Учреждение образования

«Белорусский государственный технологический университет»

Факультет информационных технологий

Кафедра программной инженерии

Специальность 6-05-0612-01 «Программная инженерия»

ОТЧЕТ

по учебной ознакомительной практике

Исполнитель

студент 1 курса 10 группы \_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись, дата) (ФИО)

Руководитель практики

от университета

преподаватель-стажер \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Север А.С.

(должность, уч. звание) (подпись, дата) (ФИО)

Отчет защищен с оценкой\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Минск 2024 г.

**17.06.2024.**

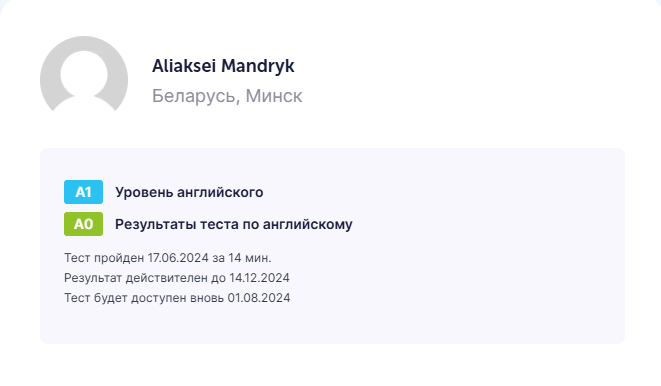
В рамках прохождения практики была проведена лекция под названием «Открытие IT Warm-Up», которая представила важные аспекты и информацию, связанную с программой практики.

Лекция началась с обзора IT Warm-Up, программы практики, которая предлагает студентам возможность получить практический опыт в компании EPAM. Были рассмотрены основные цели и задачи программы, а также перспективы дальнейшего развития студентов после ее завершения.

Далее была представлена презентация о компании EPAM и EPAM Training Center. Была рассмотрена история и достижения компании, а также ее значимость в IT-сфере. Также были представлены возможности обучения и развития, предоставляемые EPAM Training Center, что подчеркнуло важность и ценность прохождения практики в данной организации.

В ходе лекции также были представлены два важных онлайн-портала: training.by и learn.com. Портал training.by является платформой, на которой студенты могут регистрироваться, получать доступ к образовательным материалам, выполнять задания и отслеживать свой прогресс. Портал learn.com предоставляет возможность получения дополнительных знаний и навыков через онлайн-курсы и обучающие материалы.

Заключительным заданием в первом домашнем задании было прохождение теста по английскому языку (рис. 1.1).

Рисунок 1.1 – Результат теста по английскому языку

По результатам теста стало понятно, что уровень английского необходимо повышать, так как для нормальной работы программисту это уровня мало.

Так же было выполнено домашнее задание №3, которое включало в себя тест из 37 вопросов (рис. 1.2).

