# Программная инженерия

Московский Физико-Технический Институт

Автор курса: к.т.н. Иван Сергеевич Макаров

Контакты: Telegram или i.s.m.mipt@yandex.ru

# Оглавление

# 01 Общее введение и обзор технологий

2 или 3 лекции общей продолжительностью до 8 академических часов; 1 задача для самостоятельной работы.

## Программа

C++. Определение. Эволюция. Стандарты. Применения. Парадигмы программирования. Декларативное, функциональное, императивное, процедурное, структурное, объектно-ориентированное, обобщенное, параллельное и событийно-ориентированное программирование. Ядро языка. Стандартная библиотека шаблонов. STL. Boost. Дополнительные библиотеки. Инструменты разработчика. Интегрированные среды разработки. Редакторы кода. Компиляторы. Отладчики. Профилировщики. Системы автоматизации сборки. Системы контроля версий. Системы управления проектами. Важные персоны. Внешние ресурсы. Операционные системы. Windows. Visual Studio. MSVC. Решения и проекты. Файлы исходного кода. Конфигурации. Debug и release. x86 и x64. Сборка. Ошибки и предупреждения. Linux. Ubuntu. Терминал. GCC. Visual Studio Code. Расширения. Свойства. Стили кодирования. Форматирование. Именование. `snake_case`, `camelCase`. Венгерская нотация. Стили кодирования STL, Boost и Google. Комментарии. Документирование. Doxygen. Hello, world! Функция main. Консоль. Git. GitHub. SmartGit. Репозитории. Комманды. Clone, commit, push, pull. Откаты изменений. Discard, revert, reset. Ветки. Мастер. Merge и rebase. Логи. Конфликты. Индекс. Continuous integration. CI. Continuous deployment. CD.

## Материалы

- `01.01.introduction`
- `01.02.introduction.function.main`
- `01.03.introduction.standard`
- `01.04.project.tool.git`

## Литература

- B. Stroustrup, The C++ Programming Language, 4th Edition, разделы 1.2, 1.4, 44
- B. Stroustrup, A Tour of C++, 2nd Edition, разделы 1.2, 8, 16
- B. Stroustrup, The Design and Evolution of C++, разделы 1-9
- N. Josuttis, The C++ Standard Library, 2nd Edition, раздел 2.1
- S. Meyers, Effective C++, 3rd Edition, разделы 1, 53-55
- S. Meyers, More Effective C++, раздел 35
- S. Dewhurst, C++ Gotchas, разделы 1, 11, 12
- H. Sutter, C++ Coding Standards, разделы 0-4
- R. Grimm, C++20 Get the Details, разделы 1, 2

# 02 Структурное программирование

2 или 3 лекции общей продолжительностью до 8 академических часов; 13 задач для самостоятельной работы.

## Программа

Объявления и определения. Правило одного определения. ODR. Объекты. Переменные. Типы. Фундаментальные типы. Арифметические типы. Интегральные типы. `bool`, `char`, `short`, `int`, `long`, `float`, `double`. `long long`, `long double`. Размеры в байтах. `sizeof`. Двоичные форматы представления чисел. Проблемы переполнения. Одинарная и двойная точность. Дополнительный код. `signed`, `unsigned`. Проблемы переносимости. Литералы. Логические, символьные, целочисленные литералы и литералы с плавающей точкой. `true`, `false`. Суффиксы литералов. Значения. Инициализация. Неопределенное поведение. Инициализация по умолчанию, значением, копированием, списком. Унифицированная инициализация. Вывод типов. `auto`. Преобразования. Временные объекты. Явные и неявные преобразования. Сужающие и расширяющие преобразования. Целочисленные преобразования. Преобразования в старом стиле. `static_cast`. Магические числа. Константы. `const`. Особая память. `volatile`. Псевдонимы типов. `using`, `typedef`. `std::size_t`. Атрибуты. `[[maybe_unused]]`.

Операторы. Арность. Нульарные, унарные, бинарные и тернарные операторы. Приоритеты операторов. Круглые скобки. Логические операторы. Альтернативные представления. `or`, `and`, `not`. Вычисления по короткой схеме. Арифметические операторы. Операторы сравнения. Префиксные и постфиксные операторы инкремента и декремента. Порядок вычисления операндов. Побочные эффекты. Неспецифицированное поведение. Цепочки операторов. Ассоциативность. Оператор присваивания. Оператор запятая. Выражения. Поток управления. Условия. Ветвления. `if`, `else`. Тернарный оператор. `switch`, `case`, `default`. `[[fallthrough]]`. Оптимизации. `[[likely]]`, `[[unlikely]]`. Повторения. `for`, `while`, `do`. Прыжки. `continue`, `break`, `goto`. Метки. Инструкции.

Адресация памяти. Указатели. Операторы взятия адреса объектов и разыменования указателей. Нулевые указатели. `nullptr`. `std::nullptr_t`. Константные указатели. Указатели на константы. Массивы. Статические массивы. Агрегатная инициализация. Индексация. Размеры массивов. `std::size`. Арифметика указателей. Связь указателей и массивов. `std::swap`. Динамические массивы. `new`, `new[]`, `delete`, `delete[]`. Контейнеры. `std::vector`. Цикл for на основе диапазонов. Lvalue ссылки. Константные ссылки. `std::reference_wrapper`.

Функции. Области видимости функций. Объявления и определения. Forward объявления. Тела функций. Пред- и постусловия. Параметры и аргументы. Порядок вычисления аргументов. Аргументы по умолчанию. Передача аргументов по значению, указателю и ссылке. Последовательности и представления. `std::span`. Статические и динамические размеры. Возврат результатов. `return`. `[[nodiscard]]`. `void`. Завершающие типы возврата. `auto`. Встраивание функций. `inline`. Рекурсия. Возврат висячих указателей и ссылок. Локальные переменные. Статические переменные. `static`. Перегрузка функций. Сигнатуры функций. Разрешение перегрузки. Бинарный поиск. `std::midpoint`. Лексикографический порядок. Комбинированная сортировка слиянием и вставками.

## Материалы

- `02.01.statement.expression.variable.type`
- `02.02.statement.expression.operator`
- `02.03.statement.selection`
- `02.04.statement.iteration.jump`
- `02.05.pointer`
- `02.06.pointer.array`
- `02.07.pointer.array.new.delete`
- `02.08.container.vector`
- `02.09.reference.lvalue`
- `02.10.function`
- `02.11.algorithm.binary.search`
- `02.12.algorithm.lexicographic.order`
- `02.13.algorithm.insertion.merge.sorting`

## Литература

- B. Stroustrup, The C++ Programming Language, 4th Edition, разделы 6, 7, 9, 10.3-10.5, 11.1, 11.2, 11.5, 12.1-12.4
- B. Stroustrup, A Tour of C++, 2nd Edition, разделы 1, 3.6, 13.3
- N. Josuttis, The C++ Standard Library, 2nd Edition, разделы 2.2, 5.5
- S. Meyers, Effective C++, 3rd Edition, разделы 16, 20, 21, 26, 27, 30
- S. Meyers, More Effective C++, разделы 1, 2, 16, 23
- S. Meyers, Effective Modern C++, разделы 2.1, 3.1-3.3, 8.1
- S. Dewhurst, C++ Gotchas, разделы 2, 4-7, 9, 13-18, 20, 21, 31, 32, 40, 41, 44, 48, 60, 66
- H. Sutter, C++ Coding Standards, разделы 5-9, 15, 17-20, 25, 31, 77, 91, 95, 99
- R. Grimm, C++20 Get the Details, разделы 4.8, 5.2, 5.4
- M. Bancila, The Modern C++ Challenge, раздел 1

# 03 Объектно-ориентированное программирование

4 или 5 лекций общей продолжительностью до 16 академических часов; 6 задач для самостоятельной работы.

