МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное
образовательное учреждение высшего образования
«Крымский федеральный университет имени В.И. Вернадского»
Таврический колледж
(структурное подразделение)

ОТЧЕТ ПО ПРАКТИКЕ

Учебная практика по профессиональному модулю ПМ.03 Участие в интеграции программных модулей

Специальность **09.02.03 Программирование в компьютерных системах**

Обучающийся 4 курса группы **4ПКС17**

форма обучения____очная_____
(очная, заочная)
Олейников Александр Сергеевич
(фамилия, имя, отчество)
Место практики
Таврический колледж (структурное подразделение) ФГАОУ «КФУ им. В.И. Вернадского»
(наименование организации)

Срок практики с **16 марта 2023 г. по 22 марта 2023 г.**

Руководитель практики
от колледжа
преподаватель_____ _____ /___Руденко А.В.___/
должность подпись (Ф.И.О.)

Зам директора
по учебно-производственной
практике _____ /____Малюга Г.Г._____/
подпись (Ф.И.О.)

Итоговая оценка по практике _____
(отлично, хорошо, удовлетворительно)

МП
г. Симферополь, 2023 г.

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Учебная практика по программному модулю ПМ.02 "Участие в интеграции программных модулей", проводилась согласно положению о практике обучающихся, осваивающих основные профессиональные образовательные программы среднего профессионального образования, утвержденного Приказом Министерства образования Российской Федерации № 093 от 02 марта 6012 г.

Цель практики: формирование и развитие общих и профессиональных компетенций по модулю ПМ.03 Участие в интеграции программных модулей. Для достижения цели практики были поставлены следующие задачи:

- закрепить теоретические знания о технике безопасности при работе с электронно-вычислительными машинами;
- рассмотреть теоретическую информацию по выбранному языку программирования C++;
- изучить существующие алгоритмы сортировки чисел;
- Разработать техническое задание на программный продукт;
- Разработать спецификацию на программный продукт;
- Разработать функциональную диаграмму программного продукта, диаграмму потоков данных программных модулей продукта;
- Разработать функциональную схему программного продукта, составить блок-схемы программных модулей программного продукта;
- Разработать коды программных модулей программного продукта;
- Разработать пользовательский интерфейса программного продукта в визуальной среде;
- Выполнить интеграцию программных модулей в программный продукт;

- Разработать процедуру тестирования программного продукта;
- Выполнить тестирование программного продукта. Результат тестирования оформить протоколом тестирования;
- Разработать справочную систему программного продукта;
- Разработать руководства оператора (пользователя);
- Создать аккаунт в GitHub. Создать папку проекта. В папку загрузить разработанный программный комплекс, всю разработанную документацию к проекту;
- Составить отчет о выполнении;

Структура отчета по практике: содержание, введение, основная часть, состоящая из двух разделов, заключение, список использованных источников, приложения.

Первый раздел основной части, «Теоретические сведения», включает в себя информацию о технике безопасности, языке программирования C++ и сервисе GitHub.

Второй раздел – «Реализация программного комплекса» содержит описание и саму документация программного комплекса, а также включает в себя краткое описание реализации программных модулей.

РАЗДЕЛ 1. ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

1.1 Техника безопасности

В кабинете вычислительной техники (КВТ) установлена дорогостоящая, сложная и требующая осторожного и аккуратного обращения аппаратура - компьютеры (ЭВМ), принтер, другие технические средства. Исходя из этого, следует:

- бережно обращайтесь с этой техникой;
- спокойно, не торопясь, не толкаясь, не задевая столы, входите в кабинет и занимайте отведенное вам место, ничего не трогая на столах.

На рабочем месте размещены составные части ЭВМ — системный блок, клавиатура, монитор (дисплей) и др. Во время работы лучевая трубка монитора (дисплея) работает под высоким напряжением. Неправильное обращение с аппаратурой, кабелями и мониторами может привести к тяжелым поражениям электрическим током, вызвать загорание аппаратуры.

Запрещается:

- трогать разъемы соединительных кабелей;
- прикасаться к питающим проводам и устройствам заземления;
- прикасаться к экрану и к тыльной стороне монитора;
- включать и отключать аппаратуру без указания преподавателя;
- класть диск, книги, тетради на монитор и клавиатуру;
- работать во влажной одежде и влажными руками.

Перед началом работы:

1. убедитесь в отсутствии видимых повреждений рабочего места; сядьте так, чтобы линия взора приходилась в центр экрана, чтобы, не

наклоняясь пользоваться клавиатурой и воспринимать передаваемую на экран монитора информацию;

2. разместите на столе тетрадь, учебное пособие так, чтобы они не мешали работе на ЭВМ;
3. внимательно слушайте объяснения учителя и старайтесь понять цель и последовательность действий; в случае необходимости обращайтесь к преподавателю;
4. хорошо разберитесь в особенностях применяемых в работе аппаратов;
5. начинайте работу только по указанию преподавателя «Приступить к работе».

Во время работы:

1. Строго выполняйте все указанные выше правила, а также текущие указания учителя;
2. Следите за исправностью аппаратуры и немедленно прекращайте работу при появлении необычного звука или самопроизвольного отключения аппаратуры. Немедленно докладывайте об этом преподавателю;
3. Плавно нажимайте на клавиши, не допуская резких ударов;
4. Не пользуйтесь клавиатурой, если не подключено напряжение;
5. Работайте на клавиатуре чистыми руками;
6. Никогда не пытайтесь самостоятельно устранять неисправности в работе аппаратуры;
7. Не вставайте со своих мест, когда в кабинет входят посетители.

По окончании работы выключите компьютер [1].

1.2 Язык программирования C++

C++ — компилируемый статически типизированный язык программирования общего назначения.

Поддерживает такие парадигмы программирования как процедурное программирование, объектно-ориентированное программирование, обобщённое программирование, обеспечивает модульность, раздельную компиляцию, обработку исключений, абстракцию данных, объявление типов (классов) объектов, виртуальные функции. Стандартная библиотека включает, в том числе, общеупотребительные контейнеры и алгоритмы. C++ сочетает свойства как высокоуровневых, так и низкоуровневых языков. В сравнении с его предшественником — языком C, — наибольшее внимание уделено поддержке объектно-ориентированного и обобщённого программирования.[2]

C++ широко используется для разработки программного обеспечения, являясь одним из самых популярных языков программирования. Область его применения включает создание операционных систем, разнообразных прикладных программ, драйверов устройств, приложений для встраиваемых систем, высокопроизводительных серверов, а также развлекательных приложений (игр). Существует множество реализаций языка C++, как бесплатных, так и коммерческих и для различных платформ. Например, на платформе x86 это GCC, Visual C++, Intel C++ Compiler, Embarcadero (Borland) C++ Builder и другие. C++ оказал огромное влияние на другие языки программирования, в первую очередь на Java и C#.

