Муниципальное бюджетное общеобразовательное учреждение
«Средняя общеобразовательная школа №12 им академика В.И. Кудинова»
города Воткинска Удмуртской Республики

| | | |
|---|---|---|
| РАССМОТРЕНО | ПРИНЯТО | Утверждаю |
| на заседании ШМО | на педагогическом совете | директор МБОУ СОШ №12 |
| Протокол № 1 | Протокол № 1 | _____/ Г.М.Кельдибекова |
| от «30» августа 2023 г. | от «31» августа 2023 г. | приказ от 31 .08.2023 г. № 400-ос |

# РАБОЧАЯ ПРОГРАММА

## факультативного курса «Программирование на С++»

для обучающихся 8 классов

2023-2024учебный год

## Пояснительная записка

Курс рассчитан на изучение в 8 классе общеобразовательной средней школы по 68 учебных часа в год из расчета 2 учебных часа в неделю. При составлении программы использована авторская программа «Основы программирования на С++» В.Г. Тарасова, профессора кафедры программного обеспечения ИжГТУ имени М.Т. Калашникова.

Для каждого занятия подготовлен комплект задач в системе автоматической проверки решений – сайт moodle.cs.istu.ru. В системе организована регистрация участников, для каждого участника ведется учет его работы как в компьютерном классе, так и при выполнении самостоятельной работы дистанционно. Учителю доступны все решения учащихся: как ошибочные, так и прошедшие полную процедуру тестирования в автоматической системе.

***Цель программы*** – обучение программированию на языке С++ учащихся 8класса образовательных школ.

Для адаптации в современном информационном обществе важным фактором является формирование математического и алгоритмического стиля мышления, включающего индукцию и дедукцию, обобщение и конкретизацию, анализ и синтез, классификацию и систематизацию. Использование формальных языков позволяет развивать у учащихся грамотную устную и письменную речь.

**Особенностью курса** является его практическая направленность, которая служит успешному усвоению курса информатики.

Практическая значимость школьного курса программирования 8 класса состоит в том, что предметом его изучения являются количественные отношения и процессы реального мира, описанные математическими моделями в виде алгоритмов и программ на языке программирования высокого уровня. Основной целью является формирование абстрактного, логического и алгоритмического мышления.

Алгоритмические знания и умения необходимы для изучения других школьных предметов: математики, физики, химии и даже отдельных аспектов биологии.

Цель курса «Основы программирование на С++»: создание условий для изучения методов программирования на С/С++, рассмотрение различных парадигм программирования, пред- лагаемых этим языком (процедурная, функциональная, объектно-ориентированная); подготовка к использованию как языка программирования, так и методов программирования на С/С++ в учебной и последующей профессиональной деятельности в различных областях.

Задачи курса:

- формирование и развитие навыков алгоритмического и логического мышления, грамотной разработки программ;
- знакомство с основными структурами данных и типовыми методами обработки этих структур;
- приобретение навыков разработки эффективных алгоритмов и программ на основе изучения языка программирования С/С++;
- приобретение навыков поиска информации в сети Интернет, анализ выбранной информации на соответствие запросу, использование информации при решении

задач;

- формирование самостоятельности и творческого подхода к решению задач с использованием средств вычислительной техники;
- расширение кругозора обучающихся в области программирования.

## Место курса основ программирования в учебном плане

Базисный учебный план отводит на изучение информатики 1 учебный час в неделю (34 часа в год). В данной программе добавлены разделы, необходимые для успешного изучения алгоритмизации как начального этапа автоматизации производственных и информатизационных процессов, а также программирования на языке высокого уровня.

## Личностные, метапредметные и предметные результаты освоения курса основ программирования 8 класса

Личностные результаты:

- воспитание российской гражданской идентичности: патриотизма, уважения к Отечеству, осознание вклада отечественных ученых в развитие мировой науки;
- ответственное отношение к учению, готовность к саморазвитию и самообразованию;
- осознанный выбор и построение дальнейшей индивидуальной траектории образования;
- умение контролировать процесс и результат учебной деятельности;
- критичность мышления, инициатива, активность при решении алгоритмических задач.

