Министерство образования Республики Беларусь
Учреждение образования
БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерных систем и сетей
Кафедра информатики
Дисциплина: Учебная практика (ознакомительная)

ОТЧЕТ по учебной практике (ознакомительной)

Выполнил
студент: гр. 453503 Халамов Н.А.

Руководитель практики: Герчик А.В.

Минск 2025

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Учебная практика является важной составляющей образовательного процесса студентов технических специальностей. Она позволяет не только углубить теоретические знания, полученные в рамках лекционных курсов, но и сформировать навыки, необходимые для дальнейшего углубления в профессию.

Современный рынок труда предъявляет высокие требования к выпускникам программистских специальностей: они должны не только знать синтаксис языка программирования, а желательно нескольких, но и свободно владеть алгоритмическим мышлением, понимать принципы построения эффективных решений, а также уметь применять стандартные структуры данных в реальных условиях. В связи с этим особую ценность приобретают учебные активности, направленные на развитие практических умений в решении нестандартных задач.

Ознакомительная учебная практика предоставляет студентам возможность познакомиться с форматом дистанционного обучения, основанного на активном взаимодействии с учебной платформой, системами автоматического тестирования и сообществом специалистов крупных компаний. Такой формат позволяет учащимся адаптироваться к современным требованиям индустрии, где самостоятельная работа с техническими материалами, постоянное повышение квалификации и быстрая адаптация к новым инструментам стали нормой.

Целью данной практики стало освоение базовых подходов к решению алгоритмических задач, знакомство с разнообразием структур данных и развитие навыков анализа эффективности решений. В рамках практики я получаю возможность не только закрепить уже известные теоретические концепции, но и научиться применять их в новых, более сложных условиях, моделирующих реальные задачи современных компьютерных систем.

Таким образом, учебная практика служит связью между академическим образованием и профессиональной деятельностью, закладывая основы для дальнейшего углублённого изучения профильных дисциплин и участия в прикладных проектах.

1 ОПРЕДЕЛЕНИЕ ЗАДАЧ И ЦЕЛЕЙ УЧЕБНОЙ ПРАКТИКИ

1.1 Начальное определение задачи

Базовой задачей для данной учебной ознакомительной практики является приобретение и закрепление фундаментальных теоретических и практических знаний в области алгоритмизации и программирования, освоение базовых структур данных и методов решения задач с их использованием. Ключевым направлением является развитие навыков решения алгоритмических задач, характерных для олимпиад по программированию и собеседований при трудоустройстве.

Для реализации поставленным задач и целей выбран онлайн-курс “Тренировки по алгоритмам 7.0” на платформе от Young&&Yandex [1], логотип которой можно увидеть на рисунке 1.

Рисунок 1 – логотип образовательной программы Яндекс

1.2 Описание онлайн-курса

1.2.1 Курс “Тренировки по алгоритмам 7.0” создан для студентов учреждений высшего образования, знающих на базовом уровне хотя бы один язык программирования, а также желающих получить и систематизировать знания в области алгоритмов решения практических задач и проблематики использования ресурсоемких структур данных. Курс доступен для обучающихся из университетов Российской Федерации, Республики Казахстан, Республики Беларусь и других стран ближнего зарубежья.

1.2.2 На курсе используется классический формат обучения. Лекции проходят еженедельно по средам в 18:00 по московскому времени в онлайн-формате. В начале лекции преподаватель рассказывает про организационные моменты и дедлайны работы. В рамках лекции проводится разбор новой темы с презентацией и наглядным объяснением с интутивно понятными визуальными компонентами, после чего происходит диалог между преподавателем и слушателями для ответов на вопросы и разрешения сложностей, возникших при понимании темы. В 20:00 этого же дня открывается доступ к решению домашнего задания на платформе “Яндекс.Контест” [2]. Интерфейс тестирующей системы, необходимой для получения условий задач домашних заданий и их отправки на проверку, можно увидеть на рисунке 2.