Синтаксис C++ унаследован от языка C. Одним из принципов разработки было сохранение совместимости с C. Тем не менее, C++ не является в строгом смысле надмножеством C; множество программ, которые могут одинаково успешно транслироваться как компиляторами C, так и компиляторами C++, довольно велико, но не включает все возможные программы на C.

Разработчиком языка Си++ является Бьерн Страуструп. В своей работе он опирался на опыт создателей языков Симула, Модула 2, абстрактных

типов данных. Основные работы велись в исследовательском центре компании Bell Labs.

Язык Си++ является универсальным языком программирования, в дополнение к которому разработан набор разнообразных библиотек. Поэтому, строго говоря, он позволяет решить практически любую задачу программирования. Тем не менее, в силу разных причин (не всегда технических) для каких-то типов задач он употребляется чаще, а для каких-то – реже.

Си++ широко используется в системном программировании. На нем можно писать высокоэффективные программы, в том числе операционные системы, драйверы и т.п. Язык Си++ – один из основных языков разработки трансляторов.

Поскольку системное программное обеспечение часто бывает написано на языке Си или Си++, то и программные интерфейсы к подсистемам ОС тоже часто пишут на Си++.

Распределенные системы, функционирующие на разных компьютерах, также разрабатываются на языке Си++. Этому способствует то, что у широко распространенных компонентных моделей CORBA (CORBA определяет, каким образом программные компоненты, распределенные по сети, могут взаимодействовать друг с другом вне зависимости от окружающих их операционных систем и языков реализации) и COM есть удобные интерфейсы на языке Си++.

Обработка сложных структур данных – текста, бизнес-информации, Internet-страниц и т.п. – одна из наиболее распространенных возможностей применения языка. В прикладном программировании, наверное, проще назвать те области, где язык Си++ применяется мало.

Разработка графического пользовательского интерфейса на языке Си++ выполняется, в основном, тогда, когда необходимо разрабатывать сложные,

нестандартные интерфейсы. Простые программы чаще пишутся на языках Visual Basic, Java и т.п.

Программирование для Internet в основном производится на языках Java, VBScript, Perl.

Язык Си++ в настоящее время является одним из наиболее распространенных языков программирования в мире.[3]

1.2.1 История языка C++

Язык С имел процедурную парадигму, то есть предполагалось структурирование кода при помощи подпрограмм (функций и процедур).

К началу 80-х выяснилось, что для создания сложных приложений этого недостаточно. Среди предложенных удачных решений было объектно-ориентированное программирование (ООП). Его встроенная поддержка отсутствовала в С, поэтому, начиная с 1984 года, Бьярн Страуструп из Bell Labs разрабатывал новый язык программирования на основе С с поддержкой ООП, в конечном счете названный С++.

Самым привлекательным свойством С++ стала забота об обратной совместимости (backward compatibility) с С и собственными предыдущими версиями.

Разработчики зачастую могли использовать большое количество готового кода прямо в программах на новом языке и быть уверенными, что его не понадобится переписывать. Очевидная выгода от этого способствовала почти столь же широкому распространению С++, как и С. [4]

В ходе развития язык С++ претерпевал изменения и многочисленные эксперименты, иногда спорные, но большинство из них оставалось в языке. Одновременно появлялись другие языки, основанные на ООП: Objective-C

(совместим с C), Java, Object Pascal (Delphi), C#, где многие ошибки проектирования C++ были учтены и не повторялись.

Они оказались удачны для новых приложений, однако во многих случаях переписывать с C++ имеющийся код было нецелесообразно, либо новый код активно использовал старый на C++, либо требовалась одновременно производительность и кроссплатформенность. Таким образом, язык C++, в некотором смысле, «проложил дорогу» многим последующим языкам, иногда более удобным. [5]

1.3 Веб-сервис GitHub

Система контроля версий — программа, которая хранит разные версии одного документа, позволяет переключаться между ними, вносить и отслеживать изменения. Таких систем много и все они работают по принципу компьютерной игры, где вы можете вернуться к месту сохранения, если что-то пошло не так.

Git — самая популярная система контроля версий. С Git можно работать через командную строку (или терминал). В каждой системе своя встроенная программа для работы с командной строкой. В Windows это PowerShell или cmd, а в Linux или macOS — Terminal.

GitHub (или Гитхаб) — веб-сервис на основе Git, который помогает совместно разрабатывать IT-проекты. На Гитхабе разработчики публикуют свой и редактируют чужой код, комментируют проекты и следят за новостями других пользователей [6].

Создатели сайта называют GitHub «социальной сетью для разработчиков».

Кроме размещения кода, участники могут общаться, комментировать правки друг друга, а также следить за новостями знакомых.

С помощью широких возможностей Git программисты могут объединять свои репозитории — GitHub предлагает удобный интерфейс для этого и может отображать вклад каждого участника в виде дерева.

Для проектов есть личные страницы, небольшие Вики и система отслеживания ошибок.

Прямо на сайте можно просмотреть файлы проектов с подсветкой синтаксиса для большинства языков программирования.

- можно создавать приватные репозитории, которые будут видны только вам и выбранным вами людям. Раньше такая возможность была платной;
- есть возможность прямого добавления новых файлов в свой репозиторий через веб-интерфейс сервиса;
- код проектов можно не только скопировать через Git, но и скачать в виде обычных архивов с сайта;
- кроме Git, сервис поддерживает получение и редактирование кода через SVN и Mercurial;
- на сайте есть pastebin-сервис gist.github.com для быстрой публикации фрагментов кода;
- файлы из репозитория могут автоматически публиковаться в виде статического сайта с помощью GitHub Pages.

РАЗДЕЛ 2. РЕАЛИЗАЦИЯ ПРОГРАММНОГО КОМПЛЕКСА

2.1 Разработка технического задания на программный продукт.

Перед началом разработки перед командой специалистов должны быть поставлены определенные цели, которые должны быть достигнуты в программе. Такие цели описываются в техническом задании.