Метапредметные результаты:

- умение самостоятельно определять цели своего обучения, развивать мотивы и интересы своей познавательной деятельности;
- умение соотносить свои действия с планируемыми результатами;
- умение определять понятия, обобщать, устанавливать аналогии, классифицировать;
- развивать компетенции в области использования информационно-коммуникационных технологий;
- умение находить информацию в различных источниках;
- умение выдвигать гипотезы;
- понимать сущности алгоритмических предписаний;
- устанавливать причинно-следственные связи, проводить доказательные рассуждения;
- умение иллюстрировать изученные понятия и свойства алгоритмов и программ.

Предметные результаты:

- осознание значения алгоритмизации и программирования для повседневной жизни;
- развитие умений работать с математическим текстом;

- выражать свои мысли с применением терминологии компьютерной математики и теоретических основ информатики и программирования;
- владение базовым понятийным аппаратом по основным разделам содержания;
- практически значимые умения и навыки алгоритмизации и программирования, их применение к решению математических и алгоритмических задач.

В результате освоения программы учащиеся должны:

**приобрести следующие профессиональные компетенции:**

**владеть:** технологиями дистанционного обучения программированию на языках высокого уровня с применением систем автоматической проверки решений;

у**меть:** разработать и реализовать приложение консольного типа в интегрированной среде разработки программ Visual Studio (или подобной); найти и устранить логические ошибки в программе в режиме пошаговой отладки;

**знать:** структуру программы и основные типы данных, управляющие конструкции языка C++, способы создания иерархических программных систем и элементы технологии разработки программного обеспечения, приемы работы с библиотекой STL.

**Итоговой аттестацией является выполнение итоговых работ по основным разделам программы**.

# 1. УЧЕБНЫЙ ПЛАН

| Наименование разделов (модулей) | Всего часов | Обязательная аудиторная учебная нагрузка, (часов) | | | Дистанционная/самостоятельная работа, часов | Формы, виды контроля |
|---|---|---|---|---|---|---|
| | | всего | в т.ч. практические занятия | в т.ч. лабораторные занятия | | |
| **Тема 1.** История языка С++ и существующие стандарты. Потоковый ввод и вывод в языке С++. Новые элементы в С++ в представлении базовых типов данных и массивов и в работе с ними. | **24*** | **18** | **6** | **6** | **6** | |
| **Тема 2.** Строки с завершающим нулем. Класс String: свойства и методы. Массивы строк. | **24*** | **18** | **6** | **6** | **6** | Итоговая работа |
| **Тема 3.** Библиотека STL: принципы проектирования и функционирования. Контейнеры, итераторы, обобщенные алгоритмы. Последовательный контейнер **вектор**: свойства и методы. | **24*** | **18** | **6** | **6** | **6** | |
| **Тема 4.** Быстрые методы и алгоритмы сортировки последовательностей с применением последовательных контейнеров. | **8*** | **6** | **2** | **2** | **2** | |
| **Тема 5.** Двумерные и многомерные структуры. Решение задач. | **16*** | **12** | **4** | **4** | **4** | Итоговая работа |
| **Тема 6.** Алгоритмы двоичного поиска и их применение к решению задач. | **16*** | **12** | **4** | **4** | **4** | |
| **Тема 7.** Последовательные контейнеры список, стек и очередь: свойства и методы. Решение задач. | **24*** | **18** | **6** | **6** | **6** | Итоговая работа |
| **Всего:** | **136*** | **102** | **34** | **34** | **34** | |