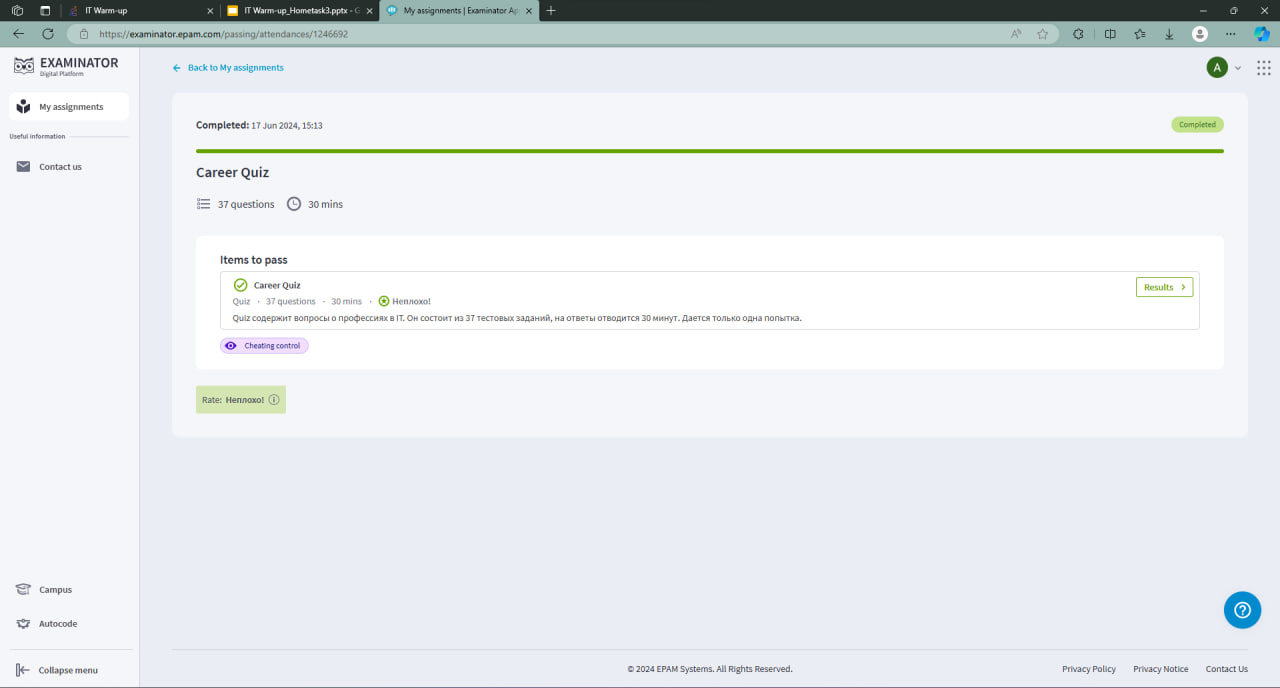


Рисунок 1.2 – Результат теста Career Quiz

После просмотра видеороликов и прохождения тестов было принято решение о выборе технологии Java. Fundamentals (рис. 1.3).



Рисунок 1.3 – Технология Java. Fundamentals

Важными аспектами выбора конкретно данного курса стало изучение Java в 4 учебном семестре и важность углубленного изучения каждого языка программирования для достижения поставленных целей.

**18.06.2024.**

Целью этого дня было выполнение домашнего задания №2, которое включало в себя работу с платформой Git и GitHub (рис. 1.4).

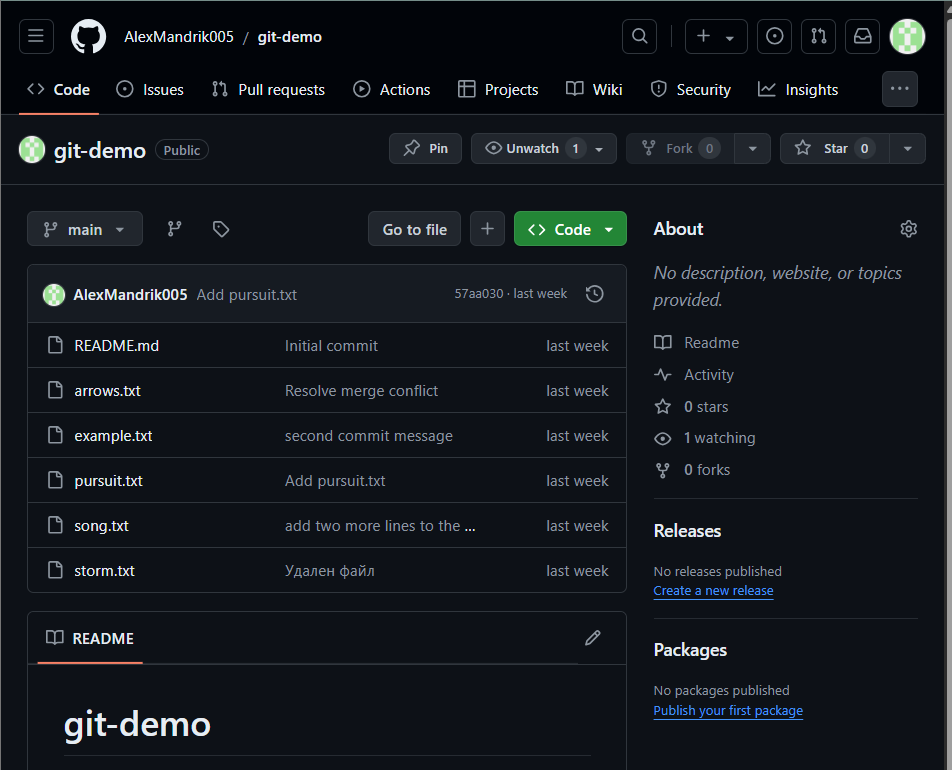


Рисунок 1.4 – Репозиторий GitHub

В Git были сгенерированы ключи ssh и локальный репозиторий был связан с удалённым. После был создан репозиторий git-demo, в который добавлены файлы с текстом. Файлы несколько раз коммитились. Во время работы постоянно использовалась команда git push (рис. 1.5).