Техническое задание (далее – ТЗ) – основной документ проекта, которым Заявитель устанавливает цели и задачи проекта, номенклатуру и назначение продуктов проекта, технические и иные значимые характеристики проектируемого производства и/или продукта проекта, порядок и последовательность необходимых стадий реализации проекта, создания продукта проекта (в том числе описание технологии) и контроля его качественных параметров.

Техническое задание содержит в себе полную информацию о разрабатываемом программном обеспечении, включая:

- функциональные требования;
- требования к интерфейсу и структуре проекта (список программных модулей, страниц сайта и т.д.);
- язык программирования;
- описание входных и выходных данных (если проект подразумевает такие сведения);
- требования к устройствам, на которых должно работать программное обеспечение;
- требования к вспомогательным программным средствам, используемым в проекте;

- дополнительные условия создания и обслуживания программного продукта;
- требования к отчетности по ходу разработки;
- приемо-сдаточные испытания;
- сроки разработки;
- другие детали проекта, подразумевающиеся заказчиком или исполнителем.

Текст ТЗ разработанного в процессе выполнения учебной практики:

ВВЕДЕНИЕ:

Наименование программы - «Сортировщик Чисел».

Назначение и область применения. Программа предназначена для сортировки целых чисел в порядке возрастания. Она содержит следующие методы сортировки:

- Сортировка пузырьком
- Шейкерная сортировка
- Сортировка расчёской
- Сортировка вставками

Программа предоставляет графический интерфейс для управления процессом сортировки в соответствии требованиям.

ТРЕБОВАНИЯ К ПРОГРАММЕ:

Требования к функциональным характеристикам. Программа должна обеспечивать возможность выполнения перечисленных ниже функций:

- Ввод вывод данных непосредственно в программе
- Сортировка 4-мя видами алгоритмов, которые выбирает сам пользователь
- Отслеживание времени выполнения операций

Требования к надежности:

Требования к обеспечению надежного функционирования программы. Надежное (устойчивое) функционирование программы должно быть

обеспечено выполнением Заказчиком совокупности организационно-технических мероприятий, перечень которых приведен ниже:

- организацией бесперебойного питания технических средств;
- использованием лицензионного программного обеспечения;
- регулярным выполнением рекомендаций Министерства труда и социального развития РФ, изложенных в Постановлении от 23 июля 1998 г. Об утверждении межотраслевых типовых норм времени на работы по сервисному обслуживанию ПЭВМ и оргтехники и сопровождению программных средств»;
- регулярным выполнением требований ГОСТ 51308-98. Защита информации. Испытания программных средств на наличие компьютерных вирусов.

Время восстановления после отказа. Время восстановления после отказа, вызванного сбоем электропитания технических средств (иными внешними факторами), не фатальным сбоем (не крахом) операционной системы, не должно превышать 30-ти минут при условии соблюдения условий эксплуатации технических и программных средств.

Время восстановления после отказа, вызванного неисправностью технических средств, фатальным сбоем (крахом) операционной системы, не должно превышать времени, требуемого на устранение неисправностей технических средств и переустановки программных средств.

Отказы из-за некорректных действий пользователей системы. Отказы программы вследствие некорректных действий пользователя при взаимодействии с программой через графический интерфейс недопустимы.

УСЛОВИЯ ЭКСПЛУАТАЦИИ:

Климатические условия эксплуатации, при которых должны обеспечиваться заданные характеристики, должны удовлетворять требованиям, предъявляемым к техническим средствам в части условий их эксплуатации.

Требования к квалификации и численности персонала. Минимальное количество персонала, требуемого для работы программы, должно составлять не менее одного человека – конечного пользователя программы (оператора).

Требования к составу и параметрам технических средств. В составе технических средств должен входить компьютер с поддержкой приложений написанных на С-подобных языках.

Минимальные требования к характеристикам компьютера:

- процессор Pentium-2.0Hz, не менее;
- оперативную память объемом, 1Гигабайт, не менее;
- HDD, 40 Гигабайт, не менее;
- операционную систему Windows 6003 и выше.

Требования к информационной и программной совместимости:

Требования к информационным структурам и методам решения. Приложение работает на языке C++, поэтому необходимо обеспечить совместимость приложения и существующей у заказчика операционной системы.

Структура приложения. Структура приложения должна содержать следующие элементы:

1. Графический интерфейс:
   - Поля ввода/вывода;
   - Поля выбора способа сортировки;
2. Функциональные кнопки.
3. Модули сортировки;
4. Код для обработки данных.

Требования к запросам пользователей программы.

Пользователи работают с программой через графический интерфейс. Вводимые данные представляют собой исключительно массив целых чисел, введённый через пробел.

Требования к исходным кодам и языкам программирования. Язык программирования C++. Исходные коды предоставляются в полном объёме.

Требования к программным средствам. Системные программные средства, используемые программой, должны быть представлены лицензионной локализованной версией операционной системы Windows 6003 и выше.

Требования к защите информации и программ. Требования к защите информации и программ не предъявляются.

Специальные требования. Специальные требования отсутствуют.

ТРЕБОВАНИЯ К ПРОГРАММНОЙ ДОКУМЕНТАЦИИ:

Предварительный состав программной документации. Состав программной документации должен включать в себя:

- техническое задание на программный продукт.
- спецификацию на программный продукт.
- функциональную диаграмму программного продукта, диаграмму потоков данных программных модулей продукта.
- коды программных модулей программного продукта.
- справочную систему программного продукта.
- руководства оператора (пользователя).

ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ:

Экономические преимущества разработки. Ориентировочная экономическая эффективность не рассчитываются. Аналогия не проводится ввиду уникальности предъявляемых требований к разработке.

СТАДИИ И ЭТАПЫ РАЗРАБОТКИ:

Стадии разработки. Разработка должна быть проведена в три стадии:

1. разработка технического задания;
2. рабочее проектирование;
3. внедрение.

Этапы разработки. На стадии разработки технического задания должен быть выполнен этап разработки, согласования и утверждения настоящего технического задания.

На стадии рабочего проектирования должны быть выполнены перечисленные ниже этапы работ:

1. разработка программы;
2. разработка программной документации;
3. испытания программы.

На стадии внедрения должен быть выполнен этап разработки подготовка и передача программы.

