

РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ

Алгоритмы и структуры данных

Трудоемкость		Семестр	Вид контроля	Контактная работа, час.	Занятий лекц. типа, час.	Лаборат. занятий, час.	Практич. занятий, час.	Конс, час.	СРО, час.
зач. ед.	час.								
3.0	108.0	1	Дифференцированный зачет	70.4	32.0	32.0	0.0	0.0	37.6
3.0	108.0	ИТОГО		70.4	32.0	32.0	0.0	0.0	37.6

РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ

Разработана: Ходненко Иван Владимирович

1. ЦЕЛИ ОСВОЕНИЯ ДИСЦИПЛИНЫ

Целью освоения дисциплины является достижение следующих результатов обучения:

Знания	Умения	Навыки
Структуры данных, Numpy, Динамическое программирование, Графы, Алгоритмы сортировки, Решение задач и применение различных подходов к ним, blas, Алгоритмы обхода графа, Алгоритмы поиска пути в графе, Временная и пространственная сложность, NP-полнота, Разделяй и властвуй	Решение задач и применение различных подходов к ним, писать простые юнит-тесты для проверки функциональности программы, Подсчет памяти, Считать О-нотацию, Оптимизация алгоритмов по памяти, Оптимизация алгоритмов по времени, Решать NP-полные задачи, Оптимизировать решение NP-полных задач, Применять динамическое программирование, Применять подход разделяй и властвуй, Решать векторные задачи используя Numpy	C/C++, Numpy, Git, Отладка программ небольшого размера, писать простые юнит-тесты для проверки функциональности программы, mlpack, Векторного мышления

2. СТРУКТУРА И СОДЕРЖАНИЕ ДИСЦИПЛИНЫ

№ раздела	Наименование раздела дисциплины	Распределение часов по дисциплине, часы						
		Контакт ная работа	Занятия лекционн ого типа	Лаборато рные занятия	Практи ческие занятия	Консу льтац ии	СРО	Всего часов
1 семестр								
1	Вспоминание основ. Изучение C синтаксиса о основ автоматического тестирования	17.6	8.0	8.0	0.0	0.0	9.4	27.0
2	Базовые структуры данных. Асимптотика	17.6	8.0	8.0	0.0	0.0	9.4	27.0
3	Углубляемся в понимание и подсчет асимптотики	17.6	8.0	8.0	0.0	0.0	9.4	27.0
4	Стратегии упрощения трудных задач	17.6	8.0	8.0	0.0	0.0	9.4	27.0
ИТОГО:		70.4	32.0	32.0	0.0	0.0	37.6	108.0

СОДЕРЖАНИЕ РАЗДЕЛОВ

№ раздела	Наименование раздела дисциплины	Содержание
1 семестр		
1	Вспоминание основ.	Введение в курс. Знакомство.

	Изучение C синтаксиса о основ автоматического тестирования	Компилятор и интерпретатор. Основы C.
		Основа C синтаксиса
		Автоматическое тестирование
2	Базовые структуры данных. Асимптотика	Статичные структуры данных
		Динамические структуры данных
		Хеширование и хеш-таблицы. Хеш функции
		Асимптотика. Учимся считать время
3	Углубляемся в понимание и подсчет асимптотики	Алгоритмы сортировки. Сложность от $O(N)$ до $O(N^2)$
		Разделяй и властвуй
		Амортизационный анализ
		Сложные задачи. NP-полные задачи
4	Стратегии упрощения трудных задач	Динамическое программирование
		Рандомизированный и жадный подходы
		Матроиды 1
		Матроиды 2

3. УЧЕБНО-МЕТОДИЧЕСКОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ

Список литературы:

1. Павлов Л. А., Перова Н. В. Структуры и алгоритмы обработки данных — Издательство "Лань", 2021 — 256 с. — ISBN 978-5-8114-7259-8 — Текст : электронный // ЭБС Лань — URL: <https://e.lanbook.com/book/156929>

Иные ресурсы:

1. Чернышев, С.А. Алгоритмы и структуры данных на языке GO : учебник. - Москва : КноРус, 2024. - 353 с.
2. Скиена С. Алгоритмы. Руководство по разработке. — 2-е изд.: Пер. с англ. — СПб.: БХВ-Петербург, 2011. — 720 с.
3. Кормен Т., Лейзерсон Ч., Ривест Р., Штайн К. Алгоритмы. Построение и анализ: [пер. с англ.]. – Издательский дом Вильямс, 2009. - 1290 с.
4. Зингардо Д. Алгоритмы на практике. / Д. Зингардо. : Питер, 2023. - 432 с.

