

ВОПРОСЫ К ЭКЗАМЕНУ/ЗАДАЧИ C++ DEVELOPER;



Экзамен состоит из билетов, билет содержит в себе 4 основных теоретических вопроса и 1 задача, а также дополнительные вопросы.

Всего - **47** вопросов для подготовки. (\pm 32-33 различных билета)

⚠ Прекрасные новости - за экзамен можно получить **60 баллов** если:

- ответить максимально идеально на основную теорию (+25 баллов)
 - 1 вопрос это 6,25 баллов;
- решить задачу (+20 баллов)
- ответить на доп вопросы (+15 баллов)

или 😞 0 баллов (если пустой лист);

БИЛЕТ ФОРМИРУЕТСЯ СЛУЧАЙНЫМ ОБРАЗОМ.

НА ПОДГОТОВКУ К ОТВЕТУ ВЫДЕЛЯЕТСЯ РОВНО **59 МИНУТ** ВРЕМЕНИ.
(ЕСЛИ ГОТОВЫ РАНЬШЕ - ЭТО ВСЕЦЕЛО ПРИВЕТСТВУЕТСЯ)

РАЗРЕШАЕТСЯ 1 РАЗ ПЕРЕБРАТЬ БИЛЕТ (БЕЗ ПОТЕРИ БАЛЛОВ) ЗА
КАЖДЫЙ ПОСЛЕД-Й ПЕРЕБОР **-20 БАЛЛОВ**.

ПРИ ПОДГОТОВКЕ МОЖНО ИСПОЛЬЗОВАТЬ КОНСПЕКТЫ ЛЕКЦИЙ!!! (НО ЭТО ВСЕ РАВНО НЕ ПОМОЖЕТ)

СПИСЫВАНИЕ КАРАЕТСЯ АВТОМАТИЧЕСКОЙ НЕ-СДАЧЕЙ ЭКЗАМЕНА!!!

ЗАПРЕЩЕНО ИСП-Е ИНТЕРНЕТА И ДР СРЕДСТВ. НО РАЗРЕШЕНО ПРИ
РЕШЕНИИ ЗАДАЧИ ИСП-ТЬ [VSCODE/CLION/Visual Studio](#).

ЭКЗАМЕН 😎 **УСПЕШНО СДАН** - ЕСЛИ ВЫ НАБРАЛИ \geq **30,5 БАЛЛОВ**.

1 БЛОК (ОСНОВЫ)

1. Версии C++. Компиляция программы. Схема компиляции программы. Структура проекта (.cpp, .h). Пример.
2. Ввод/Вывод данных. Базовый синтаксис языка C++. Пример.
3. Переменные и типы данных. sizeof(), преобразование типов данных. Вторичные типы данных size_t, ptrdiff_t, и другие - привести примеры.
4. Переменные. Операции над переменными.
5. Условная конструкция. Конструкция switch/case.
6. Операторы break/continue/return. Циклы C++.
7. Класс String. Работа со строками. Методы. Примеры.
8. Массивы данных. Виды массивов. Статический/Динамический массив.
9. Структуры данных <struct>. Unions/Enumerations. Примеры.
10. Адресная шина. Оперативная память/виртуал. адресное пространство. Размер памяти для 32/64 bit SYSTEM. Указатели и ссылки. Виды указателей. Привести простые примеры работы.
11. Функции. Описание синтаксиса функций. Виды функций. Создание/Вызов функций.
12. Функции. Перегрузка функций. Понятие рекурсии.
13. Описание функции main(). Динамические атрибуты методов (argc, argv). Для чего используется return внутри main().
14. Исключения и обработка ошибок. Классы исключений (встроенные). Собственный класс исключений. Ключевое слово throw.
15. Работа с файлами. Потоки чтения и записи данных. Сколько в одном байте бит? Препроцессор, директивы препроцессора. Работа с csv файлами.

2 БЛОК (ООП)

1. ООП. Плюсы и минусы использования ООП. Понятие класса/объекта. Свойства (атрибуты или поля) и методы класса. Вызов атрибутов и методов класса.
2. ООП. Обобщение понятий класса и объекта. Конструктор класса. Виды конструкторов. Ключевое слово this.
3. ООП. Секции класса (public/private/protected). Public/Private атрибуты и методы класса.
4. ООП. Список инициализации. Понятие деструктор. Ключевое слово default.
5. ООП. Константные/Статические методы класса. Виртуальный метод класса. Префикс OVERRIDE.
6. ООП. Концепции ООП. Таблица наследования.
7. ООП. Дружественные функции. Шаблонные классы/функции.
8. Перегрузка операторов C++. Плюсы и минусы. Правила перегрузки. Перегрузка оператора сложения/вычитания/умножения/деления.
9. Перегрузка операторов C++. Перегрузка инкремента/декремента. Перегрузка оператора вывода/ввода потока данных. Перегрузка операторов сравнения.

3 БЛОК (АИСД + STL)

1. АИСД. Плюсы и минусы. Алгоритмы сортировок, Оценка сложности программы (O-большое).
2. АИСД. стек/очередь/дек
3. АИСД. Связный списки. Односвязный список. Двусвязный список. Кольцевой список.