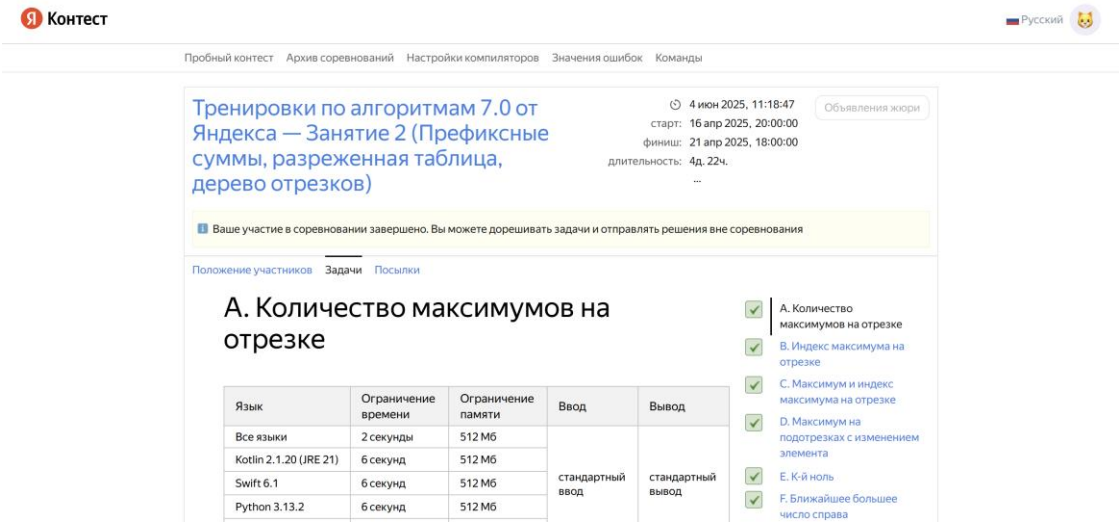

Рисунок 2 – тестирующая система Яндекс.Контест

Каждое домашнее задание состоит из 10 заданий, причем некоторые задания являются заданиями с повышенным уровнем сложности. У студента есть возможность отправить задание на проверку тестирующей системе на языках программирования C++, C#, Java, Python, Ruby, Kotlin, Go и других. Тестирующая система проверяет решение на определенном наборе тестов и в случае успеха сообщает пользователю о засчитывании решения задачи. Решения автоматически тестируются системой на правильность ответа, соответствие установленным ограничениям по времени работы и используемой памяти. Тесты являются закрытыми для студентов, за исключением нескольких примеров в условии задачи. Если решение не прошло какой-то тест, то система сообщает номер теста без указания условия теста и решение дальше не проверяется. У студента есть возможность сделать не более 100 посылок для каждой задачи.

На решение заданий отводится 5 календарных дней, после чего в понедельник проводится разбор заданий и публикация решений. В рамках разбора заданий преподаватель поясняет алгоритмы решения и используемые структуры данных, дополняя это визуальными схемами решения. В конце лекции студенты могут задать вопросы по непонятным ходам решения. Вопросы по домашним заданиям разрешено задавать в Telegram-чате курса, где также можно общаться и обмениваться знаниями и опытом с другими студентами курса.

По итогу домашние задания содержат суммарно 40 задач, по результатам решения которых составляется рейтинг курса и выдаются сертификаты о прохождении курса. 300 лучших студентов получают возможность пройти пробное техническое собеседование в Яндекс, а также очно поучаствовать в фестивале науки YoungCon в Москве.

Лектором является Михаил Густокашин – директор центра студенческих олимпиад факультета компьютерных наук НИУ ВШЭ, тренер

Чемпионов мира ICPC, председатель методической комиссии Московской олимпиады школьников по информатике, олимпиады «Высшая проба» по информатике, член жюри многих олимпиад школьников по информатике и программированию. Преподает курсы по алгоритмам и структуре данных и основам и методологии программирования.