5. Бхаргава, А. Грокаем алгоритмы. Иллюстрированное пособие для программистов и любопытствующих / А. Бхаргава. СПб. : Питер, 2024. - 288 с.

4. ОЦЕНОЧНЫЕ СРЕДСТВА ДЛЯ ПРОВЕДЕНИЯ ТЕКУЩЕЙ АТТЕСТАЦИИ ПО ДИСЦИПЛИНЕ

Порядок оценки освоения обучающимися учебного материала определяется содержанием следующих разделов дисциплины:

№ п/п	Наименование раздела дисциплины (модуля)	Оценочные средства контроля успеваемости	Тип оценочного средства
1 семестр			
1	Вспоминание основ. Изучение C синтаксиса о основ автоматического тестирования	Написание тестов для реализованных ранее функций	Лабораторная работа
		Изучение основ C	Лабораторная работа
2	Базовые структуры данных. Асимптотика	Структуры данных	Лабораторная работа
		Хеш-таблицы	Лабораторная работа
		Подсчет асимптотики	Лабораторная работа
3	Углубляемся в понимание и подсчет асимптотики	Алгоритмы сортировки	Лабораторная работа
		Разделяй и властвуй	Лабораторная работа
		Амортизационный анализ	Лабораторная работа
		NP-полные задачи	Лабораторная работа
4	Стратегии упрощения трудных задач	Динамическое программирование	Лабораторная работа
		Жадные алгоритмы	Лабораторная работа

**5. ТИПОВЫЕ КОНТРОЛЬНЫЕ ЗАДАНИЯ ИЛИ ИНЫЕ МАТЕРИАЛЫ,
НЕОБХОДИМЫЕ ДЛЯ ОЦЕНКИ ДОСТИЖЕНИЯ ЗАПЛАНИРОВАННЫХ
РЕЗУЛЬТАТОВ ОБУЧЕНИЯ**

№ оценочно го средства	Название	Тип	Ключевая точка	Мин. балл	Макс. балл	Оценивает раздел(-ы)
1 семестр						
1	Изучение основ С	Лабораторная работа	нет	0	5	Вспоминание основ. Изучение С синтаксиса о основ автоматического тестирования
2	Написание тестов для реализованных ранее функций	Лабораторная работа	нет	0	5	Вспоминание основ. Изучение С синтаксиса о основ автоматического тестирования
3	Структуры данных	Лабораторная работа	нет	0	7	Базовые структуры данных. Асимптотика
4	Хеш-таблицы	Лабораторная работа	нет	0	8	Базовые структуры данных. Асимптотика
5	Подсчет асимптотики	Лабораторная работа	нет	0	5	Базовые структуры данных. Асимптотика
6	Алгоритмы сортировки	Лабораторная работа	нет	0	5	Углубляемся в понимание и подсчет асимптотики
7	Разделяй и властвуй	Лабораторная работа	нет	0	5	Углубляемся в понимание и подсчет асимптотики
8	Амортизационный анализ	Лабораторная работа	нет	0	5	Углубляемся в понимание и подсчет асимптотики
9	NP-полные задачи	Лабораторная работа	нет	0	5	Углубляемся в понимание и подсчет асимптотики
10	Динамическое программирование	Лабораторная работа	нет	0	5	Стратегии упрощения трудных задач
11	Жадные алгоритмы	Лабораторная работа	нет	0	5	Стратегии упрощения трудных задач
12	Дифференцированный зачет	Дифференцирован ный зачет	-	0	40	Все разделы

Оценочные средства 1 семестра

[1]

Изучение основ C

Лабораторная работа

Описание:

Цель лабораторной работы – решить все выданные студенту практические задачи.

