



*Белорусский государственный университет  
информатики и радиоэлектроники*

---

## **Технологии разработки программного обеспечения**

**Преподаватель: Меженная Марина Михайловна**

К.Т.Н., доцент,

доцент кафедры инженерной психологии и эргономики  
а 609-2

[mezhennaya@bsuir.by](mailto:mezhennaya@bsuir.by)



---

*Кафедра инженерной психологии и эргономики*

# Курсовая работа по дисциплине «Технологии разработки программного обеспечения»: общая информация

---

## Как проходит защита курсовой работы?

1 этап – Вы лично демонстрируете работу программы на своем ноутбуке (во избежание проблем совместимости версий и т.д.)

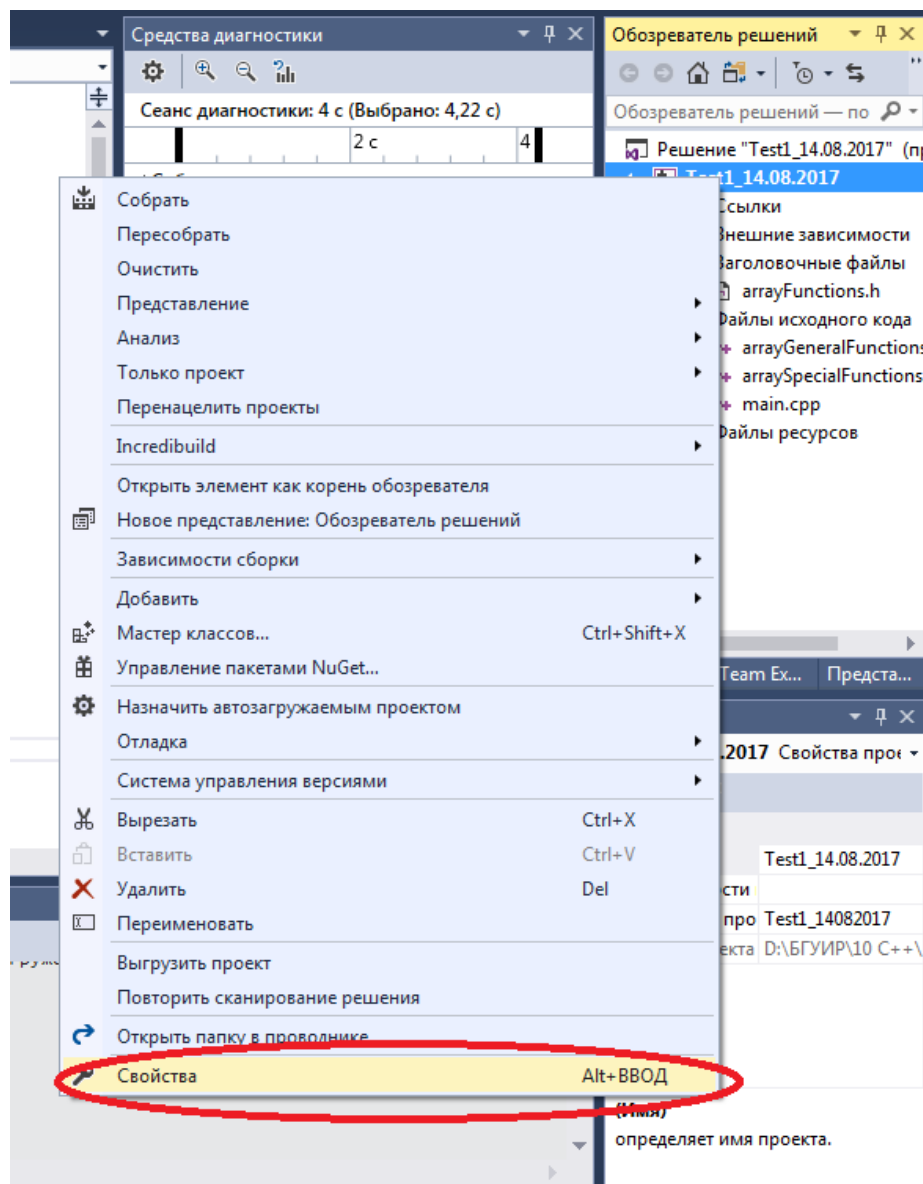
2 этап – показываете аккуратно оформленную распечатанную пояснительную записку (вкладывать лист-задание обязательно!)

3 этап – code review (просмотр кода), отвечаете на вопросы по Вашему коду.

Диск с программой НЕ НУЖЕН.



# Как сделать, чтобы release-версия гарантированно запускалась на других ПК с Windows (без Visual Studio)



# Как сделать, чтобы release-версия гарантированно запускалась на других ПК с Windows (без Visual Studio)

Конфигурация: **Активная (Release)** Платформа: **Активная (Win32)** Диспетчер конфигураций...

Свойства конфигурации

- Общие
- Отладка
- Каталоги VC++
- С/С++
  - Общие
  - Оптимизация
  - Препроцессор
  - Создание кода**
  - Язык
  - Предварительно откомпилированные файлы
  - Выходные файлы
  - Информация об исходном коде
  - Дополнительно
  - Все параметры
  - Командная строка
- Компоновщик
- Инструмент манифеста
- Генератор XML-документа
- Информация об исходном коде
- События сборки
- Настраиваемый этап сборки
- Анализ кода

Включить объединение строк	
Включить минимальное перестроение	Нет (/Gm-)
Включить C++ исключения	Да (/EHsc)
Проверка меньших типов	Нет
Основные проверки времени выполнения	По умолчанию
<b>Библиотека времени выполнения</b>	<b>Многопоточная (/MT)</b>
Выравнивание членов структуры	Многопоточная (/MT)
Проверка безопасности	Многопоточная отладка (/MTd)
Защита потока управления	Многопоточный DLL (/MD)
Включить компоновку на	Многопоточная отладка DLL (/MDd)
Включить создание параллельных потоков	<наследовать от родителя или от значений по умолчанию для проекта>
Включить расширенный набор инструкций	Не задано
Модель вычислений с плавающей точкой	Точный (/fp:precise)
Включить исключения для вычислений с плавающей точкой	
Создать образ с обновлением	

**Библиотека времени выполнения**  
определяет библиотеку времени выполнения для компоновки. (/MT, /MTd, /MD, /MDd)

для релизной - /MT  
(для отладочной версии /MTd)

OK Отмена **Применить**

# Литература по C++

Министерство образования Республики Беларусь  
Учреждение образования  
«Белорусский государственный университет  
информатики и радиоэлектроники»  
Факультет компьютерного проектирования  
Кафедра инженерной психологии и эргономики

М. М. Меженная

## **ОСНОВЫ КОНСТРУИРОВАНИЯ ПРОГРАММ. КУРСОВОЕ ПРОЕКТИРОВАНИЕ**

Рекомендовано УМО по образованию  
в области информатики и радиоэлектроники для специальностей  
1-40 05 01 «Информационные системы и технологии (по направлениям)»,  
1-58 01 01 «Инженерно-психологическое обеспечение  
информационных технологий»  
в качестве пособия



Минск БГУИР 2019

М.М. Меженная  
**Основы конструирования программ  
(актуально для Технологии  
разработки программного  
обеспечения). Курсовое  
проектирование**

