Оценочные материалы дополнительной общеразвивающей программы «Управление цифровыми продуктами и системами»

Пример оценочных материалов для промежуточной аттестации

Модуль1. Основы IT-инфраструктуры

Задание 1. Основы Linux и работа с файловой системой

В домашней директории создайте структуру каталогов и выполните операции с файлами согласно списку. Результат оформите в виде отчёта с командами и скриншотами.

Список действий:

Создать структуру: ~/exam/projects/, ~/exam/backups/, ~/exam/temp/

В projects создать файлы: readme.txt, notes.txt, todo.txt

Скопировать todo.txt в backups

Переместить notes.txt в temp

Установить на readme.txt права 640

Найти все .txt файлы в домашней директории

Эталон выполнения

Отчёт со всеми командами (например, mkdir, touch, cp, mv, chmod, find)

Скриншот вывода ls -la ~/exam

Скриншот вывода find ~ -name "*.txt" 2>/dev/null

Критерии оценки

Что проверяется	Балл
Все каталоги созданы	1
Все файлы созданы	1
Копирование и перемещение выполнены верно	1
Права 640 установлены	1
Поиск файлов выполнен	1
Итого	5

Задание 2. Bash-скрипт для резервного копирования

Напишите Bash-скрипт backup.sh, который принимает два аргумента: исходную директорию и директорию для резервного копирования.

Требования к скрипту:

Проверить, существует ли исходная директория. Если нет — вывести ошибку.

Создать директорию для резервного копирования, если её нет.

Скопировать все файлы с расширением .txt из исходной директории в целевую.

При копировании добавить к имени файла текущую дату в формате ГГГГ-ММ-ДД.

Вывести количество скопированных файлов.

Эталон выполнения

Текст скрипта (код)

Скриншот выполнения с примером

Скриншот результата (файлы с датой в имени)

Критерии оценки

Что проверяется	Балл
Проверка исходной директории	1
Создание целевой директории	1
Копирование только .txt файлов	1
Добавление даты к имени файла	1
Подсчёт и вывод количества файлов	1
Итого	5

Задание 3. Git: локальный и удалённый репозиторий

Выполните следующие действия с Git:

Настроить Git (имя и email пользователя)
Создать локальный репозиторий в папке ~/git-exam
Создать файл hello.txt с текстом «Hello, Git!» и сделать первый коммит
Создать ветку feature, переключиться на неё
Добавить в hello.txt новую строку «Feature branch work» и сделать коммит
Переключиться обратно в ветку main
Создать репозиторий на GitHub
Связать локальный репозиторий с удалённым
Отправить ветку main на GitHub

Эталон выполнения
Отчёт со всеми командами
Скриншот вывода git log --oneline
Скриншот репозитория на GitHub с коммитом
Ссылка на репозиторий (или скриншот)
Критерии оценки

Что проверяется	Балл
Настройка Git выполнена	1
Локальный репозиторий и первый коммит	1
Ветка feature и коммит в ней	1
Удалённый репозиторий создан и связан	1
Код отправлен (push) на GitHub	1
Итого	5

Задание 4. Программа на C: «Угадай число»
Напишите программу на C, которая реализует игру «Угадай число».
Требования:
Программа загадывает случайное число от 1 до 100
Пользователь вводит числа, программа даёт подсказки: «Больше», «Меньше», «Угадал!»
У пользователя не более 10 попыток
После окончания игры программа спрашивает: «Сыграем ещё? y/n»
При вводе y — новая игра, при n — выход

Эталон выполнения
Исходный код программы (файл guess.c)
Скриншот компиляции (gcc guess.c -o guess)
Скриншот выполнения программы (с примером игры)
Критерии оценки

Что проверяется	Балл
Программа компилируется без ошибок	1
Случайное число генерируется	1
Подсказки «Больше»/«Меньше» работают	1
Ограничение на 10 попыток	1
Возможность сыграть ещё	1
Итого	5

Задание 5. Массивы и функции на C
Напишите программу на C, которая работает с массивом целых чисел.
Требования:
Запросить у пользователя размер массива N (не более 20)
Заполнить массив числами, введёнными с клавиатуры
Вывести исходный массив на экран
Найти и вывести минимальный, максимальный элементы и сумму всех элементов

(каждое вычисление в отдельной функции)

Отсортировать массив по возрастанию методом «пузырька»

Вывести отсортированный массив

Подсчитать и вывести количество чётных чисел

Эталон выполнения

Исходный код программы (файл array.c)

Скриншот компиляции и выполнения (например, для N=5)

Критерии оценки

Что проверяется	Балл
Ввод N и заполнение массива	1
Функции минимума, максимума, суммы	1
Сортировка «пузырьком»	1
Вывод исходного и отсортированного массивов	1
Подсчёт чётных чисел	1
Итого	5

Задание 6. Работа с файлами на C

Напишите программу file_stats.c, которая анализирует текстовый файл.

Требования:

Программа принимает аргументы командной строки: входной файл и выходной файл

Проверяет, что передано 2 аргумента, иначе выводит инструкцию

Открывает входной файл для чтения, если файл не существует — выводит ошибку

Подсчитывает во входном файле:

количество строк

количество слов (разделители: пробел, табуляция, перевод строки)

количество символов (без учёта перевода строки)

Записывает результаты в выходной файл в формате:

text

Количество строк: X

Количество слов: Y

Количество символов: Z

Закрывает оба файла

Эталон выполнения

Исходный код программы (файл file_stats.c)

Скриншот компиляции

Пример файла input.txt

Скриншот выполнения программы

Полученный файл output.txt (содержимое)

Критерии оценки

Что проверяется	Балл
Проверка аргументов командной строки	1
Проверка существования входного файла	1
Корректный подсчёт строк, слов, символов	1
Запись результатов в выходной файл	1
Программа компилируется и работает	1
Итого	5

Модуль 2. Алгоритмизация и структурное программирование

Задание 1. Строки + аргументы командной строки (C)

Условие

Напишите программу compare.c, которая принимает из командной строки две

строки и сравнивает их без учёта регистра, не используя стандартную функцию strcasecmp.

Формат запуска:

bash

./compare "Hello" "HELLO"

Требования:

Проверить, что передано ровно 2 аргумента. Если нет — вывести Usage: ./compare <str1> <str2> и завершиться с кодом 1

Сравнить строки без учёта регистра (своя реализация)

Вывести Equal, если строки равны, иначе Not equal

Завершиться с кодом 0

Эталон выполнения

Исходный код compare.c

Код компилируется без ошибок и предупреждений

Примеры работы:

./compare Hello HELLO → Equal

./compare Hello World → Not equal

./compare Hello → Usage: ./compare <str1> <str2>

Критерии оценки

Что проверяется	Балл
Проверка количества аргументов командной строки	1
Собственная реализация сравнения строк без учёта регистра	2
Корректный вывод результата	1
Код компилируется (gcc -Wall -Wextra -Werror)	1
Итого	5

Задание 2. Структуры (C)

Условие

Определите структуру Student, содержащую:

name — строка (массив char размером 50)

age — целое число

gpa — вещественное число (средний балл)

Напишите программу, которая:

Создаёт массив из 3 студентов (заполнить в коде)

Находит студента с максимальным средним баллом

Выводит его имя и средний балл в формате: Best student: <name>, GPA: <gpa>

Эталон выполнения

Исходный код students.c

Структура определена верно

Массив из 3 студентов создан и заполнен

Функция поиска лучшего реализована

Критерии оценки

Что проверяется	Балл
Структура Student определена верно	1
Массив структур создан и заполнен	1
Поиск студента с максимальным GPA	1
Корректный вывод результата	1
Код компилируется и работает	1
Итого	5

Задание 3. Матрицы (C)

Условие

Определите структуру Matrix, содержащую:

rows — количество строк

cols — количество столбцов

data — указатель на двумерный массив (double**)

Напишите функцию matrix_sum, которая принимает две структуры Matrix и возвращает новую структуру Matrix с суммой матриц.

Требования:

Если размеры матриц не совпадают, вернуть структуру с rows = 0, cols = 0, data = NULL

Память для новой матрицы выделяется динамически

При ошибке выделения памяти — вернуть нулевую матрицу

Эталон выполнения

Исходный код matrix.c (только функция и структура, либо полная программа с примером)

Структура Matrix определена

Проверка совпадения размеров

Динамическое выделение памяти

Критерии оценки

Что проверяется	Балл
Структура Matrix определена верно	1
Проверка совпадения размеров матриц	1
Динамическое выделение памяти под data	1
Корректное сложение всех элементов	1
Обработка ошибок (NULL после malloc)	1
Итого	5

Задание 4. Матрицы + файлы (C)

Условие

Напишите программу transpose.c, которая:

Принимает из командной строки имя входного файла и имя выходного файла

Читает матрицу из входного файла (формат: первая строка — rows cols, затем rows строк по cols чисел)

Вычисляет транспонированную матрицу

Записывает результат в выходной файл в том же формате

Формат запуска:

bash

./transpose input.txt output.txt

Эталон выполнения

Исходный код transpose.c

Проверка аргументов командной строки (если меньше 2 — вывести usage)

Чтение из файла с проверкой ошибок

Динамическое выделение памяти

Транспонирование выполнено верно

Запись в выходной файл в правильном формате

Критерии оценки

Что проверяется	Балл
Проверка аргументов командной строки	1
Корректное чтение из файла	1
Динамическое выделение памяти под матрицу	1
Транспонирование выполнено верно	1
Корректная запись в выходной файл	1

Итого	5

Задание 5. Алгоритмы: сортировка (язык по выбору)

Условие

На любом языке по выбору (C, C++, Java, Kotlin, Swift, Python, Go, C#, JS) реализуйте функцию сортировки массива целых чисел методом «пузырька» (bubble sort) по возрастанию.