Задача считается решенной, если она:

- * Прошла автоматическое тестирование без ошибок.
- * Содержит программный код, соответствующий правилам оформления.

За решенную лабораторную работу студент может получить 5 баллов. Однако во время сдачи лабораторной работы студент может получить следующие штрафы:

- * До 100% штрафа за попытку сдачи чужого кода или кода, сгенерированного LLM моделью.
- * До 100% штрафа за просрочку.
- * До 50% если студент игнорирует правила написания кода и комментарии от ментора.

Предельные сроки сдачи лабораторной работы:

- * Если студент не сдал за 1 месяц - штраф 50%.
- * Если студент не сдал за 1 месяц и 2 недели - штраф 75%.
- * Если студент не сдал за 2 месяца - штраф 100%.

Пример домашнего задания.

Решить набор задач на изучение основ C: условия, циклы, функции. Всего 15 задач, здесь будет приведено только 3 в качестве примера

Задача 1. Чётное или нечётное

На вход: число int

На выход: boolean

Описание: Если число четное - то программа должна вернуть True, если нечетное - то программа должна вернуть False

Задача 2. Посчитать евклидово расстояние между 2-мя точками

На вход: 4 float числа, первые два числа обозначают 1-ю точку, вторые два обозначают 2-ю точку

На выход: float расстояние между 2-мя точками

Описание: Программа должна принимать 2 точки на вход, и считать между ними евклидово расстояния

Задача 3. Високосный год

На вход: Положительное число больше 0

На выход: boolean

Описание: Если год високосный – программа должна вернуть True, иначе False

[2]

Написание тестов для реализованных ранее функций

Лабораторная работа

Описание:

Цель лабораторной работы – решить выданные студенту практические задачи.

Задача считается решенной, если она:

- * Содержит программный код, соответствующий правилам оформления.

За решенную лабораторную работу студент может получить 5 баллов. Однако во время сдачи лабораторной работы студент может получить следующие штрафы:

- * До 100% штрафа за попытку сдачи чужого кода или кода, сгенерированного LLM моделью.
- * До 100% штрафа за просрочку.
- * До 50% если студент игнорирует правила написания кода и комментарии от ментора.

Предельные сроки сдачи лабораторной работы:

- * Если студент не сдал за 2 недели - штраф 50%.
- * Если студент не сдал за 1 месяц - штраф 75%.
- * Если студент не сдал за 2 месяца - штраф 100%.

Задача для студентов: Написать юнит-тесты для реализованных функций на прошлом задании.

Для каждой функции тесты должны покрывать:

- * Покрытие базовых сценариев. Тесты проверяют корректные входные данные, несколько типичных кейсов
- * Покрытие граничных условий. Тесты проверяют минимальные, максимальные и пограничные значения аргументов
- * Проверка некорректного ввода / ошибок. Тесты проверяют поведение функции при неверных данных (если применимо)

Тесты должны обладать:

- * Читаемость и структура тестов. Тесты логично организованы, имеют названия и понятную структуру
- * Автоматизация запуска. Есть возможность запустить все тесты одной командой (например, Makefile)
- * Репрезентативность и полнота. Количество тестов достаточно, чтобы убедиться в корректности функции

[3]

Структуры данных

Лабораторная работа

Описание:

Цель лабораторной работы – решить выданные студенту практические задачи.

Задача считается решенной, если она:

- * Прошла автоматическое тестирование без ошибок.
- * Содержит программный код, соответствующий правилам оформления.

За решенную лабораторную работу студент может получить 7 баллов. Однако во время сдачи лабораторной работы студент может получить следующие штрафы:

- * До 100% штрафа за попытку сдачи чужого кода или кода, сгенерированного LLM моделью.
- * До 100% штрафа за просрочку.
- * До 50% если студент игнорирует правила написания кода и комментарии от ментора.