1.2.3 Онлайн-курсы “Тренировки по алгоритмам 7.0” охватывают множество различных тем, среди них можно выделить основные:

– Динамическое программирование;
– Задача о рюкзаке;
– Жадный алгоритм;
– Префиксные суммы;
– Разреженные таблицы;
– Дерево отрезков;
– Метод двух указателей;
– Битовые операции;
– Дерево Фенвика;
– Исправляющие коды Хэмминга;
– Ссылочные типы данных;
– Двусвязный список;
– Стек;
– Очередь;
– Дек;
– Собеседования и трудоустройство.

1.2.4 Курс проходит в течение одного месяца, с 9 апреля по 5 мая 2025-го года. В случае успешной сдачи не менее половины задач из домашних заданий студент получает именной сертификат.

1.3 Назначение требований к студенту

Для прохождения курса был установлен ряд важных требований:

1 Посещение всех лекций курса и поиск онлайн-ресурсов по теоретическим вопросам. Самостоятельное повторение и закрепление материала спустя некоторое время после лекции.

2 Решение домашних заданий в тестирующей системе.

3 Вычисление асимптотической сложности алгоритмов решения по времени работы и памяти. Поиск оптимальных решений.

4 Разбор алгоритмов решения других участников после окончания времени сдачи задач.

5 Прояснение непонятных тем посредством адресации вопросов лектору.

2 СОДЕРЖАНИЕ ОНЛАЙН-КУРСА

2.1 Описание рассмотренных на лекциях тем

2.1.1 В рамках первой лекции была предоставлена и рассказана вводная информация о курсе, организационные моменты и важные даты и дедлайны. Было изучено динамическое программирование – методология решения задач, когда большая задача разбивается на подзадачи, которые решаются один раз, а результаты решения запоминаются для избежания повторных вычислений. Жадный алгоритм подразумавает выбор локально оптимального решения на каждом шаге в надежде получить финально оптимальное решение для всей задачи. Частным случаем задачи, использующей жадный алгоритм, является классическая задача “о рюкзаке”, когда имеются предметы определенной стоимости и определенного веса, которыми нужно набрать самую большую сумму, с ограничением по весу. Основная идея заключается в выборе предметов с максимальным соотношением цены и веса, но такое решение не всегда является эффективным. Удобно использовать массив, где можно для каждого веса хранить данные о цене и возможности его собрать, после чего выбирать оптимальный вес. Это использование метода динамического программирования.

2.1.2 Вторая лекция онлайн-курса посвящена префиксным суммам, разреженной таблице, дереву отрезков. Префиксные суммы позволяют быстро вычислить сумму элементов на отрезке массива, используя сумму всех элементов до определенного номера. Разреженные таблицы (Sparce Table) работают на двумерном массиве, в котором бинарно увеличивается число подряд идущих элементов, максимум из которых надо найти, используя предпосчитанные суммы в предыдущей строке. Максимум выбирается из строки максимальной длины, не превосходящей степень двойки, где уже посчитаны максимумы на отрезках, которые просто нужно сравнить. Дерево отрезков работает на одномерном массиве и упорядоченно хранит значения элементов, что позволяет оптизированно по памяти и времени совершать операции вставки, изменения, удаления элементов, а также совершать групповые операции, такие как нахождение максимума либо суммы на отрезке. Ячейки в дереве могут хранить максимум или сумму на отрезке для конкретных практических задач. Таким образом, префиксные суммы и разреженные таблицы отвечают на запросы очень быстро, а дерево отрезков позволяет изменять данные, в том числе быстро выполнять групповые операции сразу над несколькими элементами. На рисунке 3 приведен фрагмент лекции с обьяснением деревьев.