Требования:

Функция принимает массив (или список) и его длину

Не использовать встроенные функции сортировки языка

Функция изменяет исходный массив (не создаёт новый)

Эталон выполнения

Исходный код с функцией bubble_sort

Пример использования (в main или тесте)

Результат работы на примере: [64, 34, 25, 12, 22, 11, 90] → [11, 12, 22, 25, 34, 64, 90]

Критерии оценки

Что проверяется	Балл
Выбран корректный язык	1
Реализован именно алгоритм «пузырька»	1
Сортировка по возрастанию	1
Функция изменяет исходный массив	1
Пример работает верно	1
Итого	5

Задание 6. Алгоритмы: рекурсия (язык по выбору)

Условие

На любом языке по выбору (из того же, что и задание 5, или другом) реализуйте рекурсивную функцию для вычисления N-го числа Фибоначчи. Циклы использовать запрещено.

Формула:

fib(0) = 0

fib(1) = 1

fib(n) = fib(n-1) + fib(n-2)

Требования:

Функция принимает целое неотрицательное число n

Возвращает целое число

Обработка $n \leq 0$ — возвращать 0

Эталон выполнения

Исходный код с рекурсивной функцией fib

Пример использования (в main или тесте)

Результат для n=10: 55

Критерии оценки

Что проверяется	Балл
Функция рекурсивная (без циклов)	2
Базовые случаи (n=0, n=1) обработаны верно	1
Рекурсивный случай реализован верно	1
Результат для n=10 = 55	1
Итого	5

Модуль 3. Объектно-ориентированное программирование

Задание 1. Класс и инкапсуляция (аналог CPP1 — матрица)

Условие

Создайте класс Matrix, который представляет матрицу вещественных чисел.

Требования:

Закрытые поля: rows (int), cols (int), data (double**)

Публичные методы:

Конструктор по умолчанию (создаёт пустую матрицу 0×0)

Конструктор с параметрами Matrix(int rows, int cols) — создаёт матрицу заданного размера, заполненную нулями

Конструктор копирования

Деструктор (освобождает память)

Геттеры: get_rows(), get_cols()

Метод set_value(int row, int col, double value) — устанавливает значение

Метод get_value(int row, int col) const — возвращает значение

Метод print() const — выводит матрицу на экран

Проверка инкапсуляции: поля rows, cols, data должны быть закрыты (private).

Эталон выполнения

Файлы matrix.hpp и matrix.cpp

Класс Matrix с указанными конструкторами, деструктором и методами

Все методы реализованы корректно

Память выделяется и освобождается правильно (нет утечек)

Критерии оценки

Что проверяется	Балл
Класс определён корректно (поля private)	1
Конструкторы (по умолчанию, с параметрами, копирования)	1
Деструктор освобождает память	1
Геттеры и методы set/get реализованы	1
Метод print выводит матрицу	1
Итого	5

Задание 2. Перегрузка операторов (аналог CPP1)

Условие

Дополните класс Matrix из задания 1 перегрузкой следующих операторов:

operator+ — сложение двух матриц (возвращает новую матрицу)

operator- — вычитание двух матриц

operator* — умножение двух матриц (матричное умножение)

operator== — сравнение матриц (поэлементно)

operator= — присваивание (глубокое копирование)

Требования:

Если размеры матриц не совпадают для +, -, == — выбросить исключение std::invalid_argument

Для умножения — количество столбцов первой должно равняться количеству строк второй, иначе исключение

Оператор = должен корректно работать при самоприсваивании

Эталон выполнения

Файлы matrix.hpp и matrix.cpp с добавленными операторами

Все операторы перегружены

Обработка исключений при некорректных размерах

Оператор присваивания с проверкой на самоприсваивание

Критерии оценки

Что проверяется	Балл
Оператор + перегружен	1

Оператор - перегружен	1
Оператор * перегружен	1
Оператор == перегружен	1
Оператор = перегружен (глубокое копирование)	1
Итого	5

Задание 3. Наследование и полиморфизм

Условие

Создайте иерархию классов для геометрических фигур:

Абстрактный базовый класс Shape с чисто виртуальными методами:

double area() const = 0 — площадь

double perimeter() const = 0 — периметр

void print() const = 0 — вывод информации

Класс Rectangle (прямоугольник), наследник Shape:

Поля: width, height

Конструктор с параметрами

Реализация методов area(), perimeter(), print()

Класс Circle (окружность), наследник Shape:

Поля: radius

Конструктор с параметрами

Реализация методов area(), perimeter(), print()

Напишите функцию print_shape_info, которая принимает указатель на Shape и вызывает его метод print().

Эталон выполнения

Файлы shape.hpp, rectangle.hpp, circle.hpp

Базовый класс с чисто виртуальными методами

Классы-наследники с корректными формулами

Полиморфное поведение (через указатель на базовый класс)

Критерии оценки

Что проверяется	Балл
Абстрактный базовый класс Shape создан	1
Класс Rectangle реализован (наследование, методы)	1
Класс Circle реализован (наследование, методы)	1
Формулы площади и периметра верны	1
Функция print_shape_info демонстрирует полиморфизм	1
Итого	5

Задание 4. Шаблоны (аналог CPP2 — контейнеры)

Условие

Реализуйте шаблонный класс MyVector<T>, аналогичный std::vector.

Требования:

Шаблонный класс с параметром типа T

Закрытые поля: указатель на массив T*, размер (size), вместимость (capacity)

Конструктор по умолчанию

Конструктор с параметром size (создаёт массив заданного размера)

Деструктор

Метод push_back(const T& value) — добавляет элемент в конец (при необходимости увеличивает capacity в 2 раза)

Метод pop_back() — удаляет последний элемент

Метод size() — возвращает текущий размер

Метод at(int index) — доступ по индексу с проверкой границ

Оператор [] — доступ по индексу без проверки границ

Эталон выполнения

Файл my_vector.hpp (шаблонный класс)

Все методы реализованы

Управление памятью корректное (выделение, перевыделение, освобождение)

Шаблон работает с разными типами (int, double, string)

Критерии оценки

Что проверяется	Балл
Шаблонный класс определён	1
Конструкторы и деструктор	1
Метод push_back с увеличение capacity	1
Методы pop_back, size	1
Доступ по индексу (at с проверкой, [] без проверки)	1
Итого	5

Задание 5. Обработка исключений + умные указатели

Условие

Создайте класс SafeArray, который представляет массив фиксированного размера с безопасным доступом.

Требования:

Шаблонный класс SafeArray<T>

Конструктор принимает размер массива

Оператор [] — выбрасывает исключение std::out_of_range при выходе за границы

Метод size() — возвращает размер

Внутри для хранения данных использовать std::unique_ptr<T[]>

В конструкторе, если размер <= 0, выбросить std::invalid_argument

Эталон выполнения

Файл safe_array.hpp (шаблонный класс)

Использование std::unique_ptr для управления памятью

Проверка границ с исключениями

Исключение при некорректном размере

Критерии оценки

Что проверяется	Балл
Шаблонный класс определён	1
Использование std::unique_ptr для данных	1
Проверка границ в операторе [] с исключением	1
Проверка размера в конструкторе	1
Метод size реализован	1
Итого	5

Задание 6. Паттерн проектирования (аналог СРР3/СРР4)

Условие

Реализуйте паттерн «Одиночка» (Singleton) для класса Logger, который занимается записью сообщений в файл.

Требования:

Класс Logger имеет приватный конструктор

Статический метод getInstance() возвращает единственный экземпляр

Метод log(const std::string& message) — записывает сообщение в файл log.txt с временной меткой

Запрещено копирование и присваивание (удалить конструктор копирования и оператор присваивания)

Потокобезопасность не требуется (для упрощения)

Временная метка — текущее время в формате [ГГГГ-ММ-ДД ЧЧ:ММ:СС].

