**Лабораторна робота №4. Використання рядків типу string, символьних та рядкових величин.**

**Мета*:*** *навчитись складати й реалізовувати алгоритми та програми мовою С++ для обробки рядків типу string, символьних та рядкових величин, використовуючи оператори введення та виведення до файлу.*

***Завдання:***

1. Ознайомитися з теоретичними відомостями до цієї ЛР, переглянути теоретичні відомості до ЛР №3 та врахувати їх при виконанні завдання і в подальшій роботі.
2. Запустити середовище програмування С++ Builder (або інше доступне при виконанні вдома)
3. Продовжуємо працювати з програмою, розробленою на ЛР№3.
4. У вікні редагування доповните текст програми, відповідно до наданої постановки задачі.
5. В першому рядку програми записати в коментарі номер групи та прізвище, а також номер ЛР
6. Результати надсилати на електронну адресу викладача

ЗАУВАЖЕННЯ

Приклад рядка, сформованого програмою

1;А.В. Ovchieva;1;11534336;121;3; #20:10:5

Цей рядок містить лише дані, відділені один від одного розділювачами.

[**t.i.lumpova@gmail.com**](mailto:t.i.lumpova@gmail.com)у вигляді cpp-файлу з іменем у форматі

**<Номер групи> <Номер лабораторної><Прізвище англійською>**

Наприклад, 31-01Ivanov.cpp.

Тему в заголовку листа записати

**ООП<Номер групи>-ЛР <Номер лабораторної>-<Прізвищеанглійською>**

**Строк відсилки ЛР ІПЗ-31 11.10.2020**

**ІПЗ-32 12.10.2020**

Всі запитання, що виникнуть, надсилайте на електронну адресу викладача, Тему в заголовку листа записати

**ООП<Номер групи>-Запитання-<Прізвище англійською>**.

**Постановка задачі:**

1. Продовжуємо розроблювати додаток, який буде оброблювати інформацію про викладачів та студентів.
2. Додати позиції меню "reading from the teacher data file " (читання з файлу даних про викладачів) (4), "reading from a data file about students " (читання з файлу даних про студентів) (5), " sorting teacher data by phone number " (сортування за номером телефону даних про викладачів) (6), "sorting student data by phone number " (сортування за номером телефону даних про студентів) (7).
3. Позиції меню (4) та (5) реалізуються аналогічно: виконується читання даних зі сформованого відповідного файлу і заповнюється відповідний масив структур.
4. Позиції меню (6) та (7) реалізуються аналогічно: виконується сортування інформації в масиві структур, після чого виконується запит щодо виведення відсортованої інформації на консоль і при запиті на виведення вона виводиться. Потім робиться запит щодо перезапису файлу і при запиті га перезапис він перезаписується.
5. Дії в позиціях меню оформлюються викликом відповідної функції, де вони реалізуються.
6. Результат виконання ЛР: модифікований файл програми (срр) та два відсортованих масиви.

**Теоретичні відомості.**

1. **Щодо виклику main() з main()** в <https://ru.cppreference.com/w/cpp/language/main_function> надаються такі відомості:

# Функция main()

Программа должна иметь глобальную функцию **main**, которая является точкой старта программы. Она должна иметь одну из следующих форм:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| int **main** **()** **{** body **}** | (1) |  |  |  |  |  |  |  |  |
| int **main** **(**int argc**,** char \*argv[]**)** **{** body **}** | (2) |  |  |  |  |  |  |  |  |
| /\* другая, определённая реализацией форма, с int, в качестве возвращаемого типа \*/ | (3) |  |  |  |  |  |  |  |  |

|  |  |  |
| --- | --- | --- |
| argc | — | Неотрицательное число, представляющее количество аргументов, переданных программе из окружения, в котором запустили программу. |
| argv | — | Указатель на первый элемент массива argc + 1 указателей, последний из которых равен null, а предыдущие, если существуют, указывают на [многобайтовые строки с завершающим нулём](https://ru.cppreference.com/w/cpp/string/multibyte), которые представляют аргументы, переданные программе из окружения, в котором её запустили. Если argv[0] не пустой указатель (или, что тоже самое, argc > 0), он указывает на строку, которая представляет имя, использованное при запуске программы, или пустую строку. |
| body | — | Тело функции main. |

Имена argc и argv могут быть произвольными, так же как и запись типов этих параметров: int main(int ac, char\*\* av) так же допустимо.

Самая распространённая форма main(), определённая реализацией, имеет третий аргумент (дополнительно к argc и argv), типа char\*[], указывающий на [на массив указателей на переменные среды выполнения](http://pubs.opengroup.org/onlinepubs/9699919799/functions/exec.html).

### Объяснение

Функция main вызывается при старте программы после [инициализации](https://ru.cppreference.com/w/cpp/language/initialization) нелокальных объектов со статическим [классом памяти](https://ru.cppreference.com/w/cpp/language/storage_duration). Это точка входа в программу, которая исполняется в *гостевом* окружении (то есть с операционной системой). Точки входа в *автономные* программы (загрузчики, ядра ОС, и т.п.) зависят от реализации.