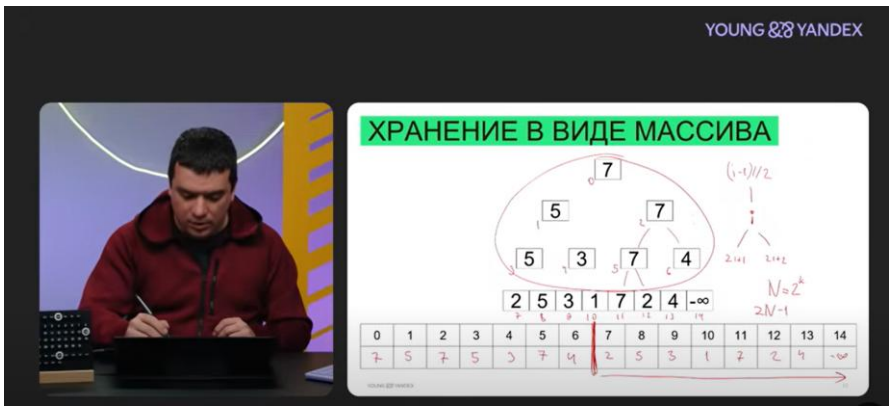

Рисунок 3 – лекция Михаила Густокашина

2.1.3 Дополнительно был проведен вебинар с рекрутерами “Как попасть в Яндекс”, где можно было узнать особенности трудоустройства в Яндекс, основные этапы отбора на стажировку и в штат, навыки, необходимые для стажера, джуна и мидл-специалиста, а также советы по составлению и улучшению резюме. На вебинаре подробно рассказали о типичных ошибках, которых стоит избегать при прохождении собеседований, а также о том, как подготовиться к каждому этапу технического и HR-интервью. Участники узнали, какие технологии и языки программирования наиболее востребованы в разных командах Яндекса, и как продемонстрировать свои знания, потенциал и желание при минимальном опыте. Рекрутеры также порекомендовали ресурсы для подготовки и подчеркнули важность участия в хакатонах и конкурсах и создания собственных проектах для начинающих специалистов.

2.1.4 Очередная третья лекция была посвящена теме битовых операций, исправляющих кодов Хэмминга, дереву Фенвика. Манипуляции с отдельными битами позволяют экономно хранить данные и совершать множество быстрых и интересных операций. Также это одна из очень популярных тем на собеседованиях, особенно для тех специалистов, которые будут работать с высоконагруженными системами. На лекции были рассмотрены применения битовых операций для исправления ошибок, которые постоянно возникают при передаче и хранении данных от таких неожиданных причин, как, например, космическое излучение. В частности, изучались коды Хэмминга — способ добавления избыточной информации к данным, который позволяет не только обнаружить, но и исправить одиночные ошибки в битовых последовательностях. Рассматривались базовые принципы построения таких кодов, включая расстановку проверочных битов и вычисление ошибки. Кроме того, отдельное внимание было уделено дереву Фенвика (Binary Indexed Tree) – эффективной структуре данных, которая позволяет за логарифмическое время производить вычисление префиксных сумм и обновление значений. Это особенно полезно в задачах, связанных с обработкой больших массивов данных в реальном времени. Таким образом, лекция охватила важные

теоретические и практические аспекты работы с битами, показав, как низкоуровневые представления данных тесно связаны с прикладными задачами в программировании, системах передачи данных и алгоритмах сжатия.

2.1.5 Четвертая лекция содержала информацию о ссылочные типах данных, таких как стек, очередь, дек. Были разобраны двусвязные списки. Двусвязные списки были изобретены очень давно и остаются незаменимыми во многих случаях, несмотря на изобретение и внедрение других способов хранения данных. Эта структура данных – одна из самых популярных на собеседованиях. В теоретической части лекции была подчёркнута важность ссылочных структур, где каждый элемент (узел) содержит не только данные, но и ссылки на соседние элементы. Это позволяет эффективно выполнять операции вставки и удаления в середине списка без необходимости сдвига остальных элементов, как в массиве. Были рассмотрены односвязные списки, где каждый элемент содержит ссылку на следующий элемент, и двусвязные списки, где каждый элемент содержит ссылку как на предыдущий элемент, так и на следующий элемент. Было уделено внимание еще трем повсеместно используемым структурам данных. Стек работает по принципу LIFO (последний вошел, первый вышел). Очередь работает по принципу FIFO (последний зашел, последний вышел). Дек является двусторонней очередью, где элементы могут удаляться и добавляться с обеих сторон.