Эталон выполнения
Файлы logger.hpp, logger.cpp
Класс Logger с приватным конструктором
Статический метод getInstance
Метод log с записью в файл
Запрет копирования
Критерии оценки

Что проверяется	Балл
Приватный конструктор	1
Статический метод getInstance (один экземпляр)	1
Метод log записывает в файл с временной меткой	1
Запрещены копирование и присваивание	1
Код компилируется и работает	1
Итого	5

Модуль 4. Современная прикладная разработка

Задание 1. Основы языка
Условие
Напишите программу, которая:
Принимает из командной строки или стандартного ввода целое число N
Создаёт массив (или список) из N случайных целых чисел в диапазоне от 1 до 100
Выводит массив на экран
Находит и выводит:
Минимальный элемент
Максимальный элемент
Сумму всех элементов
Среднее арифметическое
Требования: использовать стандартные конструкции языка (циклы, массивы/списки, функции).
Пример работы (для Python)
text
Введите N: 5
Массив: [23, 45, 12, 78, 34]
Минимум: 12
Максимум: 78
Сумма: 192
Среднее: 38.4

Эталон выполнения
Исходный код программы
Программа корректно обрабатывает ввод
Массив заполняется случайными числами
Все вычисления выполняются без использования встроенных агрегирующих функций (min, max, sum разрешены, но если их нет — плюс)
Вывод соответствует формату
Критерии оценки

Что проверяется	Балл
Корректный ввод N	1
Заполнение массива случайными числами	1
Нахождение минимума и максимума	1
Нахождение суммы и среднего	1
Корректный вывод	1

Итого	5

Задание 2. Объектно-ориентированное программирование
Условие
Создайте класс BankAccount, представляющий банковский счёт.
Требования:
Поля:

accountNumber (строка, генерируется автоматически при создании)
ownerName (строка, задаётся при создании)
balance (вещественное число, по умолчанию 0)
Методы:
deposit(amount) — пополнение счёта (сумма должна быть положительной)
withdraw(amount) — снятие со счёта (нельзя снять больше, чем есть на счёте)
getBalance() — возвращает текущий баланс
getAccountInfo() — возвращает строку с информацией о счёте
При попытке снять больше доступной суммы — выбросить исключение или вернуть ошибку.

Эталон выполнения
Исходный код класса
Демонстрация работы в main или тестах
Инкапсуляция соблюдена (поля приватные)
При попытке снять больше — ошибка обработана
Критерии оценки

Что проверяется	Балл
Класс создан, поля приватные	1
Конструктор с параметром ownerName	1
Метод deposit работает корректно	1
Метод withdraw работает с проверкой	1
Методы getBalance и getAccountInfo	1
Итого	5

Задание 3. Функциональное программирование
Условие
Напишите программу, которая:
Создаёт список строк: ["apple", "banana", "cherry", "date", "elderberry", "fig", "grape"]
С помощью функций высшего порядка (map/filter/reduce или аналоги):
Отфильтровывает строки длиной менее 5 символов
Преобразует оставшиеся строки в верхний регистр
Сортирует их по алфавиту
Объединяет в одну строку через запятую
Запрещено использовать явные циклы (for, while). Только функции высшего порядка и лямбда-выражения.
Пример результата
text
BANANA, CHERRY, ELDERBERRY, GRAPE

Эталон выполнения
Исходный код программы
Используются функции высшего порядка (map, filter, reduce / LINQ / stream / list comprehensions с условием и т.д.)
Нет явных циклов

Результат соответствует ожидаемому

Критерии оценки

Что проверяется	Балл
Фильтрация строк (длина ≥ 5)	1
Преобразование в верхний регистр	1
Сортировка по алфавиту	1
Объединение через запятую	1
Отсутствие явных циклов	1
Итого	5

Задание 4. Консольная игра (Rogue-like) с curses

Условие

Разработайте консольную игру «Лабиринт» с использованием библиотеки curses (или её аналога для выбранного языка: dotnet-curses для C#, JCurses для Java, blessed для JS, curses для Python, cinterop для Kotlin, vapor для Swift, go-curses для Go).

Игровая механика:

Игровое поле — лабиринт 10×10

Стены обозначаются #, проходы — пробел

Игрок обозначается @, выход — X

Управление стрелками (вверх, вниз, влево, вправо)

При достижении выхода — победа, выход из игры

При попытке врезаться в стену — игрок не двигается

Требования:

Использовать curses для отрисовки

Неблокирующий ввод (игра реагирует на клавиши в реальном времени)

Эталон выполнения

Исходный код игры

Корректная инициализация curses

Отображение лабиринта, игрока, выхода

Управление стрелками

Сообщение о победе при достижении выхода

Критерии оценки

Что проверяется	Балл
Лабиринт отображается корректно	1
Игрок и выход отображаются	1
Управление стрелками работает	1
Обработка столкновений со стенами	1
Победа при достижении выхода	1
Итого	5

Задание 5. Веб-приложение (CRUD + база данных)

Условие

Создайте веб-приложение с использованием выбранного фреймворка:

Язык Фреймворк

C# ASP.NET Core

Java Spring Boot

JavaScript Nest.js

Kotlin Ktor

Python Flask

Swift Vapor

Go Gin / Echo / net/http

Функциональность:

Модель Task (id, title, description, completed, created_at)

REST API эндпоинты:

GET /tasks — получить все задачи
GET /tasks/{id} — получить задачу по id
POST /tasks — создать новую задачу
PUT /tasks/{id} — обновить задачу
DELETE /tasks/{id} — удалить задачу
Данные хранятся в базе данных (любой СУБД на выбор: SQLite, PostgreSQL, MySQL)
Использовать ORM (Entity Framework, Spring Data JPA, TypeORM, Exposed, SQLAlchemy, Fluent, GORM)

Эталон выполнения
Исходный код приложения
Конфигурация подключения к БД
Миграции (если используются)
Все CRUD-эндпоинты реализованы и работают
Критерии оценки

Что проверяется	Балл
Модель Task создана	1
GET-эндпоинты работают	1
POST-эндпоинт работает	1
PUT-эндпоинт работает	1
DELETE-эндпоинт работает	1
Итого	5

Задание 6. JWT-авторизация + BrickGame v3.0 (обратная совместимость)
Условие
Разработайте веб-сервер (API) для игры BrickGame, который поддерживает:
JWT-авторизацию:
Регистрация пользователя (POST /register)
Вход (POST /login) — возвращает JWT-токен
Защищённые эндпоинты требуют валидный JWT
Игровую логику BrickGame (тетрис-лайк) на сервере или клиенте (по выбору):
Хранение рекордов пользователей в БД
Получение рекорда (GET /record — защищённый)
Сохранение нового рекорда (POST /record — защищённый)
Обратную совместимость:
API имеет две версии: v1 (старая, без JWT) и v2 (новая, с JWT)
Клиенты v1 продолжают работать без авторизации
Клиенты v2 используют JWT

Эталон выполнения
Исходный код сервера
Эндпоинты /register, /login с JWT
Защищённые эндпоинты с проверкой токена
Две версии API (v1 и v2)
Сохранение и получение рекордов из БД
Критерии оценки

Что проверяется	Балл
Регистрация и вход с JWT	1
Защищённые эндпоинты работают	1
Две версии API (обратная совместимость)	1
Сохранение рекордов в БД	1
Получение рекордов из БД	1
Итого	5

Модуль 5. Инфраструктурные решения и управление данными

Задание 1. Установка и настройка Linux

Условие

Установите Linux (Ubuntu 22.04 / аналогичный дистрибутив) на виртуальную машину и выполните базовую настройку.

Требования:

Установить систему с минимальной конфигурацией (без графического окружения или с базовым)

Создать пользователя student с паролем

Настроить sudo для пользователя student без запроса пароля

Обновить систему через пакетный менеджер (apt update && apt upgrade)

Установить пакеты: git, curl, htop, nano (или vim)

Настроить SSH-сервер: установить, запустить, добавить свой публичный ключ для доступа без пароля

Эталон выполнения

Скриншот терминала с выводом uname -a

Скриншот с выводом sudo -l (показывает права без пароля)

Скриншот с подключением по SSH через ключ

Файл /etc/hostname (содержимое)

Чек-лист со всеми выполненными пунктами

Критерии оценки

Что проверяется	Балл
Система установлена	1
Пользователь student создан, sudo настроен	1
Система обновлена, пакеты установлены	1
SSH-сервер настроен, ключ добавлен	1
Оформление отчёта (все скриншоты и файлы)	1
Итого	5

Задание 2. Настройка сети в Linux

Условие

На виртуальной машине с Linux выполните настройку сети.

Требования:

Определить и записать в отчёт:

IP-адрес виртуальной машины

MAC-адрес сетевого интерфейса

Шлюз по умолчанию

DNS-серверы

Пропинговать 3 внешних хоста (8.8.8.8, ya.ru, любой свой)

Настроить статический IP вместо DHCP (в конфигурации Netplan или interfaces)

Добавить в /etc/hosts запись: 127.0.0.1 myhost.local

Проверить маршрут до ya.ru с помощью traceroute (или tracepath)

Эталон выполнения

Вывод ip a и ip r (скриншот или текст)

Вывод ping -c 4 ya.ru

Файл конфигурации сети (Netplan или interfaces) после настройки статического IP

Вывод traceroute ya.ru (первые 10 хопов)

Пояснение ключевых параметров (IP, маска, шлюз, DNS)

Критерии оценки

Что проверяется	Балл

Определены сетевые параметры	1
Ping до внешних хостов работает	1
Статический IP настроен	1
Запись в /etc/hosts добавлена	1
traceroute выполнен	1
Итого	5

Задание 3. Docker и CI/CD

Условие

Разработайте простой CI/CD-пайплайн для одного из проектов (например, SimpleBashUtils или любой другой утилиты на C).