# Курсовая работа по дисциплине «Технологии разработки программного обеспечения»: общая информация

---

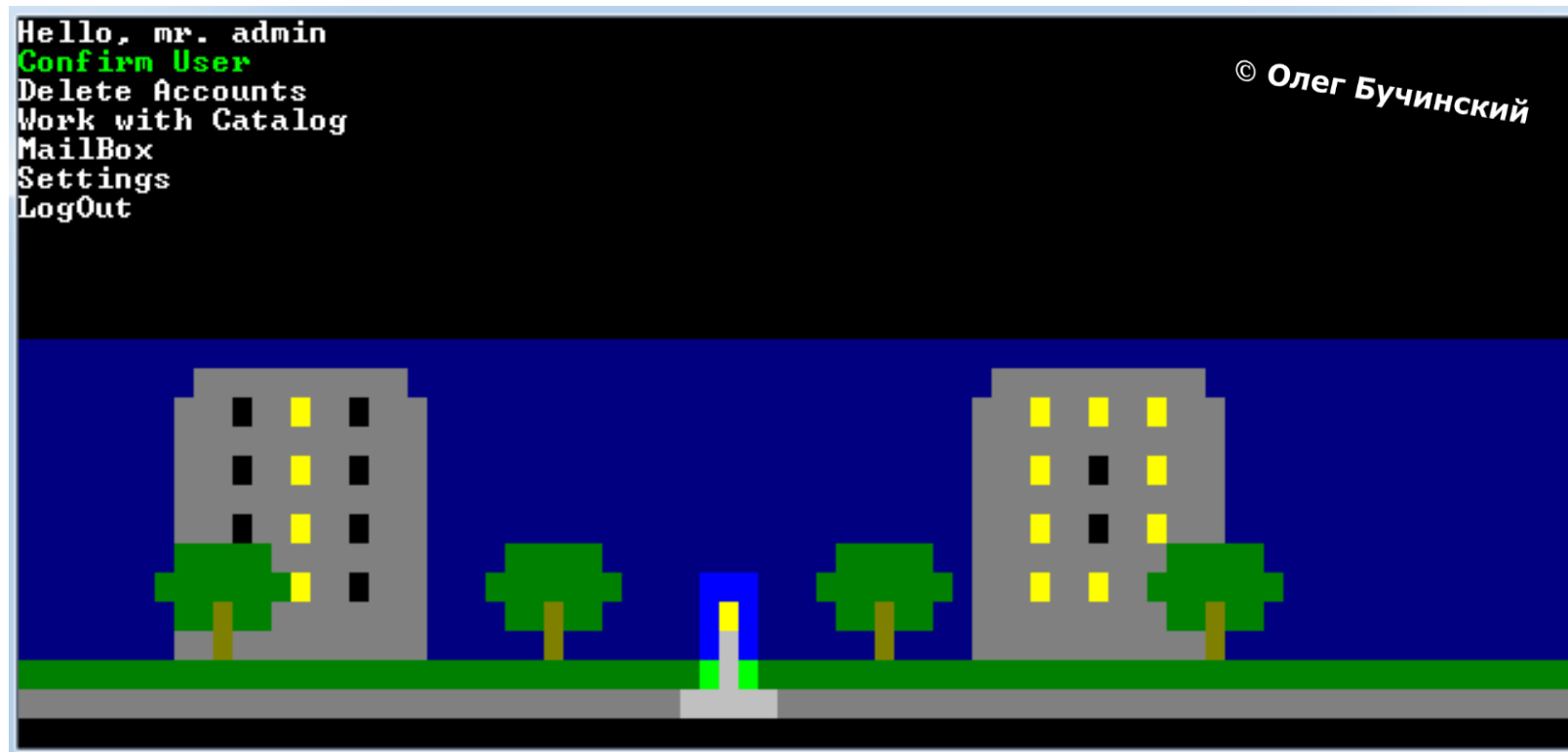
1. Тема курсовой работы выбирается из списка, приведенного в одноименном файле.
2. Язык программирования C++ **(да, это критично, поэтому Java и C# не подойдут)**.
3. Среда разработки Microsoft Visual Studio версии 2010 и выше **(нет, Builder использовать нельзя)**.
4. Вид приложения — консольное **(сосредоточьтесь на чистом коде)**.
5. Парадигма программирования — процедурная *(по согласованию с преподавателем допускается реализация программы в рамках объектно-ориентированной парадигмы программирования)*.

# Курсовая работа по дисциплине «Технологии разработки программного обеспечения».

## Исходные данные к работе:

---

Даже в консольном приложении есть место для творчества:



# Курсовая работа по дисциплине «Технологии разработки программного обеспечения».

## Исходные данные к работе:

---

Даже в консольном приложении есть место для творчества:





# Курсовая работа по дисциплине «Технологии разработки программного обеспечения».

## Исходные данные к работе:

Даже в консольном приложении есть место для творчества:

Page 1/4

© Олег Бучинский

Add

Sort

Search

Article W	Name 1	Manufacturer 1	Price 1	Size 1	Count 1
Article W	Name 2	Manufacturer 2	Price 2	Size 2	Count 2
Article W	Name 3	Manufacturer 3	Price 3	Size 3	Count 3

# Курсовая работа по дисциплине «Технологии разработки программного обеспечения».

## Исходные данные к работе:

---

Даже основываясь на примерах код можно улучшать:

// Определение количества структур в файле (при необходимости)

```
int getCountOfStructuresInFile(string file_path)
```

```
{
```

```
    ifstream file(file_path, ios::in); // Открыли текстовый файл для чтения
```

```
    int number_of_strings = 0;
```

```
    if (file.is_open())
```

```
    {
```

```
        while (file.ignore(numeric_limits<streamsize>::max(), '\n'))
```

```
            number_of_strings++;
```

```
    }
```

```
    file.close();
```

```
    return number_of_strings;
```

```
}
```

```
string buffer;  
while (getline(file, buffer))  
    number_of_strings++;
```

© Розанов Иван



## Исходные данные к работе (продолжение):

---

6.Способ организации данных – структуры (struct) (*либо поля соответствующих классов в случае объектно-ориентированного программирования*).

7. Способ хранения данных – файлы (*по согласованию с преподавателем допускается подключение баз данных*).

8.Каждая логически завершенная задача программы должна быть реализована в виде функции (*метода в случае объектно-ориентированного программирования*).

9.Построение программного кода должно соответствовать правилам, определенным в документе «C++ Code Convention».

10.Текст пояснительной записки оформляется в соответствии со стандартом предприятия СТП 01–2017.



# Напутствие

---

Каждая курсовая работа, включая программную часть, **должна быть уникальной**, т.к. подразумевает индивидуальную работу над своим заданием (об уникальности вашей работы будут свидетельствовать соответствующие тематике имена переменных, констант, функций; комментарии; проработка исключительных ситуаций; добавление «своих» функциональных возможностей и др.).

Допускается использовать классы **string** и **vector**, библиотеку **algorithm**.



# Напутствие

---

**Используйте систему контроля версий и облачные репозитории (Git + GitHub) для отслеживания изменений и резервного копирования программы на случай неисправности Вашего ПК!**



# Функциональные требования к курсовой работе

---

**Первым этапом** работы программы является авторизация – предоставление прав доступа.

В рамках данного этапа необходимо считать данные из файла с учетными записями пользователей следующего вида:

- login;
- password;
- role (данное поле служит для разделения в правах администраторов и пользователей);
- access (данное поле служит для подтверждения или блокировки администратором учетных записей).

# Функциональные требования к курсовой работе

---

## Авторизация:

После ввода пользователем своих персональных данных (логина и пароля) и сверки со считанной из файла информацией необходимо предусмотреть возможность входа:

- в качестве администратора (в этом случае, например, role = 1 && access = 1)
- в качестве пользователя (в этом случае, например, role = 0 && access = 1).

**Важно:** Если файл с учетными записями пользователей не существует, то необходимо программно создать его и записать учетные данные администратора.

# Функциональные требования к курсовой работе

---

**Регистрация новых пользователей может осуществляться двумя способами:**

- 1. администратором** в режиме работы с учетными записями пользователей (т.е. администратор сам создает для пользователя аккаунт);
- 2. самим пользователем** путем ввода желаемых логина и пароля и ожидания подтверждения администратором новой учетной записи. Для реализации этого способа в структуре учетных записей пользователей существует поле `access` (например, `access = 0` по умолчанию при попытке зарегистрироваться, администратор меняет значение на `access = 1` и тем самым подтверждает новую учетную запись: пользователь может осуществить вход в систему).



# Функциональные требования к курсовой работе

---

**Вторым** этапом работы программы является работа с данными, которые становятся доступными только после прохождения авторизации. Данные хранятся в отдельном файле и имеют вид, описанный подробно в каждом варианте к курсовой работе (информация о студентах, сотрудниках предприятия и т.д.).

Для работы с учетными записями и собственно данными должны быть предусмотрены два функциональных модуля: модуль администратора и модуль пользователя.