## Программа

Объектно-ориентированное программирование. Пользовательские типы данных. Структуры. `struct`. Данные-члены. Обычные старые данные. POD. Экземпляры. Инициализация по умолчанию. Агрегатные типы данных. Агрегатная инициализация. Назначенная инициализация. Массивы структур. Доступ к данным-членам. Операторы доступа точка и стрелочка. Константные структуры. Передача структур по ссылке. Возврат структур.

Классы. `class`. Области видимости классов. Инкапсуляция. Спецификаторы доступа. `public`, `private`. Интерфейсы и реализации. Функции-члены. Инварианты. Внутренние и внешние определения функций-членов. Константные и неконстантные функции-члены. `const`. Побитовая и логическая константность. `mutable`. Кэширование данных. Геттеры и сеттеры. Конструкторы. Конструкторы по умолчанию. Неявные конструкторы по умолчанию. Пользовательские конструкторы. Делегирующие конструкторы. Списки инициализации. Порядок инициализации данных-членов. Деструкторы. Вложенные псевдонимы и типы. Статические члены. `static`. Друзья классов. `friend`. Ограничение доступа к приватным членам. Паттерн Attorney-Client. Идиома PassKey.

Типы отношений. Композиция, агрегация, ассоциация и зависимость. Наследование. Иерархии классов. Базовые и производные классы. Порядок работы конструкторов и деструкторов. Защищенные члены. `protected`. Открытое, защищенное и закрытое наследование. Наследование интерфейсов. Является разновидностью. Наследование реализаций. Реализован посредством. Вызов унаследованных функций. Изменение спецификаторов доступа. Множественное наследование. Решение проблемы бриллианта. Виртуальное наследование. Полиморфизм. Устройство луковицы. Срезка объектов. Повышение. Виртуальные функции. `virtual`. Дополнительные спецификаторы. Проверка сигнатур. `override`. Запрет наследования или переопределения. `final`. Виртуальные деструкторы. Раннее и позднее связывание. Виртуальные указатели. Виртуальные таблицы. Ковариантные типы возврата. Абстракция. Чисто виртуальные функции. Абстрактные базовые классы. Интерфейсные классы. Пустые классы. Использование общих адресов. `[[no_unique_address]]`. Особенности компилятора MSVC. Сжатые пары. Оптимизации пустых базовых классов. EBO. Определение типов на этапе выполнения. RTTI. Понижение. `dynamic_cast`. Проверки на этапе выполнения. Анализ типов. `decltype`, `decltype(auto)`. Строковые названия. `typeid`. Проблемы ссылок. Boost.TypeIndex. `std::any`, `std::make_any`, `std::any_cast`.

Копирование и семантика перемещения. Эквивалентность и независимость копий и оригиналов. Категории выражений. История. Lvalue. Rvalue. Возможность взятия адреса. Идентифицируемость. Перемещаемость. Rvalue и временные объекты. Prvalue. Glvalue. Xvalue. Rvalue ссылки. Разрешение перегрузки. Ссылочные квалификаторы. Специальные функции-члены. Копирующие и перемещающие конструкторы и операторы присваивания. Глубокое и поверхностное копирование. Самоприсваивание. Скрытые указатели на экземпляры. `this`. Цепочки вызова функций-членов. Функции обмена. Копирование с обменом. Идиома CaS. Обеспечение свойства перемещаемости. `std::move`. Состояния объектов после перемещения. Правила генерации компиляторами специальных функций-членов. Правила нуля, трех, четырех и пяти. Реализации по умолчанию. Оптимизации кода компиляторами. Устранение избыточного копирования. Оптимизации возвращаемых значений. RVO и NRVO.

Перегрузка операторов. Явные и неявные преобразования. Явные и неявные конструкторы и операторы преобразования типов. `explicit`. Перегрузка операторов в качестве функций-членов, свободных функций и друзей класса. Перегрузка операторов ввода и вывода. Перегрузка префиксных и постфиксных операторов инкремента и декремента. Перегрузка операторов сравнения. Инфиксные и функциональные форматы вызова операторов. Дробная арифметика. Boost.Rational. Виртуальные операторы вывода в иерархиях классов. Трехстороннее сравнение. Оператор космический корабль. Переписывание выражений. Сильное, слабое и частичное упорядочение. `std::strong_ordering`, `std::weak_ordering`, `std::partial_ordering`. Лексикографические сравнения. Оптимизации операторов. Дублирование парных операторов. Устранение константности. `const_cast`.

## Материалы

- `03.01.class.struct`
- `03.02.class`
- `03.03.class.friend`
- `03.04.pattern.attorney.client.passkey`
- `03.05.class.association`
- `03.06.class.inheritance`
- `03.07.class.inheritance.multiple`
- `03.08.class.inheritance.multiple.diamond`
- `03.09.class.polymorphism`
- `03.10.class.polymorphism.vptr.vtbl`
- `03.11.class.polymorphism.type.covariant`
- `03.12.class.polymorphism.type.identification`
- `03.13.class.optimization`
- `03.14.expression.classification`
- `03.15.reference.lvalue.rvalue`
- `03.16.reference.lvalue.rvalue.function.overload`
- `03.17.implementation.container.vector`
- `03.18.class.operator.overload`
- `03.19.class.operator.overload.output`
- `03.20.class.operator.overload.spaceship`
- `03.21.class.operator.overload.optimization`

## Литература

- B. Stroustrup, The C++ Programming Language, 4th Edition, разделы 6.4, 7.7, 8.2, 13.6, 16-22
- B. Stroustrup, A Tour of C++, 2nd Edition, разделы 2.2, 2.3, 4, 5
- D. Vandevoorde, C++ Templates, 2nd Edition, разделы 21.1, B, C
- S. Meyers, Effective C++, 3rd Edition, разделы 3-5, 7, 9-12, 18, 19, 21-25, 28, 32-40
- S. Meyers, More Effective C++, разделы 2-7, 17-22, 24, 31, 33
- S. Meyers, Effective Modern C++, разделы 1.3, 1.4, 3.6, 3.11, 5
- S. Dewhurst, C++ Gotchas, разделы 10, 19, 22, 23, 30, 33, 35-39, 42, 45-47, 49-54, 56-59, 69-71, 73-87, 89-99
- H. Sutter, C++ Coding Standards, разделы 11, 26-30, 32-38, 40-42, 44, 47-50, 52, 54-56, 90, 93, 94, 100
- R. Grimm, C++20 Get the Details, разделы 4.4, 4.8
- M. Bancila, The Modern C++ Challenge, разделы 2.15, 2.16

# 04 Обобщенное программирование

4 или 5 лекций общей продолжительностью до 16 академических часов; 11 задач для самостоятельной работы.