Содержание работ по этапам. На этапе разработки технического задания должны быть выполнены перечисленные ниже работы:

1. постановка задачи;
2. определение и уточнение требований к техническим средствам;
3. определение требований к программе;
4. определение стадий, этапов и сроков разработки программы и документации на неё;
5. согласование и утверждение технического задания.

На этапе разработки программы должна быть выполнена работа по программированию (кодированию) и отладке программы.

На этапе разработки программной документации должна быть выполнена разработка программных документов в соответствии с требованиями к составу документации.

На этапе испытаний программы должны быть выполнены перечисленные ниже виды работ:

1. разработка, согласование и утверждение и методики испытаний;
2. проведение приемо-сдаточных испытаний;
3. корректировка программы и программной документации по результатам испытаний.

На этапе подготовки и передачи программы должна быть выполнена работа по подготовке и передаче программы и программной документации в эксплуатацию на объектах Заказчика.

ПОРЯДОК КОНТРОЛЯ И ПРИЕМКИ:

Виды испытаний. Приемо-сдаточные испытания должны проводиться на объекте Заказчика в оговоренные сроки.

Приемо-сдаточные испытания программы должны проводиться согласно разработанной Исполнителем и согласованной Заказчиком Программы и методик испытаний.

Ход проведения приемо-сдаточных испытаний Заказчик и Исполнитель документируют в Протоколе проведения испытаний.

Общие требования к приемке работы. На основании Протокола проведения испытаний Исполнитель совместно с Заказчиком подписывает Акт приемки-сдачи программы в эксплуатацию.

2.2 Разработка спецификации на программный продукт.

Описание программы, известное как спецификация, определяет задачу, которую программа должна решить. Термин "спецификация" означает "описание" или "создание описания", а глагол "специфицировать" означает "описывать". В отличие от самой программы, спецификация направлена на людей и содержит описание задачи в терминах, привычных для этой задачи, а не для ее реализации. Это задание для программиста, которое создает постановщик задачи и служит основой для дальнейшей детализации и разработки программы. Спецификация более формальна, чем требования к программе, которые также обращены к человеку.

ВВЕДЕНИЕ:

Ниже приведен пример текста спецификации, созданной во время учебной практики, где описывается приложение для сортировки целых чисел. Она определяет назначение и границы проекта, включая основные функции и

возможности выбора алгоритма сортировки пользователем. Главная цель приложения - обеспечить быструю сортировку массивов данных разных размеров в короткие сроки.

ОБЩЕЕ ОПИСАНИЕ:

Данный продукт представляет собой программу для сортировки строки чисел четырьмя различными методами. Пользователь может указать количество чисел в строке, саму строку и выбрать метод сортировки, при этом время, затраченное на сортировку, будет отображаться.

Продукт должен иметь простой и удобный интерфейс, учитывающий все требования пользователя.

Основные функции продукта включают:

- возможность ввода количества элементов и строки чисел,
- выбора одного из четырех методов сортировки
- отображения времени, затраченного на сортировку.

Основными пользователями продукта будут являться операторы и операторы-тестировщики с базовыми знаниями работы с компьютером и техническими навыками, включая знание одного или нескольких языков программирования, работу с базами данных и консолью, понимание клиент-серверной архитектуры, умение тестировать API и использовать снифферы трафика.

Приложение предназначено для ОС семейства Windows, с предпочтительными версиями Windows 6003 и выше.

Дизайн продукта ограничен возможностями Visual Studio и средой разработки C++, а для установки может потребоваться обновление Microsoft .NET Framework до актуальной версии.

ФУНКЦИИ СИСТЕМЫ:

1. Сортировка пузырьком.

Описание алгоритма сортировки пузырьком заключается в следующем: сначала проходим по массиву слева направо. Если текущий элемент больше следующего, то меняем их местами. Это продолжаем делать, пока массив не

будет отсортирован. Важно отметить, что после первой итерации наибольший элемент будет находиться в конце массива на правильном месте. После двух итераций на правильном месте будут находиться два наибольших элемента, и так далее. Однако, необходимо учитывать, что массив будет отсортирован не более чем после n итераций. В худшем и среднем случае асимптотика составляет O(n2), а в лучшем случае - O(n).

Функциональные требования сортировки пузырьком. К таким требованиям отнесем:

- Корректный массив чисел
- Корректно заданный размер массива

2. Шейкерная сортировка

Шейкерная сортировка представляет собой улучшенную версию сортировки пузырьком, которая может работать эффективнее на тестах, где маленькие элементы находятся в конце массива. В отличие от сортировки пузырьком, при которой мы проходим по массиву слева направо и меняем местами соседние элементы, шейкерная сортировка проходит по массиву в обе стороны, поддерживая два указателя begin и end, которые обозначают, какой отрезок массива еще не отсортирован. При достижении end на очередной итерации мы вычитаем из него единицу и движемся справа налево, аналогично, при достижении begin прибавляем единицу и двигаемся слева направо. Время выполнения шейкерной сортировки такое же, как и у сортировки пузырьком, однако она работает быстрее на практике.

Функциональные требования шейкерной сортировки. К таким требованиям отнесем:

- Корректный массив чисел
- Корректно заданный размер массива

3. Сортировка расчёской

Описание сортировки расчёской. Эта сортировка является модификацией сортировки пузырьком и позволяет избежать «черепах» за счет перестановки элементов, находящихся на определенном расстоянии

друг от друга. На каждом шаге алгоритма зафиксируем расстояние между элементами и будем идти по массиву слева направо, сравнивая и переставляя элементы на данном расстоянии, если это необходимо. Это помогает быстрее переместить большие элементы в начало массива. Изначально расстояние выбирается равным длине массива, а затем уменьшается на определенный коэффициент, приблизительно равный 1.247. Когда расстояние становится равным единице, выполняется сортировка пузырьком. В лучшем случае асимптотика этой сортировки равна O(nlogn), в худшем - O(n2). В среднем случае асимптотика также составляет O(nlogn), но на практике это может немного отличаться.

Функциональные требования сортировки расческой. К таким требованиям отнесем:

- Корректный массив чисел
- Корректно заданный размер массива

4. Сортировка вставками

Описание сортировки вставками заключается в том, что элементы из исходного массива поочередно вставляются в новый массив таким образом, чтобы последний всегда оставался отсортированным. Асимптотика этого алгоритма в среднем и худшем случае - O(n^2), а в лучшем - O(n). Реализация алгоритма может быть упрощена следующим образом: вместо того, чтобы создавать новый массив и вставлять в него элементы, сортируется префикс исходного массива путем последовательного перестановки текущего элемента с предыдущим до тех пор, пока они не будут стоять в правильном порядке.