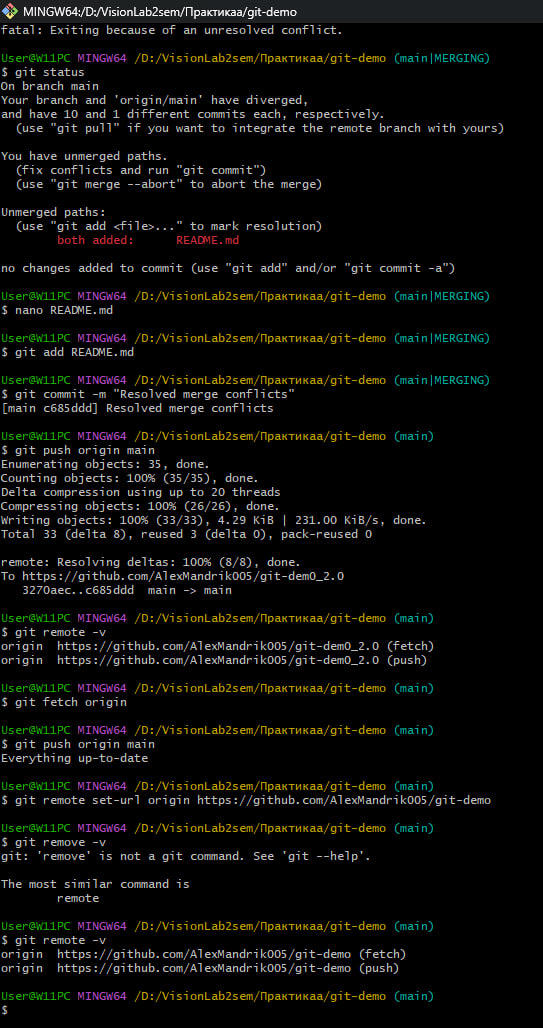


Рисунок 1.5 – Успешное использование git push

Был добавлен файл .gitignore. В репозитории создавались новые ветки, которые после смерживались. Решался конфликт смерживания веток. После создавался новый репозиторий, в который клонировалась информация исходного репозитория. Результат можно увидеть на рисунке 1.6



Рисунок 1.6 – GitHub

Умение хорошо обращаться с Git и GitHub является отличительной чертой хорошего программиста, т.к. эти платформы являются основоположными для командной работы. Несмотря на то, что студенты на данный момент уже изучали эти платформы, было изучено много новых моментов, на которые раньше не обращалось внимание.

**19.06.2024.**

В этот день была установлена IntelliJ IDEA Community Edition (рис.1.7).

Эта среда разработки позволяет писать код на языке программирования Java.

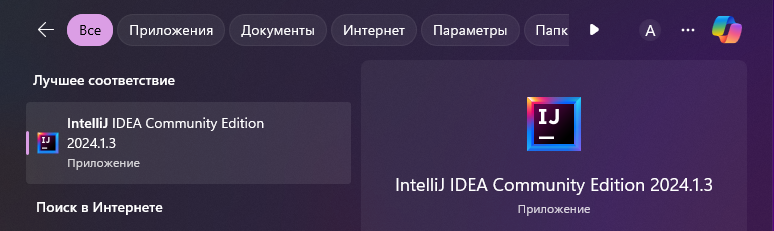


Рисунок 1.7 - IDE

При установке возникли некоторые сложности, т.к. для установки IDE необходимо было корректно установить Java (рис. 1.8).

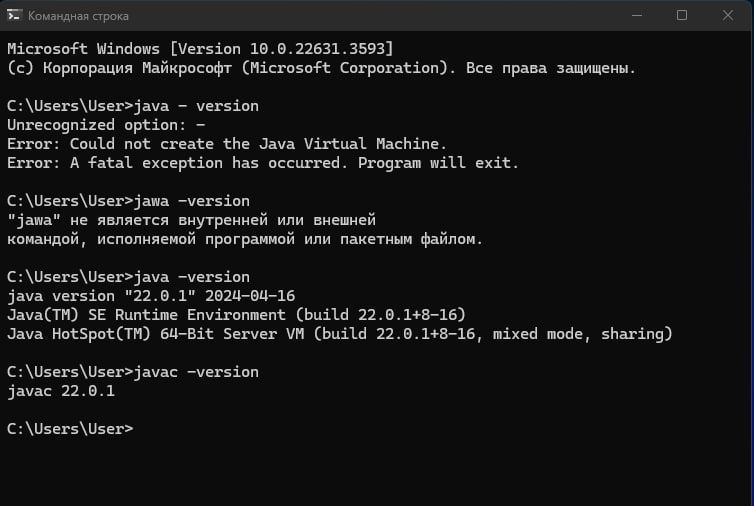


Рисунок 1.8 – Проверка Java.

Были просмотрены лекции от компании EPAM на тему Java. Fundamentals. Эти лекции включали в себя пять разделов (рис. 1.9). В лекциях рассматривались базовые элементы Java.

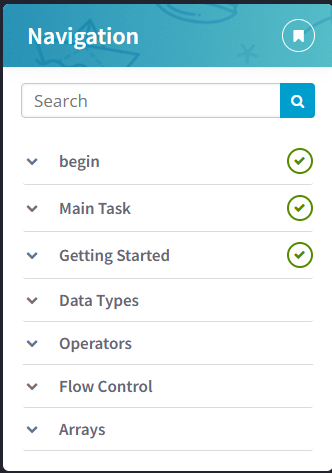


Рисунок 1.9 – Курс лекций

По каждой теме рассматривались практические примеры с полным объяснением кода программы (рис. 1.10). Наиболее короткой и важной являлась тема массивы т.к. в ней рассматривались важнейшие структурные элементы языка.