## Программа

Обощенное программирование. Шаблоны. `template`. Двухэтапная трансляция шаблонов. Модель включения. Параметры и аргументы шаблонов. Шаблоны функций. Типовые параметры шаблонов. `typename`. Аргументы шаблонов по умолчанию. Сокращенные шаблоны функций. Специализации шаблонов функций. Полные специализации. Перегрузка шаблонов функций. Нетиповые параметры шаблонов. Шаблоны для встроенных статических массивов. Вариативные шаблоны. Пакеты параметров вариативных шаблонов. Многоточие. Рекурсивное инстанцирование шаблонов. Распаковка пакетов аргументов вариативных шаблонов. Выражения свертки. Вариативный обход дерева на выражении свертки. Указатели на члены классов. Оператор двоеточие-звездочка. Оператор стрелочка-звездочка. Вариативные выражения. Шаблоны классов. Правила инстанцирования функций-членов шаблонов классов. Шаблонизированные сущности. Шаблонные параметры шаблонов. Вложенные шаблоны. Вывод аргументов шаблонов классов. Специализации шаблонов классов. Полные и частичные специализации. Друзья шаблонов классов. Шаблоны поддержки. Шаблоны псевдонимов и переменных.

Метапрограммирование шаблонов. Полнота по Тьюрингу. Генеративное программирование. Оптимизации исходного кода. Правило как есть. Вычисления на этапе компиляции. Константные выражения. Моментальные функции. `constexpr`, `consteval`, `constinit`. Ветвления на этапе компиляции. `if constexpr`. Гибридное метапрограммирование. Дробная арифметика на этапе компиляции. Определение единиц измерения результата.

Фундаментальные свойства. Модифицируемость, константность и перемещаемость. Lvalue и rvalue ссылки. Семантика перемещения в шаблонах. Идеальная передача. Пробрасывающие ссылки. `std::forward`. Обобщенные вызовы. `std::invoke`. Шаблоны специальных функций-членов классов. Ошибки подстановки, не являющиеся ошибками компиляции. Идиома SFINAE. `std::enable_if`. Передача аргументов по значению, lvalue, rvalue и пробрасывающим ссылкам. Вывод типов. Низводящие преобразования. Правила свертывания ссылок.

Шаблоны свойств. Детали реализаций. Шаблоны классов. Полные и частичные специализации. Стандартные базовые классы для констант. `std::integral_constant`, `std::false_type`, `std::true_type`. Статические константы и псевдонимы. Шаблоны псевдонимов и переменных. Свойства типов. Отношения типов. Преобразования типов. Условные шаблоны свойств. Невычисляемые контексты. `std::declval`, `decltype`. Проверка утверждений на этапе компиляции. `static_assert`. Политики. Концепты. `concept`. Ограничения. Ограничения типов аргументов шаблонов. Ограничения типов заполнителей. Предикаты на этапе компиляции. Требования. `requires`. Использование требований в объявлениях. Именованные требования. Стандартные концепты.

Вариативные гетерогенные списки типов. Вариативные гетерогенные коллекции. Пары. `std::pair`. Обобщение пар. Кортежи. `std::tuple`. Создание экземпляров кортежей. `std::make_tuple`. Классы-помощники для кортежей. `std::tuple_size`, `std::tuple_element`. Получение доступа к элементам кортежей на основе индексов и типов. `std::get`. Распаковка кортежей. `std::tie`. Заполнители. `std::ignore`. Игнорирование возвращаемых результатов nodiscard функций. Конкатенация кортежей. `std::tuple_cat`. Структурированные привязки.

## Материалы

- `04.01.template.function`
- `04.02.template.function.variadic`
- `04.03.template.function.variadic.tree.traversal`
- `04.04.template.class`
- `04.05.template.class.friend`
- `04.06.template.alias.variable`
- `04.07.template.metaprogramming`
- `04.08.template.metaprogramming.constexpr`
- `04.09.template.metaprogramming.constexpr.hybrid`
- `04.10.reference.lvalue.rvalue.forwarding`
- `04.11.pattern.sfinae`
- `04.12.template.type.inference`
- `04.13.implementation.template.type.trait`
- `04.14.implementation.template.type.concept`
- `04.15.implementation.utility.tlist`
- `04.16.implementation.utility.tuple`
- `04.17.utility.tuple`

## Литература

- B. Stroustrup, The C++ Programming Language, 4th Edition, разделы 10.4, 24.2-24.4, 25, 26, 28.2-28.7, 35.3-35.5
- B. Stroustrup, A Tour of C++, 2nd Edition, разделы 6, 7.2-7.5, 13.4, 13.9
- N. Josuttis, The C++ Standard Library, 2nd Edition, разделы 5.1, 5.4, 5.6
- D. Vandevoorde, C++ Templates, 2nd Edition, разделы 1-17, 19, 20, 23-25, 27, 28, D, E
- S. Meyers, Effective C++, 3rd Edition, разделы 41-46, 48
- S. Meyers, Effective Modern C++, разделы 1.1, 1.2, 3.3, 3.9, 5
- S. Dewhurst, C++ Gotchas, раздел 88
- H. Sutter, C++ Coding Standards, разделы 65-67
- R. Grimm, C++20 Get the Details, раздел 4.1, 4.5, 4.6
- M. Bancila, The Modern C++ Challenge, разделы 2.18, 2.19

# 05 Паттерны и технологии проектирования

2 или 3 лекции общей продолжительностью до 8 академических часов; 19 задач для самостоятельной работы.

## Программа

Объектно-ориентированное программирование. Принципы SOLID. Принцип единственности ответственности. Принцип открытости-закрытости. Принцип подстановки Барбары Лисков. Принцип разделения интерфейсов. Принцип инверсии зависимостей. Паттерны проектирования банды четырех. Порождающие паттерны. Builder. Factory. Factory method. Abstract factory. Prototype. Концепция виртуальных конструкторов для клонирования. Антипаттерны. Singleton. Умолчательные и удаленные реализации специальных функций-членов классов. `default`, `delete`. Monostate. Noncopyable. Структурные паттерны. Adapter. Bridge. Composite. Decorator. Поведенческие паттерны. Програмимрование на основе интерфейсов. Memento. Observer. State. Конечные автоматы. Циклические зависимости. Forward объявления классов. Функции undo и redo. Strategy. Template method. Невиртуальные интерфейсы. Идиома NVI. Проблемы хрупкости базовых классов. Дополнительные паттерны.

Обобщенное программирование. Стандартная библиотека шаблонов. STL. Динамический полиморфизм. Виртуальные функции. Статический полиморфизм. Шаблоны функций. Преимущества и недостатки. Странно рекурсивный шаблон проектирования. CRTP. Перевернутое наследование. Устранение виртуальности. Расширение функциональности классов. Подсчет экземпляров классов. Закрытое наследование и друзья классов. Трюк Бартона-Нэкмана. Boost.Operators. Миксины. Вариативные базовые классы. Шаблонные параметры шаблонов.