***- КОЛИЧЕСТВО ЧАСОВ ПИ ДИСТАНЦИОННОЙ РАБОТЕ**

# 1.1. УЧЕБНО-ТЕМАТИЧЕСКИЙ ПЛАН

| Наименование раздела. Тема уроков | Кол. часов | № урока | Характеристика основных видов деятельности ученика |
|---|---|---|---|
| 1 | 2 | 3 | 4 |
| **I. История языка С++ и существующие стандарты. Потоковый ввод и вывод в языке С++** | **6** | | |
| Отличие объектно-ориентированного подхода разработки программ от процедурного. | | 1-2 | *Знать*. Объектно-ориентированное программирование – подход для проектирования больших программных систем. Взаимосвязь понятий объекта и класса на примере объектов cin и cout, являющихся представителями классов iostream для потоков входной и выходной символьной информации.<br>*Уметь*. Составлять операторы ввода и вывода числовой и символьной информации для решения простых линейных алгоритмов. |
| Стиль написания исходного кода и операторы в языке С++ | | 3-4 | *Знать*. Структуру программы на языке С++. Диапазон изменения и основные операции для целых чисел. Особенности представления и обработки символов в С++. Правила записи операторов присваивания, ввода и вывода.<br>*Уметь*. Записать операторы ввода числовых и символьных значений с клавиатуры и сохранения их в переменных, операторы вывода символьных строк и значений переменных на экран монитора. Написать программу решения задачи. |
| Знакомство со средой программирования. Набор и запуск программ. | | 5-6 | *Знать*. Последовательность запуска среды программирования, создания проекта, подключения и исключения из проекта текстовых модулей с программами на С, запуска программ на выполнение.<br>*Уметь*. Создать проект в среде программирования, включить в него модуль с текстом программы, запустить программу и проанализировать результат. Зарегистрироваться в системе удаленной проверки задач, получить условие задачи, отправить решение на проверку, просмотреть результат. |
| **II. Циклы и тестовый ввод.** | **6** | | |
| Функции get(), eof() и fail() объекта cin и put() объекта cout для более точной работы с символами. | | 7-8 | *Знать*. Правило «пропуска символов-разделителей» при чтении символов с помощью объекта cin. Перегрузка функций в С++ на примере cin.get(). Средства обнаружения завершения данных в cin: биты eofbit и failbit, функции eof() и fail().<br>*Уметь*. Составлять алгоритмы, завершающие работу при завершении данных во входном потоке. |
| Циклы с пред- и постусловием при вводе символьной информации. | | 9-12 | *Знать*. Новые возможности инициализации массивов в С++.<br>*Уметь*. Написать программу решения задачи обработки |

| Наименование раздела. Тема уроков | Кол. часов | № урока | Характеристика основных видов деятельности ученика |
|---|---|---|---|
| Решение задач. | | | числовых и символьных последовательностей с применением циклов с пред- и постусловием. |
| **III. Системы счисления.** | **6** | | |
| Позиционная система счисления: алфавит, основание, представление целых чисел и преобразования | | 13-14 | *Знать.* Основы двоичной и шестнадцатеричной систем счисления, алгоритмы перевода чисел между двоичной, десятичной и шестнадцатеричной системами счисления. Внутреннее представление целых чисел и символов в памяти компьютера.<br>*Уметь.* Выполнять преобразования целых чисел из внешнего представления (текстовой десятичной записи) во внутреннее и наоборот. |
| Новые элементы в C++ в представлении целых чисел и в работе с ними. Решение задач. | | 15-18 | *Знать*. Встроенные типы данных для работы с целыми числами, правила записи и операции с целыми в языке C++.<br>*Уметь*. Разработать и отладить алгоритм и программу преобразования целого числа из одной системы счисления в другую. Написать программу решения задачи, опирающейся на свойства целого и значения отдельных разрядов числа. |
| **IV. Строки в стиле C.** | **6** | | |
| Строки с завершающим нулем. Представление в памяти, инициализация в C++11, ввод и вывод. Работа с указателями. | | 19-20 | *Знать.* Массив символов как хранилище (контейнер) строк в стиле C. Ввод и вывод строк. Понятие указателя и работа с ним.<br>*Уметь.* Разработать и отладить программу обработки строк с применением «индуктивного» способа построения алгоритма. |
| Функции для строк. Решение задач. | | 21-24 | *Знать*. Назначение и параметры основных функций для обработки строк: длина строки, копирование, сравнение строк, поиск подстрок.<br>*Уметь*. Разработать и отладить программу обработки строк с применением функций для строк. |
| **V. Класс String.** | **6** | | |
| Введение в класс string. Инициализация строк в C++11, ввод и вывод. Присваивание, сравнение и вероятные ошибки. Основные функции. | | 25-26 | *Знать.* Правила инициализации объектов типа string, операции присваивания, сцепления, сравнения, ввода и вывода. Синтаксические правила для вызова функций-членов класса string.<br>*Уметь*. Разработать и отладить программу обработки строк с применением функций для строк. |
| Функции вставки и замены для строк. Применение обобщенных алгоритмов к объектам string. Решение задач. | | 27-30 | *Знать*. Назначение и параметры функций-членов класса string: вставка и замена строк. Правила применения обобщенных алгоритмов (reverse, swap) для объектов string.<br>*Уметь*. Разработать и отладить программу обработки строк с применением функций для объектов string. |
| **VI. Массивы String.** | **6** | | |