Требования:

Написать Dockerfile, который:

Создаёт образ на основе ubuntu:latest

Устанавливает gcc, make, git

Копирует исходный код проекта

Выполняет сборку (make)

Запускает тесты

Написать конфигурацию CI/CD (GitLab CI или GitHub Actions):

build — сборка Docker-образа

test — запуск тестов внутри контейнера

Загрузить образ в реестр (Docker Hub / GitHub Container Registry)

В отчёте описать этапы пайплайна и приложить скриншоты

Эталон выполнения

Файл Dockerfile

Файл .gitlab-ci.yml или .github/workflows/ci.yml

Скриншот успешного выполнения пайплайна

Ссылка на образ в реестре (или скриншот)

Инструкция по сборке и запуску

Критерии оценки

Что проверяется	Балл
Dockerfile корректен, образ собирается	1
CI/CD конфигурация написана	1
Пайплайн проходит успешно (build + test)	1
Образ загружен в реестр	1
Отчёт с описанием и скриншотами	1
Итого	5

Задание 4. SQL: создание таблиц и вставка данных

Условие

Работа с базой данных (PostgreSQL / SQLite / MySQL).

Часть 1. Создание таблиц

Создайте таблицы для учёта студентов и их оценок:

sql

-- Таблица students

-- id (PRIMARY KEY), name (TEXT), email (TEXT UNIQUE), enrolled_at (DATE)

-- Таблица courses

-- id (PRIMARY KEY), title (TEXT), hours (INTEGER)

-- Таблица grades

-- id (PRIMARY KEY), student_id (FOREIGN KEY → students.id),

-- course_id (FOREIGN KEY → courses.id), grade (INTEGER), date (DATE)

Часть 2. Вставка данных

Вставьте:

3 студентов
2 курса
5 оценок (у каждого студента хотя бы по одной)

Эталон выполнения
SQL-скрипт создания таблиц (CREATE TABLE)
SQL-скрипт вставки данных (INSERT)
Скриншот результата SELECT * FROM students;, SELECT * FROM courses;, SELECT * FROM grades;
Критерии оценки

Что проверяется	Балл
Таблица students создана с ограничениями	1
Таблица courses создана	1
Таблица grades создана с внешними ключами	1
Вставлены 3+ студента, 2+ курса, 5+ оценок	1
Все данные корректны (внешние ключи ссылаются на существующие записи)	1
Итого	5

Задание 5. SQL: выборка, фильтрация, группировка, JOIN
Условие
На основе таблиц из задания 4 напишите SQL-запросы:
Вывести всех студентов, зачисленных после 2024-01-01 (если дат нет — можно добавить)
Вывести список курсов с количеством часов больше 30
Вывести среднюю оценку каждого студента (имя + средний балл)
Вывести список студентов, у которых есть хотя бы одна оценка ниже 3 (или ниже 60, в зависимости от шкалы)
Вывести для каждого курса: название курса, максимальную оценку, минимальную оценку, количество студентов
Требования: использовать WHERE, GROUP BY, AVG, MIN, MAX, COUNT, JOIN.

Эталон выполнения
5 SQL-запросов (текст)
Скриншоты результатов выполнения каждого запроса
Критерии оценки

Что проверяется	Балл
Запрос 1 (фильтрация по дате)	1
Запрос 2 (фильтрация по часам)	1
Запрос 3 (средняя оценка студента)	1
Запрос 4 (оценки ниже порога)	1
Запрос 5 (агрегации по курсам)	1
Итого	5

Задание 6. SQL: обновление и удаление данных
Условие
На основе таблиц из задания 4 выполните операции модификации данных:
UPDATE: увеличить количество часов всех курсов на 10%
UPDATE: студенту с id = 1 изменить email на newemail@example.com
DELETE: удалить все оценки ниже 3 (или ниже 60)
DELETE: удалить студента с id = 3 (предварительно удалив его оценки, используя ON DELETE CASCADE или отдельный DELETE)
SELECT после каждого изменения, чтобы убедиться в корректности
Требования: показать SQL-запросы и результаты до/после.

Эталон выполнения

SQL-запросы на UPDATE и DELETE (текст)

Скриншоты SELECT до и после каждого изменения

Критерии оценки

Что проверяется	Балл
UPDATE часов курсов (+10%)	1
UPDATE email студента	1
DELETE низких оценок	1
DELETE студента (с каскадом или предварительным удалением оценок)	1
Результаты подтверждены SELECT	1
Итого	5

Модуль 6. Старт бизнеса

Задание 1. Что такое Lean Startup

Условие

Выберите любую бизнес-идею (реальную или учебную). Опишите её в 2–3 предложениях.

Требования:

Сформулируйте гипотезу ценности (предположение о том, какую проблему решает продукт и для кого)

Сформулируйте гипотезу роста (предположение о том, как продукт будет привлекать новых пользователей)

Опишите MVP (минимально жизнеспособный продукт) — какие минимальные функции нужны для проверки гипотез

Постройте цикл Build → Measure → Learn для этого MVP (что делаем → как измеряем → что узнаём)

Эталон выполнения

Текстовый документ (или слайд) с описанием бизнес-идеи

Две гипотезы (ценность и рост)

Описание MVP (3–5 пунктов, что входит)

Схема или текст цикла Build → Measure → Learn

Критерии оценки

Что проверяется	Балл
Бизнес-идея описана чётко	1
Гипотеза ценности сформулирована корректно	1
Гипотеза роста сформулирована корректно	1
MVP описан (минимальный набор функций)	1
Цикл Build → Measure → Learn проработан	1
Итого	5

Задание 2. Как быстро тестировать гипотезы с помощью Custdev

Условие

Для той же бизнес-идеи (из задания 1) разработайте план Custdev-интервью.

Требования:

Сформулируйте 3 гипотезы, которые нужно проверить через Custdev (например, о проблеме, о решении, о цене)

Напишите скрипт интервью (10–15 вопросов). Вопросы должны быть открытыми, не наводящими, без продажи

Опишите портрет респондента (кто целевая аудитория, где её искать, критерии отбора)

Укажите, сколько интервью нужно провести для проверки гипотез (минимум 5–10)

Опишите, как будете анализировать результаты (выявление паттернов, подтверждение/опровержение гипотез)

Эталон выполнения

Текстовый документ с тремя гипотезами

Скрипт интервью (10–15 вопросов)

Портрет респондента (3–5 критериев)

План анализа результатов

Критерии оценки

Что проверяется	Балл
Три гипотезы сформулированы (проблема, решение, цена)	1
Скрипт интервью готов (вопросы открытые, ненаводящие)	1
Портрет респондента описан	1
План по количеству интервью указан	1
Метод анализа результатов описан	1
Итого	5

Задание 3. Как оценивать рынок и избежать типичных ошибок

Условие

Для той же бизнес-идеи (из задания 1) проведите оценку рынка.

Требования:

Рассчитайте TAM (Total Addressable Market) — общий объём рынка для вашего продукта

Рассчитайте SAM (Serviceable Available Market) — доступный сегмент рынка

Рассчитайте SOM (Serviceable Obtainable Market) — реально достижимая доля (например, 5–10% от SAM)

Укажите источники данных для расчётов (статистика, отчёты, аналитика)

Опишите 3 типичные ошибки при оценке рынка (и как вы их избежали в своём расчёте)

Эталон выполнения

Таблица с цифрами TAM, SAM, SOM (в деньгах или единицах)

Источники данных (ссылки или названия отчётов)

Список 3 ошибок с пояснением, как избежали

Критерии оценки

Что проверяется	Балл
TAM рассчитан (с обоснованием)	1
SAM рассчитан (с обоснованием)	1
SOM рассчитан (с обоснованием)	1
Источники данных указаны	1
3 типичные ошибки описаны	1
Итого	5

Задание 4. Как масштабировать бизнес за пределы региона

Условие

Для той же бизнес-идеи (из задания 1) разработайте стратегию масштабирования.

Требования:

Выберите один регион для первой экспансии (соседний город, страну или международный рынок)

Обоснуйте выбор региона по 5 критериям (размер рынка, близость, язык/культура, конкуренция, регуляторика)

Опишите адаптацию продукта под новый регион (язык, функциональность,

упаковка, цена)

Составьте roadmap масштабирования по шагам с указанием сроков (первые 3–6 месяцев)

Опишите риски масштабирования (минимум 3) и способы их снижения

Эталон выполнения

Название выбранного региона

Обоснование по 5 критериям (таблица или текст)

Список адаптаций продукта

Дорожная карта (пошагово, с датами)

Риски и меры снижения

Критерии оценки

Что проверяется	Балл
Регион выбран и обоснован	1
5 критериев оценки региона применены	1
Адаптация продукта описана	1
Roadmap составлен	1
Риски и снижения описаны	1
Итого	5

Задание 5. Как начать бизнес с франшизой

Условие

Проведите анализ франшизы (выберите реальную или гипотетическую).