2.2 Описание лекций с разборами и решениями задач

На этих лекциях подробно разбирались все задачи последних домашних заданий по теме. Лектор давал советы по решению задач, визуально пояснял решение, приводил примеры, разбирал тестовые случаи и проверял краевые значения. Студенты могли задавать вопросы и прояснять трудности, возникшие при попыткам решения сложных алгоритмических задач.

2.3 Описание дополнительных источников получения информации

В качестве вспомогательных ресурсов, которые полезны для более глубокого понимания рассматриваемых тем и прояснения сложных моментов, были выбраны несколько интернет-источников. В качестве теоретического материала использовались лекции преподавателя Алгоритмов и структур данных МФТИ Руховича Ф.Д. [3], а также статьи с описанием алгоритмов в хэндбуке от Яндекс Образования [4]. В качестве ресурсов, включающих дополнительные задачи по темам для закрепления пройденного материала, использовались платформа для закрепления знаний по алгоритмам CodeRun и сайт с задачами по разным темам LeetCode [5]. Также использовались олимпиадные задачи с платформы Информатикс [6].

3 ВЫПОЛНЕНИЕ ДОМАШНИХ ЗАДАНИЙ ОНЛАЙН-КУРСА

3.1 Введение, жадный алгоритм и задача о рюкзаке

Первое домашнее задание содержало задачи по теме первой лекции. Все они были решены на языке программирования C++. В первой задаче для того, чтобы оптимально расставить аудитории и группы, они сортируются по возрастанию, и на каждом этапе выбирается самый рабочий вариант. Это классическое использование жадного алгоритма. Во второй задаче необходимо разбить массив на минимальное количество отрезков так, чтобы для каждого элемента в отрезке выполнялось условие: значение элемента не меньше длины отрезка, которому он принадлежит. Она была решена с использованием жадного алгоритма, который на каждом шаге пытается максимально расширить отрезок, и сохраняет длины отрезков. В следующей задаче было сказано решить “задачу о рюкзаке”, набирая максимальный вес, не превышающий установленный лимит. Она была решена созданием массива с сохранением тех весов, которые можно набрать, последовательно добавляя предметы к уже набранным весам, и выбором максимального веса в конце. Следующая задача была усложнена необходимостью набрать максимальную стоимость. Алгоритм похож, но надо сравнивать текущую стоимость набора веса с той, которая получится при добавлении предмета, и выбирать наибольшую. Следующая задача была усложнена необходимостью вывести номера предметов, которыми набирается максимальная стоимость. Решение было организовано созданием дополнительного массива, хранящего номера последних добавленных предметов к каждому весу, где в конце посредством отнимания от веса с максимальной стоимостью весов последнего добавленного предмета получается ответ.

3.2 Префиксные суммы, разреженная таблица, дерево отрезков

Первая задача требует эффективного выполнения множества запросов на нахождение максимума и количества максимумов на отрезках массива. Она была решена с использованием дерева отрезков, которое в каждом своем узле хранило информацию о максимуме и количестве максимумов на отрезке. Запросы рекурсивно проходили по дереву для достижения результата с помощью сравнения результатов у дочерних узлов. Следующая задача похожа по смыслу, но нужно вывести не количество максимумов на отрезке, а индекс первого максимума. Для этого модифицировали структуру узла, а алгоритмы остались те же. Еще одна задача требовала обьединить предыдущие и использовать оба случая. В следующей задаче добавлялось условие возможности запроса на обновление значения элемента. Это было решено рекурсивным опусканием и подниманием по дереву с обновлением узлов. Еще одна задача требовала найти определенный по счету нуль на отрезке. Я хранил количество нулей на отрезке в дереве и затем рекурсивно спускался вниз,

проверяя нужное количество нулей. Вариантом задачи было нахождение максимального числа на отрезке справа от заданного элемента. Всегда шли в левое поддерево для удобного получения ближайшего элемента. Также были задачи с операциями на отрезке. В одной задаче нужно было прибавить значение на отрезке. Построил дерево отрезков и использовал ленивое обновление, когда сохранял добавление для детей, и обновлял их значения только при непосредственном взаимодействии с ними, чтобы эффективно выполнять массовые прибавления на отрезке и получать значения отдельных элементов без перебора всего массива. Еще одна задача просила обновить значения и также найти максимум на отрезке. Тут мы прибавляли сохраненные значения при обращении к детям, находящимся в отрезке. Для всех задач массив дополняется до размера, равного степени двойки, для удобного построения дерева отрезков. Эти подходы позволяют выполнять каждый запрос за логарифмическое время, обеспечивая высокую производительность даже при большом числе запросов.