| Наименование раздела. Тема уроков | Кол. часов | № урока | Характеристика основных видов деятельности ученика |
|---|---|---|---|
| Массивы String. | | 31-32 | *Знать.* Правила объявления и инициализации массивов string, применения функций к элементам массивов.<br>*Уметь.* Разработать и отладить программу обработки строк с применением функций для массивов string. |
| Понятие структуры. Массивы структур. Решение задач. | | 33-35 | *Знать*. Правила объявления и инициализации структур, работы с полями структуры; работы с массивами структур.<br>*Уметь*. Разработать и отладить программу обработки символьной информации с применением структур и массивов структур. |
| Контрольная работа 1 | | 36 | Применять полученные знания и умения при решении примеров и задач. |
| **VII. Векторы почти как массивы. Создание, методы.** | **6** | | |
| Отличие STL от других библиотек. Контейнеры последовательностей: вектор. Инициализация, ввод и вывод. | | 37-38 | *Знать.* Правила объявления и инициализации векторов, операции над ячейками вектора и векторами в целом. Функции-члены класса вектор `push_back(), capacity()`.<br>*Уметь.* Разработать и отладить программу обработки последовательности чисел с хранением ее в контейнере вектор. |
| Функции-члены класса вектор `clear(), swap()`. Решение задач. | | 39-42 | *Знать*. Назначение и параметры функций-членов класса вектор `clear(), swap()`.<br>*Уметь*. Разработать и отладить программу обработки последовательности чисел с хранением ее в контейнере вектор, применяя функции-члены класса вектор. |
| **VIII. Простые итераторы, алгоритмы.** | **6** | | |
| Итераторы – связующий элемент между контейнерами и алгоритмами: определения, классификация. Операции над итераторами. | | 43-44 | *Знать.* Назначение и классификацию итераторов, правила их объявления, диапазоны и допустимые операции. Функции класса вектор, возвращающие значения итераторов: `begin(), end()`.<br>*Уметь.* Разработать и отладить программу обработки последовательности чисел с хранением ее в контейнере вектор, выполняя доступ к элементам вектора с использованием итераторов. |
| Обобщенные алгоритмы: определения, классификация, примеры. Решение задач. | | 45-48 | *Знать*. Классификацию алгоритмов. Назначение и основные параметры обобщенных алгоритмов, правила использования в С++ (`adjacent_difference(), copy(), transform ()` ).<br>*Уметь*. Разработать и отладить программу обработки последовательности чисел с хранением ее в контейнере вектор, выполняя доступ к элементам вектора с использованием итераторов и обработку с использованием |

| Наименование раздела. Тема уроков | Кол. часов | № урока | Характеристика основных видов деятельности ученика |
|---|---|---|---|
| | | | обобщенных алгоритмов. |
| **IX. Входные, выходные, потоковые итераторы.** | **6** | | |
| Обратный итератор. Выходные и потоковые итераторы. | | 49-50 | *Знать.* Назначение, объявление и операции с выходными итераторами. Особенности выходных потоковых итераторов.<br>*Уметь.* Разработать и отладить программу обработки последовательности чисел с хранением ее в контейнере вектор, выполняя доступ к элементам вектора с использованием выходных итераторов и обработку с использованием обобщенных алгоритмов (`copy()`). |
| Входные и потоковые итераторы. Итераторы вставки. Решение задач. | | 51-54 | *Знать.* Назначение, объявление и операции со входными итераторами. Особенности входных потоковых итераторов и итераторов вставки.<br>*Уметь.* Разработать и отладить программу обработки последовательности чисел с хранением ее в контейнере вектор, выполняя доступ к элементам вектора с использованием входных итераторов и обработку с использованием обобщенных алгоритмов (`copy(), find()`). |
| **X. Использование сортировки.** | **6** | | |
| Модели вычислительной сложности алгоритмов – О-обозначения. Алгоритмы сортировки и их сложность. | | 55-56 | *Знать.* Способ оценки вычислительной сложности алгоритмов с помощью О-обозначений, оценки сложности распространенных алгоритмов сортировки. Обозначения параметров и правила использования обобщенного алгоритма `sort()` из библиотеки STL.<br>*Уметь.* Оценивать вычислительную сложность алгоритма решения задачи, применять функцию `sort()`. |
| Сортировки векторов структур по разным полям. Решение задач. | | 57-60 | *Знать*. Правила применения функции `sort()` для векторов встроенных типов данных, а также векторов из структур.<br>*Уметь*. Применять функцию `sort()` для упорядочивания значений как в порядке возрастания, так и в порядке убывания. Написать программу решения задачи. |
| **XI. Двумерные и многомерные структуры.** | **12** | | |
| Особенности применения контейнеров STL для хранения и обработки двумерных данных. | | 61-62 | *Знать.* Правила объявления и инициализации двумерных векторов, доступа к отдельным элементам.<br>*Уметь.* Разработать и отладить программу обработки последовательности чисел с хранением ее в двумерном контейнере вектор. |
| Решение задач с обработкой двумерных | | 63- | *Знать*. Правила объявления и инициализации двумерных векторов, доступа к отдельным элементам, особенности |