# Функциональные требования к курсовой работе

---

Модуль **администратора** включает следующие подмодули (с указанием функциональных возможностей):

## **1. Управление учетными записями пользователей:**

- просмотр всех учетных записей;
- добавление новой учетной записи;
- редактирование учетной записи;
- удаление учетной записи.

# Функциональные требования к курсовой работе

---

Модуль **администратора** включает следующие подмодули (с указанием функциональных возможностей):

## 2.Работа с данными:

а) режим редактирования:

- просмотр всех данных;
- добавление новой записи;
- удаление записи;
- редактирование записи;

б) режим обработки данных:

- выполнение индивидуального задания;
- поиск данных (как минимум по трем различным параметрам);
- сортировка (как минимум по трем различным параметрам).

# Функциональные требования к курсовой работе

---

Модуль **пользователя** включает следующие подмодули (с указанием функциональных возможностей):

## **Работа с данными:**

- просмотр всех данных;
- выполнение индивидуального задания;
- поиск данных (как минимум по трем различным параметрам);
- сортировка (как минимум по трем различным параметрам).

# Модульная структура программы

Пример:

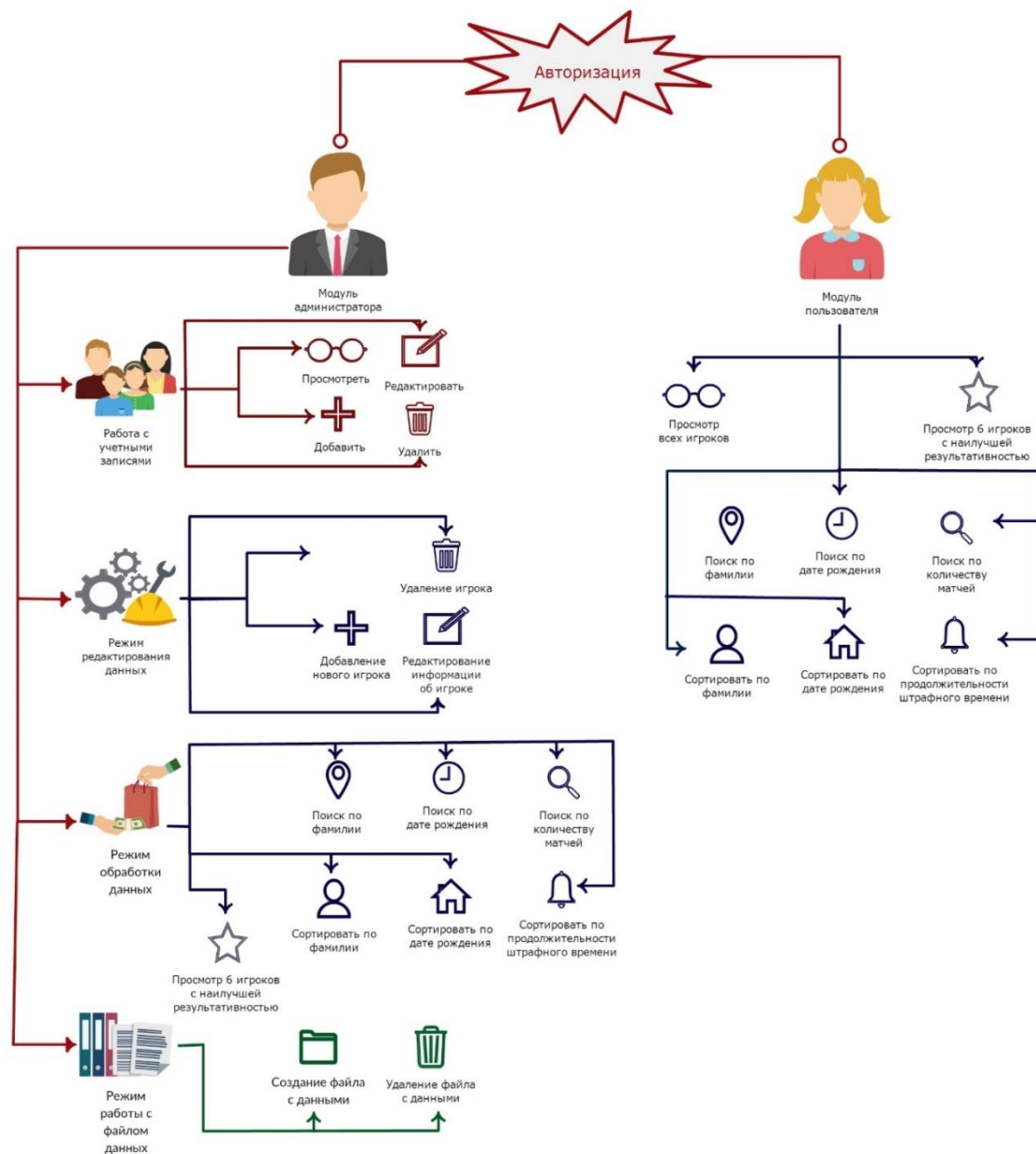
*(не для копирования  
в Вашу курсовую!)*



# Модульная структура программы

Еще пример:

*(опять же не для копирования в Вашу курсовую!)*



# Примеры

---

Для реализации перечисленных модулей/подмодулей необходимо создавать меню с соответствующими пунктами .

```
2
----- Вход пользователя -----

Введите логин - пользователь
Введите пароль- 12345

----- Меню пользователя -----

1.Просмотр данных;
2.Поиск и фильтрация;
3.Сортировка;
0.Выход в главное меню.
```

Пример авторизации и меню для пользователя

# Примеры

---

```
----- Поиск студента -----  
  
ФИО искомого студента Петр  
  
Информация о найденных студентах  
  
-----  
| № |      ФИО      | № гр. | Матем. | Физ | Инф. | Ср.б. | Актив. | Стипендия |  
-----  
| 1 | Петров      | 12    | 4      | 5    | 5    | 4.67  | 1      | 7500.00  |  
-----  
| 2 | Петрова     | 21    | 5      | 3    | 5    | 4.33  | 0      | 0.00     |  
-----
```

Пример поиска





# Примеры

```
-----Меню сортировки -----
1.По алфавиту;
2.По среднему баллу;
3.По стипендии
0.Назад.
1
```

Сортированная информация

№	ФИО	№ гр.	Матем.	Физ	Инф.	Ср.б.	Актив.	Стипендия
1	Антонов	21	3	5	5	4.33	1	5000.00
2	Иванов	12	5	5	5	5.00	0	6250.00
3	Петров	12	4	5	5	4.67	1	7500.00
4	Петрова	21	5	3	5	4.33	0	0.00
5	Сидоров	11	4	5	4	4.33	1	7500.00
6	Чернов	23	4	4	5	4.33	0	6250.00

Пример сортировки

# Функциональные требования к курсовой работе

---

## Предусмотреть:

- обработку исключительных ситуаций:
  - введенные пользователем данные не соответствуют формату поля (*например, символы в числовом поле*)
  - введенные пользователем данные некорректны (*например, отрицательная цена товара*)
  - файл с данными для чтения не существует
  - ничего не найдено по результатам поиска
  - номер удаляемой записи выходит за пределы массива/вектора
  - логин новой учетной записи уже существует

# Функциональные требования к курсовой работе

---

## Предусмотреть:

- возможность возврата назад (навигация);
- запрос на выполнение необратимых действий, а именно, подтверждение удаления вида «Вы действительно хотите удалить файл (запись)?»;
- обратную связь с пользователем, например, вывод сообщения об успешности создания файла/удаления записи/ и т.д.

# Требования к программной реализации

---

1. Все переменные и константы должны иметь осмысленные имена в рамках тематики варианта к курсовой работе.
2. Имена функций должны быть осмысленными и строиться по принципу глагол+существительное. Если функция выполняет какую-либо проверку и возвращает результат типа `bool`, то ее название должно начинаться с глагола `is` (например, `isFileExist`, `isUnicLogin`).
3. ~~goto~~ под запретом.

