МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«МУРМАНСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Кафедра математики,
информационных систем
и программного обеспечения

МЕТОДИЧЕСКИЕ УКАЗАНИЯ К ВЫПОЛНЕНИЮ САМОСТОЯТЕЛЬНОЙ РАБОТЫ

По дисциплине: Б1.О.09.01 Программирование
название дисциплины

для направления подготовки 09.03.01 Информатика и вычислительная техника
направленность (профиль) «Программное обеспечение вычислительной техники и автоматизированных систем»
квалификация выпускника бакалавр

Мурманск
2020

Составитель - Скрябин Антон Васильевич, старший преподаватель кафедры математики, информационных систем и программного обеспечения

Методические указания к выполнению курсовой работы рассмотрены и одобрены на заседании кафедры-разработчика:

математики, информационных систем и программного обеспечения

«24» ноября 2020 г., протокол № 4.

ОБЩИЕ МЕТОДИЧЕСКИЕ УКАЗАНИЯ

Данные методические указания разработаны в соответствии с ФГОС ВО по направлению подготовки

09.03.01 Информатика и вычислительная техника
код и наименование направления подготовки

утверждённого 19.09.2017 г. № 929 и учебного плана направления подготовки
(дата, номер приказа Минобрнауки РФ) (обозначение или наименование документа университетского уровня)

09.03.01 Информатика и вычислительная техника (направленность программы: Программное обеспечение вычислительной техники и автоматизированных систем)

Целью дисциплины «Программирование» является формирование компетенций в соответствии с ФГОС ВО по направлению подготовки 09.03.01 Информатика и вычислительная техника и учебным планом в составе ОПОП по направлению подготовки 09.03.01 Информатика и вычислительная техника, направленность (профиль): Программное обеспечение вычислительной техники и автоматизированных систем, что предполагает формирование у обучающегося знаний об основных методах программирования, принципах тестирования ПО.

Задачи дисциплины: ознакомить с основными структурами и типами, применяемыми при разработке программного обеспечения, с особенностями разработки консольных и оконных приложений в современных средствах разработки, а также со способами анализа эффективности разработанных программ.

Процесс изучения дисциплины направлен на формирование общепрофессиональных компетенций в соответствии с ФГОС ВО по направлению подготовки 09.03.01 Информатика и вычислительная техника:

- способен применять естественнонаучные и общеинженерные знания, методы математического анализа и моделирования, теоретического и экспериментального исследования в профессиональной деятельности (ОПК-1);
- способен понимать принципы работы современных информационных технологий и программных средств, в том числе отечественного производства, и использовать их при решении задач профессиональной деятельности (ОПК-2);
- способен разрабатывать алгоритмы и программы, пригодные для практического применения (ОПК-8);
- способен разрабатывать требования и проектировать программное обеспечение (ПК-1).

В результате изучения дисциплины обучающийся должен:

- ***знать***: основные принципы разработки программ; правила определения требований на разработку программы; основные критерии эффективности программ; назначение и основные принципы работы современных средств разработки программ.
- ***уметь***: выявлять требования к программному обеспечению; разрабатывать программы в соответствии с предъявляемыми требованиями; определять необходимый программный инструментарий в соответствии с решаемой задачей; выполнять тестирование разработанных программ.
- ***владеть***: навыками проектирования программного обеспечения; навыками разработки программ с использованием соответствующего инструментария; навыками разработки прикладных программ; навыками оценки эффективности разработанных программ.

МЕТОДИЧЕСКИЕ УКАЗАНИЯ К САМОСТОЯТЕЛЬНОЙ РАБОТЕ ПО ВЫПОЛНЕНИЮ РАСЧЕТНО-ГРАФИЧЕСКОЙ РАБОТЫ

РАСЧЕТНО-ГРАФИЧЕСКАЯ РАБОТА №1

Целью РГР №1 является получение навыков разработки программного средства и описания процесса разработки.

Тематика программного средства выбирается самостоятельно. Возможно продолжение работы над тематиками, которые обсуждались на практических занятиях. В результате выполнения РГР необходимо предоставить:

1. программное средство в виде исполняемого файла и исходного кода;
2. описание процесса разработки программного средства в виде текстового документа в соответствии с приведённым ниже планом;
3. презентацию, представляющую результаты работы.