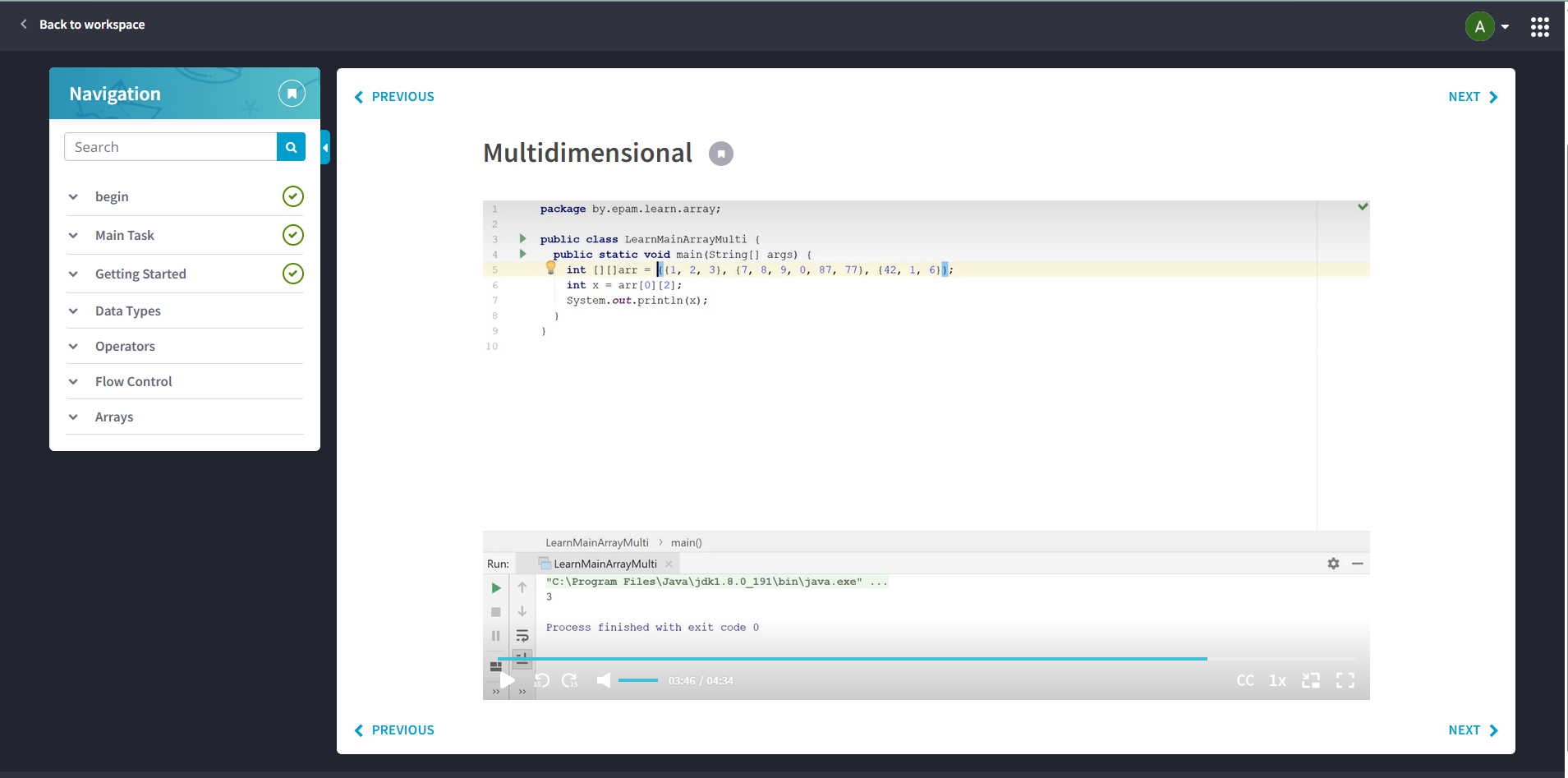


Рисунок 1.10 – Видеоролик курса

После просмотра видеороликов необходимо было пройти тест (рис. 1.11), в котором авторы всячески пытались подловить пользователей.