3.3 Битовые операции, исправляющие коды Хэмминга, дерево Фенвика

На третьей лекции были рассмотрены задачи, связанные с битовыми операциями, кодами Хэмминга и деревом Фенвика. В первой задаче нужно было подсчитать количество единиц в двоичном представлении числа. Мы проверяем равен ли последний бит единице и сдвигаем вправо пока число не станет нулем. Это равносильно проверке остатка и делению на 2. Вторая задача связана с восстановлением массива по матрице попарных побитовых И: поскольку побитовая операция И ограничивает возможные биты в исходных числах, можно пройтись по всем парам и для каждого бита определить, в каких элементах он должен быть установлен, после чего восстановить массив по битам. В третьей задаче нужно перевести число из десятичной системы счисления в двоичную, произвести циклические сдвиги и выбрать из результатов операций максимальный в десятичной записи. В следующей задаче нужно догадаться произвести операцию XOR и воспользоваться свойством перемены мест элементов в уравнении с операцией. Также одна задача требует реализации дерева Фенвика со стандартной операцией вычисления суммы на отрезке, а также необходимостью поменять значение отдельно взятого элемента. Все эти задачи акцентируют внимание на эффективной работе с битами, что важно для оптимизации хранения данных, исправления ошибок передачи и повышения производительности в задачах низкоуровневого программирования.

3.4 Двусвязный список, очередь, стек, дек

В первых трех задачах четвертого домашнего задания нужно было реализовать три структуры данных. В реализации стека были описаны

следующие функции: push (добавление элемент в начало стека), pop (удаление и возврат верхнего элемента стека), back (возврат верхнего элемента стека), size (получение размера стека), clear (полная очистка стека), exit(выход из программы). На рисунке 4 приведена модель работы стека.

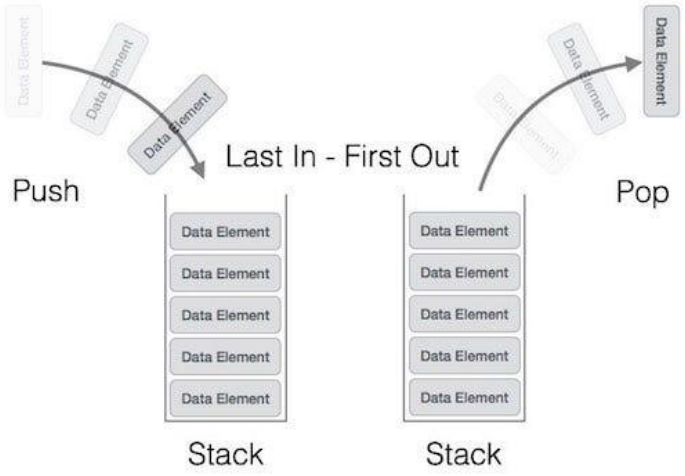

Рисунок 4 – модель работы стека

В реализации очереди были описаны следующие функции: push (добавление элемент в начало очереди), pop (удаление и возврат первого элемента очереди), front (возвратпервого элемента очереди), size (получение размера очереди), clear (полная очистка очереди), exi t(выход из программы). В реализации дека были описаны следующие функции: push_front (добавление элемент в начало дека),push_back (добавление элемента в конец дека), pop_fron t(удаление и возврат первого элемента дека),pop_back (удаление и возврат последнего элемента дека),front (возврат первого элемента дека), back (возврат последнего элемента дека), size (получение размера дека), clear (полная очистка дека), exit (выход из программы). В следующей задаче нужно было моделировать поведение операционной системы Windows при переключении между приложениями с помощью комбинации Alt+Tab. Моделирование было организовано с помощью дека и добавления окон в очередь с обоих концов. В еще одном задании нужно было найти минимальное количество копилок, которые нужно разбить, чтобы открыть все остальные. Ключи от копилок находятся внутри других копилок, создавая зависимости между ними. Суть решения состояла в том, что задача эквивалентна нахождению количества циклов в ориентированном графе. Я использовал массив для отметки посещенных копилок, и для каждой непосещенной копилки начинал обход, пока не обнаруживал цикл. В одной задаче я использовал структуру системы непересекающихся множеств для определения минимального количества мостов, которые нужно построить, чтобы все острова стали связанными (образовали одну связную компоненту). Мосты строятся в заданном порядке, и нужно найти момент, когда граф островов станет связным, то есть останется одна компонента связности.