Предельные сроки сдачи лабораторной работы:

- * Если студент не сдал за 1 месяц - штраф 50%.
- * Если студент не сдал за 1 месяц и 2 недели - штраф 75%.
- * Если студент не сдал за 2 месяца - штраф 100%.

Задание делится на 2 этапа: базовый и дополнительный.

В рамках базового этапа студентам необходимо реализовать структуры данных:

- * Динамический массив.
- * Связный список (однонаправленный и двунаправленный).

Пример задачи для дополнительного этапа:

1) Подсчет значений в массиве

На вход: arr = [1, 2, 1, 3]

На выход: [2, 1, 2, 1]

Ограничения:

- 1) Кол-во значений во входном массиве - до $1e9$
- 2) Сложность работы алгоритма не должна превышать $O(N)$

[4]

Хеш-таблицы

Лабораторная работа

Описание:

Цель лабораторной работы – решить выданные студенту практические задачи.

Задача считается решенной, если она:

- * Прошла автоматическое тестирование без ошибок.
- * Содержит программный код, соответствующий правилам оформления.

За решенную лабораторную работу студент может получить 8 баллов. Однако во время сдачи лабораторной работы студент может получить следующие штрафы:

- * До 100% штрафа за попытку сдачи чужого кода или кода, сгенерированного LLM моделью.
- * До 100% штрафа за просрочку.
- * До 50% если студент игнорирует правила написания кода и комментарии от ментора.

Предельные сроки сдачи лабораторной работы:

- * Если студент не сдал за 2 недели - штраф 50%.
- * Если студент не сдал за 1 месяц - штраф 75%.
- * Если студент не сдал за 2 месяца - штраф 100%.

Задание включает 2 этапа: обязательный и дополнительный

В рамках обязательного этапа студентам необходимо реализовать следующие структуры данных:

- 1) Множество
- 2) Хеш-таблицу

В рамках дополнительного задания студентам выдается одно из заданий на выбор.

Пример задания:

Удаление дубликатов из массива

На вход: $arr = [1, 2, 3, 1]$

На выход: $[1, 2, 3]$

Ограничения:

- 1) Кол-во значений во входном массиве - до $1e9$
- 2) Сложность работы алгоритма не должна превышать $O(N)$

[5]

Лабораторная работа

Описание:

Подсчет асимптотики

Цель лабораторной работы – подсчитать асимптотику для ранее реализованных задач. лабораторная работа считается решенной, если она:

- * Содержит оценку асимптотики для выданных студенту задач, которую принял ментор.

За решенную лабораторную работу студент может получить 5 баллов. Однако во

время сдачи лабораторной работы студент может получить следующие штрафы:

- * До 100% штрафа за попытку сдачи чужого решения или решения, сгенерированного LLM моделью.
- * До 100% штрафа за просрочку.

Предельные сроки сдачи лабораторной работы:

- * Если студент не сдал за 2 недели - штраф 50%.
- * Если студент не сдал за 1 месяц - штраф 75%.
- * Если студент не сдал за 2 месяца - штраф 100%.

В рамках задачи студенту необходимо выбрать из списка, решить (написать работающий код с тестами) и проанализировать 3 задачи с учетом следующих ограничений:

- * Каждая задача должна иметь отличающую асимптотику
- * Должны быть асимптотики: $O(1)$, $O(N)$, $O(N^2)$

Для каждой задачи для каждой строки с кодом, должен находиться комментарий с оценкой этой строки

В конце функции с решенной задачи должна быть приведена асимптотика для этой задачи.

[6]

Алгоритмы сортировки

Лабораторная работа

Описание:

Цель лабораторной работы – решить все выданные студентам практические задачи.

Задача считается решенной, если она:

- * Содержит программный код, соответствующий правилам оформления.
- * Содержит отчет, удовлетворяющий требованиям курса.

За решенную лабораторную работу студент может получить 5 баллов. Однако во время сдачи лабораторной работы студент может получить следующие штрафы:

- * До 100% штрафа за попытку сдачи чужого кода или кода, сгенерированного LLM моделью.
- * До 100% штрафа за просрочку.
- * До 50% если студент игнорирует правила написания кода и комментарии от ментора.
- * До 50% если студент не понимает и не может ответить на вопросы по отчету.

