Федеральное государственное бюджетное

образовательное учреждение высшего образования

«Саратовский государственный технический университет

имени Гагарина Ю.А.»

Институт прикладных информационных

технологий и коммуникаций

Кафедра Информационная безопасность

автоматизированных систем

Направление подготовки 10.03.01 Информационная безопасность

Расчётно-графическая работа

по дисциплине «Языки программирования»

Тема: «Игра «Tower defense»»

Выполнил: студент 1 курса

учебной группы с-ИБС11

очной формы обучения

Катаржин Михаил Антонович

Проверил: ассистент каф. ИБС

Романчук С. П.

Саратов 2020

**Аннотация**

Выполнение расчётно-графической работы можно свести к нескольким обобщённым областям, а именно: ознакомление с новой средой разработки QtCreator; ознакомление и изучение работы с GUI в QtCreator; изучение алгоритмов нахождения кратчайших путей между двумя точками в пространстве, способном содержать между ними непреодолимые преграды; разработка классов ( возможно наследуюмых и наследующих ) для различных типов нападающих, обороняющихся и препятсвующих движению объектов; разработка карт местности игрового поля.

Выполнение данной работы позволит и обязует обучиться работе с выводом окон для приложения, интерактивных виджетов и графических элементов, взаимодействию программ с периферийными устройствами отличными от клавиатуры, а именно мышь компьютерная, логике действия алгоритмов для конкретной задачи.

**Содержание**

**Введение**

**Тема расчётно-графической работы:**

Игра «Tower defense»»

**Цель расчётно-графической работы:**

Используя среду разработки Qt Creator создать рекреационное интерактивное приложения с стратегическим и эконмическим уклоном типа Tower defense на языке С++.

**Задачи рассчётно-графической работы:**

* Изучение основных возможностей QtCreator;
* Изучение GUI в QtCreator;
* Подбор алгоритма для рассчёта перемещения вражеских юнитов от точек появления;
* Создание нескольких карт для игрового пространства;
* Создание класса «вражеского юнита» с уникальными характеристиками;
  + Создание дополнительных классов вражеских юнитов с уникальными характеристиками;
* Создание класса «башни» для уменьшения единиц целостности вражеских юнитов;
  + Создание дополнительных классов башен с уникальными характеристиками;
* Создание и назначение спрайтов для «вражеских юнитов» и «башни»;
* Добавление возможности улучшения «башни» за игровую валюту;
* Создание класса «препятсвия» на «дороге» для задержания «вражеских юнитов» до уничтожения «препятствия»;
* Разработка меню приложения с выбром карт в окне игры;
* Добавление возможности улучшения башен от количества убитых вражеских юнитов;

**Видение итога выполнения работы:**

В результате выполнения работы ожидается получить рекреационное приложение двумерное с видом сверху башенной защиты с возможностю выбрать несколько различных карт, где нападали бы различные по скорости, прочности, урону и сопротивлениям врагам. Защита состояла бы из подбора места размещения и типа башен и препятствий способных к усовершенствоанию. Поражение засчитывается при уничтожении «препятсвия» - врат - преграждающего крайнюю к концу пути клетку дороги. Возможность приостановки, перезапуска и завершения игры с возвращением в меню выбора карт или завершения программы.

**Теоретическая часть**

**Выбор языка программирования: С++**

C++ - компилируемый, статически типизированный язык программирования общего назначения.

Поддерживает такие парадигмы программирования, как процедурное программирование, объектно-ориентированное программирование, обобщённое программирование. Язык имеет богатую стандартную библиотеку, которая включает в себя распространённые контейнеры и алгоритмы, ввод-вывод, регулярные выражения, поддержку многопоточности и другие возможности. C++ сочетает свойства как высокоуровневых, так и низкоуровневых языков.

Язык С++ возник в начале 1980-х годов, когда сотрудник фирмы Bell Labs Бьёрн Страуструп придумал ряд усовершенствований к языку C под собственные нужды. Когда в конце 1970-х годов Страуструп начал работать в Bell Labs над задачами теории очередей, он обнаружил, что попытки применения существующих в то время языков моделирования оказываются неэффективными, а применение высокоэффективных машинных языков слишком сложно из-за их ограниченной выразительности. Так, язык Симула имеет такие возможности, которые были бы очень полезны для разработки объемного программного обеспечения, но работает слишком медленно, а язык BCPL достаточно быстр, но слишком близок к языкам низкого уровня и не подходит для разработки объемного программного обеспечения. Страуструп дополнил язык C возможностями работы с классами и объектами. В результате практические задачи моделирования оказались доступными для решения как с точки зрения времени разработки (благодаря использованию Симула-подобных классов), так и с точки зрения времени вычислений (благодаря быстродействию C).