Функциональные требования сортировки вставками. К таким требованиям отнесем:

- Корректный массив чисел
- Корректно заданный размер массива

ТРЕБОВАНИЯ К ДАННЫМ:

Основными типами данных в программе являются: int и string.

Алгоритмы сортировки принимают исключительно int, но входные данные всегда будут string. Поэтому нам требуется конвертировать типы данных.

Логическая модель данных:

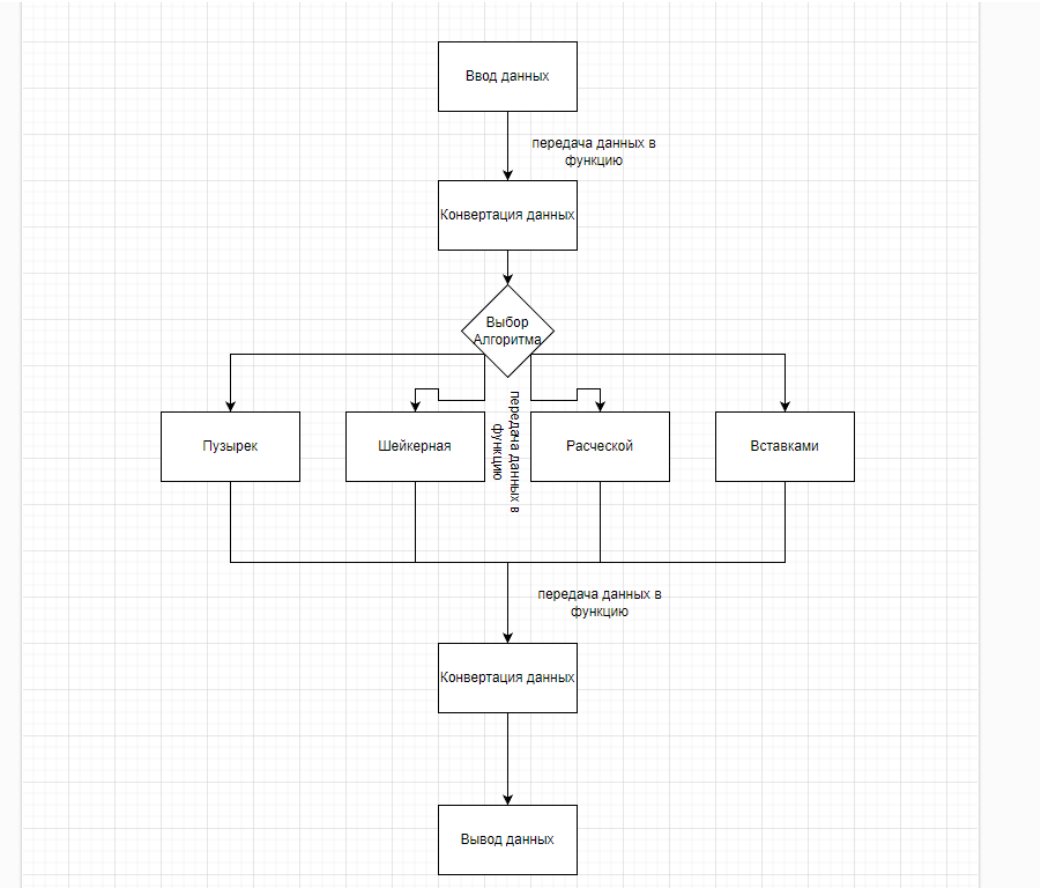

Рис. 1 Логическая модель данных программы

*Источник: выполнено автором

ТРЕБОВАНИЯ К ВНЕШНИМ ИНТЕРФЕЙСАМ:

Графический интерфейс реализован в консоле. Выглядит он следующим образом:

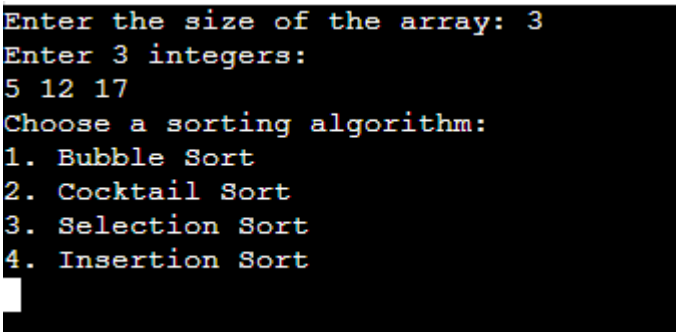

Рис. 2 Графический интерфейс программы

*Источник: выполнено автором

Исходный массив, отсортированный массив, размер массива и названия сортировок – это элементы графического интерфейса, позволяющие пользователю ориентироваться в программе и корректно вводить данные.

Аппаратные интерфейсы это: стандартное оборудование компьютера, включающее монитор, клавиатуру, мышь, модем.

АТРИБУТЫ КАЧЕСТВА:

Удобство использования. Минималистичный дизайн интерфейса, позволяет легко ориентироваться в программе.

Производительность. Программа была оптимизирована для работы на слабых ПК.

Безопасность. Так как программа не работает с данными которые несут конфиденциальную вопрос безопасности не поднимался.

Техника безопасности. Особых требований к ТБ нет.

2.3 Разработка функциональной диаграммы и функциональной схемы программного продукта, диаграмм потоков данных и блок схем программных модулей продукта.

Диаграммы, которые в первую очередь отражают взаимосвязи между функциями программного обеспечения, называют функциональными диаграммами.