| Наименование раздела. Тема уроков | Кол. часов | № урока | Характеристика основных видов деятельности ученика |
|---|---|---|---|
| числовых и символьных данных. | | 66 | применения обобщенных алгоритмов.<br>*Уметь*. Разработать и отладить программу обработки последовательности чисел с хранением ее в двумерном контейнере вектор. |
| Особенности применения контейнеров STL для хранения и обработки простых изображений. | | 67-68 | *Знать.* Правила объявления и инициализации двумерных векторов для хранения и обработки простых изображений, доступа к отдельным элементам.<br>*Уметь.* Разработать и отладить программу обработки простого изображения с хранением его в двумерном контейнере вектор. |
| Решение задач с обработкой простых изображений. | | 69-71 | *Знать*. Правила объявления и инициализации двумерных векторов для хранения и обработки простых изображений, доступа к отдельным элементам, особенности применения обобщенных алгоритмов.<br>*Уметь*. Разработать и отладить программу обработки простого изображения с хранением его в двумерном контейнере вектор. |
| Контрольная работа 2 | | 72 | Применять полученные знания и умения при решении примеров и задач. |
| **XII. Двоичный поиск в массиве.** | **6** | | |
| Задача поиска информации; линейный и логарифмический (двоичный) поиск в массиве (векторе): алгоритмы и сравнение. | | 73-74 | *Знать.* Алгоритмы и параметры вычислительной сложности основных алгоритмов поиска в массиве (векторе), области применимости алгоритмов поиска.<br>*Уметь.* Разработать и отладить программу с применением двоичного поиска в контейнере вектор. |
| Обобщенные алгоритмы STL, связанные с двоичным поиском. Решение задач. | | 75-78 | *Знать*. Правила применения функций двоичного поиска `binary_search()`, `upper_bound()` и `lower_bound()` для векторов.<br>*Уметь*. Разработать и отладить программу с применением функций двоичного поиска в контейнере вектор. |
| **XIII. Двоичный поиск в по ответу.** | **6** | | |
| Задачи двоичного поиска по ответу. | | 79-80 | *Знать.* Способ двоичного поиска по ответу как средство уменьшения вычислительной сложности алгоритма решения задач; условия его применения.<br>*Уметь.* Разработать и отладить программу с применением двоичного поиска по ответу. |
| Решение задач методом двоичного поиска по ответу. | | 81-84 | *Знать*. Структуру алгоритма двоичного поиска по ответу.<br>*Уметь.* Разработать и отладить программу с применением двоичного поиска по ответу. |
| **XIV. Контейнер список.** | **6** | | |