## Требования к программной реализации

---

4. Код не должен содержать неименованных числовых констант (так называемых «магических» чисел), неименованных строковых констант (например, имен файлов и др.). Подобного рода информацию следует выносить в глобальные переменные с атрибутом `const`. По правилам хорошего стиля программирования тексты всех информационных сообщений, выводимых пользователю в ответ на его действия, также оформляются как константы.

5. Код необходимо комментировать (как минимум в части объявления структур, массивов/векторов, заголовков функций, нетривиальной логики).

# Требования к программной реализации

---

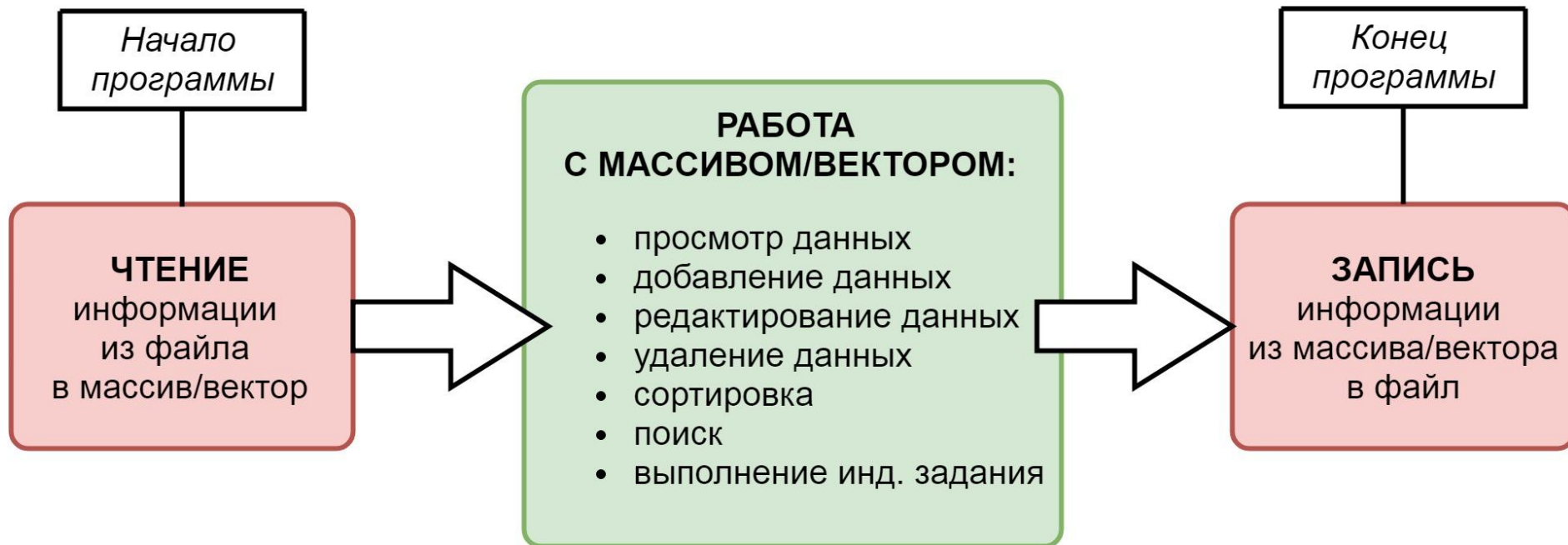
6. Код не должен дублироваться – для этого существуют функции!

7. Одна функция решает только одну задачу (например, не допускается в одной функции считывать данные из файла и выводить их на консоль – это две разные функции!). При этом внутри функции возможен вызов других функций.

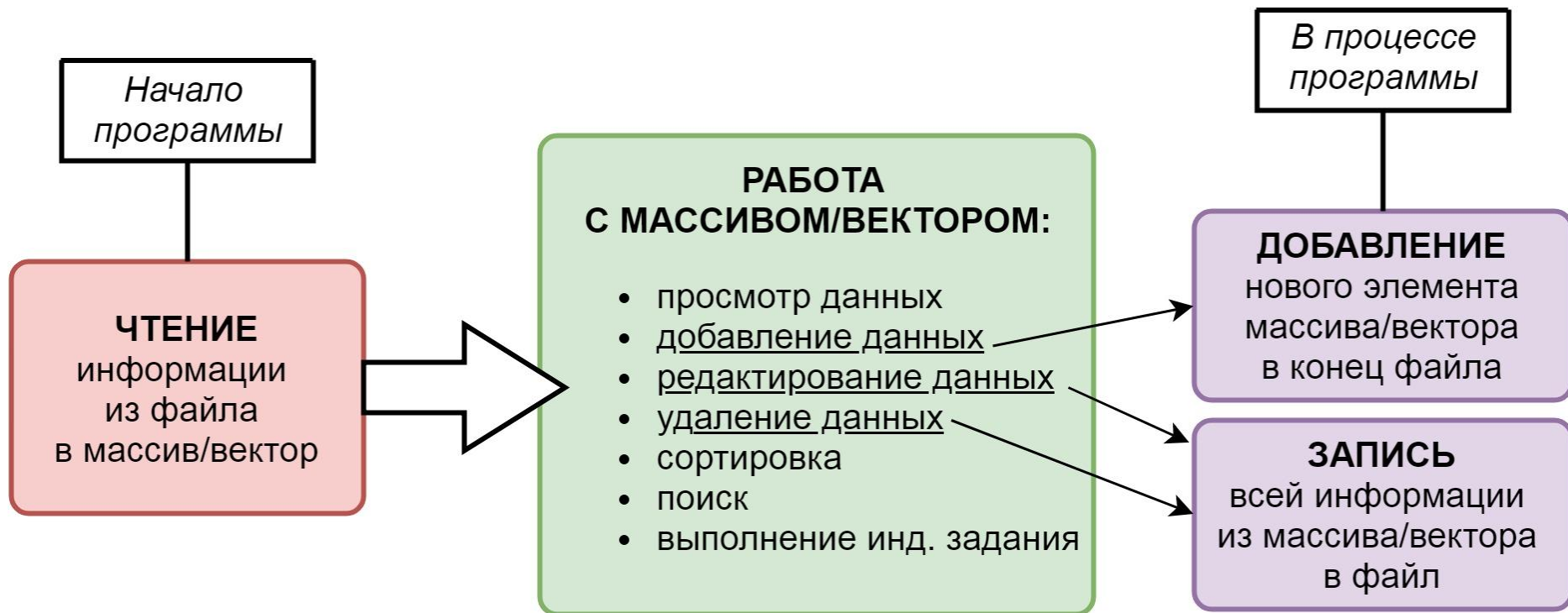
8. **Выполнение операций чтения/записи в файл должно быть сведено к минимуму** (т.е. после однократной выгрузки данных из файла в массив/вектор дальнейшая работа ведется с этим массивом/вектором, а не происходит многократное считывание данных из файла в каждой функции).