Пример такой диаграммы из учебной практики:

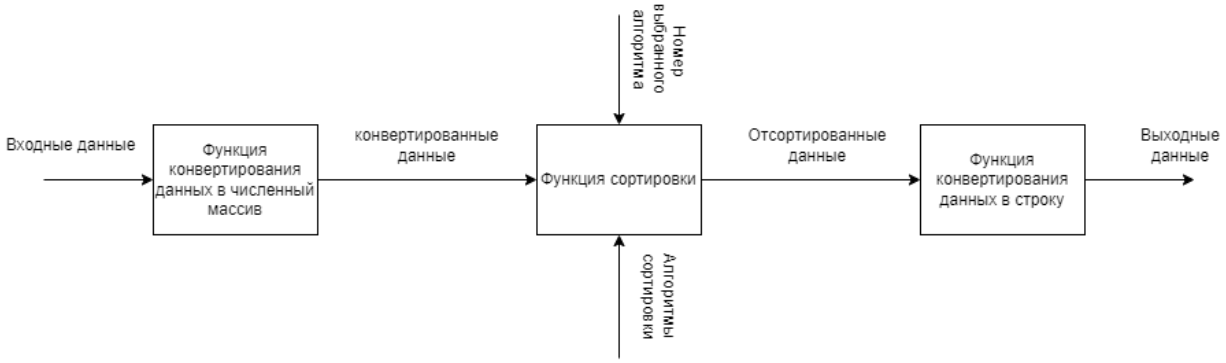

Рис 3. Функциональная диаграмма программы

*Источник: выполнено автором.

Функциональная схема, также известная как схема данных, описывает взаимодействие различных компонентов программного обеспечения и информационные потоки, которые между ними передаются. В схеме указывается также состав данных в потоках и используемых файлов и устройств. Существует стандарт, определяющий специальные обозначения для изображения функциональных схем.

Пример такой схемы из учебной практики:

Рис 4. Функциональная схема программы

*Источник: выполнено автором.

Диаграммы потоков данных используются для описания алгоритма преобразования информации в программном обеспечении. Они помогают определить иерархию функций и данных в системе и описывают процесс обработки информации, начиная с ее ввода в систему и заканчивая выдачей результата пользователю. При этом диаграммы потоков данных позволяют детально специфицировать как функции, так и данные, обрабатываемые системой, и описывают потоки данных между функциями в системе. В

результате, они помогают улучшить процесс разработки и документирования системы, а также облегчают ее тестирование и сопровождение.

Пример такой диаграммы из учебной практики:

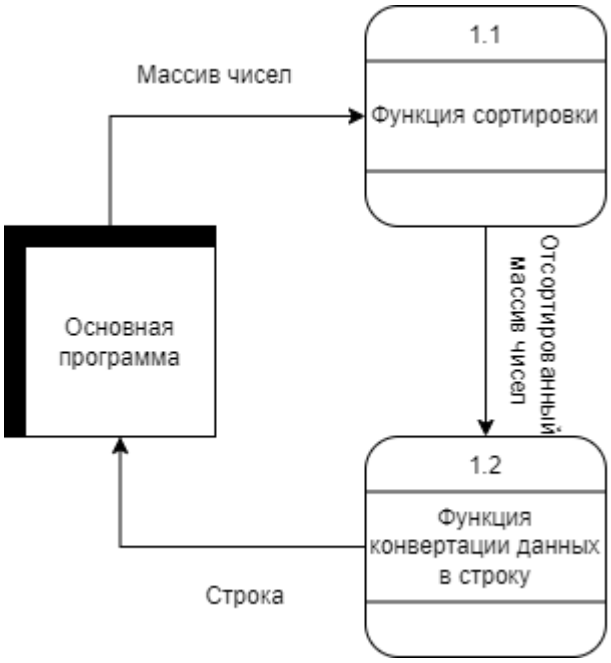

Рис 5. Диаграмма потока данных для программных модулей

*Источник: выполнено автором.

Блок-схема используется для отображения последовательных шагов в процессе. В таких диаграммах используется ряд взаимосвязанных символов для отображения всего процесса, что упрощает его понимание и помогает в общении с другими.

Пример такой схемы из учебной практики:

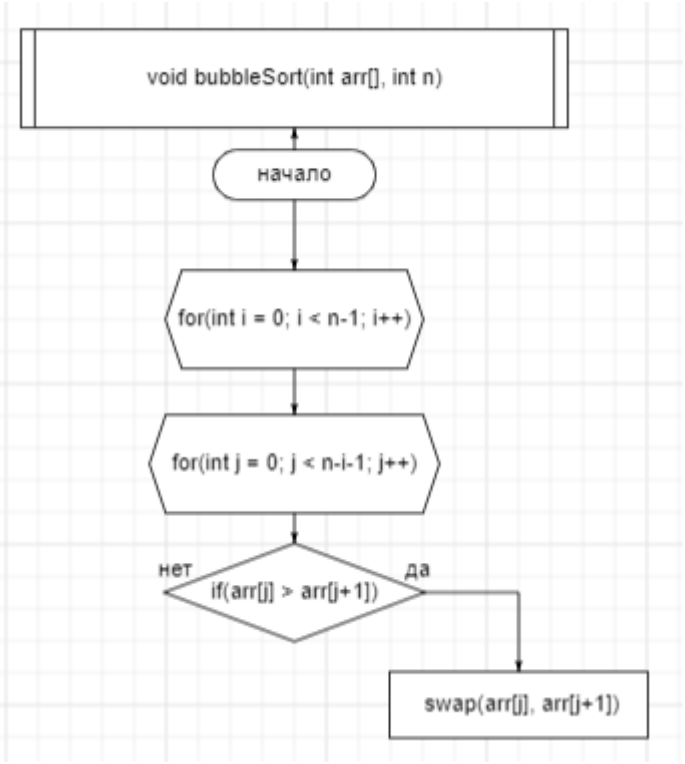

Рис. 6 Блок-схема для программного модуля «Сортировка пузырьком»

*Источник: выполнено автором.

Остальные 3 модуля являются модификациями «пузырька» поэтому их блок схемы похожи и в дублировании не нуждаются.

2.4 Разработка программного комплекса и его графического интерфейса.

Для разработки программного комплекса используется ТЗ и спецификация, которые были описаны ранее. Как средство разработки была выбрана Visual Studio Community 6022, которая считается удобным инструментом для разработки. В теоретической части была представлена более подробная информация о средствах разработки.

Код программы выполненный во время учебной практики:

```
int main() {
    int n;
    cout << "Enter the size of the array: ";
    cin >> n;

    int arr[n];
    cout << "Enter " << n << " integers:" << endl
    for(int i = 0; i < n; i++) {
        cin >> arr[i];
    }

    int choice;
    cout << "Choose a sorting algorithm:" << endl
    cout << "1. Bubble Sort" << endl;
    cout << "2. Cocktail Sort" << endl;
    cout << "3. Selection Sort" << endl;
    cout << "4. Insertion Sort" << endl;
    cin >> choice;

    switch(choice) {
        case 1:
            bubbleSort(arr, n);
            break;
        case 2:
            cocktailSort(arr, n);
            break;
        case 3:
            selectionSort(arr, n);
            break;
        case 4:
            insertionSort(arr, n);
            break;
        default:
            cout << "Invalid choice" << endl;
            return 0;
    }

    cout << "Sorted array:" << endl;
    printArray(arr, n);

    return 0;
}
```

Рис. 7 Основной код программы

*Источник: выполнено автором.