| **Наименование раздела. Тема уроков** | **Кол. часов** | **№ урока** | **Характеристика основных видов деятельности ученика** |
|---|---|---|---|
| Контейнеры последовательностей: список. Инициализация, ввод и вывод. | | 85-86 | *Знать.* Правила объявления и инициализации списков, операции над ячейками списка и списками в целом. Функции-члены класса список: `push_back(), erase(), front(), insert()`.<br>*Уметь.* Разработать и отладить программу обработки последовательности чисел с хранением ее в контейнере список. |
| Итераторы входные, выходные и однонаправленные. Решение задач. | | 87-90 | *Знать*. Назначение и параметры функций-членов класса список: `splice()`.<br>*Уметь*. Разработать и отладить программу обработки последовательности чисел с хранением ее в контейнере список. |
| **XV. Контейнер стек.** | **6** | | |
| Контейнеры последовательностей: стек. Инициализация, ввод и вывод. | | 91-92 | *Знать.* Правила объявления и инициализации стеков. Функции-члены класса стек: `push(), pop(), top(), size()`.<br>*Уметь.* Разработать и отладить программу обработки последовательности чисел с хранением ее в контейнере стек. |
| Вычислительные задачи с обработкой данных в обратном порядке «последний пришел – первым обслужен». | | 93-96 | *Знать*. Модель памяти LIFO «последний пришел – первым обслужен», условия ее применения.<br>*Уметь.* Разработать и отладить программу обработки последовательности чисел с хранением ее в контейнере стек. |
| **XVI. Контейнеры очередь, дек.** | **6** | | |
| Контейнеры последовательностей: очередь и дек. Инициализация, ввод и вывод. | | 97-98 | *Знать.* Правила объявления и инициализации очередей (деков). Функции-члены класса очередь: `push(), pop(), front(), size()`.<br>*Уметь.* Разработать и отладить программу обработки последовательности чисел с хранением ее в контейнере очередь (дек). |
| Вычислительные задачи с обработкой данных в порядке поступления «первый пришел – первым обслужен» | | 99-101 | *Знать*. Модель памяти FIFO «первый пришел – первым обслужен», условия ее применения.<br>*Уметь.* Разработать и отладить программу обработки последовательности чисел с хранением ее в контейнере очередь (дек). |
| Контрольная работа 3 | | 102 | Применять полученные знания и умения при решении примеров и задач. |

## 1.2. КАЛЕНДАРНЫЙ УЧЕБНЫЙ ГРАФИК

Изучение курса проходит в течение 34 учебных недель по 3 учебных часа в неделю. Курс логически разбивается на 17 учебных блоков, каждый из которых включает 2 часа теоретических (лекционного типа), 2 часа практических и 2 часа лабораторных (в компьютерном классе) занятий.

## 2.3 СОДЕРЖАНИЕ

**Тема 1.** История языка C++. Обеспечение совместимости языков C и C++ на основе стандартов C++ ISO (C++98/03 и C++11), поддержка идеи переносимого языка программирования. Препроцессор C++, файл iostream; имена заголовочных файлов, пространства имен. Вывод в C++ с помощью cout. Ввод информации с использованием cin.
Целочисленные типы short, int, long и long long. Типы без знаков. Тип char: символы и короткие целые числа. Тип bool. Квалификатор const. Правила инициализации массивов.
Двоичная система счисления и представление целых чисел в памяти ЭВМ.
Альтернативы массивам - шаблонный класс array (C++11). Сравнение массивов и объектов array.
*Практическая часть:*
Изучение основных приемов работы учителя и учащегося с системой автоматической проверки решений – сайт moodle.cs.istu.ru.
Создание проекта в среде программирования Visual Studio, выполнение нескольких заданий в рамках одного проекта.

**Тема 2.** Строки с завершающим нулем. Представление в памяти, инициализация, функции для строк. Введение в класс string. Инициализация строк в C++11, ввод и вывод. Присваивание, сравнение и вероятные ошибки. Выражения отношений. Сравнение строк в стиле C и строк класса string.
Функции для строк (string): вставка, замена, удаление, поиск. Массивы строк.
*Практическая часть:*
Объявления строк, ввод и вывод. Работа с библиотекой функций для обработки строк. Пошаговая отладка программ с функциями в среде программирования Visual Studio.

**Тема 3.** Отличие STL от других библиотек. Последовательные контейнеры: вектор. Инициализация, ввод и вывод. Итераторы прямого доступа.
Обобщенные алгоритмы STL. Неизменяющие и изменяющие алгоритмы над последовательностями (на примере вектора). Алгоритмы, связанные с сортировкой, и обобщенные числовые алгоритмы.
*Практическая часть:*
Объявление векторов, ввод и вывод, операции над элементами и с векторами в целом. Работа с библиотекой обобщенных алгоритмов STL.

**Тема 4.** Реализация и сравнение сортировок: пузырьковая, поразрядным группированием и быстрсорт (quicksort).

*Практическая часть:*
Применение обобщенных алгоритмов сортировки из библиотеки STL.

**Тема 5.** Особенности применения контейнеров STL для хранения и обработки двумерных данных. Решение задач с обработкой двумерных числовых и символьных данных.
*Практическая часть:*
Объявление двумерных векторов, ввод и вывод, операции над элементами и с векторами в целом. Работа с библиотекой обобщенных алгоритмов STL.