Параметры функции main в варианте с двумя параметрами позволяют передать произвольные многобайтовые строки из среды выполнения (обычно известных как *аргументы командной строки*), указатели argv[1] .. argv[argc-1] ссылаются на первые символы этих строк. argv[0] это указатель на первый символ многобайтовой строки с завершающим нулём, которая содержит имя, используемое при вызове программы (или пустую строку "", если это не поддерживается средой выполнения). Эти строки изменяемые, хотя их изменения не распространяются назад в среду выполнения: они могут использоваться, например, в [std::strtok](https://ru.cppreference.com/w/cpp/string/byte/strtok). Размер массива, на который указывает argv, равен, по меньшей мере, argc+1, и последний элемент массива argv[argc] гарантированно является пустым указателем.

Функция main обладает следующими специальными свойствами:

1) Она нигде не может быть использована в программе,

а) в частности её нельзя вызывать рекурсивно;

б) нельзя взять её адрес.

2) Её нельзя объявлять и нельзя перегружать: фактически имя main зарезервировано в глобальном пространстве имён для функций (хотя это имя может быть использовано для именования классов, пространств имён, перечислений и любых сущностей не в глобальных пространствах имён, за исключением того, что функция с именем 'main' не может быть объявлена со [связыванием](https://ru.cppreference.com/mwiki/index.php?title=cpp/language/%D1%81%D0%B2%D1%8F%D0%B7%D1%8B%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5%D0%BC&action=edit&redlink=1) для языка C в любом пространстве имён (начиная с C++17)).

3) Её нельзя объявить как удалённую или определить со [связыванием](https://ru.cppreference.com/mwiki/index.php?title=cpp/language/%D1%81%D0%B2%D1%8F%D0%B7%D1%8B%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5%D0%BC&action=edit&redlink=1) для языка C (начиная с C++17), [inline](https://ru.cppreference.com/w/cpp/language/inline), [static](https://ru.cppreference.com/w/cpp/language/static) или [constexpr](https://ru.cppreference.com/w/cpp/language/constexpr).

4) В теле функции main не обязателен [оператор return](https://ru.cppreference.com/w/cpp/language/return): при завершении функции main без оператора return эффект будет тот же самый, как при выполнении return 0;.

5) Выполнение return (или неявного return при достижении конца функции main) эквивалентно нормальному выходу из функции (которое уничтожает объекты с автоматическим классом памяти) с последующим вызовом [std::exit](http://ru.cppreference.com/w/cpp/utility/program/exit) с тем же самым аргументом, который был передан в [return](https://ru.cppreference.com/w/cpp/language/return). ([std::exit](http://ru.cppreference.com/w/cpp/utility/program/exit) уничтожает статические объекты и завершает программу).

6) (начиная с C++14) Тип возвращаемого значения функцией main не может быть выведен (auto main() {...} не разрешено).

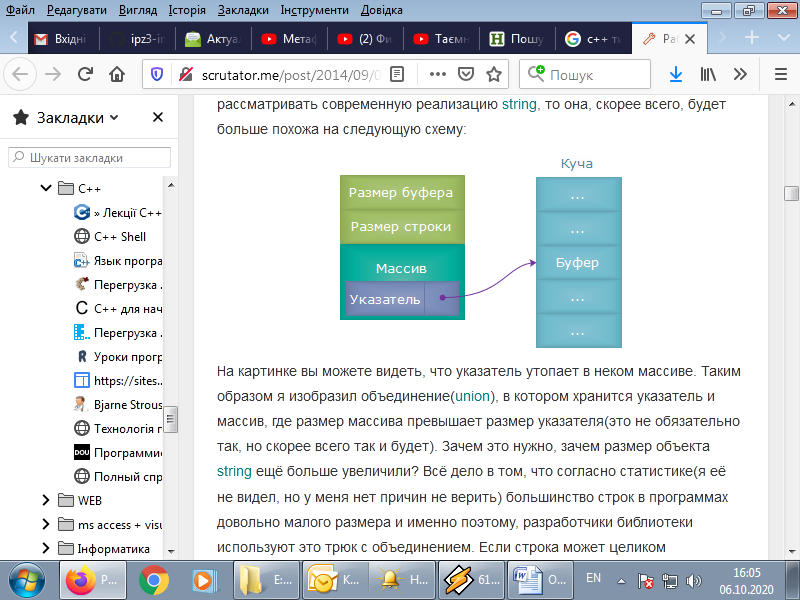
Окрім того стандартом мови забороняється будь-яке використання функції main:

**6.6.1 main function [basic.start.main]**  
3 The function main shall not be used within a program.

1. **Спосіб збереження рядка string**

За матеріалами <http://scrutator.me/post/2014/09/02/cpp_strings_basics.aspx>

В статті розглядається спосіб збереження рядка **string**



Масив – це позначене місце зберігання даних класу **string**, якщо кількість символів не перевищує розміру масиву. В іншому випадку в "купі" виділяється буфер, розмір, якого більший за дійсний розмір рядка, і покажчик вказує адресу цього буферу.