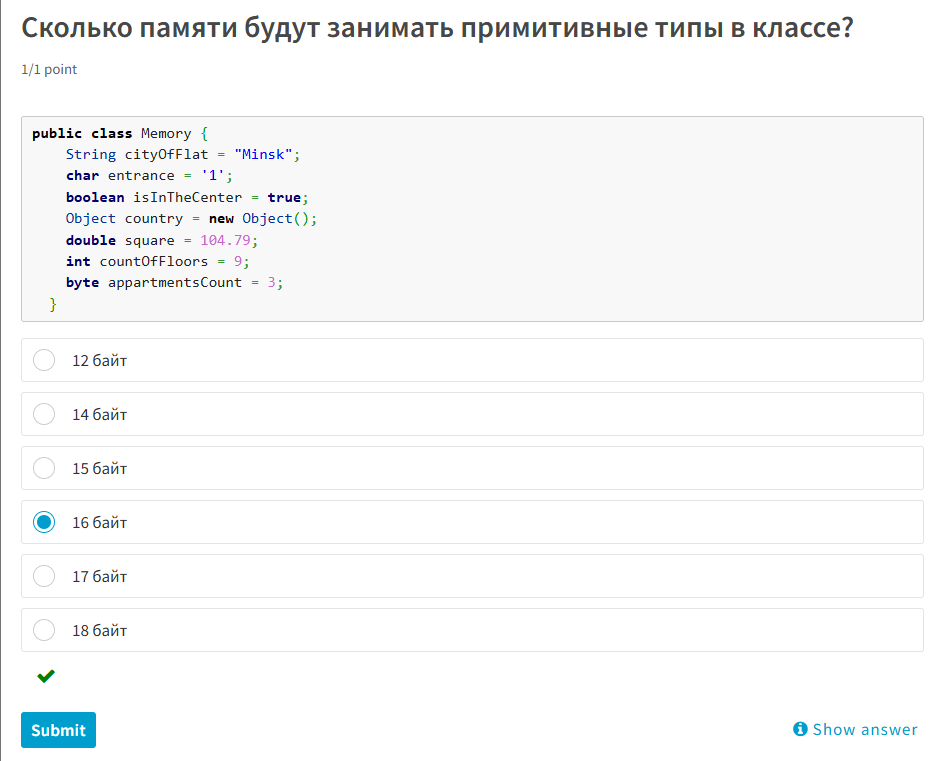


Рисунок 1.11 – Пример вопроса теста

В видеороликах описывались базовые аспекты языка программирования: базовые типы данных, инициализация переменных, литералы, логика и унарные операторы. Все задания проделывались самостоятельно в IDEA (рис. 1.12).

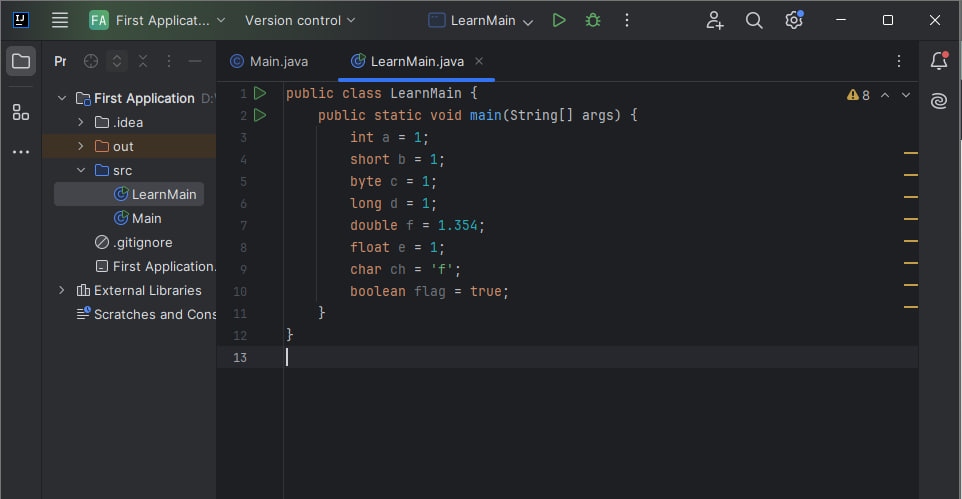


Рисунок 1.12 – Типы данных и инициализация

В конечном итоге можно сказать, что синтаксис языка программирования Java похож с синтаксисом языка программирования C++.

Базовые элементы, такие как типы данных, унарные операторы, так же очень похожи.

**20.06.2024.**

В этот день было принято решение приступить к выполнению основных практических заданий курса (рис. 2.1), которые являются фундаментальными для этого языка программирования.

