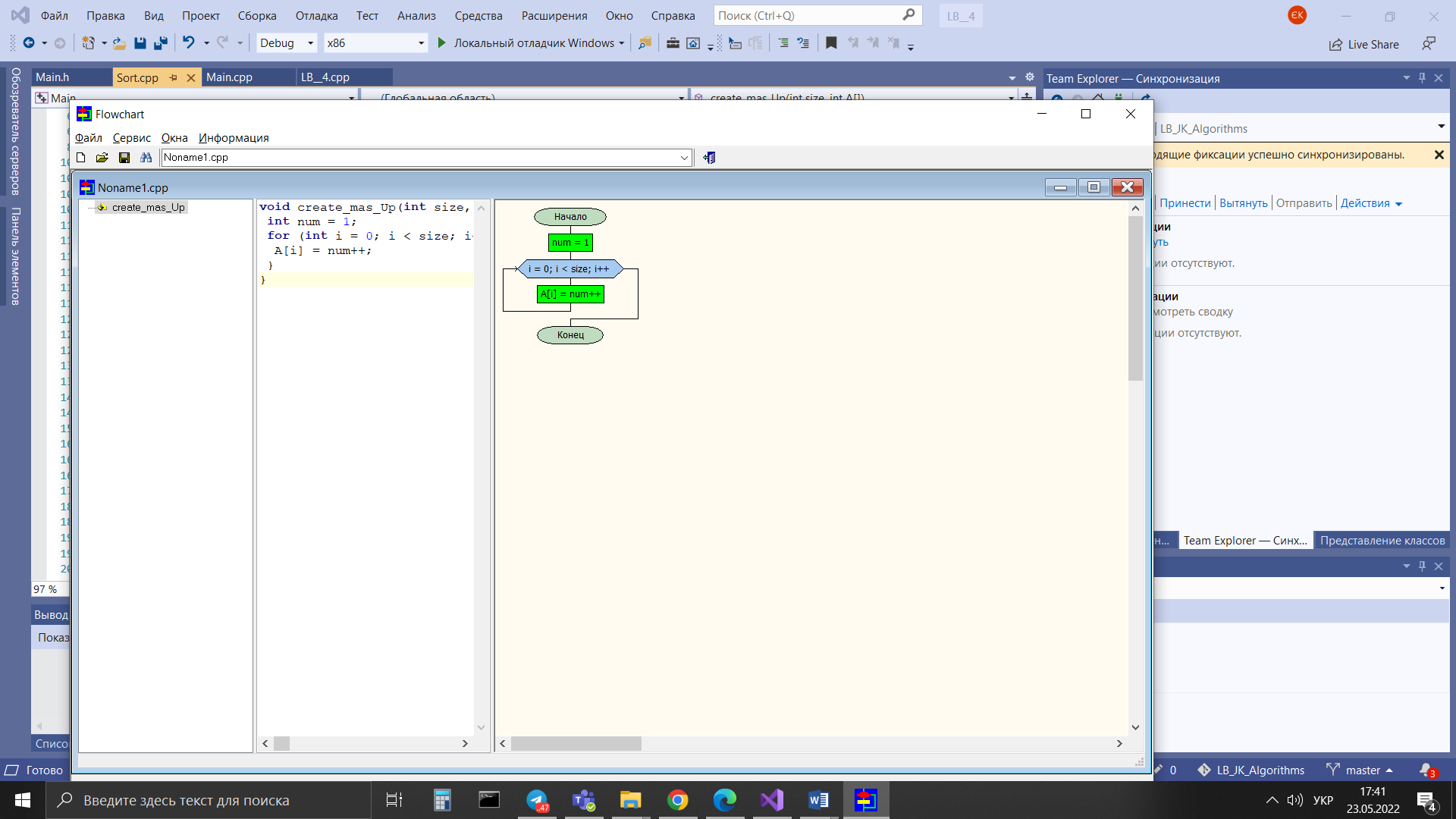


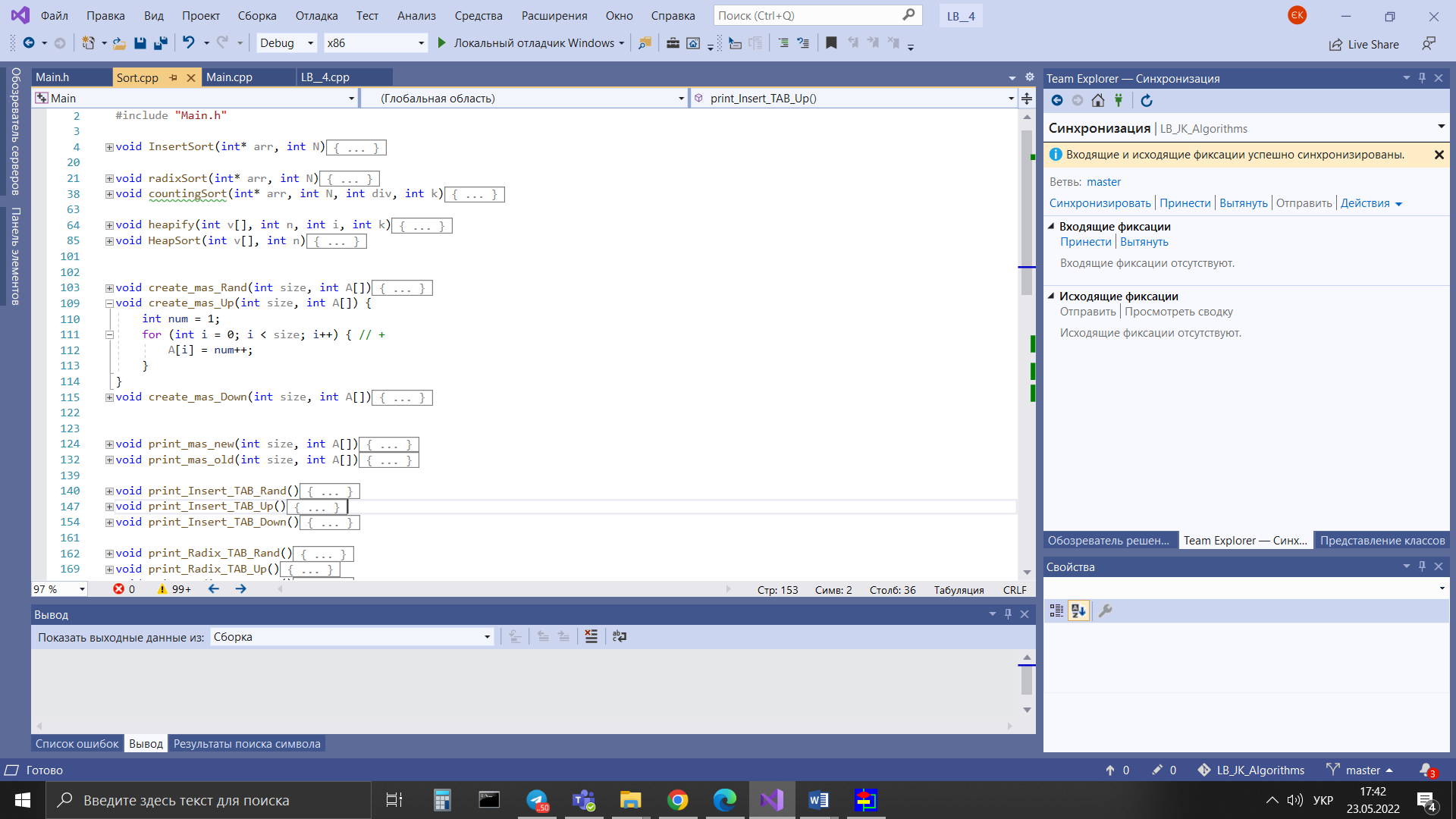
1. Функція для створення рандомних елементів масива