C++ — один из самых мощных и востребованных языков программирования. Ежедневно на нём пишут сотни приложений, зачастую использующих GUI. В противовес этому часто ставится скорость написания кода, которая, например, у интерпретируемых языков на порядок выше. Однако очевиден выбор, что лучше для конечного пользователя: время разработки приложения или его скорость работы?

Скорость исполнения кода — пожалуй, главный аргумент в пользу того, почему C++ был, есть и будет востребован в ИТ. В первую же очередь, огромное сообщество — плодородная почва для появления качественной литературы. По C++ есть несколько фундаментальных книг, по которым училось несколько поколений, есть новые, учитывающие все свежие изменения и актуальное ПО, есть масса интернет-ресурсов для обучения.

Компиляторы C++ есть на каждой операционной системе, большинство программ легко переносится с платформы на платформу, со средой разработки и библиотеками у вас точно не возникнет проблем. C++ — это демонстрация идеи классического программирования, когда 90% мыслей связано с кодом и лишь 10% с используемой периферией. Достаточно просто вспомнить, где он используется: микроконтроллеры, IoT, роботы, десктопные и мобильные приложения, веб, игры, системы моделирования, прогнозирования, обработки статистики и в нейронных сетях. Везде. Просто не существует такой области программирования, где C++ был бы бесполезен.

C++ имеет колоссальное сообщество программистов, которые постоянно делятся библиотеками, шаблонами и кодами, приходят на помощь новичкам и опытным коллегам.

С++ полезен в качестве фундамента для обучения. Java, JavaScript, C #, как и огромное количество других популярных языков программирования, содержит в основе принципы C++. Именно поэтому учебные программы многих вузов содержат курс «плюсплюс», который идёт сразу за получением основ на C. К примеру, принцип работы Java, одного из мощнейших языков современности, достаточно сложно понять, если не начать обучение с основ, которые впервые появились именно в C++. Кроме того, популярно мнение, что если вы сможете изучить C++, то любой другой язык не вызовет у вас никаких затруднений. Это что-то вроде того, как легко пытаться пересесть с механической коробки передач на автоматическую и как сложно проделать путь в обратном направлении.

Область его применения включает создание операционных систем, разнообразных прикладных программ, драйверов устройств, приложений для встраиваемых систем, высокопроизводительных серверов, а также 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.

**Выбор среды разработки: Qt Creator**

Qt Creator (ранее известная под кодовым названием Greenhouse) — кроссплатформенная свободная IDE для разработки на С, С++ и QML. Разработана Trolltech (Digia) для работы с фреймворком Qt. Включает в себя графический интерфейс отладчика и визуальные средства разработки интерфейса как с использованием QtWidgets, так и QML. Поддерживаемые компиляторы: GCC, Clang, MinGW, MSVC, Linux ICC, GCCE, RVCT, WINSCW

Основная задача Qt Creator — упростить разработку приложения с помощью фреймворка Qt на разных платформах. Поэтому среди возможностей, присущих любой среде разработки, есть и специфичные, такие как отладка приложений на QML и отображение в отладчике данных из контейнеров Qt, встроенный дизайнер интерфейсов: как на QML, так и на QtWidgets.

В Qt Сreator реализовано автодополнение, в том числе ключевых слов, введённых в стандарте C++11 (начиная с версии 2.5), подсветка кода (её определение аналогично таковому в Kate, что позволяет создавать свои виды подсветок или использовать уже готовые). Также, начиная с версии 2.4, есть возможность задания стиля выравнивания, отступов и постановки скобок.

Qt представляет собой целый набор инструментов для быстрого и удобного проектирования GUI. Qt привлекает программистов в основном тем, что она является быстрой, удобной, гибкой и кроссплатформенной. Во многом Qt обогнала даже SFML: библиотека доступна как на Windows, Linux и Mac, так и на мобильных платформах — Windows Mobile, Android и iOS. С помощью Qt были разработаны такие известные приложения, как: KDE, Opera, Google Earth и Skype. Впервые Qt был опубликован в мае 1995 года.

За счёт использования собственного фреймворка и мощного инструментария Qt позволяет быстро и удобно создавать собственные кроссплатформенные приложения. Кроме того, команды разработчиков получают возможность работать на разных платформах, используя при этом общие инструменты для разработки и отладки.

**Практическая часть**

**Заключение**

**Приложения**

**Литература**

<https://ru.wikipedia.org>

<https://ravesli.com>

<https://www.youtube.com>

<https://github.com>

<https://habr.com>

<https://prog-cpp.ru>