Предельные сроки сдачи лабораторной работы:

- * Если студент не сдал за 2 недели - штраф 50%.
- * Если студент не сдал за 1 месяц - штраф 75%.

* Если студент не сдал за 2 месяца - штраф 100%.

Задача: найти и реализовать 3 алгоритма сортировки. Для каждого алгоритма должны быть написаны тесты

* Первый алгоритм сортировки должен иметь сложность $O(N^2)$

* Второй алгоритм сортировки должен иметь сложность $O(N \cdot \log(N))$

* Третий алгоритм сортировки должен иметь сложность $O(N)$

Нужно доказать что выбранные студентами алгоритмы имеют именно такую сложность

[7]

Разделяй и властвуй

Лабораторная работа

Описание:

Цель лабораторной работы – подсчитать асимптотику для выданных студенту задач. лабораторная работа считается решенной, если она:

* Содержит оценку асимптотики для выданных студенту задач, которую принял ментор.

За решенную лабораторную работу студент может получить 5 баллов. Однако во время сдачи лабораторной работы студент может получить следующие штрафы:

* До 100% штрафа за попытку сдачи чужого решения или решения, сгенерированного LLM моделью.

* До 100% штрафа за просрочку.

Предельные сроки сдачи лабораторной работы:

* Если студент не сдал за 2 недели - штраф 50%.

* Если студент не сдал за 1 месяц - штраф 75%.

* Если студент не сдал за 2 месяца - штраф 100%.

Задача включает в себя 3 подзадачи:

* Решить рекуррентное соотношение для задач основным методом

* Решить рекуррентное соотношение для задач методом деревьев

* Решить рекуррентное соотношение для задач методом подстановок

Пример задач:

* Бинарный поиск

* Быстрая сортировка

* Merge sort

[8]

Амортизационный анализ

Лабораторная работа

Описание:

Цель лабораторной работы – выполнить амортизационный анализ для выданных студенту функций. Лабораторная работа считается решенной, если она:

- * Содержит оценку амортизационный анализ для выданных студенту функций, одобренных ментором.

За решенную лабораторную работу студент может получить 5 баллов. Однако во время сдачи лабораторной работы студент может получить следующие штрафы:

- * До 100% штрафа за попытку сдачи чужого решения или решения, сгенерированного LLM моделью.
- * До 100% штрафа за просрочку.

Предельные сроки сдачи лабораторной работы:

- * Если студент не сдал за 2 недели - штраф 50%.
- * Если студент не сдал за 1 месяц - штраф 75%.
- * Если студент не сдал за 2 месяца - штраф 100%.

Задача: оценить выданную студенту задачу несколькими способами:

- * Метод потенциалов.
- * Метод усреднения.
- * Метод бухгалтерского отчета.

Примеры задач, которую необходимо оценить:

- * Вставка в динамическом массиве.
- * Стек с операцией MULTIPOP.
- * Бинарный счётчик.
- * Очередь из двух стеков.

[9]

NP-полные задачи

Лабораторная работа

Описание:

Цель лабораторной работы – решить выданные студенту практические задачи.

Задача считается решенной, если она:

- * Содержит программный код, соответствующий правилам оформления.
- * Содержит отчет, удовлетворяющий требованиям лабораторной работы.

За решенную лабораторную работу студент может получить 5 баллов. Однако во время сдачи лабораторной работы студент может получить следующие штрафы:

- * До 100% штрафа за попытку сдачи чужого кода или кода, сгенерированного LLM моделью.
- * До 100% штрафа за просрочку.

- * До 50% если студент игнорирует правила написания кода и комментарии от ментора.
- * До 50% если студент не понимает и не может ответить на вопросы по отчету.

Предельные сроки сдачи лабораторной работы:

- * Если студент не сдал за 2 недели - штраф 50%.
- * Если студент не сдал за 1 месяц - штраф 75%.
- * Если студент не сдал за 2 месяца - штраф 100%.