## Материалы

- `05.01.pattern.builder`
- `05.02.pattern.factory`
- `05.03.pattern.prototype`
- `05.04.pattern.singleton.monostate.noncopyable`
- `05.05.pattern.adapter`
- `05.06.pattern.bridge`
- `05.07.pattern.composite`
- `05.08.pattern.decorator`
- `05.09.pattern.memento`
- `05.10.pattern.observer`
- `05.11.pattern.state`
- `05.12.pattern.strategy`
- `05.13.pattern.template.method`
- `05.14.template.polymorphism`
- `05.15.pattern.crtp`
- `05.16.pattern.crtp.observer`
- `05.17.pattern.crtp.operator.overload`
- `05.18.pattern.crtp.mixin`
- `05.19.pattern.crtp.mixin.constructor`
- `05.20.pattern.crtp.mixin.inheritance.variadic`

## Литература

- B. Stroustrup, The C++ Programming Language, 4th Edition, раздел 27
- D. Vandevoorde, C++ Templates, 2nd Edition, разделы 18, 21, 5.8
- S. Meyers, Effective C++, 3rd Edition, разделы 6, 35
- S. Meyers, More Effective C++, разделы 25, 26, 32
- S. Meyers, Effective Modern C++, разделы 3.5, 4.5
- S. Dewhurst, C++ Gotchas, разделы 72, 76, 90, 92
- H. Sutter, C++ Coding Standards, разделы 39, 50, 53, 64
- R. Grimm, C++20 Get the Details, раздел 4.3
- M. Bancila, The Modern C++ Challenge, раздел 8
- E. Gamma, Design Patterns, разделы 1, 3-6
- B. Schaeling, The Boost C++ Libraries, 2nd Edition, раздел 72

# 06 Организация проектов и библиотек

2 или 3 лекции общей продолжительностью до 8 академических часов; 2 задачи для самостоятельной работы.

## Программа

Многофайловые программы. Файлы исходного кода. Заголовочные файлы. Объявления и определения. ODR. Видимость идентификаторов. Этапы сборки. 9 фаз трансляции. Препроцессинг. Препроцессоры. Генерация кода. Директивы препроцессоров. `include`. Директории поиска. Транзитивные включения. Макроопределения. Объектные и функциональные макросы. `define`, `undef`. Многострочные макросы. Проблемы макросов. Побочные эффекты. Предопределенные макросы. `DEBUG`, `NDEBUG`. Стандартные предопределенные макросы и идентификаторы. `FILE`, `LINE`, `DATE`, `TIME`. `func`. `std::source_location`. Условная компиляция. `if`, `defined`, `else`, `endif`. Защиты от повторного включения. `pragma once`. Отключение предупреждений. `pragma warning`. Триграфы. Юниты трансляции. Компиляция. Компиляторы. Частичная перекомпиляция. Уменьшение зависимостей на этапе компиляции. Указатели на реализации. Идиома PImpl. Накладные расходы. Предкомпилированные заголовки. Объектные файлы. Файлы библиотек. Компоновка. Компоновщики. Внешнее и внутреннее связывание. Ошибки компоновщиков. Многократно определенные внешние символы. Неразрешенные внешние символы. Глобальные переменные и константы. `extern`. Анонимные пространства имен. Определения классов и шаблонов в заголовочных файлах. Исключения из ODR. Пространства имен. `namespace`. Конфликты имен. Области видимости пространств имен. Глобальное пространство имен. Оператор разрешения области видимости. Квалифицированные имена. Поиск, зависящий от аргументов. Поиск Кёнига. Аддитивность пространств имен. Вложенные пространства имен. Псевдонимы пространств имен. Встроенные пространства имен. Поддержка версий. Встроенные переменные. `inline`. Стандартное пространство имен. Using объявления. `using`.

Системы автоматизации сборки. CMake. CMakeLists. Проекты. Требования к стандартам языков. Переменные. Списки. Циклы. Пакеты. Директории включаемых файлов и библиотек. Целевые объекты. Исполняемые файлы. Библиотеки. Свойства. Флаги компиляторов и компоновщиков. Определения препроцессоров. Подготовка сборки. Shell скрипты. Bash. Пакетные менеджеры. Модули. Объявления модулей. `module`. Фрагменты модулей. Глобальные фрагменты модулей. Приватные фрагменты модулей. Интерфейсы и реализации модулей. Экспорт объявлений и определений. Одиночный и групповой экспорт. `export`. Импортирование модулей. `import`. Классы и шаблоны в модулях. Подмодули и разделы. Модули стандартной библиотеки шаблонов. STD.

Boost. Сборка и использование. Статические библиотеки. Интерфейсы в заголовочных файлах. Реализации в библиотечных файлах. Динамические библиотеки. Явное и неявное связывание библиотек. Экспорт и импорт символов и псевдонимов. `extern "C"`. Boost.DLL. Обновление динамических библиотек на этапе выполнения.

## Материалы

- 06.01.project.translation
- 06.02.project.header
- 06.03.project.source
- 06.04.project.source.main
- 06.05.pattern.pimpl
- 06.06.project.header.precompiled
- 06.07.project.source.precompiled
- 06.08.project.tool.cmake
- 06.09.project.module
- 06.10.project.module.source
- 06.11.project.module.submodule
- 06.12.project.library
- 06.13.project.library.static
- 06.14.project.library.shared

## Литература


- B. Stroustrup, The C++ Programming Language, 4th Edition, разделы 12.6, 14, 15
- B. Stroustrup, A Tour of C++, 2nd Edition, разделы 3.2-3.4
- N. Josuttis, The C++ Standard Library, 2nd Edition, разделы 4.1, 4.2
- D. Vandevoorde, C++ Templates, 2nd Edition, разделы 9.3, 13.2, A
- S. Meyers, Effective C++, 3rd Edition, разделы 2, 4, 31
- S. Dewhurst, C++ Gotchas, разделы 3, 8, 25-28, 55
- H. Sutter, C++ Coding Standards, разделы 10, 16, 21-24, 43, 57-59, 61, 63
- R. Grimm, C++20 Get the Details, раздел 4.2

# 07 Обработка ошибок и исключений

2 или 3 лекции общей продолжительностью до 8 академических часов; 6 задач для самостоятельной работы.

## Программа

Обработка ошибок. Инварианты. Пред- и постусловия. Проверка утверждений на этапе выполнения. `assert`. Проверка утверждений на этапе компиляции. `static_assert`. Ошибки на этапе выполнения. Повторения. Нормальные завершения работы. `std::exit`, `std::atexit`. Ненормальные завершения работы. `std::abort`. Коды возврата. Глобальные переменные. Дополнительные аргументы. Неудобства кодов возврата. Объединнения. `union`. Анонимные объединения. Варианты. `std::variant`. Опциональные значения. `std::optional`.

Исключения. Стандартная иерархия исключений. `std::exception`. `std::system_error`. Совместимость с кодами возврата. `std::error_code`. Пользовательские исключения. Наследование и виртуальные функции. Выброс исключений. `throw`. Повторный выброс исключений. `[[noreturn]]`. Ловушки для исключений. `try`. Ловушки на уровне функций. Обработчики исключений. `catch`. Обработчики всех исключений. Указатели для исключений. `std::exception_ptr`. Текущее исключение. `std::current_exception`. Порядок расположения обработчиков исключений. Небуферизованный поток символьного вывода для сообщений об ошибках. `std::cerr`. Особенности обработки исключений в конструкторах и деструкторах. Отсутствие исключений. `noexcept`. Невычисляемые контексты. `std::declval`. Пробрасывание исключений. Эффективность исключений. Преобразование исключений. Сворачивание стека. Принцип нулевых накладных расходов. Необработанные исключения. `std::terminate`. Гарантии безопасности исключений. Отстутствие гарантий. Базовая гарантия. Строгая гарантия. Гарантия отсутствия исключений. Вариант безопасного относительно исключений интерфейса стека.

