Министерство образования и науки Российской Федерации
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ Н.Г.ЧЕРНЫШЕВСКОГО»

Кафедра дискретной математики
и информационных технологий

# Разработка лабораторных работ по дисциплине «Технологии программирования»

АВТОРЕФЕРАТ БАКАЛАВРСКОЙ РАБОТЫ

студента 4 курса 421 группы
направления 09.03.01 «Информатика и вычислительная техника»
факультета компьютерных наук и информационных технологий
Мусатова Ивана Алексеевича

Научный руководитель
ассистент каф. ДМиИТ _____ М.В. Белоконь
подпись, дата

Зав. кафедрой
к. ф.-м.н., доцент _____ Л.Б. Тяпаев
подпись, дата

Саратов 2018

# ВВЕДЕНИЕ

Актуальность бакалаврской работы обуславливается тем, что технологии программирования представляют собой инженерную дисциплину, являющуюся неотъемлемой частью современного программирования. Они позволяют заметно упрощать процесс разработки, улучшать надежность программных продуктов и тем самым увеличивать производительность труда и результативность, что значительно повышает шансы проектов на коммерческий успех.

Целью бакалаврской работы является создание комплекса лабораторных работ по дисциплине «Технологии программирования». Для достижения поставленной цели необходимо решить следующие задачи:

1. Рассмотреть использование системы контроля версий для различных проектов, понятия и свойства объектно-ориентированного программирования, работу с единицами трансляций, разработку графического интерфейса пользователя для разных проектов, тестирование и отладку программных продуктов.
2. Разработать лабораторные работы с использованием системы контроля версий Git, сервера для хранения удаленных репозиториев Bitbucket, IDE Qt Creator и встроенного конструктора Qt Designer, а также компилятора GCC в комплекте с GNU Symbolic Debugger.

# 1 Основные этапы развития технологий программирования

За всю историю становления, в технологии программирования произошла огромная масса событий, выдвигалось множество различных идей, кардинально меняющих методы и подходы в процессе написания исходных кодов программ, а также заметно упрощающих разработку программных продуктов на любом ее этапе.

Историю развития технологии программирования можно разделить на три основных периода: дореволюционный, революционный, послереволюционный [18].

## 1.1 Дореволюционный период

Стоимость электронно-вычислительных машин была очень высокой, их количество было малым, эксплуатация преподносила множество трудностей, а сфера применения в основном ограничивалась на специальных целях, таких как оборона и космос. Программирование, как профессия являлось узконаправленным, и в данной отрасли были задействованы специалисты только той области, в которой применялись компьютеры.

Основными технологическими идеями, продвигаемыми в этом периоде, становятся появление языков программирования и компиляторов для них, а также выделение модульного программирования как важной части этапа разработки программ и основы для сбора и накопления библиотек программ для их дальнейшего повторного использования.

## 1.2 Период революции в программировании

К середине шестидесятых годов электронно-вычислительные машины стали дешевле, более компактнее и производительнее, а область применения компьютеров расширилась. Эксплуатация программного обеспечения и программных продуктов перестала быть доступным только небольшому, определенному кругу лиц, а сама профессия программиста приобрела статус массовой.

Такие резкие перемены предполагали не только положительные последствия, но и отрицательные в виде низкой надежности используемого пользователями программного продукта и низкого уровня коммерчески успешных проектов у разработчиков.

Для решения всех скопившихся проблем и возникающих вопросов в срочном порядке были предложены множество новых идей и подходов для выхода из сложившейся ситуации.

### 1.3 Послереволюционный период

Революция в данной сфере была успешной – в кратчайшие сроки технология программирования предстала как инженерная дисциплина, и многие ее аксиомы стали применяться в практическом использовании. Надежность программных продуктов заметно повысилась за счет развития практики систематического тестирования. Основные идеи, предложенные в шестидесятые годы, все более эволюционировали и совершенствовались.

В современности применение консервативных усовершенствованных идей вполне хватает для организации программирования на персональных компьютерах, но в обозримом будущем все с большим развитием и распространением встроенных систем, глобально превосходящих по количеству персональные компьютеры, может сложиться ситуация по нехватке методов организации программирования, что повлечет новые революционные изменения в технологии программирования.

## 2 Дисциплина «Технологии программирования»

Целью изучения данной дисциплины является теоретическая и практическая подготовка студентов в области разработки программных продуктов с использованием необходимых технических, алгоритмических, программных и технологических средств.