Пример задачи: Кластеризация массива посредством полного перебора всех комбинаций значений этого массива

Описание: Дан на вход массив, необходимо его разбить на K кластеров таким образом, чтобы минимизировать расстояния между каждым элементом подмассива и средним значением в подмассиве. Должен сохраняться порядок. Метрика - $\sum(\text{abs}(\text{arr_cluster}[i] - \text{arr_cluster_mean}))$. Каждый подмассив на выходе должен содержать как минимум один элемент

Пример ограничений:

- * Количество кластеров - 6
- * Алгоритмическая сложность не должна быть меньше чем $O(2^N)$ (подразумевает перебор всех возможных наборов кластеров)
- * Максимальное кол-во элементов - до 25
- * Алгоритм не должен работать больше часа

[10]

Лабораторная работа

Динамическое программирование

Описание:

Цель лабораторной работы – решить выданные студенту практические задачи.

Задача считается решенной, если она:

- * Прошла автоматическое тестирование без ошибок.
- * Содержит программный код, соответствующий правилам оформления.

За решенную лабораторную работу студент может получить 5 баллов. Однако во время сдачи лабораторной работы студент может получить следующие штрафы:

- * До 100% штрафа за попытку сдачи чужого кода или кода, сгенерированного LLM моделью.
- * До 100% штрафа за просрочку.
- * До 50% если студент игнорирует правила написания кода и комментарии от ментора.

Предельные сроки сдачи лабораторной работы:

- * Если студент не сдал за 2 недели - штраф 50%.
- * Если студент не сдал за 1 месяц - штраф 75%.
- * Если студент не сдал за 2 месяца - штраф 100%.

Пример задачи: дан массив целых чисел $A[1..n]$.

Найдите длину наибольшей возрастающей подпоследовательности (элементы не обязаны идти подряд, но должны идти в порядке возрастания индексов и значений).

Вход: [3, 1, 5, 2, 6, 4, 9]

Выход: 4 (например – 1,2,4,9)

[11]

Лабораторная работа

Описание:

Жадные алгоритмы

Цель лабораторной работы – решить выданные студенту практические задачи.

Задача считается решенной, если она:

- * Прошла автоматическое тестирование без ошибок.
- * Содержит программный код, соответствующий правилам оформления.
- * Содержит отчет.

За решенную лабораторную работу студент может получить 5 баллов. Однако во время сдачи лабораторной работы студент может получить следующие штрафы:

- * До 100% штрафа за попытку сдачи чужого кода или кода, сгенерированного LLM моделью.
- * До 100% штрафа за просрочку.
- * До 50% если студент игнорирует правила написания кода и комментарии от ментора.

Предельные сроки сдачи лабораторной работы:

- * Если студент не сдал за 2 недели - штраф 50%.
- * Если студент не сдал за 1 месяц - штраф 75%.
- * Если студент не сдал за 2 месяца - штраф 100%.

Пример задачи: дан набор отрезков (интервалов) на числовой прямой — каждый задаётся началом $s[i]$ и концом $e[i]$.

Нужно выбрать максимальное количество непересекающихся отрезков.

Вход: [(1,3), (2,5), (3,9), (6,8), (9,10)]

Выход: [(1,3), (6,8), (9,10)] → 3 отрезка

Что нужно от студента

- * Сформулировать жадное свойство: Чтобы выбрать как можно больше отрезков, нужно сортировать их по времени окончания и всегда выбирать следующий,

который не пересекается с последним выбранным.

- * Реализовать алгоритм:
- * Отсортировать по end
- * Идти по списку и добавлять отрезок, если $start \geq end_последнего$
- * Оценить сложность алгоритма $O(n \log n)$
- * Проверить на работа алгоритма на написанных тестах

Дифференцированный зачет

Диф. зачет состоит из двух частей:

- 1) Опрос по двум теоретическим вопросам.
- 2) Решение практической задачи.