Отладка. Обнаружение и воспроизведение проблем. Стратегии отладки. Комментирование. Вывод. Логгирование. Трасировка. Boost.Log. Модульное тестирование. Разработка модульных тестов. Boost.Test. Варианты использования. Деревья тестов. Мастер-тесты. Группы тестов. Отдельные тесты. Тесты на основе данных. Наборы данных и выборки. Операции с наборами данных. Шаблонные тесты. Boost.MPL. Параметризованные тесты. Адаптеры тестов. Состояния программ. Интегрированные отладчики. Режим debug. Пошаговое выполнение. Step into, over и out. Run to cursor. Start и continue. Точки останова. Отслеживание значений переменных. Стек вызовов. Профилирование. Интегрированные профилировщики. Режим release. Загрузка процессора. Использование памяти. Утечки памяти. Снимки кучи. Рефакторинг и оптимизации кода. Инструменты Google. Замеры времени. Google.Benchmark. Форматы вывода. Статистическая стабильность результатов. Передача аргументов. Аппроксимация алгоритмической сложности. Предотвращение оптимизаций компиляторов. Обработка ошибок. Полезные макросы. Разработка модульных тестов. Google.Test. Отдельные тесты. Утверждения. Ожидания. Адаптеры тестов. Параметризованные тесты. Запуск тестов. Фреймворк для фиктивных объектов.

## Материалы

- `07.01.error.assertion.termination`
- `07.02.class.union`
- `07.03.utility.variant.optional`
- `07.04.error.exception`
- `07.05.error.exception.stack.interface`
- `07.06.project.tool.boost.logging`
- `07.07.project.tool.boost.testing`
- `07.08.project.tool.profiler.cpu`
- `07.09.project.tool.profiler.memory`
- `07.10.project.tool.google.benchmark`
- `07.11.project.tool.google.testing`

## Литература

- B. Stroustrup, The C++ Programming Language, 4th Edition, разделы 8.3, 13.1, 13.2, 13.4, 13.5, 30.4
- B. Stroustrup, A Tour of C++, 2nd Edition, разделы 2.4, 3.5, 13.5
- N. Josuttis, The C++ Standard Library, 2nd Edition, раздел 4.3
- D. Vandevoorde, C++ Templates, 2nd Edition, раздел 26
- S. Meyers, Effective C++, 3rd Edition, разделы 8, 29
- S. Meyers, More Effective C++, разделы 9-15
- S. Meyers, Effective Modern C++, раздел 3.8
- S. Dewhurst, C++ Gotchas, разделы 64, 65
- H. Sutter, C++ Coding Standards, разделы 14, 51, 62, 68-75, 97
- B. Schaeling, The Boost C++ Libraries, 2nd Edition, раздел 62

# 08 Особенности математических вычислений

2 или 3 лекции общей продолжительностью до 8 академических часов; 6 задач для самостоятельной работы.

## Программа

Манипуляторы. `std::showbase`. `std::oct`, `std::dec`, `std::hex`, `std::boolalpha`. Двоичные, восьмеричные, десятичные и шестнадцатиричные литералы. Побитовые операторы. Арифметика битового сдвига. Целочисленные типы фиксированного размера. `std::int#_t`, `std::uint#_t`. Оперирование битами и байтами. Хранение битов и байтов. `std::bitset`. `std::byte`. `std::to_integer`. `std::as_bytes`. Endianess. Little и big endian. `std::endian`. Каламбуры типов. Реинтерпретирование битов. `std::bit_cast`. `reinterpret_cast`. Вопросы безопасности. Дополнительные функции. `std::bit_ceil`, `std::bit_floor`. Битовые поля. Двоичный код Грея. Кодирование и декодирование. Индийский алгоритм возведения в степень. Отслеживание переполнений. Перечисления. `enum`. Перечисления с областью видимости. Явные преобразования. Нижележащие типы. Перечисления без области видимости. Флаги состояний. Числа с плавающей точкой. Научная нотация. Точность. `std::scientific`, `std::defaultfloat`, `std::fixed`. `std::setprecision`. Форматирование вывода. `std::setw`, `std::right`, `std::left`, `std::setfill`. Максимумы, минимумы, бесконечность, NaN и отрицательный ноль. Стандарт IEEE-754. `std::numeric_limits`, `errno`. Сравнение числе с плавающей точкой. Константы эпсилон. Реализации и оптимизации Дональда Кнута. Расширенная точность чисел с плавающей точкой. Boost.Multiprecision. Сложные математические функции. Boost.Math. Комплексные числа. `std::complex`.

Генерация псевдослучайных чисел. Последовательности псевдослучайных чисел. Начальные значения. Источники энтропии. Недетерминированные генераторы. `std::random_device`. Генераторы случайных значений. Период, вероятность и производительность. 32-битные и 64-битные вихри Мерсенна Мацумото и Нишимуры. `std::mt19937`. Распределения случайных чисел. Равномерное, Бернулли, Пуассона, нормальное и выборочное распределение. `std::normal_distribution`. Генерация случайных чисел C. `std::srand`, `std::rand`. Методы Монте-Карло. Вычисление значения числа Пи. `std::uniform_real_distribution`. Управление временем. Библиотека chrono. `std::chrono`. Часы. Эпохи. Эпоха Unix. Такты. Системные часы. Монотонные часы. Друиге часы. Моменты времени. Интервалы времени. Замеры времени. Хронометр. Библиотека datetime. Календарные даты. Преобразования в моменты и интервалы времени. Арифметика календаря. Дни недели. Часовые пояса. Стандартные литералы. `std::literals`. Пользовательские литералы. Реализация операторов для литералов.

## Материалы

- `08.01.number.integer`
- `08.02.algorithm.binary.gray.code`
- `08.03.algorithm.indian.exponentiation`
- `08.04.class.enumeration`
- `08.05.number.floating.point`
- `08.06.number.floating.point.multiprecision`
- `08.07.number.complex`
- `08.08.number.random.generation`
- `08.09.algorithm.monte.carlo`
- `08.10.algorithm.monte.carlo.simplex`
- `08.11.time.chrono`
- `08.12.time.chrono.calendar`
- `08.13.implementation.operator.literal`

## Литература

- B. Stroustrup, The C++ Programming Language, 4th Edition, разделы 6.2, 8.4, 35.2, 40.2-40.5, 40.7
- B. Stroustrup, A Tour of C++, 2nd Edition, разделы 2.5, 13.4, 13.7, 14
- N. Josuttis, The C++ Standard Library, 2nd Edition, разделы 5.3, 5.5, 5.7, 12.5, 17, S.1, S.2
- S. Meyers, More Effective C++, раздел 2
- S. Meyers, Effective Modern C++, раздел 3.4
- H. Sutter, C++ Coding Standards, раздел 92
- R. Grimm, C++20 Get the Details, разделы 5.4, 5.5
- M. Bancila, The Modern C++ Challenge, разделы 1.10, 1.13, 2.17, 2.22, 5

# 09 Низкоуровневое управление памятью

3 или 4 лекции общей продолжительностью до 12 академических часов; 16 задач для самостоятельной работы.