В результате изучения дисциплины технологии программирования студенты должны будут иметь представление о: жизненном цикле программного обеспечения, командной работе, планировании и управлении проектами, тестировании и обеспечении качества программного продукта, чтении и использовании документации. Они также смогут самостоятельно получать необходимые для работы знания из предметной области, составлять техническое задание для проектов, осуществлять выбор необходимых алгоритмов для реализации, выбирать необходимые для работы программные и инструментальные средства для разработки программного обеспечения, создавать графический пользовательский интерфейс, выполнять интеграцию проектов, организовывать тестирование и проверку стабильности разрабатываемого программного продукта.

### 2.1 Лабораторная работа № 1. Знакомство с Git

В ходе лабораторной работы происходит ознакомление с системой контроля версий Git и ее базовыми командами. Для этого, производится клонирование учебного репозитория example, происходит полный его разбор, изменение исходного кода и фиксация изменений с оставлением комментариев к проделанной работа. Итоговая версия репозитория загружается на сервер хранения удаленных репозиториев Bitbucket.

### 2.2 Лабораторная работа № 2. Применение основных принципов объектно-ориентированного программирования с использованием языка программирования C++ и системы контроля версий Git

Лабораторная работа состоит из двух основных заданий. В первом необходимо произвести разбор и научиться применению основных принципов объектно-ориентированного программирования, а именно: полиморфизма, инкапсуляции и наследования с помощью учебного репозитория oop. Во втором, основываясь на репозитории CoffeeMachine – проект вендингого автомата по разливу напитков, расширить функциональность уже существующих классов и их методов, а также добавить новые функции и классы.

### 2.3 Лабораторная работа № 3. Работа с единицами трансляций, интерфейсом и реализацией классов

В данной лабораторной работе происходит исследование методов разбиения исходных текстов программ на дополнительную единицу трансляций, при помощи создания нового заголовочного файла oop.h и переноса объявлений всех классов и глобальных функций в данный файл, а также создание oop.cpp и перенос в него всех не подставляемых методов классов и глобальных функций. А также добавление стража включения. Под самостоятельную работу отводится разбиение на дополнительную единицу трансляции, собственной версии репозитория CoffeeMachine, полученной по итогам прошлой лабораторной работы.

### 2.4 Лабораторная работа № 4. Создание графического интерфейса пользователя для простых приложений

Целью данной лабораторной работы является освоение начальных навыков по проектированию и разработке графического интерфейса пользователя для определенного приложения, при помощи встроенного в Qt Creator конструктора Qt Designer. В ходе данной работы происходит создание проекта SaveTextToFile по шагам с их полным разбором. Также студентам по вариантам дается самостоятельная работа по созданию простого графического приложения с использованием Qt Creator.

### 2.5 Лабораторная работа № 5. Продолжение работы с графическим интерфейсом. Разработка графического интерфейса пользователя для проекта CoffeeMachine

Данная лабораторная работа является продолжением предыдущей и ознакомляет обучающихся с более продвинутым уровнем навыков по разработке графического интерфейса пользователя, путем полного разбора примера создания графического интерфейса для проекта CoffeeMachine. Под самостоятельную часть отводится задание по созданию графического интерфейса пользователя для репозитория CoffeeMachine, полученного в результат выполнения лабораторных работ №2, № 3. Данную работу необходимо выполнить, основываясь на наработках, представленных в репозитории GraphicCoffeeMachine.

### 2.6 Лабораторная работа № 6. Освоение навыков отладки программ. Работа с Debugger

Данная лабораторная работа является заключительной и в ней происходит освоение навыков отладки программ и исправления различных, найденных в коде ошибок и предупреждений из учебного репозитория DebugExample, при помощи GNU Symbolic Debugger компилятора GCC в Qt Creator.

# ЗАКЛЮЧЕНИЕ

В выпускной квалификационной работе были рассмотрены основные этапы развития, понятия и аксиомы дисциплины «Технологии программирования». Был разработан комплекс лабораторных работ по дисциплине «Технологии программирования». В процессе написания составлено шесть лабораторных работ по разных темам и с различным уровнем сложности выполнения.

В результате выполнения лабораторных работ студенты смогут научиться:

- работать с широко использующейся в данной время системой контроля версий Git и сервером для хранения удаленных репозиториев Bitbucket;
- применять свойства и признаки объектно-ориентированного программирования в программных продуктах;
- разбивать программы на дополнительную единицу трансляции с применением стража включения;
- применять методы создания и доработки графического интерфейса пользователя для разных проектов при помощи IDE Qt Creator в комплекте со встроенным конструктором Qt Designer;
- тестировать и отлаживать программы при помощи использования компилятора GCC в комплекте с GNU Symbolic Debugger.

## СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ


1 Википедия [Электронный ресурс]: свободная энциклопедия / текст доступен по лицензии Creative Commons Attribution-ShareAlike : Wikimedia Foundation Inc, некоммерческой организации. Электронные данные (1469331 стастей, 564892 страниц, 209358 загружаемых файлов). Wikipedia®, 2001 – URL: https://ru.wikipedia.org/wiki (дата обращения 03.05.2018). Загл. с экрана. Яз. рус.