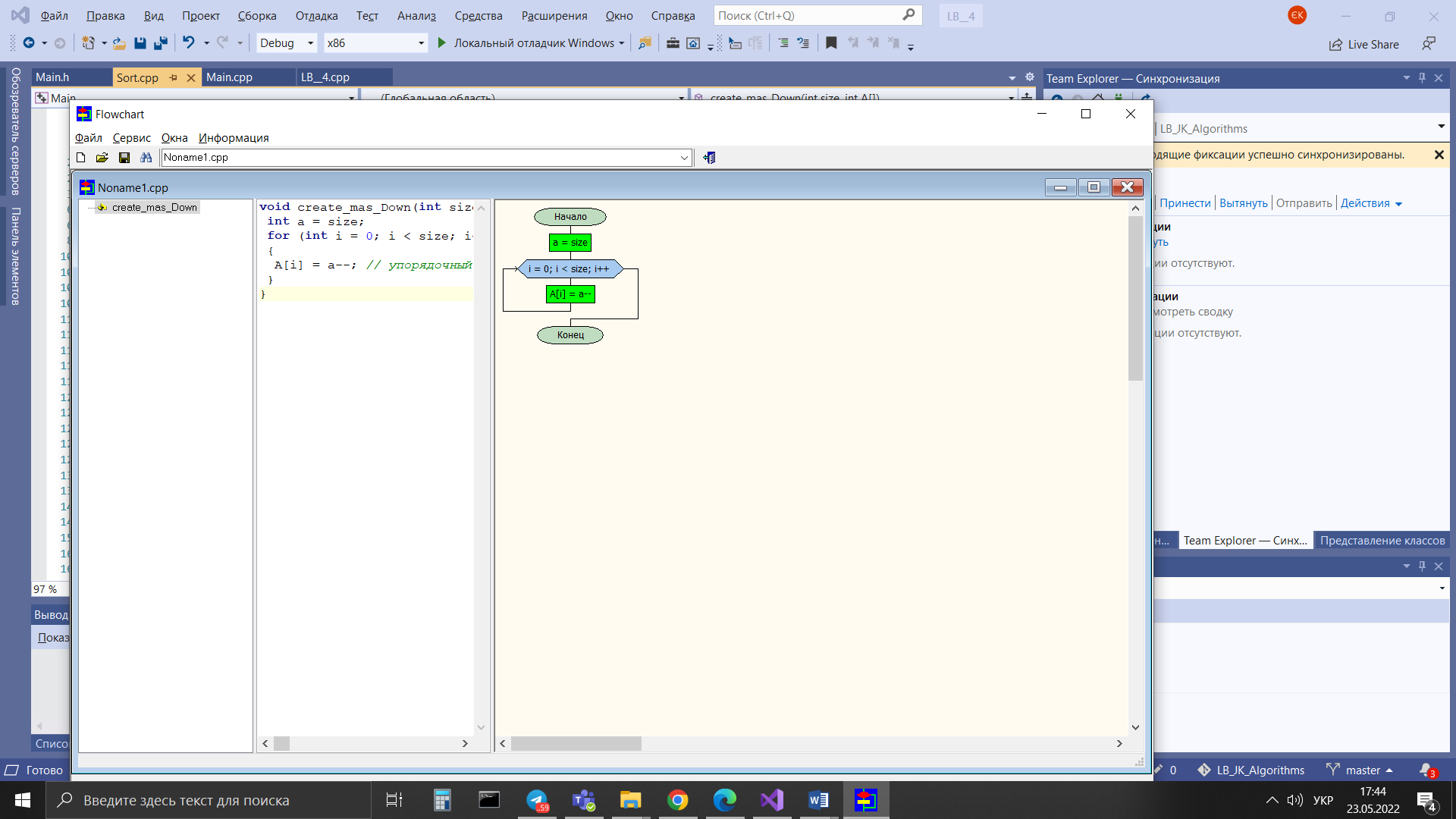


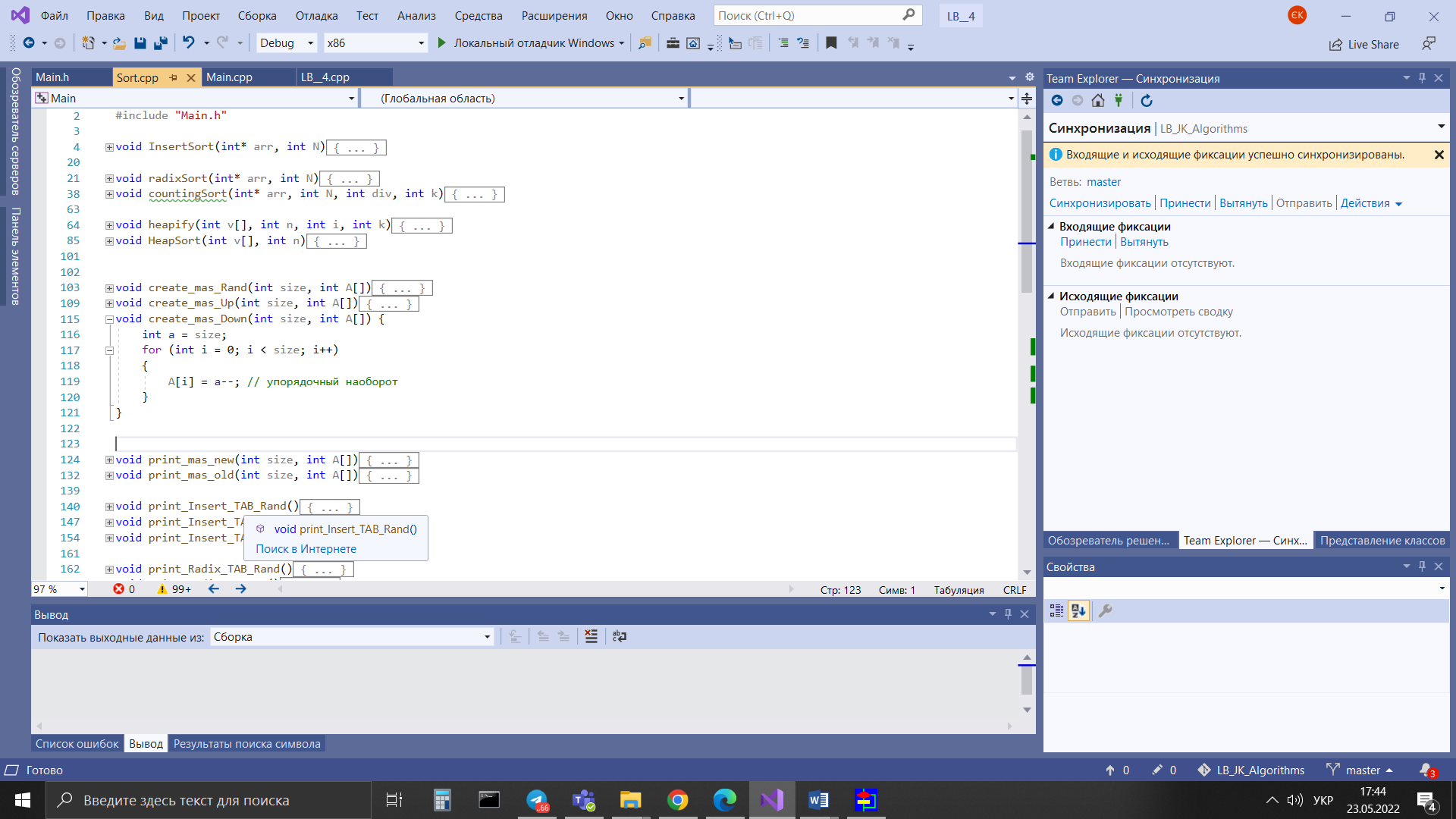
1. Функція для створення упорядкованних елементів масиву