Вопросы в опросе должны быть из разных блоков (не может быть 2 вопроса по графам). После выдачи вопросов студенту дается 10 минут на подготовку. Как только студент готов или время на подготовку вышло, студент приглашается к защите. В рамках защиты студент отвечает на вопросы. Преподаватель в свою очередь может задавать дополнительные вопросы для уточнения деталей и проверки знаний студента в смежных областях.

Примеры вопросов в устном опросе:

- 1) Назовите базовые типы данных и их размер. Как может варьироваться размер типов в зависимости от архитектуры?
- 2) Динамические структуры данных. Перечислите основные динамические структуры данных.
- 3) Что такое хеш-функция. Каким образом она используется в хеш-таблицах? Что такое коллизии и каким способом они могут решаться?
- 4) Что такое рекуррентное соотношение? Приведите примеры.
- 5) Как посчитать сложность рекурсивного алгоритма методом деревьев?

Всего за каждый вопрос по теории студент может получить 10 баллов.

Критерии оценки за ответ на теоретический вопрос:

- * Студент смог ответить на вопрос – 3 балла максимум.
- * Студент смог ответить на уточняющие вопросы в пределах темы вопроса – 3 балла максимум.
- * Студент смог ответить на уточняющие вопросы в смежных темах – 4 балла максимум.
- * Если студент не может ответить на вопросы или не понимает тот материал, про который рассказывает, преподаватель может снизить балл.

Например, вопрос: "Динамические структуры данных. Перечислите основные динамические структуры данных."

Чтобы получить 3 БАРС студент должен ответить, например: "Динамические структуры данных - структуры, способные изменять свой размер и структуру связей между элементами во время выполнения программы, а память

под них выделяется и освобождается по мере необходимости, а не заранее фиксируется при компиляции. Примеры динамических структур данных: массив, список односвязный и двусвязный, стек, очередь".

Чтобы получить еще 3 БАРС студент должен ответить на дополнительные вопросы от преподавателя в рамках темы вопроса. Примеры вопросов:

- * Какая асимптотическая сложность основных операций у массива и связного списка?
- * Приведите задачи, где использование связного списка будет эффективнее нежели использования массива.
- * В чем отличие стека от очереди?

Чтобы получить еще 4 БАРС, студент должен ответить на вопросы по смежным темам. Примеры вопросов:

- * Что такое амортизационная сложность и почему она применима для массива?
- * Что такое хеш-таблицы и чем они похожи на массивы?
- * Какие методы сортировки существуют для сортировки массивов с асимптотической сложностью $O(N)$

За решение практической задачи студент может получить максимум 20 БАРС. На написание кода студенту даётся не более 30 минут.

Пример задачи: реализовать функцию для стандартной нормализации массива целых положительных чисел не превышающих 100.

Критерии оценки практической задачи:

- * Студент написал рабочий код – 5 баллов максимум
- * Студент выбрал оптимальный типы данных согласно заданным условиям, и может объяснить свой выбор – 5 баллов максимум
- * Алгоритм студента имеет оптимальную асимптотическую сложность для выданной ему задачи – 5 баллов максимум
- * Студент выделил базовые и краевые ситуации и написал для них unit-тесты – 5 баллов.

За что могут быть снижены баллы:

- * Оформление кода не в соответствии со стандартами разработки кода на C.
- * Неосмысленное именование переменных и функций.
- * Если студент не может объяснить строки написанного им кода.

Пример программного кода с описанием некоторых ошибок:

```
float* normalize_array(short int* arr, size_t size) {
    // Баллы будут снижены, так как в условии говорится что максимальное число 100,
    // достаточно было использовать unsigned char тип данных

    int a = arr[0], b = arr[0]; // Баллы будут снижены, так как студент вместо
    // поясняющих названий min, max дал ничего не говорящие названия
    // баллы будут снижены, так как используется сложное выражение которое нужно
    // разбить
    for (int i = 1; i < size; i++) a = arr[i] < a ? arr[i] : a;
    // баллы будут снижены, так как использован лишний цикл
    for (int i = 1; i < size; i++) b = arr[i] > b ? arr[i] : b;
```

```
// баллы будут снижены, так как нет логического отделения между циклами, что
усложняет чтение

// ...

}
```