## Программа

Управление памятью. Сборщики мусора. Захват ресурса есть инициализация. Идиома RAII. Интеллектальные указатели. Совместное владение. `std::shared_ptr`. Подсчет ссылок. Контрольные блоки. `std::make_shared`. Работа со встроенными статическими массивами. Политики освобождения. `std::default_delete`. Циклические зависимости. `std::weak_ptr`. Сильные и слабые ссылки. Эксклюзивное владение. `std::unique_ptr`. `std::make_unique`. Производители и потребители данных. `std::enable_shared_from_this`. Итераторы. Категории итераторов. Однонаправленные и двунаправленные итераторы. Итераторы произвольного доступа. Свойства и тэги итераторов. Обобщенные функции. `std::advance`, `std::next`, `std::prev`, `std::distance`, `std::iter_swap`. Адаптеры итераторов. Обратные итераторы. Facade. Фасады для итераторов. Boost.Iterator.

Процессы. Память. Виртуальное адресное пространство. Физическое адресное пространство. Трансляция адресов. Таблицы страниц. Блоки управления памятью. Буферы трансляции адресов. Ошибки трансляции адресов. Ядро операционной системы. Пространство ядра. Ошибка page fault. Свободное пространство. Стек. Ошибка stack overflow. Сегмент отображения в память. Куча. Диспетчер памяти. Сегмент данных. Инициализированные и неинициализированные статические объекты. Сегмент кода. Ошибка segmentation fault. Продолжительность хранения объектов. Время жизни автоматических, динамических, статических, временных и локальных для потоков объектов. Гранулярность доступа к памяти. Выравнивание данных. `alignof`, `alignas`. Обработка ошибок выделения памяти. `std::set_new_handler`. Запрет на выброс исключений. `std::nothrow`. Выделение неинициализированной памяти. `operator new`, `operator delete`. Размещающая версия оператора new. Явный вызов деструкторов. Перегрузка операторов new и delete. Аллокаторы. Стандартные аллокаторы. `std::allocator`. Свойства аллокаторов. `std::allocator_traits`. Фрагментация памяти. Пользовательские аллокаторы. Линейные аллокаторы. `std::align`, `std::max_align_t`, `std::align_val_t`. Arena, Stack, Chain и Block аллокаторы. Аллокаторы без использования динамической памяти. Полиморфные аллокаторы. `std::pmr`.

## Материалы

- 09.01.pointer.smart
- 09.02.pointer.smart.shared.this
- 09.03.implementation.pointer.smart.unique
- 09.04.implementation.pointer.smart.shared
- 09.05.iterator.classification
- 09.06.implementation.iterator.forward.list
- 09.07.implementation.iterator.forward.list.facade
- 09.08.pattern.facade
- 09.09.memory.address.space
- 09.10.memory.address.translation
- 09.11.memory.alignment
- 09.12.memory.operator.new.handler
- 09.13.memory.operator.new.placement
- 09.14.implementation.utility.optional
- 09.15.memory.allocator.standard
- 09.16.memory.allocator
- 09.17.memory.allocator.block
- 09.18.implementation.allocator.arena
- 09.19.implementation.allocator.stack
- 09.20.implementation.allocator.chain
- 09.21.implementation.allocator.block
- 09.22.implementation.allocator.fixed
- 09.23.implementation.allocator.arena.adaptor
- 09.24.memory.allocator.polymorphism

## Литература

- B. Stroustrup, The C++ Programming Language, 4th Edition, разделы 33.2-33.4, 34.3-34.6
- B. Stroustrup, A Tour of C++, 2nd Edition, разделы 12.2, 12.3, 13.2, 13.6
- N. Josuttis, The C++ Standard Library, 2nd Edition, разделы 4.6, 5.2, 9, 19, S.3
- S. Meyers, Effective C++, 3rd Edition, разделы 13-15, 17, 47, 49-52
- S. Meyers, More Effective C++, разделы 4, 8, 27-29
- S. Meyers, Effective Modern C++, разделы 4.1-4.4
- S. Dewhurst, C++ Gotchas, разделы 24, 29, 43, 61-63, 67, 68
- H. Sutter, C++ Coding Standards, разделы 13, 45, 46, 60
- M. Bancila, The Modern C++ Challenge, раздел 2.21

# 10 Коллекции объектов и контейнеры

2 или 3 лекции общей продолжительностью до 8 академических часов; 5 задач для самостоятельной работы.

## Программа

Контейнеры. Семантика значений, указателей и ссылок. Безопасность относительно исключений. Порядок элементов. Последовательные контейнеры. Векторы. `std::vector`. Размер и емкость. Доступ к элементам. Свободные функции. `std::size`, `std::ssize`. Неконстантные итераторы. `std::begin`, `std::end`. Константные итераторы. `std::cbegin`, `std::cend`. Фиктивный элемент в конце. Копирование и перемещение коллекций. Итераторы для перемещения. `std::make_move_iterator`. Вставка и удаление элементов. Конструкторы. Конструирование на месте. Инициализаторы в фигурных скобках. `std::initializer_list`. Проблемы вывода типов. Сложность основных операций. Двусторонние очереди. `std::deque`. Статические массивы. `std::array`. Размещение на стеке. `std::to_array`. Двусвязные списки. `std::list`. Односвязные списки. `std::forward_list`. Определение среднего узла списков за один проход. Инвалидация итераторов. Proxy. Специализация вектора для логических значений. Разновидности прокси объектов. Интеллектуальные указатели. Отложенная загрузка. Многомерные массивы. Приоритет строк или столбцов. Размещение многомерных массивов в памяти. Линеаризация многомерных массивов. Boost.MultiArray. Векторы, матрицы и тензоры. Boost.uBLAS. Численные массивы. `std::valarray`. Адаптеры контейнеров. Последним пришел - первым ушел. LIFO. Стеки. `std::stack`. Первым пришел - первым ушел. FIFO. Очереди. `std::queue`. Очереди с приоритетами. `std::priority_queue`. Циклический буфер. Boost.CircularBuffer. Линеаризация циклического буфера.

Сбалансированные бинарные деревья поиска. Красно-черные деревья. Черная высота. Множества. `std::set`, `std::multiset`. Строгое слабое упорядочение. Антисимметричность, транзитивность, иррефлексивность и транзитивность эквивалентности. Вставка, удаление и поиск элементов. Подсказки. Бинарный поиск. Сложность основных операций. Извлечение узлов. Ассоциативные массивы. Словари. `std::map`, `std::multimap`. Замена ключей. `std::pair`, `std::make_pair`. Неупорядоченные контейнеры. Хеш функции. `std::hash`. Комбинированные хеши. Boost.ContainerHash. Хеш таблицы. `std::unordered_#`. Объекты и слоты. Коэффициенты загруженности. Способы разрешения коллизий. Метод цепочек. Итераторы хеш таблиц. Метод открытой адресации. Рехеширование. Сложность основных операций в лучшем и худшем случаях. Контейнеры с множественными интерфейсами. Boost.MultiIndex. Биассоциативные массивы. Boost.Bimap. Flyweight. Boost.Flyweight.