1. Функція для створення неупорядкованних елементів масиву





Таблиці замірів робіт алгоритмів сортування:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Неупорядкований масив | | | | | | |
|  | Кількість елементів в масиві | | | | | |
| 10 | 100 | 1000 | 10000 | 20000 | 30000 |
| Сортування вставками | 0.000010 | 0.001000 | 0.029000 | 2.624000 | 10.582000 | 23.754000 |
| Порозрядного сортування | 0.001000 | 0.001000 | 0.001000 | 0.004000 | 0.005000 | 0.007000 |
| Пірамідального сортування | 0.000020 | 0.000100 | 0.002000 | 0.020000 | 0.042000 | 0.065000 |

Таблиця 1 – Неупорядкований масив (заповнений випадковими числами).

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Упорядкований масив | | | | | | |
|  | Кількість елементів в масиві | | | | | |
| 10 | 100 | 1000 | 10000 | 20000 | 30000 |
| Сортування вставками | 0.001000 | 0.001000 | 0.001000 | 0.001000 | 0.001000 | 0.001000 |
| Порозрядного сортування | 0.000010 | 0.001000 | 0.001000 | 0.003000 | 0.006000 | 0.007000 |
| Пірамідального сортування | 0.001000 | 0.001000 | 0.002000 | 0.022000 | 0.043000 | 0.072000 |

Таблиця 2 – Упорядкований масив (заповнений числами від 1 до 10…).

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Упорядкований масив в зворотному напрямку | | | | | | |
|  | Кількість елементів в масиві | | | | | |
| 10 | 100 | 1000 | 10000 | 20000 | 30000 |
| Сортування вставками | 0.001000 | 0.001000 | 0.073000 | 5.759000 | 22.692000 | 51.159000 |
| Порозрядного сортування | 0.001000 | 0.001000 | 0.001000 | 0.003000 | 0.006000 | 0.007000 |
| Пірамідального сортування | 0.001000 | 0.001000 | 0.002000 | 0.024000 | 0.045000 | 0.060000 |

Таблиця 3 – Упорядкований масив в зворотному напрямку (заповнений числами від 10… до 1).

Графіки замірів робіт алгоритмів сортування:

Графік 1 – Неупорядкований масив (заповнений випадковими числами).

Графік 2 – Упорядкований масив (заповнений числами від 1 до 10…).

Графік 3 – Упорядкований масив в зворотному напрямку (заповнений числами від 10… до 1).

На діаграмах зображені результати порівнянь трьох методик, які були заповнені різним способом:

Неупорядкований масив:

Найшвидше сортування з трьох запропонованих це Порозрядна, так як витрачає саме мінімальний час роботи. При обробці 30 000 елементів, які були заповнені рандомним способом, алгоритму знадобилося лише 0.007000 секунд (Графік 1). І для порівняння візьмемо сортування Вставками, для обробки точно такої ж кількості елементів, програмі доводиться витрачати 23.754000 секунди, це практично в 3500 разів більше ніж Порозрядний метод сортування.

Упорядкований:

Найшвидше сортування для масиву який був заповнений від 1 до 10...., це сортування Вставками, тут вона показу себе в кращому вигляді (Графік 2).

Упорядкований у зворотному вигляді:

А ось в даному випадку вже простежується те, що Порозрядне сортування знову лідирує серед всіх сортувань, їй треба було обробити 30.000 елементів за 0.007000 секунд, а при тій же кількості сортування Вставками витратила 51.159000 секунд (Графік 3), що є великою різницею для написання і оптимізації коду, практично в 7300 разів більше ніж Порозрядна.

Висновок:

У кожного алгоритму сортування своя тимчасова і просторова складність. Використовувати можна будь-який з представлених алгоритмів в залежності від поставлених завдань. Але на мою суб'єктивну думку найкращим алгоритмом є порозрядне сортування. З нею зручно працювати і вона найменше витрачає часу на обробку даних.