Требования:

Выберите 3 франшизы из интересующей ниши (можно реальные из открытых каталогов)

Составьте сравнительную таблицу по параметрам:

Паушальный взнос

Роялти (процент или фиксированная сумма)

Инвестиции (стартовые)

Срок окупаемости (по данным франчайзора)

Обучение и поддержка (есть/нет, что входит)

Рассчитайте примерную прибыль и срок окупаемости для одной из франшиз

Составьте чек-лист из 10 вопросов для проверки франчайзора (что спросить перед покупкой)

Сделайте вывод: в каком случае франшиза выгоднее открытия бизнеса «с нуля» (2–3 аргумента)

Эталон выполнения

Сравнительная таблица на 3 франшизы

Расчёт окупаемости (по одной франшизе)

Чек-лист из 10 вопросов

Вывод (2–3 аргумента)

Критерии оценки

Что проверяется	Балл
3 франшизы выбраны, таблица составлена	1
Расчёт окупаемости выполнен	1
Чек-лист из 10 вопросов готов	1
Вывод обоснован (2–3 аргумента)	1
Все разделы оформлены аккуратно	1
Итого	5

Модуль 7. Рост бизнеса

Задание 1. Продуктовый подход и работа с гипотезами

Условие

Для вашего бизнес-проекта (или выбранного учебного проекта) выполните продуктовый анализ.

Требования:

Сформулируйте 5–7 гипотез по структуре: «Если мы сделаем X, то получим Y, потому что Z»

Проведите приоритизацию гипотез методом ICE (Impact, Confidence, Ease) или RICE (Reach, Impact, Confidence, Effort) — заполните таблицу с баллами

Выберите топ-3 гипотезы для проверки

Для одной гипотезы (на выбор) опишите эксперимент: что делаем, как измеряем, какой результат считаем успехом (метрики)

Назовите 3 типа метрик, которые нужно отслеживать при проверке гипотез (например, качественные, количественные, продуктовые)

Эталон выполнения

Список из 5–7 гипотез

Таблица приоритизации (ICE или RICE) с баллами

Топ-3 гипотезы

Описание эксперимента для одной гипотезы (шаги, метрики успеха)

3 типа метрик с пояснением

Критерии оценки

Что проверяется	Балл
5–7 гипотез сформулированы	1
Приоритизация выполнена (ICE/RICE)	1
Топ-3 гипотезы выбраны обоснованно	1
Эксперимент описан (что, как, метрики успеха)	1
3 типа метрик названы	1
Итого	5

Задание 2. Custdev и концепция Jobs to be Done (JTBD)

Условие

Для вашего бизнес-проекта примените концепцию JTBD.

Требования:

Опишите 3 «работы» (jobs) , которые клиент «нанимает» ваш продукт выполнить (функциональные, эмоциональные, социальные)

Для каждой работы опишите: ситуацию (контекст), мотивацию (почему это важно), ожидаемый результат, альтернативы (как клиент решает проблему сейчас)

Напишите скрипт JTBD-интервью (5–7 вопросов), направленный на выявление «работ»

Объясните, чем JTBD отличается от Custdev (традиционного подхода) — 2–3 ключевых отличия

Укажите, как результаты JTBD-анализа повлияют на ваш продукт (что измените)

Эталон выполнения

3 работы (job) с описанием ситуации, мотивации, результата, альтернатив

Скрипт JTBD-интервью

Таблица или текст отличий JTBD от Custdev

План изменений в продукте

Критерии оценки

Что проверяется	Балл
3 работы описаны полно	1
По каждой работе — ситуация, мотивация, результат,	1

альтернативы	
Скрипт JTBD-интервью готов	1
Отличия JTBD от Custdev объяснены	1
Влияние на продукт описано	1
Итого	5

Задание 3. Анализ рынка

Условие

Проведите углублённый анализ рынка для вашего бизнес-проекта.

Требования:

Проведите PESTEL-анализ (Политические, Экономические, Социальные, Технологические, Экологические, Юридические факторы) — минимум по 2 фактора на категорию

Проведите анализ 5 сил Портера (угроза новых игроков, заменители, власть поставщиков, власть покупателей, интенсивность конкуренции)

Постройте карту конкурентов (две оси: например, «цена» и «качество/функциональность»). Разместите 5–10 конкурентов + свой продукт

Выделите 3 конкурентных преимущества вашего продукта (чем отличаетесь)

Сделайте SWOT-анализ (сильные и слабые стороны, возможности и угрозы)

Эталон выполнения

Таблица PESTEL (по 2+ фактора на категорию)

Таблица или текст по 5 силам Портера

Карта конкурентов (нарисованная или табличная)

3 конкурентных преимущества

SWOT-матрица

Критерии оценки

Что проверяется	Балл
PESTEL-анализ выполнен	1
5 сил Портера проанализированы	1
Карта конкурентов построена	1
3 конкурентных преимущества выделены	1
SWOT-анализ выполнен	1
Итого	5

Задание 4. Продажи через ценность (Value Selling)

Условие

Разработайте стратегию продаж через ценность для вашего бизнес-проекта.

Требования:

Заполните Value Proposition Canvas (канву ценностного предложения):

Профиль клиента: боли, выгоды, работы

Карта ценности: продукты/услуги, облегчение болей, создание выгод

Сформулируйте ценностное предложение (1–2 предложения) по шаблону: «Мы помогаем [клиент] [решить проблему] с помощью [уникальное решение], в отличие от [альтернатива]»

Напишите скрипт SPIN-вопросов (ситуация, проблема, извлечение, ценность) — 5–7 вопросов для разговора с клиентом

Подготовьте 3 возражения клиентов и ответы на них (с переводом из цены в ценность)

Напишите питч (1–2 минуты, текст) для презентации ценности вашего продукта

Эталон выполнения

Заполненная канва Value Proposition Canvas (таблица или схема)

Ценностное предложение (1–2 предложения)

Скрипт SPIN-вопросов (5–7)

3 возражения и ответы

Питч (текст)

Критерии оценки

Что проверяется	Балл
Value Proposition Canvas заполнена	1
Ценностное предложение сформулировано	1
SPIN-скрипт готов	1
3 возражения и ответы проработаны	1
Питч готов	1
Итого	5

Задание 5. Финансовые модели и юнит-экономика

Условие

Постройте финансовую модель и рассчитайте юнит-экономику для вашего бизнес-проекта.

Требования:

Рассчитайте CAC (Customer Acquisition Cost) — стоимость привлечения одного клиента (укажите допущения)

Рассчитайте LTV (Lifetime Value) — сколько денег приносит один клиент за всё время

Рассчитайте ARPU (Average Revenue Per User) — средняя выручка на одного клиента

Вычислите соотношение LTV / CAC (должно быть > 3 для здорового бизнеса)

Постройте финансовую модель на 12 месяцев (по месяцам):

Доходы (по источникам)

Расходы (постоянные и переменные)

Прибыль / убыток

Денежный поток (cash flow)

Точка безубыточности (break-even point)

Допущения: можно использовать реалистичные цифры (или учебные). Главное — логика расчёта.

Эталон выполнения

Расчёт CAC с допущениями

Расчёт LTV с допущениями

Расчёт ARPU

Соотношение LTV / CAC

Финансовая модель на 12 месяцев (таблица Excel / Google Sheets или в текстовом виде)

Критерии оценки

Что проверяется	Балл
CAC рассчитан (с допущениями)	1
LTV рассчитан (с допущениями)	1
ARPU рассчитан	1
LTV / CAC рассчитан и прокомментирован	1
Финансовая модель на 12 месяцев построена	1
Итого	5

Модуль 8. Продуктовый подход в бизнесе

Задание 1. Личное видение. Устойчивая мотивация основателя

Условие

Сформулируйте личное видение как основателя и проанализируйте свою мотивацию.

Требования:

Напишите личное видение (1–2 абзаца) на 5–10 лет: что вы создаёте, для кого, как меняете мир, какими будете через 5–10 лет

Сформулируйте миссию (одно предложение) — зачем существует ваш проект

Определите 3 источника вашей мотивации (внутренние: смысл, автономия, мастерство, признание, влияние)

Опишите 3 признака выгорания и 3 способа профилактики (личных, не общих)

Составьте список ритуалов (ежедневных или еженедельных) для поддержания энергии и фокуса

Эталон выполнения

Текст личного видения

Миссия (одно предложение)

3 источника мотивации

3 признака выгорания + 3 способа профилактики

Список ритуалов

Критерии оценки

Что проверяется	Балл
Личное видение сформулировано	1
Миссия сформулирована	1
3 источника мотивации названы	1
Выгорание: признаки + профилактика	1
Ритуалы описаны	1
Итого	5

Задание 2. Технологическое предпринимательство. Основные понятия

Условие

Проведите анализ вашего проекта (или учебного) через призму технологического предпринимательства.

Требования:

Дайте определение технологического стартапа (своими словами) и отличие от традиционного малого бизнеса (3 отличия)

Определите тип инновации вашего продукта (продуктовая, процессная, маркетинговая, организационная) — объясните почему

Назовите технологическую платформу вашего продукта (свою или партнёрскую) — например, iOS, Android, AWS, Telegram Bot API

Опишите жизненный цикл технологического стартапа (5 этапов: pre-seed → seed → early → growth → exit) с примером для своего проекта

Перечислите 3 способа защиты интеллектуальной собственности (патент, ноу-хау, коммерческая тайна) и укажите, какой подходит для вашего проекта

Эталон выполнения

Определение технологического стартапа + 3 отличия

Тип инновации с обоснованием

Технологическая платформа

Жизненный цикл стартапа с примерами

3 способа защиты ИС

Критерии оценки

Что проверяется	Балл
Определение и отличия	1
Тип инновации определён	1
Технологическая платформа названа	1

Жизненный цикл описан	1
Защита ИС описана	1
Итого	5

Задание 3. Стратегия подрыва отраслей с помощью технологий

Условие

Проанализируйте стратегию подрыва для вашего проекта.