## Материалы

- `10.01.container.vector.deque.array.list.memory`
- `10.02.container.vector.deque.array.list`
- `10.03.container.vector.deque.array.list.sorting`
- `10.04.pattern.proxy.vector`
- `10.05.container.multidimensional`
- `10.06.container.multidimensional.multiarray`
- `10.07.container.multidimensional.ublas`
- `10.08.container.multidimensional.valarray`
- `10.09.algorithm.longest.common.subsequence`
- `10.10.container.adaptor`
- `10.11.container.adaptor.stack`
- `10.12.container.circular.buffer.memory`
- `10.13.container.circular.buffer`
- `10.14.container.red.black.tree.memory`
- `10.15.container.red.black.tree`
- `10.16.implementation.function.hash`
- `10.17.container.hash.table.memory`
- `10.18.container.hash.table`
- `10.19.implementation.class.functor.hash.equal`
- `10.20.container.multiindex.bimap`
- `10.21.pattern.flyweight`

## Литература


- B. Stroustrup, The C++ Programming Language, 4th Edition, разделы 11.3, 13.4, 29, 31, 34.2
- B. Stroustrup, A Tour of C++, 2nd Edition, разделы 11, 13.4
- N. Josuttis, The C++ Standard Library, 2nd Edition, разделы 5.1, 7, 8, 12
- S. Meyers, More Effective C++, раздел 30
- S. Meyers, Effective Modern C++, разделы 2.2, 3.1, 3.7, 8.2
- S. Dewhurst, C++ Gotchas, раздел 34
- H. Sutter, C++ Coding Standards, разделы 76, 78-83
- R. Grimm, C++20 Get the Details, раздел 5.3
- B. Schaeling, The Boost C++ Libraries, 2nd Edition, разделы 12, 13, 16, 19, 66

# 11 Итераторы и алгоритмы на диапазонах

2 или 3 лекции общей продолжительностью до 8 академических часов; 13 задач для самостоятельной работы.

## Программа

Типы функций. Указатели на функции. Использование функций в качестве аргументов. Шаблоны с идеальной передачей. Обобщенные вызовы. `std::invoke`. Вызываемые объекты. Внутренние состояния. Функции. Статические переменные. Глобальные переменные. Дополнительные аргументы. Функторы. Функциональные объекты. Данные-члены классов. Перегрузка оператора вызова. Предикаты. `std::less`, `std::greater`. `std::negate`, `std::plus`. Лямбда выражения. Замыкания. Списки захватов. Захваты с перемещением. Обобщенные захваты лямбда выражений. `mutable`. Стирание типов. `std::function`. Привязки. `std::bind`. Шаблонизированные лямбда-выражения. Захват указателей на экземпляры классов. Command. Visitor. Двойная диспетчеризация. Посетители на вариативных базовых классах и вариантах. `std::variant`, `std::visit`. Гетерогенные посетители. `std::any`. Вариативные последовательности. `std::integer_sequence`, `std::make_integer_sequence`.

Адаптеры итераторов. Итераторы вставки. `std::front_inserter`, `std::back_inserter`, `std::inserter`. Алгоритмы стандартной библиотеки шаблонов. Алгоритмы для пакетов. Алгоритмы поиска. Алгоритмы копирования, перемещения и обмена. Алгоритмы преобразования. Алгоритмы генерации. Алгоритмы удаления. Алгоритмы перемешивания. Алгоритмы для выборок. Алгоритмы упорядочения. Алгоритмы сортировки. Алгоритмы бинарного поиска. Алгоритмы слияния. Алгоритмы для множеств. Алгоритмы для кучи. Алгоритмы определения максимумов и минимумов. Алгоритмы лексикографических сравнений. Алгоритмы перестановки. Численные алгоритмы. Алгоритмы для неинициализированной памяти. Цикл for на основе диапазонов. Структурированные привязки. Алгоритмы на диапазонах. Диапазоны. `std::ranges`. Представления. `std::views`. Проекции. Адаптеры и алгоритмы. Библиотека для графов. Boost.Graph. Списки смежности. Матрицы инцидентности. Вершины. Ребра. Направления. Свойства. Посетители. Обход в ширину. BFS. Обход в глубину. DFS. Веса ребер. Алгоритм Дийкстры поиска кратчайших путей. Утилита AT&T Graphviz для визуализации графов.

## Материалы

- `11.01.pointer.function`
- `11.02.class.functor`
- `11.03.class.functor.lambda.expression`
- `11.04.pattern.command`
- `11.05.pattern.visitor`
- `11.06.pattern.visitor.utility.variant`
- `11.07.pattern.visitor.utility.any`
- `11.08.implementation.utility.variant`
- `11.09.template.metaprogramming.cartesian.product`
- `11.10.algorithm.iterator.adaptor.inserter`
- `11.11.algorithm.standard`
- `11.12.algorithm.standard.range.view`
- `11.13.algorithm.graph`
- `11.14.algorithm.graph.example`
- `11.15.algorithm.graph.traversal.bfs`
- `11.16.algorithm.graph.traversal.dfs`
- `11.17.algorithm.graph.search.dijkstra`

## Литература

- B. Stroustrup, The C++ Programming Language, 4th Edition, разделы 11.4, 12.5, 20.6, 32, 33.5, 33.6, 40.6
- B. Stroustrup, A Tour of C++, 2nd Edition, разделы 6.3, 12.5-12.8, 13.8
- N. Josuttis, The C++ Standard Library, 2nd Edition, разделы 4.4, 9.4, 10, 11
- D. Vandevoorde, C++ Templates, 2nd Edition, разделы 11.1, 22
- S. Meyers, Effective Modern C++, раздел 6
- H. Sutter, C++ Coding Standards, разделы 84-89
- R. Grimm, C++20 Get the Details, разделы 4.7, 5.1, 5.3
- M. Bancila, The Modern C++ Challenge, раздел 2.20
- B. Schaeling, The Boost C++ Libraries, 2nd Edition, раздел 31
- J. Siek, The Boost Graph Library, разделы 1-16

# 12 Кодирование символов и парсинг текстов

2 или 3 лекции общей продолжительностью до 8 академических часов; 14 задач для самостоятельной работы.

## Программа

Символы. `char`. Таблица ASCII. Знаковые и беззнаковые символьные типы. Специальные символы. Кодировки. Многобайтовые и широкие кодировки. 7-битная и 8-битная кодировки ASCII. ISO-Latin-1. Unicode. Универсальные наборы символов. UCS. Универсальные форматы преобразования. UTF. UTF-8, UTF-16, UTF-32. `char8_t`, `char16_t`, `char32_t`, `wchar_t`. Endianness. Big и little endian. `std::endian`. Знак порядка байтов. BOM. Проблемы кракозябр. Строки. `std::basic_string`, `std::char_traits`. Псевдонимы для строк. `std::string`, `std::u8string`, `std::u16string`, `std::u32string`, `std::wstring`. Ввод строк. `std::getline`. Пропуск пробельных символов. `std::ws`. Вывод строк в кавычках. `std::quoted`. Стандартные строковые литералы. Индексы и итераторы. `std::stoi`, `std::to_string`. C-строки. Массивы символов. Завершающий нуль-символ. Функции для C-строк. `std::strlen`. Нечувствительные к регистру строки. Оптимизация маленьких строк. SSO. Представления. `std::string_view`. Преимущества и опасности. Интернационализация. I18N. Локализация. L10N. Локали. `std::locale`, `setlocale`. Классическая локаль C. Подключение локалей к потокам ввода-вывода. Фацеты. `std::use_facet`. Фацеты для пунктуации и валют. `std::numpunct`, `std::moneypunct`. Фацеты для дат и времени. `std::time_put`, `std::time_get`. Преобразование кодировок между UTF. Boost.Locale.