# Выполнение операций чтения/записи в файл должно быть сведено к минимуму: способ 1

---



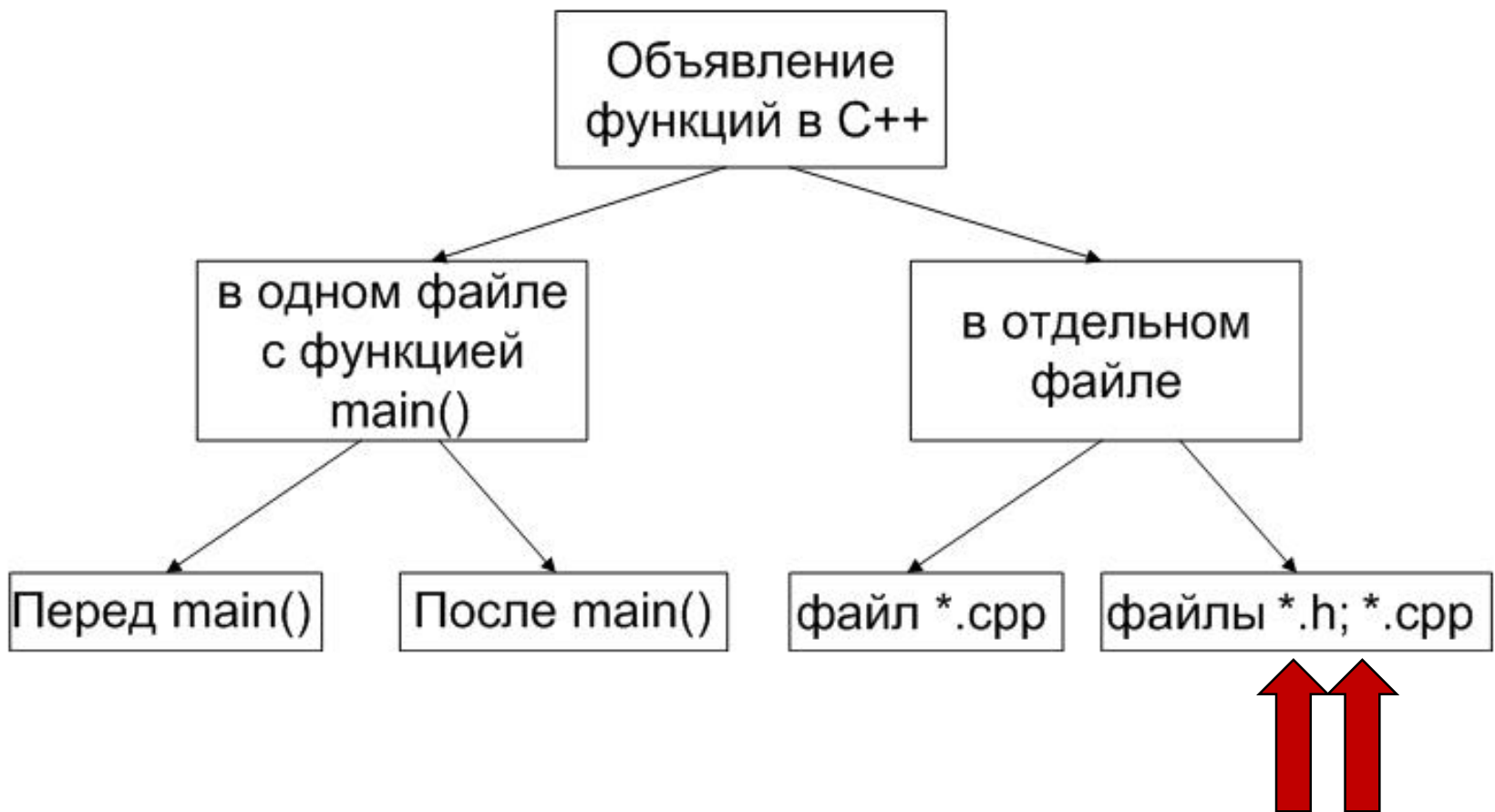
# Выполнение операций чтения/записи в файл должно быть сведено к минимуму: способ 2





# Выносите код логически независимых модулей в отдельные .cpp файлы и подключайте их с помощью заголовочных .h файлов

---



**Код должен быть логичным и понятным...  
как минимум для Вас самих!**

---



# Способ улучшения кода - refactoring

---



# Структура пояснительной записки

---

Титульный лист (*см. образец на гугл-диске или персон.странице в Методических указаниях к выполнению курсовой работы*)

Задание по курсовой работе (*будет выдано*)

Содержание

1. Требования к программе
2. Конструирование программы
  - 2.1 Разработка модульной структуры программы
  - 2.2 Выбор способа организации данных
  - 2.3 Разработка перечня пользовательских функций программы

# Структура пояснительной записки

---

- 3. Разработка алгоритмов работы программы
  - 3.1 Алгоритм функции main
  - 3.2 Алгоритм функции ... *(по вашему выбору)*
  - 3.3 Алгоритм функции ... *(по вашему выбору)*
- 4. Описание работы программы (см. пояснения ниже)
  - 4.1 Авторизация
  - 4.2 Модуль администратора
  - 4.3 Модуль пользователя
  - 4.4 Исключительные ситуации

Приложение (обязательное): листинг кода с комментариями (приводится ВЕСЬ код с Вашим авторским форматированием и комментариями).

# Структура пояснительной записки

---

**Требования** к программе включают:

- полный текст Вашего варианта задания,
- исходные данные для курсовой работы из документа *Методические указания к выполнению курсовой работы*,
- функциональные требования к конкретно вашей курсовой работе (рекомендуется взять за основу материал из документа *Методические указания к выполнению курсовой работы* и расширить его для своей темы, например, прописать индивидуальное задание, разновидности поиска и сортировки, + конкретизировать возможные исключительные ситуации),
- требования к программной реализации (из документа *Методические указания к выполнению курсовой работы*).

# Структура пояснительной записки

---

Не пишите в требования то, что Вы точно не реализуете в действительности. Для надежности после завершения работы над программой вернитесь к требованиям и проверьте соответствие вашей программы тому, что Вы заявляли.

# Структура пояснительной записки

---

**Разработка структуры программы** подразумевает графическое представление структуры программы с указанием модулей, подмодулей и их функциональных возможностей. Пример приведен ранее на одноименном слайде.

*Для объектно-ориентированного программирования в данном подразделе приводится UML диаграмма классов!*

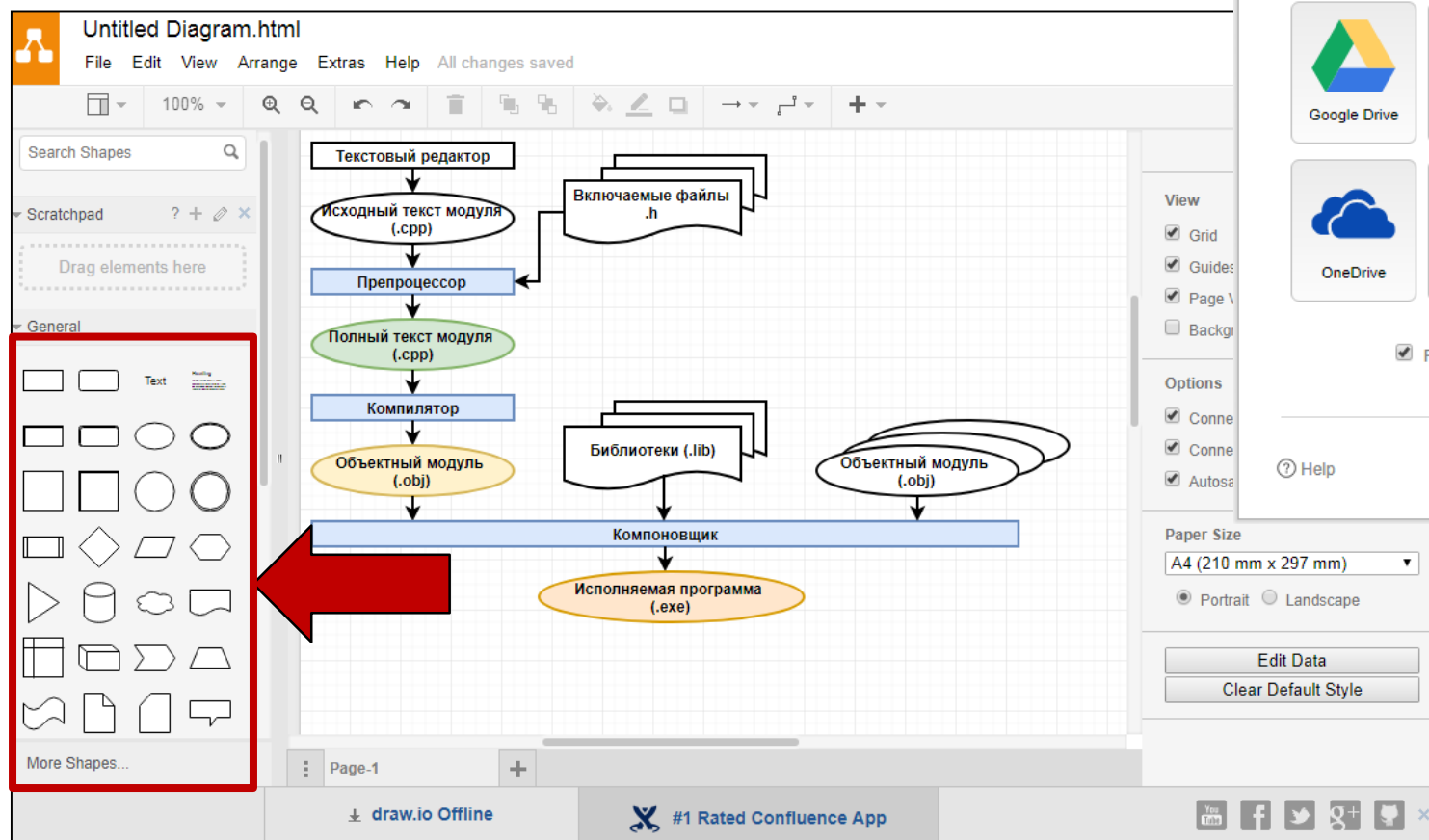
Графическая среда может быть любой; в качестве рекомендации обращаю ваше внимание на online-редакторы, например:

[http://www.newart.ru/htm/flash/risovalka\\_42.php](http://www.newart.ru/htm/flash/risovalka_42.php)



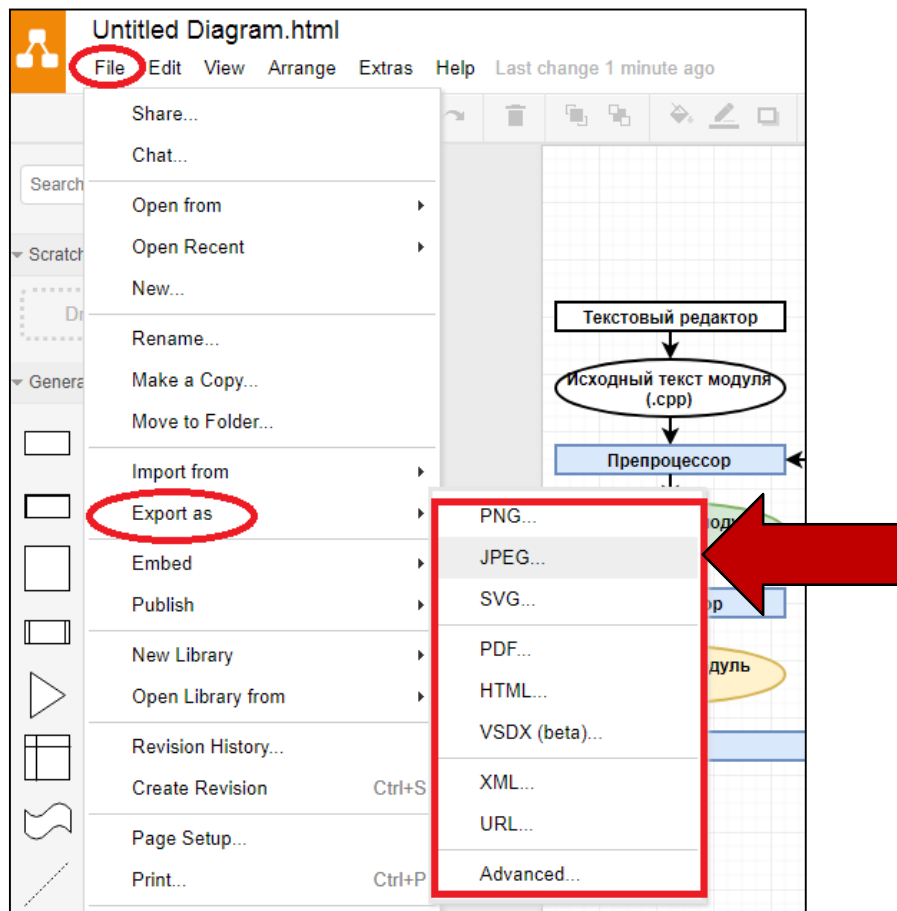
# Векторный редактор диаграмм:

[http://www.newart.ru/htm/flash/risovalka\\_42.php](http://www.newart.ru/htm/flash/risovalka_42.php)



# Векторный редактор диаграмм:

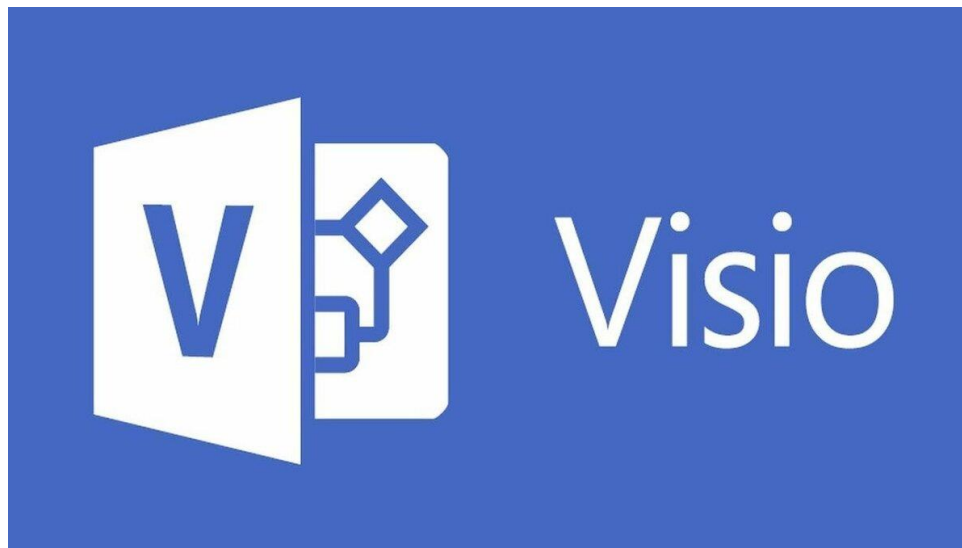
[http://www.newart.ru/htm/flash/risovalka\\_42.php](http://www.newart.ru/htm/flash/risovalka_42.php)



# Разрабатываем алгоритмы:

---

Используйте **Microsoft Visio**



# Структура пояснительной записки

---

## Выбор способов организации данных:

- в качестве выбора способа описания входных данных: приводится описание типов struct (1. для учетных записей пользователей, 2. для данных) с указанием конкретных полей. *В случае объектно-ориентированного программирования приводятся названия предполагаемых классов и содержащихся в них полей. При работе с базой данных дополнительно приводится структура таблиц.*
- в качестве способа объединения входных данных: указывается использование массивов(стат./дин.)/векторов, а также их выбранная область видимости (локальные/глобальные).

## Структура пояснительной записки

---

**Разработка перечня пользовательских функций программы** подразумевает перечисление и краткие комментарии прототипов функций, необходимых для реализации программы.

Прототипы функций рекомендуется разбить на тематические группы в соответствии с модульной структурой программы.

Выносите эти тематические группы в отдельные .cpp файлы и подключайте их через заголовочные .h файлы.

*В случае использования объектно-ориентированного программирования приводятся методы для классов.*

Код функций (методов) не приводится, т.к. на этом этапе он еще не существует.