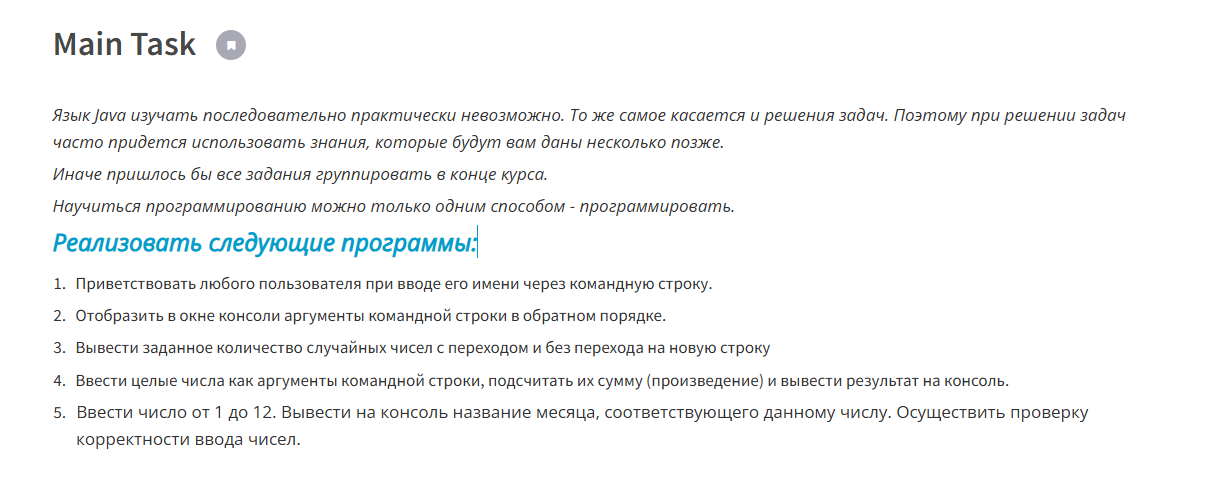


Рисунок 2.1 – Практические задания курса

В процессе выполнения заданий возникли проблемы с аргументами командной строки. Если сравнить работу с командной строкой в С++ и Java, то в Java всё реализовано гораздо сложнее, однако функционал больше.

Первые четыре задания ориентированы на работу с командной строкой и передачу аргументов через неё (рис. 2.2). К сожалению в курсе не описывалась передача аргументов через командную строку и пришлось искать дополнительную информацию через Google.

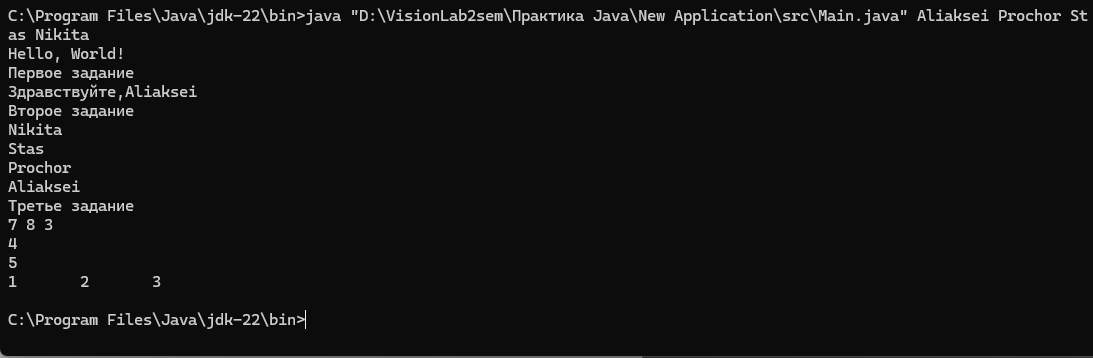


Рисунок 2.2 – Результат выполнения

В данном случае программа Main.java выполняется полностью через командную строку. Код программы в IDE представлен на рисунке 2.3.

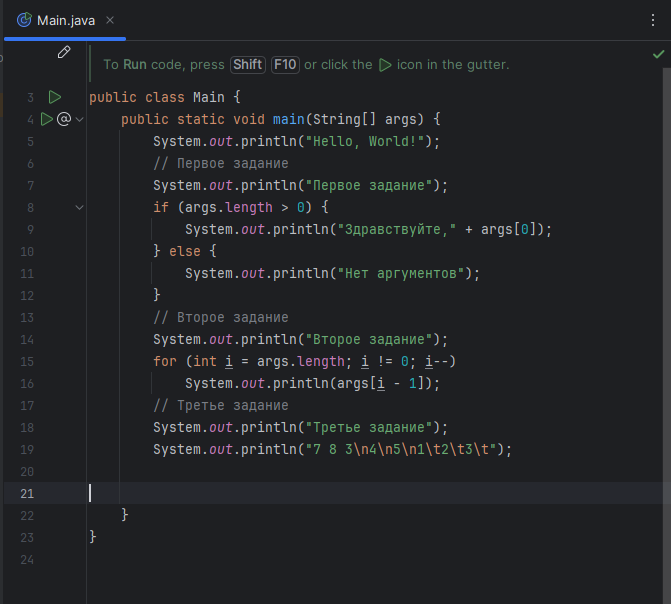


Рисунок 2.3 – Код программы

В целом эти задания не вызывали трудностей, за исключение самого процесса передачи аргументов. Благодаря интернету удалось узнать все аспекты данного процесса и решить все поставленные задачи.