4. АИСД. Деревья. Виды деревьев. Отличие бинарного дерева от BST?
5. АИСД. ХЕШ-таблица. Методы решения коллизий.
6. АИСД. Вектор + PAIR.
7. АИСД. Множество. Операции над множеством.
8. STL, STD. `std::stack`, `std::queue`, `std::deque`, `std::pair`;
9. STL, STD. `std::list`, `std::vector`, `std::set/map`;
10. STL, STD. `std::unordered_set/unordered_map`. Iterators.
11. STL, STD. `std::chrono`, `std::thread`, `std::mutex`, `std::regex`.
12. STL, STD. `std::future`, `std::promise`, `std::functional`, `std::algorithm`.

4 БЛОК (WinAPI и прочее)

1. WinAPI. Описание. Схема работы. Хендлы (handles). Типы handles.
2. WinAPI. window procedure. Создание окна (window).
3. WinAPI. работа с событиями и сообщениями. Обработка сообщений окна.
Работа с сетью, звуком, таймером и тп.
4. WinAPI. Работа с ресурсами системы. Многопоточное программирование.
5. WinAPI. Работа с DLL. Работа с реестром. Обработка ошибок и исключений.
`GetLastError()`;
6. WinAPI. Что такое UI, GUI? Графический интерфейс и блок элементов управления (кнопки, текстовое поле ввода, списки и другое).
7. Server part. Схема работы <localhost>. <server request>, HTTP, connect to server.
create-localhost-server. SOCKET.
8. Системное программирование. Концепции/Описание и примеры.
9. Параллельное программирование. Концепции/Описание и примеры.
10. Функциональное программирование. Концепции/Описание и примеры.
11. OS(SHELL). Интернет HTTP. Коды ответов HTTP. Схема работы интернета.
12. * Зачем нужен C++? Сферы применения языка C++.

5 БЛОК (ПРАКТИКА)

- 1 практическая задача на любую randomную тему, которые мы изучили/прошли.

Примеры практических задач;

1. Дан массив данных (который задает пользователь), найти минимальный положительный элемент в этом массиве.
2. Дан массив данных (который задает пользователь), найти максимальный элемент в этом массиве.
3. <задача на стек данных> Напишите функцию, которая принимает на вход строку, содержащую только символы '(', ')', '{', '}', '[' и ']', и определяет, является ли расстановка скобок в этой строке корректной.
4. <задача на стек данных> Напишите функцию, которая проверяет балансировку символов в строке. В строке могут присутствовать только символы '(' и ')'. Функция должна вернуть True, если каждой открывающей скобке соответствует закрывающая, и False в противном случае.
5. Напишите функцию которая принимает массив вещественных чисел и возвращает два самых наибольших отрицательных числа в этом массиве данных.
6. <задача с leetcode> Учитывая вектор целых чисел **nums** и целое число **target**, верните индексы двух чисел так, чтобы их сумма составляла **target**. Вы можете предположить, что каждый вход будет иметь ровно одно решение, и вы не можете использовать один и тот же элемент дважды. Вы можете вернуть ответ в любом порядке.

Пример 1:

```
Ввод: nums = [2,7,11,15], target = 9
Выход: [0,1]
Объяснение: поскольку nums[0] + nums[1] == 9, мы возвращаем [0, 1].
```

Пример 2:

```
Ввод: nums = [3,2,4], цель = 6
Выход: [1,2]
```

Пример 3:

```
Ввод: nums = [3,3], цель = 6
Выход: [0,1]
```

Задачи на файлы:

- открыть файл .csv - прочитать его содержимое вывести на экран.
- открыть файл .csv - внести в него какие то изменения, сохранить изменения.
- открыть файл .txt - прочитать и вывести его содержимое.
- открыть файл .txt - записать изменения в файл.
- попробуйте сделать операцию объединения двух текстовых файлов в один единый файл.

Типовая задача:

создать функцию foo() с чтением из потока данных argv целых чисел до тех пор пока не встретиться отрицательное число.

** стоит отметить что задача может быть на любую из пройденных тем, советую повторить ООП, АИСД (деревья, хеш-таблицы). Также работу с файлами .txt, .csv, и др*

Таблица-1 Оценивание ученика:

Имя Фамилия ученика	Задача (20б)	T1 (6,25б)	T2 (6,25б)	T3 (6,25б)	T4 (6,25б)	Доп1 (5б)	Доп2 (5б)	Доп3 (5б)	Итого:

**пометка T1, T2, T3, T4 - основные теоретические вопросы по билету; по 4 блоку дополнительные вопросы не ЗАДАЮ!!!!*

Таблица-2 Оценивание задачи:

Номер билета	Работоспособность кода (согласно условию и требованиям) (9б)	Соблюдение всех требований условия задачи (6б)	Соблюдение синтаксически х норм. (2б)	Грамотное описание функций, правильная передача аргументов функций. (2б)	Читаемость кода, оптимизирован ность, оценка сложности программного кода (1б)