Регулярные выражения. `std::regex`. Грамматика ECMA-script. Паттерны. Сырые строки. Совпадения с паттернами. `std::regex_match`. Поиск по паттернам. Контейнер совпадений. `std::regex_search`. Группы и подобразцы. `std::match_results`. Префиксы и суффиксы. Последовательный обход совпадений. Итераторы регулярных выражений. `std::regex_iterator`. Токены. `std::regex_token_iterator`. Замены по паттернам. `std::regex_replace`. Флаги регулярных выражений. `std::regex_constants`. Грамматики. Парсеры рекурсивного спуска. Калькулятор Бьерна Страуструпа. Инструкции, объявления переменных, выражения, термы и первичные элементы. Потоки и токены. Расширенная форма Бэкуса-Наура. EBNF. Boost.Spirit. Парсеры строк. Правила и последовательности. Атрибуты. Парсер для комплексных чисел. Парсер для римских чисел. Парсер для классов. Адаптеры для классов. Boost.Fusion. Калькулятор на абстрактных синтаксических деревьях. AST.

## Материалы

- `12.01.character.encoding`
- `12.02.character.encoding.unicode.utf`
- `12.03.character.string.view`
- `12.05.locale.facet`
- `12.06.locale.facet.utf.converter`
- `12.07.parser.regex.ecma.script`
- `12.08.parser.regex`
- `12.09.parser.grammar.arithmetic`
- `12.10.parser.grammar.arithmetic.calculator`
- `12.11.parser.spirit.number.integer`
- `12.12.parser.spirit.number.complex`
- `12.13.parser.spirit.number.romanus`
- `12.14.parser.spirit.fusion.class.structure`
- `12.15.parser.spirit.grammar.arithmetic.calculator`

## Литература

- B. Stroustrup, The C++ Programming Language, 4th Edition, разделы 36, 37, 39
- B. Stroustrup, A Tour of C++, 2nd Edition, раздел 9
- N. Josuttis, The C++ Standard Library, 2nd Edition, разделы 13, 14, 16
- R. Grimm, C++20 Get the Details, раздел 5.6
- M. Bancila, The Modern C++ Challenge, раздел 3

# 13 Потоки ввода-вывода и сериализация

2 или 3 лекции общей продолжительностью до 8 академических часов; 5 задач для самостоятельной работы.

## Программа

Библиотека IOStream. Иерархия классов потоков. `std::ios_base`. Состояния потоков. `good`, `eof`, `fail`, `bad`. Проверки ввода. Внедрение исключений. Флаги форматирования. Манипуляторы. `std::endl`. Пользовательские манипуляторы. Перегрузка операторов ввода и вывода. `std::basic_ios`. Специализации шаблонов классов потоков для двух символьных типов. Буферы потоков. `std::basic_streambuf`. Потоки ввода и вывода. `std::basic_istream`, `std::basic_ostream`, `std::basic_iostream`. Виртуальное наследование. Глобальные экземпляры классов потоков ввода и вывода. `std::cin`, `std::cout`, `std::cerr`, `std::clog`. Стандартные заголовочные файлы. Файловые потоки ввода и вывода. `std::fstream`. Режимы открытия файлов. `in`, `out`, `trunc`, `app`, `ate`, `binary`. Произвольный доступ в файлах. Строковые потоки ввода и вывода. `std::stringstream`. Итераторы буферов потоков ввода и вывода. `std::istreambuf_iterator`, `std::ostreambuf_iterator`. Окончание ввода. Символ конца ввода. EOF. Библиотека потоков C. `stdin`, `stdout`, `stderr`. `std::clearerr`. Синхронизация с stdio. Итераторы потоков ввода и вывода. `std::istream_iterator`, `std::ostream_iterator`. Библиотека форматирования. `std::format`, `std::format_to`. Пользовательские форматы. `std::formatter`.

## Материалы

- `13.01.stream.class.inheritance`
- `13.02.stream.class`
- `13.03.stream.class.file`
- `13.04.stream.class.string`
- `13.05.stream.iterator.buffer.input.output`
- `13.06.algorithm.erase.comment`
- `13.07.algorithm.erase.comment.test`
- `13.08.stream.output.format`

## Литература

- B. Stroustrup, The C++ Programming Language, 4th Edition, раздел 38
- B. Stroustrup, A Tour of C++, 2nd Edition, разделы 10, 12.4
- N. Josuttis, The C++ Standard Library, 2nd Edition, раздел 15
- S. Meyers, More Effective C++, раздел 25
- M. Bancila, The Modern C++ Challenge, разделы 3.23, 4, 9
- B. Kernigan, The C Programming Language, 3nd Edition, task 1.23

# 14 Параллельное программирование

4 или 5 лекций общей продолжительностью до 16 академических часов; 0 задач для самостоятельной работы.

## Программа

## Материалы

-

## Литература

- B. Stroustrup, The C++ Programming Language, 4th Edition, разделы 41, 42
- B. Stroustrup, A Tour of C++, 2nd Edition, разделы 12.9, 15
- N. Josuttis, The C++ Standard Library, 2nd Edition, разделы 4.5, 18
- S. Meyers, Effective Modern C++, разделы 3.10, 7
- H. Sutter, C++ Coding Standards, раздел 12
- R. Grimm, C++20 Get the Details, разделы 6, 7

# 15 Компьютерные сети и сетевые технологии

3 или 4 лекции общей продолжительностью до 12 академических часов; 0 задач для самостоятельной работы.

## Программа

## Материалы

-

## Литература

-

# 16 Встраивание технологий других языков

2 или 3 лекции общей продолжительностью до 8 академических часов; 0 задач для самостоятельной работы.

## Программа

Ассемблеры. Язык ассемблера Intel x86. Диалект Microsoft Macro Assembler. Память, регистры и дизассемблированный код. Стек. Соглашения о вызовах. `cdecl`, `pascal`, `stdcall`. Вызывающие и вызываемые. Порядок передачи аргументов. Возвращаемые значения. Очистка стека. Встраивание ассемблерного кода. `asm`. Регистры общего назначения. `eax`, `ebx`, `ecx`, `edx`. Регистры для индексов. `esi`, `edi`. Указатели для стека. `esp`. Указатели для фреймов. `ebp`. Инструкции по перемещению данных. `push`, `pop`, `mov`. Арифметические инструкции. `add`, `sub`, `inc`, `dec`. Инструкции потока управления. Прыжки. `jmp`. Метки. Сравнения. `cmp`. Вызовы. `call`. Адресация памяти. Язык ассемблера, процессоры, стек, регистры и инструкции для чисел с плавающей точкой.

Длинная арифметика. Способы хранения цифр. Перегрузка арифметических операторов. Быстрое произведение Карацубы. Прочие операции. Расширенные целые числа. Boost.Multiprecision. Встраивание Python. Подключение интерпретатора. Python C/C++ API. Boost.Python. Глобальный блокировщик интерпретатора. GIL. Потоки и процессы. Импорт функций. Вызов функций. Передача аргументов. Получение результатов. Обработка исключений. Инструменты Python. Пакет math. Вычисление факториалов. Пакет matplotlib. Визуализация.

## Материалы

- `16.01.embedding.assembler`
- `16.02.implementation.class.big.integer`
- `16.03.number.integer.multiprecision`
- `16.04.embedding.python.wrapper.header`
- `16.05.embedding.python.wrapper.source`
- `16.06.embedding.python.caller`
- `16.07.embedding.python.script`

## Литература


- R. Hyde, The Art of Assembly Language, 2nd Edition, разделы 1-6