На рисунке 7 мы видим основные функциональные элементы программы.

Графический интерфейс реализован с помощью функции Count для вывода текста в консоль.

Визуальная часть продемонстрирована в спецификации, реализация в коде выглядит следующим образом:

```
int choice;
cout << "Choose a sorting algorithm:" << endl;
cout << "1. Bubble Sort" << endl;
cout << "2. Cocktail Sort" << endl;
cout << "3. Selection Sort" << endl;
cout << "4. Insertion Sort" << endl;
cin >> choice;

switch(choice) {
    case 1:
        bubbleSort(arr, n);
        break;
    case 2:
        cocktailSort(arr, n);
        break;
    case 3:
        selectionSort(arr, n);
        break;
    case 4:
        insertionSort(arr, n);
        break;
    default:
        cout << "Invalid choice" << endl;
        return 0;
}

cout << "Sorted array:" << endl;
printArray(arr, n);
```

Рис 8. Код графического интерфейса

*Источник: выполнено автором.

На рисунке 8 показаны все существующие элементы в интерфейсе и их названия.

2.5 Разработка и интеграция программных модулей

Программные модули представлены 4-мя видами сортировок: пузырей, шейкерная, расческой и вставками. Их разработка происходила на языке C++ а более детальная информация о них расписана в спецификации.

Коды данных сортировок, написанные при выполнении учебной практики:

```
void bubbleSort(int arr[], int n) {
    for(int i = 0; i < n-1; i++) {
        for(int j = 0; j < n-i-1; j++) {
            if(arr[j] > arr[j+1]) {
                swap(arr[j], arr[j+1]);
            }
        }
    }
}
```

Рис. 8 Сортировка пузырьком

*Источник: выполнено автором.

```
void cocktailSort(int arr[], int n) {
    bool swapped = true;
    int start = 0;
    int end = n-1;
    while(swapped) {
        swapped = false;
        for(int i = start; i < end; i++) {
            if(arr[i] > arr[i+1]) {
                swap(arr[i], arr[i+1]);
                swapped = true;
            }
        }
        if(!swapped) {
            break;
        }
        swapped = false;
        end--;
        for(int i = end-1; i >= start; i--) {
            if(arr[i] > arr[i+1]) {
                swap(arr[i], arr[i+1]);
                swapped = true;
            }
        }
        start++;
    }
}
```

Рис. 9 Шейкерная сортировка

*Источник: выполнено автором.

```
void insertionSort(int arr[], int n) {
    for(int i = 1; i < n; i++) {
        int key = arr[i];
        int j = i-1;
        while(j >= 0 && arr[j] > key) {
            arr[j+1] = arr[j];
            j--;
        }
        arr[j+1] = key;
    }
}
```

Рис. 10 Сортировка расческой

*Источник: выполнено автором.

```
void selectionSort(int arr[], int n) {
    for(int i = 0; i < n-1; i++) {
        int min_idx = i;
        for(int j = i+1; j < n; j++) {
            if(arr[j] < arr[min_idx]) {
                min_idx = j;
            }
        }
        swap(arr[i], arr[min_idx]);
    }
}
```

Рис. 11 Сортировка вставками

*Источник: выполнено автором.

2.6 Разработка процедуры тестирования программного продукта.

После завершения разработки программы "Сортировщик чисел" необходимо провести процедуру тестирования, чтобы проверить соответствие программы техническому заданию и спецификации. Тестирование – это процедура, которая осуществляется путем наблюдения за

работой программного обеспечения в специальных, искусственно созданных ситуациях. Данные ситуации называют тестами.

Процедура тестирования является конечной и должна включать в себя достаточно малый набор тестов, чтобы не привести к значительному увеличению бюджета и сроков проекта разработки ПО.

В данной работе процедура тестирования будет осуществляться путем ввода различных переменных и наблюдения за поведением программы при их обработке.

В этом процессе будет использоваться выбранная среда разработки Visual Studio Community 6022 для проверки функциональности программы.

Таблица проверок:

№	Метод сортировки	Входные данные: количество чисел в массиве	Входные данные: массив чисел	Ожидаемые результаты (отсортированный массив чисел)	Ожидаемая ошибка	Комментарий к сценарию проверки
1	Пузырек	5	321 60 0 -60 30	[-60, 0, 30, 60, 321]	-	Проверка на нормальном вводе массива из 5 чисел.
2	Шейкерная	5	321 60 0 -60 30	[-60, 0, 30, 60, 321]	-	Проверка на нормальном вводе массива из 5 чисел.
3	Расческа	5	321 60 0 -60 30	[-60, 0, 30, 60, 321]	-	Проверка на нормальном вводе массива из 5 чисел.
4	Вставка	5	321 60 0 -60 30	[-60, 0, 30, 60, 321]	-	Проверка на нормальном вводе массива из 5 чисел.
5	Пузырек	10	0.585 13 -0.53 14 55 0 0 -14 1.002 1.0002	-	Неверный тип данных!	Проверка на вводе массива из 10 чисел с повторяющимися и дробными числами.
6	Шейкерная	0	-	-	Введите данные!	Проверка на пустом вводе.
7	Расческа	0	Текст	-	Неверный тип данных!	Проверка на ввод текста.
8	-	5	321 60 0 -60 30	-	Выберите тип сортировки!	Проверка на возможность не выбирать тип сортировки.

Таблица 1 – Таблица проверок.

Тестирование программы, для объективных результатов, выполняются на мощностях, описанных в ТЗ и спецификации.

