**Курсовая работа по дисциплине**

**«Основы конструирования программ»**

**Структура пояснительной записки**

Титульный лист *(образец приведен в одноименном файле)*

Задание по курсовой работе *(заполненное и подписанное студентом и преподавателем)*

Содержание

1. Требования к программе *(см. пояснения ниже)*
2. Конструирование программы
	1. Разработка структуры программы *(см. пояснения ниже)*
	2. Выбор способа организации данных *(см. пояснения ниже)*
	3. Разработка перечня пользовательских функций программы *(см. пояснения ниже)*

3. Разработка алгоритмов работы программы *(см. пояснения ниже)*

3.1 Алгоритм функции main

3.2 Алгоритм функции … *(выбирается из перечня пользовательских функций)*

3.3 Алгоритм функции … *(выбирается из перечня пользовательских функций)*

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).*

**Требования к программе** включают:

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

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.

**Разработка структуры программы** подразумевает графическое представление структуры программы с указанием модулей, подмодулей и их функциональных возможностей. *Для объектно-ориентированного программирования в данном подразделе приводится UML диаграмма классов!*

 

Рисунок 1 – Пример структуры программы

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

http://www.newart.ru/htm/flash/risovalka\_42.php

**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/векторов/…, а также их выбранная область видимости (локальные/глобальные).

**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. *В случае использования объектно-ориентированного программирования приводятся методы для классов.* Код функций (методов) не приводится, т.к. на этом этапе он еще не существует.

В сущности, второй раздел впоследствии будет представлять собой код программы до главной функции main (здесь расположены пользовательские типы данных – структуры, глобальные переменные массивов или векторов, прототипы всех предполагаемых функций).

**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.

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.

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 быт любой; в качестве рекомендации обращаю ваше внимание на online-редакторы, например:

http://www.newart.ru/htm/flash/risovalka\_42.php

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

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