**Тема 6.** Задача поиска информации; линейный и логарифмический (двоичный) поиск: алгоритмы и сравнение. Задачи двоичного поиска по ответу.
*Практическая часть:*
Применение обобщенных алгоритмов двоичного поиска для обработки массивов и векторов из библиотеки STL. Решение задач.

**Тема 7.** Контейнеры последовательностей: список. Инициализация, ввод и вывод. Итераторы входные, выходные и однонаправленные. Контейнеры последовательностей: стек и очередь. Инициализация, ввод и вывод. Вычислительные задачи с обработкой данных в порядке поступления «первый пришел – первым обслужен» и в обратном порядке «последний пришел – первым обслужен».
*Практическая часть:*
Объявление очереди и стека, ввод и вывод, операции над элементами и с очередями и стеками в целом.

**Итоговый контроль. Количество часов – 1 час.**

## 2. УСЛОВИЯ РЕАЛИЗАЦИИ ПРОГРАММЫ

### 3.1. Материально-технические условия и информационное обеспечение для реализации модулей программы

Учебная лекционная аудитория, компьютерный класс (15 – 20 рабочих мест). Компьютеры объединены в локальную сеть и имеют выход в Интернет, установлена операционная система Windows, Web-browser, MS Visual Studio, текстовый процессор Word.
Необходим высокоскоростной канал для подключения к сети Интернет.

### 3.2. Требования к кадровому обеспечению учебного процесса

При проведении лабораторно-практических занятий в компьютерном классе совместно с преподавателем работает инженер-программист, обеспечивающий работоспособность рабочих станций и программного обеспечения как на рабочих местах, так и на удаленном сервере.

**3.3. Требования к учебно-методическому обеспечению учебного процесса**

Необходимые теоретические сведения и наборы задач к темам курса готовятся к загрузке на электронные ресурсы moodle.cs.istu.ru и bacs.cs.istu.ru. Необходимы финансовые средства на программно-техническую и организационно-методическую поддержку этих ресурсов.

# 3. ОЦЕНКА КАЧЕСТВА ОСВОЕНИЯ ОБРАЗОВАТЕЛЬНОЙ ПРОГРАММЫ

Оценка качества освоения образовательной программы проводится по результатам собеседования по одному-двум вопросам из нижеследующего списка. Также могут быть предложены задачи для самостоятельного решения.

1. Препроцессор C++ и файл iostream. Имена заголовочных файлов.
2. Вывод в C++ с помощью cout. Манипулятор endl. Символ новой строки. Конкатенация с помощью cout.
3. Операторы объявления и переменные. Операторы присваивания.
4. Использование cin. Построчное чтение ввода.
5. Смешивание строкового и числового ввода.
6. cin и cout: признак класса.
7. Пространства имен. Местоположение директивы using в программах с множеством функций.
8. Циклы и текстовый ввод. Применение для ввода простого cin.
9. Выбор используемой версии cin. get (). Условие конца файла.
10. Целочисленные типы short, int, long и long long. Типы без знаков.
11. Выбор целочисленного типа. Целочисленные литералы. Определение компилятором C++ типа константы.
12. Тип char: символы и короткие целые числа.
13. Тип bool. Квалификатор const. Объявления auto в C++11.
14. Правила инициализации массивов.
15. Альтернативы массивам - шаблонный класс array (C++11).
16. Сравнение массивов и объектов array.
17. Строковый тип с нулевым символом-завершителем. Выделение подстроки. Нахождение первого вхождения буквы.
18. Функции для работы со строками.
19. Введение в класс string. Инициализация строк в C++11.
20. Выражения отношений. Формы строковых литералов.
21. Присваивание, сравнение и вероятные ошибки.
22. Сравнение строк в стиле C. Сравнение строк класса string.
23. STL – обобщенное программирование: связь контейнеров с итераторами.
24. Компоненты STL. Обобщенный алгоритм STL reverse со строкой и массивом.
25. Компоненты STL. Обобщенный алгоритм STL find с массивом и вектором.
26. Компоненты STL. Обобщенный алгоритм STL find со списком.
27. Компоненты STL. Обобщенный алгоритм STL merge.
28. Классификация итераторов STL.
29. Диапазоны итераторов. Входные и выходные итераторы.
30. Одно- и двунаправленные итераторы.
31. Итераторы с произвольным доступом.
32. Двоичный поиск заданного элемента в массиве.
33. Интегрированная среда разработки. Техпроцесс создания программы.