Презентация. По итогам работы над проектом необходимо подготовить презентацию для *устного выступления*, в котором необходимо рассказать о том, как проходила работа. В презентацию выносится информация, которая будет в наглядной форме представлять позиции из выступления. Объем презентации не ограничен. Основные критерии: наглядность, краткость. Содержимое слайдов не должно быть полной копией устного выступления, а должно содержать основные моменты и визуальный материал для лучшего понимания того, что произносится докладчиком (то есть вами). Выступление необходимо отрепетировать, чтобы по длительности оно не превышало 7-8 минут.

План работы и описания процесса разработки

Постановка задачи. Нужно сформулировать задачу на разработку на языке заказчика. Например, «*Необходимо разработать программу для решения квадратного уравнения*».

Здесь же приводится дополнительная информация, если она есть и может повлиять на разработку. Например, что эта *программа будет использоваться в процессе обучения пятиклассников* (такая информация может быть применена в определении требований к точности производимых программой расчетов или требований к способу организации интерфейса программы).

Другими словами, в этом разделе закладывается *идея* будущей программы. Детализация и формализация этой идеи происходит в дальнейших этапах.

Анализ предметной области. В этом разделе необходимо собрать всю необходимую для разработки информацию о процессе, для которого разрабатывается программа. Процесс рассматривается не с точки зрения реализации в программе, а в общем, т.е. в этом разделе не должны использоваться фразы, связанные с работой в программе («после нажатия на кнопку…», «пользователь вводит данные» и т.д.). Процесс описывается так, как он происходит сейчас, без написанной программы.

В идеале каждое слово из постановки задачи должно быть раскрыто и формализовано. Совет: представьте, что вы не знаете ни одного слова из постановки задачи и вам нужно с ними разобраться – результат этого разбора и будет представлять собой анализ предметной области.

Возвращаясь к примеру с *разработкой программы для решения квадратных уравнений*, в этом разделе необходимо:

- описать, что такое *«квадратное уравнение»;*

- выявить все его компоненты, их характеристики и особенности (коэффициенты *a, b, c* и особые случаи при разных значениях этих коэффициентов);

- описать, что означает *«решить квадратное уравнение»*;

- описать способы решения квадратных уравнений с пояснением всех возникающих терминов и формул.

Если в постановке задачи были заявлены еще какие-то общие требования (например, *«использование программы в процессе обучения пятиклассников»*), то они также должны быть полностью рассмотрены и формализованы (например, выявить, на что нужно делать акцент при обучении пятиклассников способам решения квадратных уравнений).

Полученная в результате выполнения этого этапа информация используется для составления требований к программе! Обратите внимание на этот критерий. Если информация не приводит к появлению каких-то требований (например, годы жизни Франсуа Виета), то она не является необходимой (либо вы забыли сформировать требование по этой информации). Верно и обратное: если при формировании требований вы сформировали некоторое требование и не знаете, на основании какой информации из анализа предметной области оно получено, то значит, что вы забыли что-то описать в анализе предметной области.

Выдвижение требований. Этот раздел подводит итог анализу предметной области. Необходимо составить список требований, исходя из информации, выявленной при анализе предметной области. Каждое требование должно быть сформулировано четко и однозначно и должно быть проверяемым (вы должны понимать, как будет проверяться реализация этого требования в итоговой программе). Например, требование «интерфейс программы должен быть удобным» не является проверяемым, т.к. не существует способа проверки удобства интерфейса. Но это требование можно представить в виде одного или нескольких требований, которые в вашей программе будут реализовывать «удобство интерфейса» (например, наличие всплывающих подсказок, наличие графических иконок у кнопок и т.д.).

Требования можно разбить на следующие группы:

«Группы пользователей»

Для каких обобщенных групп пользователей будет предназначена программа. Обратите внимание, что группы пользователей отличаются друг от друга не социальным статусом или ролью входящих в нее людей, а списком задач, которые будут им доступны в программе. Например, в уже любимой нам программе для решения квадратного уравнения учащиеся класса 5«А» и учащиеся класса 5«Б» будут объединены в одной группе пользователей «Учащиеся». А другой группой пользователей может быть, например, «Учителя», если для учителей в программе будет реализован какой-то специальный функционал. Если же такого специального функционала не предусмотрено, то в программе будет единственная группа пользователей. В таком случае можно указать, что среди потенциальных пользователей не выделено специфических групп.