ЗАКЛЮЧЕНИЕ

После решения задач курса мной был получен сертификат о прохождении курса, который в будущем можно использователь при прохождении отбора на стажировки и в летние школы Яндекса, копия сертификата приведена на рисунке 5.

Рисунок 5 – Сертификат о прохождении курса Халамовым Н.А.

Прохождение курса "Тренировки по алгоритмам 7.0" позволило мне глубоко освоить ключевые алгоритмы и структуры данных, что значительно расширило мои профессиональные возможности. Особое внимание уделялось практическому применению знаний – каждое теоретическое понятие отрабатывалось на серии задач разного уровня сложности. В процессе обучения я научился анализировать временную и пространственную сложность алгоритмов, выбирать оптимальные структуры данных для конкретных задач и реализовывать эффективные решения. Это сформировало системный подход к решению алгоритмических проблем, который применим как в учебных, так и в реальных профессиональных задачах.

Освоенные технологии и методы имеют непосредственное практическое значение для карьеры в IT. Полученные знания регулярно проверяются на технических собеседованиях ведущих компаний, а умение работать с алгоритмами является базовым требованием для многих позиций в разработке [7]. Особенно ценным оказался опыт решения задач, моделирующих реальные ситуации — от переключения между окнами приложений до оптимизации маршрутов между островами. Эти навыки позволяют не только успешно

проходить отбор, но и создавать производительные решения в профессиональной деятельности.

Для дальнейшего развития я планирую продолжить углубленное изучение алгоритмов через специализированные платформы и литературу. Практика на таких ресурсах, как LeetCode и Codeforces, поможет закрепить полученные знания. Особое внимание стоит уделить графовым алгоритмам и сложным структурам данных, которые открывают новые возможности для решения нетривиальных задач. Полученная база создает прочный фундамент для освоения более продвинутых тем, включая машинное обучение и распределенные системы, делая этот курс важной ступенью в профессиональном становлении.

Отчёт по учебной практике оформлен в соответствии с СТП 01-2024 [8].

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

[1] Тренировки по алгоритмам [Электронный ресурс]. – Режим доступа : [Электронный ресурс]. – Режим доступа : https://yandex.ru/yaintern/training/algorithm-training.

[2] Яндекс Контест [Электронный ресурс]. – Режим доступа : https://contest.yandex.ru/.

[3] Алгоритмы и структуры данных (1 курс, осень 2024) – Рухович Ф.Д. [Электронный ресурс]. – Режим доступа : https://www.youtube.com/playlist?list=PL4_hYwCyhAvZobs9MJdTG0hFjT6C7ZzHk.

[4] Хэндбук от Яндекс Образования [Электронный ресурс]. – Режим доступа : https://education.yandex.ru/handbook/algorithms.

[5] Leetcode [Электронный ресурс]. – Режим доступа : https://leetcode.com/.

[6] Информатикс [Электронный ресурс]. – Режим доступа : https://informatics.msk.ru/.

[7] Алгоритмические собеседования в Яндексе: как подготовиться и чего ожидать [Электронный ресурс]. – Режим доступа : https://education.yandex.ru/journal/algoritmicheskie-sobesedovaniya-v-yandekse-kak-podgotovitsya-i-chego-ozhidat.

[8] Стандарт предприятия – Минск : БГУИР, 2024. – 178 с.