Зачет не является обязательным процессом для сдачи курса. Зачет весит 10 баллов, которые суммируются с баллами за лабы и проект. Максимум в семестре можно получить 55 первичных баллов (10 баллов за зачет – это доп к лабам и проекту), оценка переводится во вторичные 10 баллов линейно и непрерывно.

Вопросы на 3

1. Фундаментальные и пользовательские типа данных. Хранение целых чисел в памяти компьютера, прямой и дополнительный код.
2. Числа с плавающей точкой: хранение и взаимодействие с ними.
3. Модель памяти Фон-Неймана. Стек локальных переменных и куча.
4. Сырые указатели и ссылки – сходства, отличия, зачем нужны. Арифметика указателей.
5. Явное и неявное приведение типов. C-style casts. Преобразования числовых типов с потерей точности и без.

Вопросы на 4

1. Инкапсуляция: что это такое и зачем используется. Идея объединения методов и данных. Инвариант класса и консистнтное состояние. Модификторы доступа public, private.
2. Наследование: что это такое и зачем используется. Модификатор доступа protected. Ключевые слова final и override. Виртуализация и таблица виртуальных функций. Виртуальный деструктор: для чего нужен.
3. Полиморфизм – концепция и примеры статического и динамического полиморфизма в языке.
4. Ключевые слова class и struct, const и static методы в классах. Указатель this.
5. Пространства имен, их использование. Аддитивность пространств имён.

Вопросы на 5

1. Шаблоны, двухэтапное инстанцирование шаблона и разрешение зависимых имен. Использование явного `this`→.
2. Разрешение типов в шаблонном параметре. Ключевое слово `auto` и принцип его работы. Особенности разрешения уточненных и не уточненных типов, уточненного и не уточненного `auto`.
3. Полная и частичная специализация шаблона: что такое, какой синтаксис и для чего используется частичная специализация.
4. Последовательные контейнеры STL: array, vector, list, deque, list, forward list. Принцип работы, асимптотики доступа и добавления. Сравнение deque и vector.

5. Алгоритмы STL: несколько примеров кода, подобранных под разные алгоритмы. Зачем придумали `std::array`?

6. Итераторы. Методы `begin` и `end` в контейнерах. Range-based-for и как он раскрывается на самом деле. Инвалидация итераторов на примере методов `std::vector` и `std::list`. Валидные итераторы и диапазоны. Категории итераторов. Примеры контейнеров с соответствующими категориями итераторов.

Вопросы на 6

1. Ключевое слово `decltype`, принцип работы, отличие от `auto`.

2. Адаптеры stack и queue для контейнеров в STL.

3. Ассоциативные контейнеры STL: set, map, multiset, multimap. Хеш-таблица. Принцип работы, асимптотики доступа и добавления.

4. Исключения. Ментальная модель throw и catch. Порядок использования catch-блоков.

5. ООП и память. Правило трех: в чем заключается и какие проблемы призвано решить.

6. Схема сборки многофайлового проекта. Этапы компиляции. Препроцессинг. Разделение на файлы .hpp и .cpp. Линкер и примеры ошибкок линковки. CMake – что такое и для чего нужен. Примеры CMakeLists для ваших проектов.

Вопросы на 7

1. Кэш процессора и его эффекты на перфоманс на примере перемножения матриц.

2. Вариабельные шаблоны, пакет параметров в шаблоне. Пример использования пачки параметров через fold-expressions. Пример использования пакчи параметров для реализации функции invoke.

3. Кортеж `std::tuple`: для чего нужен, примеры использования. Код функции вывода кортежа в поток.

4. Нетиповые параметры шаблонов. Рекурсивное инстанцирование. Условный оператор времени компиляции (`if constexpr`). Вычисление n-го числа Фибоначчи на этапе компиляции с примером кода.

5. Ключевое слово `constexpr` применительно к переменным и функциям. Что означает, как используется. Что можно использовать внутри `constexpr` функций. Ключевое слово `consteval` и его отличие от `constexpr` с примерами кода.

6. Проблема ромба и виртуальное наследование. Интерфейсные классы, наследование от интерфейсных классов.

Вопросы на 8

1. Идиома RAII: суть, какие проблемы решает и для чего используется.
2. Семантика перемещения а шаблонах. Идеальная передача с помощью `std::forward`.
3. Принцип SFINAE с примерами.
4. RTTI, typeinfo и dynamic_cast: для чего используются и какую проблему решают. Пример места, где static_cast не может сработать.
5. Функторы и лямбда-функции. Что это такое и как связаны две эти вещи.
6. Статический интерфейс к шаблонным классам и функциями с помощью `std::void_t`

Вопросы на 9

1. `std::decay` и его отношение к ключевому слову `auto`. Пример кода для `std::decay`.
2. Reference collapsing, универсальные ссылки, условия контекста сворачивания ссылок.
3. Гарантии безопасности исключений. Пример на каждую из гарантий безопасности. Принцип "критической линии".
4. Семантика перемещения и ссылок на rvalue. Правило пяти для классов.
5. Перегрузка операторов вывода, пре- и пост- инкремента и декремента
6. Вспомогательные классы для метапрограммирования: `std::is_same`, `std::is_base_of`, `std::enable_if`, `std::void_t`. Принцип работы, примеры возможной реализации.

Вопросы на 10

1. Вещи, которые можно найти в пустом классе. Правило нуля.
2. lvalue, prvalue, xvalue, glvalue, rvalue и их отношения с `decltype`.