«Функциональные требования»

В этой группе перечисляются функции будущей программы, т.е. список того, что программа должна будет уметь делать. При этом, если ранее были выделены отдельные группы пользователей, то приводятся как функции, специфичные для каждой группы, так и функции общие для всех групп пользователей. Обратите внимание на формулировку требований. Например, ошибочным будет требование *«обучать пользователей решать квадратные уравнения»*, поскольку программа не является кем-то, кто может обучать. А формулировка *«выводить теоретические сведения по заданной пользователем теме»*, скорее всего, отразит то, что в данном случае имелось в виду. Другими примерами

функциональных требований могут быть *F1: «вычисление дискриминанта по заданным коэффициентам квадратного уравнения», F2: «вывод на экран решения заданного квадратного уравнения»* и т.д. При этом вы должны понимать (хотя бы в общих чертах), что будет поступать на вход этой функции и что будет результатом ее выполнения. Это потребуется вам при составлении спецификаций функций.

«Требования к данным»

Если из анализа предметной области становятся понятны какие-то особенности входных, выходных или промежуточных данных, то их необходимо зафиксировать в виде требований. Требования к данным могут включать состав входной и выходной информации, способы ввода/вывода данных, форму представления, ограничения, особые случаи и т.д.

«Требования к интерфейсу»

Из анализа предметной области также могут быть выведены требования к интерфейсу программы, т.е. способам взаимодействия пользователя и будущей программы. Они могут включать требование оконного или консольного приложения, требования к элементам управления, их оформлению, расположению содержанию и т.д. Вопросы разработки интерфейса, как содержащиеся к требованиях к интерфейсу, так и не содержащиеся в них, будут решаться на этапе проектирования.

«Требования к окружению»

Также анализ предметной области может привести к появлению требований, которые связаны не с характеристиками разрабатываемой программы, а с тем, что ее будет окружать – аппаратному и программному обеспечению. Например, программа будет применяться в организации, в которой на компьютерах установлена Windows 95. В таком случае необходимость работы программы в этой операционной системе должна быть отражена в требованиях к окружению. Не забывайте, что требования обуславливаются анализом предметной области. Если из этого анализа не становятся очевидными какие-то требования к окружению, то придумывать их с целью «просто, чтобы были» не нужно.

НЕ в каждой группе обязательно должны быть требования. Еще раз: требования появляются из анализа предметной области.

Проектирование. Это большой раздел, посвященный решению вопроса о том, как реализовать требования. Можно разбить его на несколько разделов.

Проектирование функций

Для каждой функции из функциональных требований составляется *спецификация*, т.е. описание входных и выходных данных (состав, назначение, свойства), а также описание связи между ними в виде формулы или сценария получения выходных данных из входных.

Проектирование данных

Обобщая информацию всех спецификаций функций, формируется общий набор данных, который будет храниться во внешних файлах и проектируется структура этих файлов. Возможно, что файлов будет несколько: входной файл, содержащий входные данные, и выходной файл, содержащий результаты работы программы. При проектировании формата хранения данных определяется то, какие данные будут храниться в проектируемом файле и как эти данные будут в нем расположены. Разработчику программы необходимо спроектировать формат хранения данных для того, чтобы «объяснить» программе, в каких позициях файла находятся те или иные данные и

как отличить одни данные от каких-то других. Это «объяснение» будет отражено в алгоритмах, которые проектируются далее.

Проектирование алгоритмов

При проектировании алгоритма программы рекомендуется использовать метод пошаговой детализации, который начинается с составления общего алгоритма программы включающего самые обобщенные действия, а затем аналогично составляются алгоритмы для обобщенных действий (подпрограмм) общего алгоритма. Алгоритмы для подпрограмм также могут включать в себя какие-то обобщенные действия, для которых впоследствии составляются алгоритмы. Процесс детализации общего алгоритма заканчивается, когда все подпрограммы представлены в виде собственных алгоритмов.

Алгоритмы представляются в виде блок-схем. Подробнее про элементы блок-схем читайте в документе «Алгоритмы и кодирование».

Если программа предполагает большое количество функционала, то бывает полезно разбить ее на модули, скомпоновав близкие по смыслу или какому-то другому признаку функции в отдельных модулях. В результате приводится *структура программы*.