Требования:

Объясните концепцию подрывных инноваций (Кристенсен): поддерживающие vs подрывные (2–3 предложения)

Назовите 3 примера известных подрывных инноваций (Netflix, Uber, Airbnb, Zoom и др.) — кратко, по 1–2 предложения на пример

Выберите лидера в вашей отрасли (конкретную компанию) и опишите 3 его «слепые зоны» (что он игнорирует: сегмент, функцию, цену, бизнес-модель)

Сформулируйте гипотезу подрыва для вашего проекта: с какого сегмента заходите (low-end или new-market), почему лидеры не ответят

Нарисуйте график «эволюция технологии vs требования рынка» (качественно: линия требований рынка, линия производительности продукта лидера, линия вашего стартапа)

Эталон выполнения

Объяснение подрывных инноваций

3 примера

Лидер в отрасли + 3 слепые зоны

Гипотеза подрыва

График (рисунок или текстовое описание)

Критерии оценки

Что проверяется	Балл
Концепция объяснена	1
3 примера названы	1
Лидер + 3 слепые зоны	1
Гипотеза подрыва сформулирована	1
График построен	1
Итого	5

Задание 4. Подрывные инновации или как стартапу победить лидера рынка

Условие

Разработайте конкретную стратегию, как стартап может победить лидера рынка.

Требования:

Выберите стратегию подрыва (low-end disruption — дешёвый сегмент, или new-market disruption — новый рынок) и обоснуйте

Опишите 3 этапа подрыва для вашего случая: вход → скрытый рост → переключение → вытеснение

Назовите 3 барьера для лидера (почему он не сможет ответить быстро) — например, бизнес-модель, каналы продаж, культура, KPI

Опишите точку входа: недопотреблённые клиенты / переобслуженные клиенты / новые сценарии

Напишите асимметричную войну: в чём ваше преимущество, которое лидер не может скопировать (2–3 пункта)

Эталон выполнения

Выбранный тип подрыва с обоснованием

4 этапа подрыва (описание)

3 барьера для лидера

Точка входа

Асимметричное преимущество (2–3 пункта)

Критерии оценки

Что проверяется	Балл
Тип подрыва выбран и обоснован	1
Этапы подрыва описаны	1
3 барьера для лидера названы	1
Точка входа определена	1
Асимметричное преимущество описано	1
Итого	5

Задание 5. Стратегия голубого океана

Условие

Примените стратегию голубого океана к вашему проекту.

Требования:

Объясните концепцию голубого океана (Ким и Моборн) vs красного океана (2–3 предложения)

Постройте стратегическую канву (график кривой ценности) для вашей отрасли:

Ось X: 5–7 факторов конкуренции (цена, качество, удобство, сервис, ассортимент, скорость, дизайн)

Ось Y: уровень предложения (низкий → высокий)

Нанесите кривые: лидеры отрасли (1–2), ваш продукт (текущий), ваш продукт (голубой океан)

Заполните матрицу «убрать — уменьшить — повысить — создать» (минимум 3 пункта в каждой категории)

Сформулируйте новую кривую ценности (чем вы отличаетесь от конкурентов)

Назовите один из 6 путей создания голубого океана, который вы использовали

Эталон выполнения

Объяснение стратегии голубого океана

Стратегическая канва (таблица или описание графика)

Матрица «убрать-уменьшить-повысить-создать»

Новая кривая ценности

Путь создания голубого океана

Критерии оценки

Что проверяется	Балл
Концепция объяснена	1
Стратегическая канва построена	1
Матрица заполнена (3+ в каждой)	1
Новая кривая ценности сформулирована	1
Путь создания указан	1
Итого	5

Задание 6. Факторы успеха стартапа. Команда основателей

Условие

Проведите анализ факторов успеха вашего стартапа и спроектируйте команду основателей.

Требования:

Назовите топ-5 причин провала стартапов по исследованиям CB Insights (списком)

Опишите идеальный профиль команды из 3 основателей для вашего проекта (роли: бизнес, технологии, продукт — или «хакер, бизнесмен, дизайнер»)

Рассчитайте vesting (4 года + cliff 1 год) для 3 сооснователей с разными долями (например, 40%, 30%, 30%)

Напишите 5 культурных ценностей вашего стартапа (по 1–2 предложения на каждую)

Составьте профиль первого сотрудника (кого наймёте первым, какие навыки, почему именно его)

Эталон выполнения

Топ-5 причин провала

Идеальный профиль команды (3 роли)

Расчёт vesting

5 культурных ценностей

Профиль первого сотрудника

Критерии оценки

Что проверяется	Балл
5 причин провала названы	1
Профиль команды описан	1
Vesting рассчитан	1
5 культурных ценностей сформулированы	1
Профиль первого сотрудника описан	1
Итого	5

Задание 7. Клиент и его потребности. Востребованный продукт

Условие

Проведите анализ соответствия вашего продукта рынку (Product-Market Fit).

Требования:

Объясните понятие Product-Market Fit (PMF) и перечислите 3 признака достижения PMF (органический рост, retention, сарафанное радио)

Постройте Customer Journey Map (CJM) для вашего продукта: 5–7 этапов от осознания проблемы до лояльности, для каждого этапа укажите действия клиента, боли и возможности

Заполните Kano-модель для вашего продукта: по 3 потребности каждого типа (базовые, линейные, восторгающие)

Проведите опрос на PMF (вопрос: «Как сильно вы расстроитесь, если продукт исчезнет?») — представьте гипотетические результаты (например, 45% сказали «очень расстроюсь»). Сделайте вывод о готовности к PMF

Напишите план из 5 итераций по достижению PMF (что будете менять в продукте и как проверять)

Эталон выполнения

Объяснение PMF, 3 признака

CJM (таблица или схема)

Kano-модель (9 потребностей: 3+3+3)

Опрос PMF с гипотетическими результатами и выводом

План из 5 итераций

Критерии оценки

Что проверяется	Балл
PMF объяснён, 3 признака названы	1
CJM построена (5–7 этапов)	1
Kano-модель заполнена (9 потребностей)	1
Опрос PMF и вывод	1
План из 5 итераций	1
Итого	5

Модуль 9. Азбука стартапа

Задание 1. Выбор организационно-правовой формы и расчёт налоговой нагрузки

Условие

Для вашего бизнес-проекта выберите организационно-правовую форму и рассчитайте налоговую нагрузку.

Требования:

Сравните 3 организационно-правовые формы (например, ИП, ООО, самозанятость) по 5 критериям:

Сложность регистрации

Ответственность (полная / в пределах уставного капитала)

Возможность вывести прибыль

Статус для контрагентов (корпоративный или нет)

Стоимость регистрации и обслуживания

Выберите оптимальную форму для вашего проекта и обоснуйте выбор

Сравните 3 налоговых режима (например, УСН 6% (доходы), УСН 15% (доходы минус расходы), НПД (самозанятость), ОСНО) по 4 критериям:

Ставка налога

Объект налогообложения

Ограничения (по доходам, численности сотрудников)

Сложность отчётности

Рассчитайте налоговую нагрузку для выбранного режима на примере:

Доходы за месяц: 1 000 000 руб.

Расходы (аренда, зарплата, материалы): 600 000 руб.

Покажите расчёт налога (цифры)

Сделайте вывод: какая форма и режим подходят для стартапа на ранней стадии

Эталон выполнения

Сравнительная таблица 3 ОПФ

Обоснование выбора

Сравнительная таблица 3 налоговых режимов

Расчёт налоговой нагрузки (с цифрами)

Вывод по стартапу

Критерии оценки

Что проверяется	Балл
Сравнение ОПФ (3 формы, 5 критериев)	1
Выбор и обоснование ОПФ	1
Сравнение налоговых режимов (3 режима, 4 критерия)	1
Расчёт налоговой нагрузки (цифры)	1
Вывод	1
Итого	5

Задание 2. Налоговая модель и управляемость бизнеса с первого дня

Условие

Разработайте налоговую модель для вашего стартапа с учётом управляемости бизнеса.