2 IDE Qt Creator [Электронный ресурс] // Qt [Электронный ресурс ]. URL: https://www.qt.io (дата обращения 05.05.2018). Загл. с экрана. Яз. англ.

3 Система контроля версий Git [Электронный ресурс] // Git-scm [Электронный ресурс]. URL: https://git-scm.com/about (дата обращения 05.05.2018). Загл. с экрана. Яз англ.

4 Команды Git [Электронный ресурс] // Хабрахабр [Электронный ресурс]. URL: https://habr.com/post/60347/ (дата обращения 06.06.2018). Загл. с экрана. Яз. рус.

5 Алексеев, Е. Р. Программирование на языке C++ в среде Qt Creator. [Текст] // Е. Р. Алексеев, Г. Г. Злобин, Д. А. Костюк, О. В. Чеснокова, А. С. Чмыхало – Москва: ALT Linux, 2015. – 448с. (дата обращения 07.05.2018)

6 Компилятор GCC [Электронный ресурс] // The GNU Compiler [Электронный ресурс]. URL: http://gcc.gnu.org (дата обращения 07.05.2018). Загл. с экрана. Яз. англ.

7 Butbucket Tutorial [Электронный ресурс] // Atlassian [Электронный ресурс]. URL: https://confluence.atlassian.com/bitbucket/tutorials-755338051.html (дата обращения 08.05.2018). Загл. с экрана. Яз. англ.

8 Шлее, М. Qt 5.10. Профессиональное программирование на C++. [Текст] // Макс Шлее – СПб: БХВ-Петербург, 2018. – 1072 с. (дата обращения 10.05.2018).

9 Чакон С. Git для профессионального программиста. [Текст] // Скотт Чакон, Бен Штрауб – СПб: Питер, 2016 – 496 с. (дата обращения 11.05.2018).
10 Git и Bitbucket за 20 минут [Электронный ресурс] // Голобурдин [Электронный ресурс]. URL: http://goloburdin.blogspot.com/2013/11/git-bitbucket-20.html (дата обращения 11.05.2018). Загл. с экрана. Яз. рус.
11 Страуструп, Б. Программирование: принципы и практика использования C++ [Текст] // Бьярне Страуструп – Москва: Вильямс, 2011. – 1248 с. (дата обращения 12.05.2018).
12 Фазы трансляций [Электронный ресурс] // Microsoft Library [Электронный ресурс]. URL: https://msdn.microsoft.com/ru-ru/library/bxss3ska.aspx (дата обращения 13.05.2018). Загл. с экрана. Яз. рус.
13 Сигналы и слоты [Электронный ресурс] // CrossPlatform [Электронный ресурс]. URL: http://doc.crossplatform.ru/qt/4.3.2/signalsandslots.html (дата обращения 15.05.2018). Загл. с экрана. Яз. рус.
14 Бланшет, Ж. Qt 4: Программирования GUI на C++ [Текст] // Жасмин Бланшет, Марк Саммерфилд – Москва: КУДИЦ-ПРЕСС, 2008. – 718 с. (дата обращения 17.05.2018)
15 Технология программирования и отладки [Электронный ресурс] // НОУ ИНТУИТ [Электронный ресурс]. URL: https://www.intuit.ru/studies/courses/41/41/lecture/1245%3Fpage%3D3 (дата обращения 20.05.2018). Загл. с экрана. Яз. рус.
16 Мэйерс С. Эффективное использование C++. [Текст]: 55 верных способов улучшить структуру и код ваших программ // Скотт Мэйерс – Москва: ДМК-Пресс, 2017 – 300 с. (дата обращения 21.05.2018).
17 Исправление ошибок компиляции – процесс компиляции [Электронный ресурс] // CppStudio [Электронный ресурс]. URL: http://cppstudio.com/post/2674/ (дата обращения 22.05.2018). Загл. с экрана. Яз. рус.

18 Новиков Ф.А. Учебно-методическое пособие по дисциплине «Технологические подходы к разработке программного обеспечения» [Текст] // Ф. А. Новиков – Санкт-Петербург, 2007 – 137 с. (дата обращения 23.05.2018).

19 Qt Componets для десктопа [Электронный ресурс] // Хабрахабр [Электронный ресурс]. URL: https://habr.com/post/133754/ (дата обращения 24.05.2018). Загл. с экрана. Яз. рус.

20 The Real History of the GUI [Электронный ресурс] // SitePoint [Электронный ресурс]. URL: https://www.sitepoint.com/real-history-gui/ (дата обращения 25.05.2018). Загл. с экрана. Яз. англ.