Проектирование интерфейса

В этом разделе описываются и аргументируются принятые решения по организации интерфейса программы. Проектные решения можно представить в виде эскиза интерфейса программы с объяснением выбора, назначения и расположения элементов в нем.

Реализация. В этом разделе описывается полученная программная реализация. Описание файлов с кодом программы: назначение, описание переменных и функций. Описываются проблемы, возникавшие при реализации, и как они были решены, возможные исключительные ситуации при работе программы и способы их обработки. Приводятся скриншоты работы программы.

Отметим, что для программ с большим функционалом допустима реализация только части этого функционала.

РАСЧЕТНО-ГРАФИЧЕСКАЯ РАБОТА №2

Расчетно-графическая работа является одним из основных видов самостоятельной работы по теме «Анализ программ» и ставит **целью** закрепление теоретического материала и развитие практических навыков по данной теме.

Для выполнения работы обучающемуся необходимо выбрать программный код для анализа. Наиболее подходящим вариантом является программная реализация, полученная в рамках РГР №1 дисциплины, однако программы, реализованные в течение семестра, также могут выступать целью для анализа. Перед началом выполнения работы обучающийся должен согласовать анализируемый программный код с преподавателем.

В ходе выполнения РГР для выбранной задачи необходимо провести тестирование программы, оценить стиль программирования, временную и объемную сложности, а также сделать выводы о возможности и способах повышения эффективности анализируемой программы.

Во *введении* описывается задача, решаемая в анализируемой программе, и ставятся общие цели на работу.

В *основной части* выделяется несколько частей:

1) Тестирование программы.

В этом разделе необходимо подготовить и описать несколько тестовых наборов для проведения тестирования анализируемой программы. Тестовые наборы создаются с учетом правил их составления. Затем для одного или нескольких наборов проводится

тестирование программы в виде таблицы трассировки. По результатам проведения тестирования оформляются соответствующие выводы.

2) Оценка стиля программирования.

Этот раздел содержит анализ стиля программирования анализируемого кода. Оценка проводится по рассмотренным на лекционных занятиях критериям. Допускается анализ отдельного фрагмента программы, выполняющего наиболее важные функции. Результаты оценки оформляются в виде таблицы. Заканчивается эта часть подведением итогов о стиле программирования анализируемого кода.

3) Оценка временной сложности.

В этом разделе для анализируемой программы составляются временные профили, на основании которых выводится аналитический профиль программы, являющийся базой для оценки временной сложности. По результатам оценки необходимо также сделать выводы о временной эффективности программы и способах ее повышения при необходимости/возможности.

4) Оценка объемной сложности.

В этом разделе для анализируемой программы составляются схема вызовов, на основании которой выполняется оценка объемной сложности. По результатам оценки необходимо также сделать выводы об используемой программой памяти и способах повышения ее использования при необходимости/возможности.

В *заключении* подводятся итоги, перечисляются достигнутые цели, отмечаются ключевые моменты по проделанной работе.

Общие рекомендации по оформлению результатов работ

Обучающиеся оформляют результаты, полученные в ходе выполнения РГР, в форме пояснительных записок, отвечающих следующим требованиям:

1. У работы должен быть титульный лист (Приложение 1).
2. Следующим листом должно идти содержание с указанием страниц разделов.
3. Разделы пояснительной записки: введение, основная часть, заключение, список использованных источников, приложения (при необходимости).
4. Параметры страницы: А4, поля 30 см слева, 15 см снизу и сверху, 20 см справа.
5. Текст: шрифт Times New Roman, 12 pt. Заголовки выделять жирным.
6. Абзац: отступ 1,25 см; выравнивание – по ширине; полуторный интервал.
7. Рисунки и таблицы должны быть обязательно подписаны и иметь ссылку в тексте. Рисунки подписываются снизу по центру «Рис. № - Подпись к рисунку». Аналогичные надписи к таблицам размещаются над ними.

Приложение 1. Образец титульного листа РГР

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«МУРМАНСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»
(ФГАОУ ВО «МГТУ»)

Кафедра математики,
информационных систем и
программного обеспечения

Расчетно-графическая работа

по дисциплине

«Программирование»

Выполнил:
студент группы ИВТбNNз-NN
Фамилия И.О.

Проверил:
преподаватель кафедры МИС и ПО
Фамилия И.О.

Мурманск
20__