Требования:

Опишите налоговую модель вашего бизнеса:

Какие налоги платите (НДС, налог на прибыль/доходы, страховые взносы, НДФЛ)

Периодичность уплаты (ежемесячно, ежеквартально)

Кто отвечает за расчёт и уплату (вы сами, бухгалтер, аутсорсинг)

Опишите систему учёта с первого дня:

Как фиксируете доходы и расходы (таблица Excel, облачный сервис, бухгалтерская программа)

Как работаете с первичными документами (договоры, акты, счета, чеки)

Как храните документы (бумажные / электронные)

Составьте график налоговых платежей на первый квартал (по месяцам, с суммами — учебными)

Опишите 3 риска налоговых ошибок на старте и способы их предотвращения

Назовите 3 онлайн-сервиса для бухгалтерии стартапа (например, Тинькофф Бизнес, Мой налог, Контур.Эльба, 1С:Фреш)

Эталон выполнения

Налоговая модель (какие налоги, периодичность, ответственность)

Система учёта с первого дня

График налоговых платежей (3 месяца)

3 риска и их предотвращение

3 онлайн-сервиса

Критерии оценки

Что проверяется	Балл
Налоговая модель описана	1
Система учёта описана	1
График платежей составлен	1
3 риска и предотвращение	1
3 онлайн-сервиса названы	1
Итого	5

Задание 3. Финансовое управление бизнесом

Условие

Разработайте систему финансового управления для вашего стартапа.

Требования:

Составьте прогноз движения денежных средств (Cash Flow) на 6 месяцев (по месяцам):

Поступления от клиентов

Операционные расходы (аренда, зарплата, реклама, налоги)

Инвестиционные расходы (оборудование, ПО)

Сальдо на конец месяца

Рассчитайте cash runway (на сколько месяцев хватит денег) при текущем остатке, например, 2 000 000 руб.

Назовите 3 финансовых метрики для стартапа (не CAC/LTV, а другие, например, gross margin, burn rate, месячный рост выручки)

Составьте 3 сценария финансовой модели: пессимистичный, реалистичный, оптимистичный (отличаются по выручке и расходам) — таблица на 6 месяцев

Напишите чек-лист финансового контроля на каждый месяц (5 пунктов, что проверять и контролировать)

Эталон выполнения

Cash Flow на 6 месяцев (таблица)

Cash runway (расчёт)

3 финансовые метрики (с пояснением)

3 сценария (пессимистичный, реалистичный, оптимистичный)

Чек-лист финансового контроля (5 пунктов)

Критерии оценки

Что проверяется	Балл
Cash Flow на 6 месяцев	1
Cash runway рассчитан	1
3 финансовые метрики названы	1
3 сценария построены	1
Чек-лист финансового контроля	1

Итого	5

Задание 4. Масштабирование и защита бизнеса

Условие

Разработайте стратегию масштабирования и защиты вашего бизнеса.

Требования:

Опишите модель масштабирования: органический рост, франчайзинг, лицензирование, выход в новые регионы, партнёрства (выберите 1–2 и обоснуйте)

Назовите 3 юридических аспекта масштабирования (регистрация в новом регионе, налоги, защита ИС, договоры с партнёрами, трудовое законодательство)

Составьте карту рисков масштабирования (минимум 5 рисков) и для каждого — план смягчения

Опишите способы защиты бизнеса:

Интеллектуальная собственность (товарный знак, патент, авторское право)

Договоры с контрагентами (NDA, оферта, договор услуг)

Безопасность данных (152-ФЗ, хранение персональных данных)

Напишите план защиты бизнеса на первый год (пошагово, что сделать в первую очередь)

Эталон выполнения

Модель масштабирования с обоснованием

3 юридических аспекта

Карта рисков (5+ рисков + смягчение)

Способы защиты бизнеса

План защиты на первый год

Критерии оценки

Что проверяется	Балл
Модель масштабирования обоснована	1
3 юридических аспекта названы	1
Карта рисков (5+ рисков)	1
Способы защиты бизнеса описаны	1
План защиты на первый год	1
Итого	5

Пример оценочных материалов для итоговой аттестации

Часть 1. Теоретическая часть (письменные ответы)

Формат: слушатель письменно отвечает на 20 вопросов. Каждый ответ должен быть содержательным (3–5 предложений, чётко по существу). Максимум — 20 баллов (1 балл за каждый полный и правильный ответ).

Модуль 1. Основы IT-инфраструктуры

Вопрос 1. Какие основные команды Linux используются для навигации по файловой системе, просмотра содержимого каталогов и управления файлами? Опишите их назначение.

Вопрос 2. Что такое Git? Опишите базовый цикл работы с Git (init, add, commit, push) и объясните, зачем нужна каждая команда.

Вопрос 3. Что такое Docker? Для чего используются Dockerfile и команды docker build и docker run?

Модуль 2. Алгоритмизация и структурное программирование

Вопрос 4. Что такое указатель в языке C? Как происходит работа с динамической памятью (malloc, free)? Почему важно освобождать память?

Вопрос 5. Объясните, как работает пузырьковая сортировка (bubble sort). Какова её временная сложность в лучшем и худшем случаях?

Вопрос 6. Что такое рекурсия? Приведите пример рекурсивной функции и объясните, что такое «базовый случай» и почему он необходим.

Модуль 3. Объектно-ориентированное программирование

Вопрос 7. Объясните три основных принципа ООП: инкапсуляция, наследование, полиморфизм. Приведите примеры.

Вопрос 8. Что такое перегрузка операторов в C++? Для чего она используется? Приведите пример перегрузки оператора + для класса Matrix.

Вопрос 9. Что такое абстрактный класс и чисто виртуальные методы? Когда и зачем их используют?

Модуль 4. Современная прикладная разработка

Вопрос 10. Что такое JWT (JSON Web Token) и как он используется для авторизации в веб-приложениях? Опишите структуру токена.

Вопрос 11. Назовите и кратко охарактеризуйте основные парадигмы программирования: процедурную, объектно-ориентированную, функциональную. В чём их различия?

Модуль 5. Инфраструктурные решения и управление данными

Вопрос 12. Какие типы JOIN существуют в SQL? Объясните разницу между INNER JOIN, LEFT JOIN, RIGHT JOIN. Приведите примеры.

Вопрос 13. Что такое агрегационные функции в SQL (COUNT, SUM, AVG, MIN, MAX)? Для чего используется GROUP BY и HAVING?

Модуль 6. Старт бизнеса

Вопрос 14. Что такое Lean Startup? Объясните цикл Build → Measure → Learn и понятие MVP (Minimum Viable Product).

Вопрос 15. Что такое Custdev (Customer Development) и как с его помощью проверять гипотезы? Опишите основные принципы проведения интервью с потенциальными клиентами.

Модуль 7. Рост бизнеса

Вопрос 16. Что такое юнит-экономика? Объясните метрики CAC, LTV, ARPU. Какое соотношение LTV / CAC считается здоровым для бизнеса?

Вопрос 17. Что такое продуктовый подход и работа с гипотезами? Как приоритизировать гипотезы (например, методом ICE или RICE)?

Модуль 8. Продуктовый подход в бизнесе

Вопрос 18. Что такое стратегия голубого океана? Опишите инструменты: стратегическая канва и матрица «убрать — уменьшить — повысить — создать».

Вопрос 19. Что такое подрывные инновации (disruptive innovation) по Кристенсену? Приведите пример и объясните, как стартап может победить лидера рынка.

Модуль 9. Азбука стартапа

Вопрос 20. Какие основные налоговые режимы существуют для малого бизнеса (например, УСН 6%, УСН 15%, НПД)? В чём их различия и как выбрать подходящий для стартапа?

Часть 2. Практическая часть

Форма выполнения: письменная. Слушатель выполняет одно задание по выбору (5 вариантов). Код пишет самостоятельно. В эталоне — только описание требований и критерии.

Задания

Вариант А. Структурное программирование на C

Разработайте программу на языке C, которая:

Принимает из командной строки имя входного файла и имя выходного файла

Читает матрицу вещественных чисел из входного файла (формат: первая строка — rows cols, затем rows строк по cols чисел)

Вычисляет сумму всех элементов матрицы, находит минимальный и максимальный элемент

Записывает результат в выходной файл

Освобождает всю динамически выделенную память

Вариант Б. Объектно-ориентированное программирование на C++

Создайте шаблонный класс DynamicArray с методами: push_back, pop_back, size, at (с проверкой границ и выбросом исключения), оператор []. Продемонстрируйте работу всех методов в функции main.

Вариант В. Базы данных и SQL

Напишите 5 SQL-запросов для базы данных интернет-магазина (таблицы products и orders):

Товары с ценой от 1000 до 5000

Количество проданных единиц по каждому товару

Топ-3 самых продаваемых товара

Товары, которые ни разу не заказывались

Общая выручка за все время

Вариант Г. Прикладная разработка (язык по выбору)

Разработайте консольный Todo-менеджер на выбранном языке (Python, C#, Java, JavaScript, Go, Kotlin, Swift) с операциями:

Добавить задачу (название, описание, статус: новая / в работе / выполнена)

Показать все задачи

Показать задачи по статусу

Отметить задачу как выполненную

Удалить задачу

Сохранить задачи в файл

Загрузить задачи из файла

Используйте ООП (класс Task).

Вариант Д. DevOps и Linux

Опишите пошагово (с командами и пояснениями) выполнение следующих задач:

Установка Linux на виртуальную машину

Создание пользователя admin и добавление его в группу sudo

Настройка статического IP-адреса

Установка Docker

Запуск контейнера с Nginx (проброс порта 8080)

Написание Dockerfile для приложения, выводящего "Hello, DevOps!"