# Структура пояснительной записки

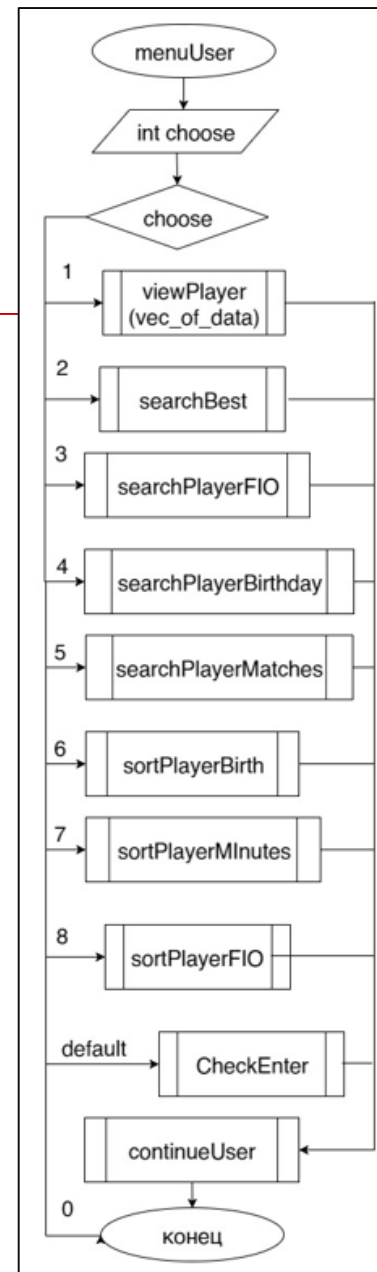
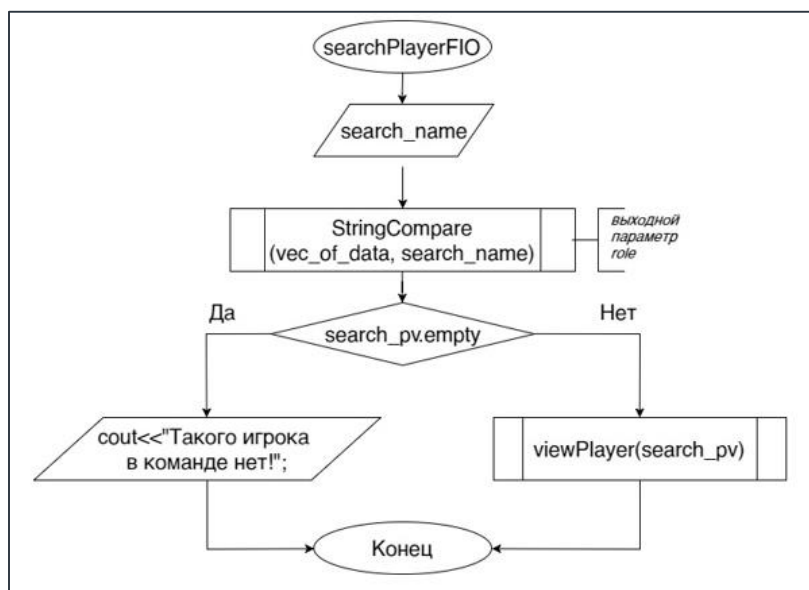
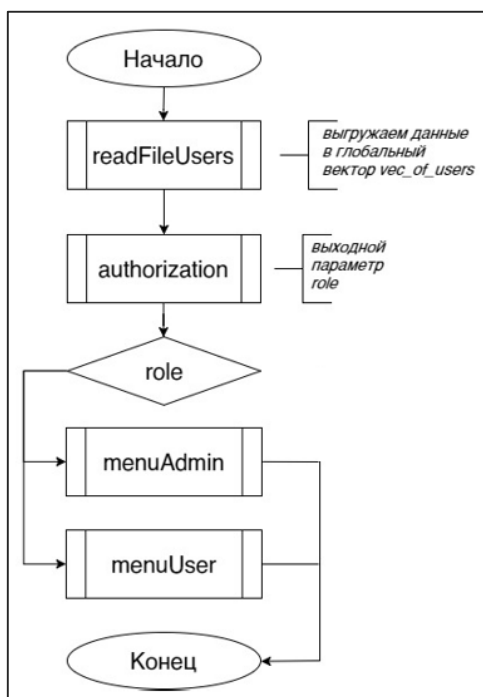
---

**Разработка алгоритмов работы программы** включает блок-схемы алгоритмов с кратким словесным описанием их работы для функции main и двух любых пользовательских функций (*в случае использования объектно-ориентированного программирования разрабатываются алгоритмы для двух любых методов классов + алгоритм функции main*).

Обратите внимание: так как алгоритмы разрабатываются до непосредственного кодирования программы, то они не могут содержать просто копии строчек кода. Алгоритм может содержать словесные инструкции с упоминанием имен структур, массивов/векторов, функций.

# Структура пояснительной записки

## Разработка алгоритмов работы программы



# Структура пояснительной записки

---

**Главный критерий хорошего алгоритма:** если вы можете дать ваш алгоритм коллеге (одногруппнику) и он по вашему алгоритму без дополнительных разъяснений с вашей стороны напишет код, значит алгоритм достиг своей цели.

Признаком хорошего алгоритма также является его относительная компактность.



# Структура пояснительной записки

---

Алгоритм должен быть оформлен согласно ГОСТ 19.701-90 Схемы алгоритмов, программ, данных и систем (вспомогательная информация по данной теме доступна по ссылке <http://bit.ly/1skqzkw>).

Графическая среда для разработки алгоритмов может быть любой, например, тот же online-редактор:

*[http://www.newart.ru/htm/flash/risovalka\\_42.php](http://www.newart.ru/htm/flash/risovalka_42.php)*

# Структура пояснительной записки

---

**Описание работы программы** подразумевает краткое словесное описание работы программы со скриншотами консоли.

**Объем пояснительной записки не регламентируется, основной критерий в данном случае – это качество, а не количество.**

# Оформляем документы в Word (например, пояснительную записку по стандарту):

**bestzmest.by**



## РЕДАКТОР ДОКУМЕНТОВ

О РЕСУРСЕ

О КОМПАНИИ

ВОЗМОЖНОСТИ

НОВОСТИ

Загрузка документа

Формирование требований

Список требований

Список документов

Личная информация

Информация о компании

Состояние счета

## Список документов

- Any -

Новый

В обработке

Готов

Ошибка обработки

Поиск

Удалить...

Обработать

<input type="checkbox"/>	Название	Размер	Дата загрузки	Состояние	Требование	Percent	
<input type="checkbox"/>	poyasnitelnaya_zapiska1.docx	158.27 KB	13.09.2019	В обработке	STP BSUIR	2	



Меженная М.М.

# Оформляем документы в Word (например, пояснительную записку по стандарту):

**bestzmest.by**

**1 Готовим документ в соответствии с минимальными требованиями (это обязательно, иначе не получится корректный результат)**

**РЕДАКТОР ДОКУМЕНТОВ**

О РЕСУРСЕ О КОМПАНИИ ВОЗМОЖНОСТИ НОВОСТИ

**Загрузка документов**

Требования: STP BSUIR

\*  
☐ poyasnitelnaya\_zapiska1.docx

Удалить Начать обработку

Обзор... Файл не выбран. Загрузить

**ВНИМАНИЕ**

**Требования к входному документу для корректной работы основных функций:**

- загружать нужно всё одним файлом без титульного листа, листа задания, реферата и содержания, созданного в ручную
- заголовки первого уровня (главы и разделы) должны быть набраны БОЛЬШИМИ (ЗАГЛАВНЫМИ) БУКВАМИ (комбинация "SHIFT+F3" делает выделенный фрагмент текста заглавными буквами), Документ должен содержать как минимум один такой заголовок

# Оформляем документы в Word (например, пояснительную записку по стандарту):

**bestzmest.by**

**2 Загрузка  
документа:**  
Обзор →  
Загрузить →  
Активировать  
чек-бокс файла →  
Начать обработку

**РЕДАКТОР ДОКУМЕНТОВ**

О РЕСУРСЕ О КОМПАНИИ ВОЗМОЖНОСТИ НОВОСТИ

**Загрузка документов**

Требования: STP BSUIR

ВНИМАНИЕ

Требования к входному документу для корректной работы основных функций:

- загружать нужно всё одним файлом без титульного листа, листа задания, реферата и содержания, созданного в ручную
- заголовки первого уровня

Файл: poyasnitelnaya\_zapiska1.docx

Удалить Начать обработку

Обзор... Файл не выбран. Загрузить

2

# Оформляем документы в Word (например, пояснительную записку по стандарту):

**bestzmest.by**

**3 Список документов:**  
В разделе Готов  
дождаться  
окончания  
обработки и  
скачать результат

**РЕДАКТОР ДОКУМЕНТОВ**


О РЕСУРСЕ О КОМПАНИИ ВОЗМОЖНОСТИ НОВОСТИ

Список документов

- Агу - Новый В обработке Готов Ошибка обработки

Поиск

Удалить... Обработать

	Название	Размер	Дата загрузки	Состояние	Требование	Percent	
<input type="checkbox"/>	poyasniteinaya_zapiska1.docx	158.27 KB	13.09.2019	Готов	STP BSUIR	100	

3

# Оформляем документы (пояснительную записку):

**bestzmest.by**

**ДО**

## ТРЕБОВАНИЕ К ПРОГРАММЕ

Полный текст варианта задания:

Программа предоставляет сведения о товарах, имеющихся на складе: наименование товара; количество единиц товара; цена единицы товара; дата поступления товара на склад; ФИО зарегистрировавшего товар.