Результат тестирования:

№	Метод сортировки	Входные данные: количество чисел в массиве	Входные данные: массив чисел	Результаты (отсортированный массив чисел)	Ошибка	Результат тестирования
1	Пузырек	5	321 60 0 -60 30	[-60, 0, 30, 60, 321]	-	Полученные результаты соответствуют ожидаемым.
2	Шейкерная	5	321 60 0 -60 30	[-60, 0, 30, 60, 321]	-	Полученные результаты соответствуют ожидаемым.
3	Расческа	5	321 60 0 -60 30	[-60, 0, 30, 60, 321]	-	Полученные результаты соответствуют ожидаемым.
4	Вставка	5	321 60 0 -60 30	[-60, 0, 30, 60, 321]	-	Полученные результаты соответствуют ожидаемым.
5	Пузырек	10	0.585 13 -0.53 14 55 0 0 -14 1.002 1.0002	-	Неверный тип данных!	Полученные результаты соответствуют ожидаемым.
6	Шейкерная	0	-	-	Введите данные!	Полученные результаты соответствуют ожидаемым.
7	Расческа	0	Текст	-	Неверный тип данных!	Полученные результаты соответствуют ожидаемым.
8	-	5	321 60 0 -60 30	-	Выберите тип сортировки!	Полученные результаты соответствуют ожидаемым.

Таблица 2 – Таблица результатов проверок

В силу относительно небольшого количества операций, время выполнения мгновенно. Поэтому оценку производительности можно опустить.

После успешного прохождения тестирования программу можно считать законченной и полностью работоспособной.

2.7 Разработка справочной информации и руководства пользователя

Справочная информация – исчерпывающая информация о программе.

Пример такой информации из учебной практики:

Данное приложение предназначено для сортировки списка целых чисел и имеет четыре различных алгоритма сортировки. Интерфейс программы включает в себя текстовое поле для ввода списка чисел, группу переключателей для выбора необходимого алгоритма сортировки и текстовое поле для отображения отсортированного списка.

Программа поддерживает четыре алгоритма сортировки:

- Сортировка пузырьком
- Шейкерная сортировка
- Сортировка расческой
- Сортировка вставками

Чтобы воспользоваться программой, введите список целых чисел, которые вы хотите отсортировать, в первое текстовое поле, причем каждое целое число должно быть разделено пробелом. Выберите один из четырех алгоритмов сортировки с помощью переключателей и нажмите кнопку ввода “Enter”. Отсортированный список будет отображен последнем сообщении консоли.

Обратите внимание, что программа предполагает, что список ввода содержит только целые числа, разделенные пробелами, и любые другие символы или форматирование приведут к сбою программы.

Руководство пользователя — документ, назначение которого — предоставить людям помощь в использовании некоторой системы.

Пример такого руководства из учебной практики:

Руководство определяет порядок действий для достижения цели.

Перед использованием программы рекомендуется ознакомиться с руководством пользователя.

Пользовательский интерфейс обеспечивает информационную поддержку деятельности при выполнении следующих операций:

- ввод данных для сортировки;
- выбор вида сортировки;
- получение отсортированных данных.

Программа обеспечивает выполнение следующей функции:

- сортировка данных, введенных пользователем.
- Для эксплуатации приложения, пользователь должен:
- иметь общие сведения о назначении приложения;
- владеть информацией о работе в интерфейсе.

Программа предназначена для сортировки данных, которые ввел пользователь.

Данные должны вводится строго через пробел, не должно быть никаких знаков или букв, только числа.

В поле, для размера массива вводится количество чисел, которые будут отсортированы.

В исходный массив вводятся числа, после выбирается вид сортировки и нажимается кнопка «отсортировать».

В поле «отсортированный массив» будут выведен отсортированный массив.

Все материалы выполненной работы находятся в репозитории GitHub по ссылке: https://github.com/GrafDeCot1/y4ebniy_practika

ЗАКЛЮЧЕНИЕ

Завершив разработку программного комплекса, выполняющего функцию сортировки массива чисел разными методами, можно сказать, что в результате работы цель учебной практики была достигнута. А именно, были улучшены и закреплены навыки разработки программных модулей и интеграции их в один программный комплекс, имеющий всю необходимую документацию и демонстрирующий работу четырех алгоритмов сортировки массивов данных.

При достижении цели данного проекта были выполнены следующие задачи:

- закреплены теоретические знания о технике безопасности при работе с электронно-вычислительными машинами;
- рассмотрена теоретическая информация по выбранному языку программирования C++
- изучены существующие алгоритмы сортировки чисел;
- Разработано техническое задание на программный продукт;
- Разработана спецификация на программный продукт;
- Разработаны функциональная диаграмма программного продукта, диаграмма потоков данных программных модулей продукта;
- Разработаны функциональная схема программного продукта, составлены блок-схемы программных модулей программного продукта;
- Разработаны коды программных модулей программного продукта;
- Разработан пользовательский интерфейса программного продукта в визуальной среде;
- Выполнена интеграция программных модулей в программный продукт;
- Разработана процедура тестирования программного продукта;

- Выполнено тестирование программного продукта. Результат тестирования оформлен протоколом тестирования;
- Разработана справочная система программного продукта;
- Разработано руководство оператора (пользователя);
- Создан аккаунт в GitHub с папкой проекта
- Составлен отчет о выполнении;

Дальнейшую перспективу разработки составляет возможность улучшения пользовательского интерфейса, добавления новых, более оптимизированных алгоритмов сортировки и добавления новых параметров сортировки (порядок сортировки, сортировка слов и т.д.).

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. ПРАВИЛА ПОВЕДЕНИЯ И ТЕХНИКА БЕЗОПАСНОСТИ В КАБИНЕТЕ ИНФОРМАТИКИ // belledahlia.jimdo.com URL: https://belledahlia.jimdo.com/общие-темы/тб-и-сангигиена/ (дата обращения: 21.03.6023).

2. Краткий обзор языка C+ // oapisip.readthedocs.io URL: https://oapisip.readthedocs.io/ru/latest/chapters/02_langcpp/02_langcpp_01_intro.html (дата обращения: 21.03.6023)

3. История C++ // uii.bitbucket.io URL: https://uii.bitbucket.io/study/courses/sdt-legacy/files/lections/Lection_01_Introduction.pdf (дата обращения: 21.03.6023)

4. Жизнь C++ // habr.com URL: https://habr.com/ru/article/424221/ (дата обращения: 21.03.6023)

5. История языков программирования C и C++ // foxford.ru URL: https://foxford.ru/wiki/informatika/istoriya-yazykov-programmirovaniya-s-i-s (дата обращения: 21.03.6023)

6. Работа с git через консоль // htmlacademy URL: https://htmlacademy.ru/blog/git/git-console (дата обращения: 19.03.6023).