Вариант Е. Бизнес-план стартапа (предпринимательство)

Разработайте бизнес-план для технологического стартапа (идею выберите самостоятельно).

Требования:

Опишите бизнес-идею (продукт, целевая аудитория, проблема)

Сформулируйте гипотезу ценности и гипотезу роста

Опишите MVP (минимально жизнеспособный продукт)

Проведите анализ рынка (TAM, SAM, SOM с обоснованием)

Рассчитайте юнит-экономику (CAC, LTV, ARPU, соотношение LTV/CAC)

Опишите стратегию привлечения первых клиентов

Назовите 3 основных риска и способы их снижения

Формат сдачи: аналитический отчёт (до 5 страниц) или презентация (до 10 слайдов).

Вариант Ж. Стратегия стартапа (предпринимательство)

Выберите реальный стартап (или создайте свой) и разработайте стратегию развития.

Требования:

Краткое описание стартапа (продукт, рынок, конкуренты)

Примените стратегию голубого океана: заполните матрицу «убрать — уменьшить — повысить — создать»

Постройте стратегическую канву (кривую ценности) для вашего продукта и конкурентов

Сформулируйте стратегию подрыва (low-end или new-market disruption) — как победить лидера

Опишите факторы успеха стартапа (команда, продукт, рынок)

Составьте roadmap на первый год (по кварталам)

Формат сдачи: аналитический отчёт или презентация.

Вариант З. Франшиза и масштабирование (предпринимательство)

Проведите анализ открытия бизнеса по франшизе.

Требования:

Выберите 3 реальные франшизы из одной ниши (например, общепит, образование, услуги)

Составьте сравнительную таблицу (паушальный взнос, роялти, инвестиции, срок окупаемости)

Выберите лучшую франшизу и обоснуйте выбор

Рассчитайте примерную прибыль и точку безубыточности

Опишите план масштабирования: как открыть вторую и третью точку

Назовите 3 юридических аспекта, которые нужно проверить перед покупкой франшизы

Составьте чек-лист вопросов к франчайзору (минимум 10 вопросов)

Формат сдачи: аналитический отчёт с таблицами и расчётами.

Критерии оценивая и шкалы приведены в п.2.5.1

Промежуточная аттестация проводится в форме экзамена по модулю в виде выполнения практического здания по пройденным темам. Для успешного прохождения ПА слушатель выполняет по одной индивидуальной работе по каждому из модулей и размещает полный комплект материалов в ЛМС, согласно утверждённым критериям. Формат демонстрации не предусмотрен.

Критерии оценивания промежуточной аттестации:

- соответствие заданию модуля и полнота выполнения всех пунктов;
- корректность решений и валидность результатов;
- качество оформления;
- соблюдение академической добросовестности (оригинальность, корректные заимствования, отсутствие совпадений, не объяснённых ссылками);
- соблюдение сроков сдачи в ЛМС.

Шкалы оценивания:

Баллы/Критерии	**соответствие заданию модуля и полнота выполнения всех пунктов**	**корректность решений и валидность результатов;**	**качество оформления;**	**соблюдение академической добросовестности**	**соблюдение сроков сдачи в ЛМС.**

5	Выполнены все пункты задания модуля в полном объёме, нет пропусков, работа полностью соответствует заданию	Решения технически/логически верны, результаты воспроизводимы и обоснованы, обработка ошибок корректна	Структурировано, читаемо, единый стиль, код/текст отформатированы, есть комментарии/пояснения, файлы названы логично, материалы упакованы аккуратно	Работа полностью оригинальна (или все заимствования явно указаны ссылками), нет совпадений с другими работами, нет плагиата	--
4	Выполнены все основные пункты, но отсутствует 1 незначительный элемент / незначительная часть	Решения в целом верны, но есть 1–2 незначительные ошибки / неточности, не влияющие на основной результат	Хорошее оформление, но есть мелкие недочёты: местами нарушен стиль, не все файлы подписаны, отсутствуют отдельные комментарии	Единичные совпадения, объяснённые корректными ссылками; общая оригинальность высокая (>80%)	--
3	Выполнено 70–80% задания, отсутствуют 2–3 пункта или один значительный блок	Решения работают, но есть ошибки в краевых случаях / не обработаны исключения / результаты частично неверны	Среднее качество: структура есть, но оформление «грязное» (лишние файлы, разный стиль, мало комментариев)	Есть заимствования без ссылок (10–20% работы), или незначительные совпадения с другими слушателями	Работа сдана строго в установленный срок (включая все материалы)
2	Выполнено 50–69% задания, отсутствует более 3 пунктов или ключевая часть	Решения содержат системные ошибки, результаты невалидны или не воспроизводятся	Оформление плохое: трудно читать, нет структуры, неясно, что к чему относится	Значительные заимствования без ссылок (20–40%), или заметные совпадения с другими работами	Незначительная задержка (до 3 календарных дней)
1	Выполнено менее 50% задания, работа не соответствует заданию	Решения не работают, результаты отсутствуют или противоречивы	Оформление отсутствует (набор файлов без пояснений, нечитаемый код/текст)	Более 40% работы скопировано (без ссылок), или работа несамостоятельна	Задержка от 4 до 7 календарных дней
0	Работа не сдана или не относится к модулю	Решения не представлены или полностью неверны	Материалы не загружены в ЛМС или загружены с нарушением формата (не открываются)	Полный плагиат (работа целиком не своя, или копия другой работы)	Задержка более 7 календарных дней или работа не сдана

Итоговая шкала оценивания

Отлично	21–23 балла
Хорошо	17–20 баллов
Удовлетворительно	14–16 баллов
Неудовлетворительно	менее 14 баллов

Слушателям, получившие по результатам промежуточной аттестации оценку «неудовлетворительно» устанавливаются сроки повторной промежуточной аттестации. Если обучающийся не ликвидировал академическую задолженность при прохождении повторной промежуточной аттестации в первый раз, ему предоставляется возможность пройти повторную промежуточную аттестацию во второй раз.

Первая повторная промежуточная аттестация и вторая повторная промежуточная аттестация проводятся комиссией. Состав комиссии формируется департаментом образовательных программ.

Итоговая аттестация (ИА): в рамках итоговой аттестации проводятся итоговые экзамены на знание теоретической части и выполнение практических заданий.

Критерии оценивания:

Теоретическая часть:

– Правильность ответов — процент верных ответов от общего числа вопросов
– Полнота охвата — отвечены ли вопросы по всем ключевым темам модуля

Практическая часть:

– Соответствие заданию — выполнение всех пунктов задания
– Корректность решения — работоспособность, отсутствие ошибок
– Валидность результатов — результаты верны и воспроизводимы
– Качество кода/решения — читаемость, структура, комментарии, обработка ошибок
– Оформление — инструкции, логичная структура материалов

Общие требования (для теории и практики):

– Соблюдение формата — требования к объёму, времени, способу сдачи
– Академическая добросовестность — отсутствие плагиата, списывания, подсказок

Шкалы оценивания:

Баллы/Критерии	5	4	3	2	1	0
Правильность ответов	90–100% верных ответов	75–89% верных ответов	50–74% верных ответов	менее 50% верных ответеов	—	—
Полнота охвата	Ответы охватывают все ключевые темы курса	Ответы охватывают основные темы, но есть 1–2 пропущенных раздела	Ответы охватывают менее 70% ключевых тем	Ответы не охватывают ключевые темы или тест отсутствует	—	—
Соответствие заданию	Выполнены все пункты задания	Выполнены все основные, пропущен 1 незначительный	Выполнено 70–80%	Выполнено менее 70%	—	—
Корректность решения	Решение верно, ошибок нет	1–2 незначительные ошибки	Решение работает, но есть ошибки	Решение не работает или содержит системные ошибки	—	—
Валидность результатов	Результаты верны, воспроизводимы,	В целом верны, но 1–2 результата не обоснованы	Результаты частично верны или частично не воспроизводят	Результаты отсутствуют или неверны	—	—

	обоснованы		ся			
Качество реализации	Код/решение структурированы, читаемы, есть комментарии, обработка ошибок	Хорошо, но есть мелкие недочёты	Удовлетворительно, но есть «грязь»	Нечитаемо/не компилируемо	—	—
Оформление	Есть инструкция, логичные названия, аккуратная упаковка	Хорошо, но есть 1–2 недочёта	Средне: что-то есть, но неполно	Оформление отсутствует	—	—
Соблюдение формата	—	—		Существенные отклонения Минус 2 балла	Незначительные отклонения Минус 1 балл	Формат полностью соблюдён 0 баллов
Академическая добросовестность	—		Более 30% заимствований или копирование чужой работы – работа не принимается	Значительные заимствования (10–30%) Минус 2 балла	Незначительные заимствования (до 10%) Минус 1 балл	Ответ/решение оригинальны 0 баллов

Итоговая шкала оценивания

Отличо	45 – 50 баллов
Хорошо	35 – 44 балла
Удовлетворительно	25 – 34 балла
Неудовлетворительно	0 – 24 балла

Лица, не прошедшие итоговую аттестацию или получившие на итоговой аттестации неудовлетворительные результаты, вправе повторно пройти итоговую аттестацию в сроки, определяемые академией.