Индивидуальное задание: вывести в алфавитном порядке список товаров, хранящихся более  $x$  месяцев, стоимость которых превышает  $y$  рублей ( $x, y$  вводятся с клавиатуры).

Кроме того, реализовать авторизацию для входа в систему, функционал администратора и пользователя

### Исходные данные

1. Разработка программы учета товаров на складе.
2. Язык программирования C++.
3. Среда разработки Microsoft Visual Studio.
4. Вид приложения – консольное.
5. Парадигма программирования – процедурная.
6. Способ организации данных – структуры (struct).
7. Способ хранения данных – файлы.
8. Каждая логически завершенная подзадача программы реализована в виде отдельной функции.
9. Построение программного кода соответствует соглашению о коде «C++ Code Convention».
10. К защите курсовой работы представляются: консольное приложение и пояснительная записка.
11. Текст пояснительной записки оформляется в соответствии со стандартом предприятия СТП 01–2017.
12. Используются классы string, vector и библиотека algorithm.

### Функциональные требования

Первый этап – авторизация. В рамках этого этапа считаны данные из файла с учетными записями пользователей следующего вида:

- login;
- password;
- role (0 – пользователь, 1 – администратор);
- access (поле для подтверждения учетной записи или ее блокировки);

По умолчанию access = 0 при попытке зарегистрироваться; далее администратор меняет значение на access = 1 и тем самым подтверждает новую учетную запись: пользователь может осуществить вход в систему.

По соображениям безопасности вводимый пароль отображается звездочками.

наименование,  
количество товара на складе,  
его цена,  
дата поступления на склад,  
ФИО зарегистрировавшего товар.

Работа с данными может осуществляться через два предусмотренных модуля:

Модуль администратора со следующими подмодулями:

1. Управление учетными записями пользователей:

- просмотр всех учетных записей;
- добавление новой учетной записи;
- редактирование учетной записи;
- удаление учетной записи;
- подтверждение учетной записи;
- блокировка учетной записи.

Допускается создание учетной записи нескольких администраторов, при этом запрещается удаление учетной записи администратора или изменение его роли/доступа, если он один. Это позволит исключить ситуацию удаления всех администраторов (что сделало бы дальнейшее редактирование учетных записей и данных невозможным).

2. Работа с данными:

А) режим редактирования

- просмотр всех данных;
- добавление новой записи;
- удаление записи;
- редактирование записи;

Б) режим обработки данных:

выполнение индивидуального задания (вывод в алфавитном порядке список товаров, хранящихся более  $x$  месяцев, стоимость которых превышает  $y$  рублей ( $x, y$  вводятся с клавиатуры));  
поиск данных (по наименованию, по цене, по дате, по ФИО зарегистрировавшего товар);  
сортировка (наименование товара и ФИО по алфавиту, цена на выбор в порядке убывания или возрастания)

2. Модуль пользователя включает подмодуль работы с данными со следующими функциональными возможностями:

- просмотр всех данных;
- вывод в алфавитном порядке список товаров, хранящихся более  $x$  месяцев, стоимость которых превышает  $y$  рублей ( $x, y$  вводятся с клавиатуры);
- поиск данных (по наименованию, по дате, по ФИО)
- сортировка (наименование и ФИО (по алфавиту), цена (по возрастанию и убыванию)).



# Оформляем документы в Word (например, пояснительную записку по стандарту):

**bestzmest.by**

## ПОСЛЕ

### СОДЕРЖАНИЕ

Введение.....	2
1 Требование к программе .....	3
1.1 Исходные данные .....	3
1.2 Функциональные требования.....	3
1.3 Требования к программной реализации.....	6
2 Конструирование программы .....	7
2.1 Модульная структура программы .....	7
2.2 Выбор способа организации данных.....	7
2.3 Перечень пользовательских функций .....	8
3 Алгоритмы работы программы.....	12
4 Описание работы программы.....	13
4.1 Регистрация и авторизация.....	13
4.2 Модуль администратора.....	15
4.3 Модуль пользователя.....	17
4.4 Исключительные ситуации .....	18

### 1 ТРЕБОВАНИЕ К ПРОГРАММЕ

Полный текст варианта задания:

Программа предоставляет сведения о товарах, имеющихся на складе: наименование товара; количество единиц товара; цена единицы товара; дата поступления товара на склад; ФИО зарегистрировавшего товар.

Индивидуальное задание: вывести в алфавитном порядке список товаров, хранящихся более  $x$  месяцев, стоимость которых превышает  $y$  рублей ( $x, y$  вводятся с клавиатуры).

Кроме того, реализовать авторизацию для входа в систему, функционал администратора и пользователя.

#### 1.1 Исходные данные

- 1 Разработка программы учета товаров на складе.
- 2 Язык программирования C++.
- 3 Среда разработки Microsoft Visual Studio.
- 4 Вид приложения – консольное.
- 5 Парадигма программирования – процедурная.
- 6 Способ организации данных – структуры (struct).
- 7 Способ хранения данных – файлы.
- 8 Каждая логически завершенная подзадача программы реализована в виде отдельной функции.
- 9 Построение программного кода соответствует соглашению о коде «C++ Code Convention».
- 10 К защите курсовой работы представляются: консольное приложение и пояснительная записка.
- 11 Текст пояснительной записки оформляется в соответствии со стандартом.





# Критерии оценки

---

Способ оценивания	Максимально возможный результат	Критерии оценки
1. Демонстрация программы 2. Просмотр записки 3. Просмотр кода (code review)	4 балла	1. Реализация базового функционала (работа с данными: просмотр, добавление, удаление, редактирование, выполнение индивидуального задания, поиск, сортировка). 2. Оформленная в соответствии с требованиями записка, но <u>без</u> алгоритмов. 3. Владение кодом (можете объяснить <u>свой</u> код).

# Критерии оценки

Способ оценивания	Максимально возможный результат	Критерии оценки
Демонстрация программы	+ 1 балл (итог: <b>5 баллов</b> )	Реализация полного функционала (авторизация, работа с учетными записями: просмотр, добавление, удаление, редактирование).
	+ 2 балла (итог: <b>6 баллов</b> )	Реализация продвинутого функционала, а именно: реализация обратной связи с пользователем (запрос на подтверждение удаления, сообщения об успешности выполнения действий), <u>обработка исключительных ситуаций</u> (проверка форматов вводимых данных, проверка существования номера записи для редактирования/удаления, проверка на уникальность нового логина и т.д.); другие любые Ваши творческие подходы по усовершенствованию программы.



# Критерии оценки

Способ оценивания	Максимально возможный результат	Критерии оценки
Просмотр записки	+1 балл (итог: <b>7 баллов</b> )	Частичная реализация алгоритмов (т.е. 1 или 2 вместо 3) или наличие в алгоритмах ошибок.
	+2 балла (итог: <b>8 баллов</b> )	Реализованы 3 алгоритма без ошибок.



# Критерии оценки

Способ оценивания	Максимально возможный результат	Критерии оценки
Просмотр кода (code review)	+1 балл (итог: <b>9 баллов</b> )	Качество кода: хорошее (нет дублирования одного и того же кода: вместо этого – функции; каждая функция решает одну задачу; осмысленные имена переменных, функций).
	+2 балла (итог: <b>10 баллов</b> )	Качество кода: отличное (функция main не перегружена кодом; сведены к минимуму операции по чтению и записи в файл; код снабжен комментариями; нет «хардкода»: вместо этого – константы; код разбит на модули в виде отдельных .cpp файлов, которые подключаются посредством заголовочных .h файлов; все действия логичны и понятны).