Оценки *«отлично»* заслуживает учащийся, обнаруживший всестороннее, систематическое и глубокое знание учебно-программного материала, умение свободно выполнять задания, предусмотренные программой, усвоивший основную и знакомый с дополнительной литературой, рекомендованной программой. Как правило, оценка «отлично»

выставляется учащимся, усвоившим взаимосвязь основных понятий дисциплины в их значении для приобретаемой профессии, проявившим творческие способности в понимании, изложении и использовании учебно-программного материала.

Оценки *«хорошо»* заслуживает учащийся, обнаруживший полное знание учебно-программного материала, успешно выполняющий предусмотренные в программе задания, усвоивший основную литературу, рекомендованную в программе. Как правило, оценка «хорошо» выставляется учащимся, показавшим систематический характер знаний по дисциплине и способным к их самостоятельному пополнению и обновлению в ходе дальнейшей учебной работы и профессиональной деятельности.

Оценки *«удовлетворительно»* заслуживает учащийся, обнаруживший знания основного учебно-программного материала в объеме, необходимом для дальнейшей учебы и предстоящей работы по специальности, справляющийся с выполнением заданий, предусмотренных программой, знакомый с основной литературой, рекомендованной программой. Как правило, оценка «удовлетворительно» выставляется учащимся, допустившим погрешности в ответе на экзамене и при выполнении экзаменационных заданий, но обладающим необходимыми знаниями для их устранения под руководством учителя.

Оценка *«неудовлетворительно»* выставляется учащемуся, обнаружившему пробелы в знаниях основного учебно-программного материала, допустившему принципиальные ошибки в выполнении предусмотренных программой заданий. Как правило, оценка «неудовлетворительно» ставится учащимся, которые не могут продолжить обучение или приступить к профессиональной деятельности по окончании вуза без дополнительных занятий по соответствующей дисциплине.

### 4.1. Примерные задачи для самостоятельного решения

1. Разные форматы вывода в языке C++: одно- и многострочный.
2. Напечатать заданное расположение наборов отрезков, перпендикулярных осям координат.
3. Напечатать изображение «игрового» поля с использованием от 1 до 5 символов.
4. Обработать числовую последовательность с формированием результатов в виде символьных кодов.
5. Обработать числовую последовательность с формированием результатов в виде искомых числовых наборов и текстовых пояснений.
6. Обработать числовую последовательность с преобразованием символьных последовательностей в числа и обратно.
7. Обработать числовую последовательность с преобразованием значений из одной системы счисления в другую.

## 5. КОНТРОЛЬНЫЕ ИЗМЕРИТЕЛЬНЫЕ МАТЕРИАЛЫ

Для каждого занятия подготовлен комплект задач в системе автоматической проверки решений – сайт moodle.cs.istu.ru. В системе организована регистрация участников, для каждого участника ведется учет его работы как в компьютерном классе, так и при выполнении самостоятельной работы дистанционно. Учителю доступны все решения учащихся: как ошибочные, так и прошедшие полную процедуру тестирования в автоматической системе.

## 6. УЧЕБНО-МЕТОДИЧЕСКОЕ ОБЕСПЕЧЕНИЕ ПРОГРАММЫ

**Перечень рекомендуемых учебных изданий, Интернет-ресурсов, дополнительной литературы.**

**Основная литература**


1. С. Прата. Язык программирования C++. Лекции и упражнения, 6-е изд. : Пер. с англ. - М. : ООО "И.Д. Вильямс", 2012. - 1248 с.
2. Д. Р. Мюссер, Ж. Дж. Дердж, А. Сейни. C++ и STL: справочное руководство, 2-е изд. (серия C++ in Depth).: Пер. с англ. - М.: 000 "И.Д. Вильямс", 2010. — 432 с.
3. Электронный ресурс moodle.cs.istu.ru.


**Дополнительная литература**


1. Б. Керниган, Д. Ритчи. Язык программирования Си.\Пер. с англ., 3-е изд., испр. - СПб.: "Невский Диалект", 2001.
2. В. Давыдов. Visual C++. - СПб.: Изд-во «БХВ», 2008.



***Разработчик программы:*** В.Г.Тарасов, профессор кафедры программного обеспечения ИжГТУ имени М.Т.Калашникова

**Согласовано:**

Зав. кафедрой «Программное обеспечение» И.О. Архипов

Заместитель проректора по учебной работе и
международной деятельности В.А. Цапок