**21.06.2024.**

Началась разработка четвёртого и пятого заданий, которые включали в себя арифметические операции с аргументами командной строки и работу с двумерными массивами соответственно.

Для корректного функционирования было принято решение использовать обработку исключений try catch (рис. 2.4).

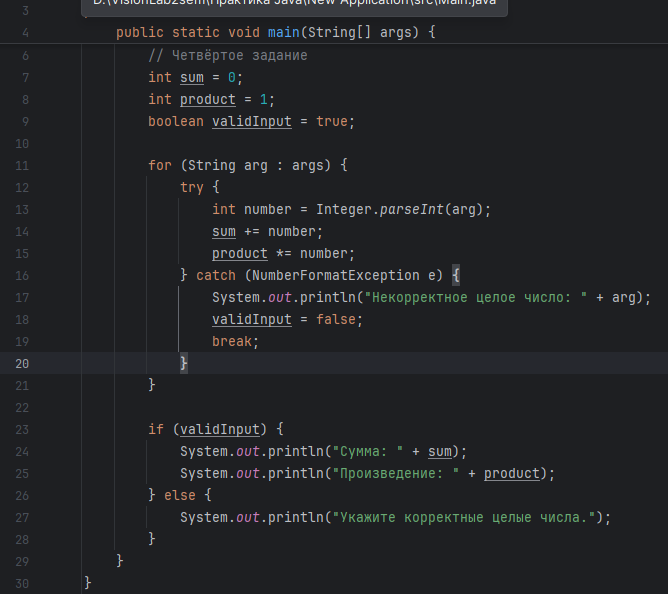


Рисунок 2.4 – Код четвёртого задания

В результате выполнения программы в консоль выводится сумма и произведение аргументов (рис. 2.5), что соответствует поставленной задаче.

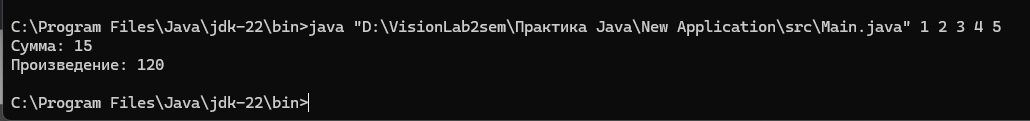


Рисунок 2.5 – Результат в командной строке

Пятое задание является гораздо более интересным. На базе него уже можно выполнять некоторые полезные действия, например поиск элементов по индексу. В данном случае при вводе пользователем числа в пределах от 1 до 12 включительно на экране появится название соответствующего числу месяца (рис. 2.6).

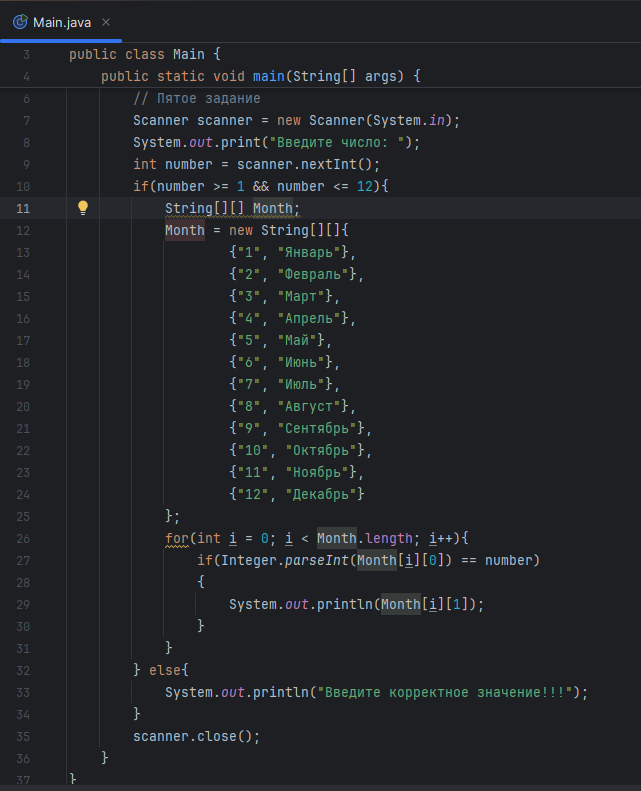
.

Рисунок 2.6 – Код программы

Для решения данной задачи пришлось возвращаться к разделу видеоуроков, т.к. не всё сразу получалось (рис. 2.7).

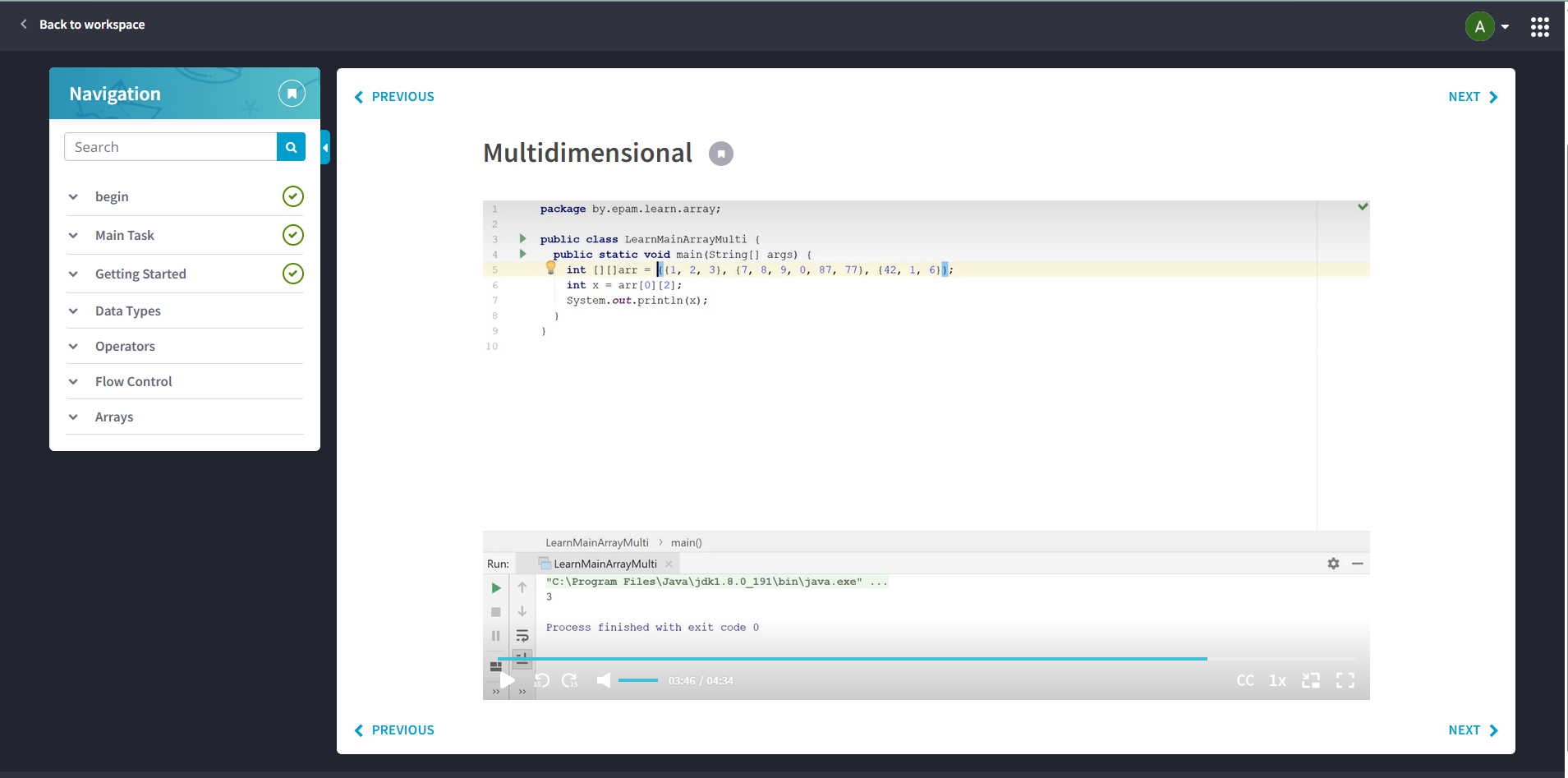


Рисунок 2.7 – Курс лекций

Программа выполняется полностью корректно. Реализована проверка корректности вводимых данных. Если пользователь вводит не корректное значение, программа попросит его ввести значение верно. На рисунке 2.8 приведён пример выполнения программы.

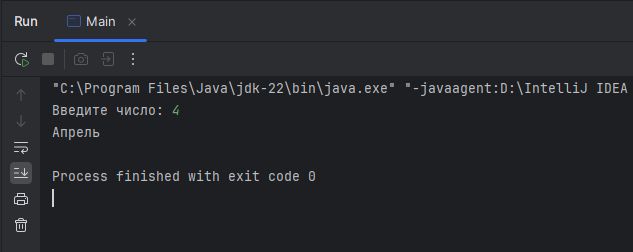


Рисунок 2.8 – Результат выполнения

Итогом это дня является повторение передачи аргументов через командную строку и изучение синтаксиса и практическое применение массивов в Java. В этот день были изучены как одномерные, там и двумерные массивы, но работать пришлось в данном случае с двумерными. Работа с массивами данных в Java оказалась практически полностью похожа на ту же работу в С++, что значительно упростило задачу.

**22.06.2024.**

В этот день было принято решение начать делать опциональные задания курса. Всего курс подразумевает выполнение семи опциональных задач. Для успешного выполнения этого опционального задания достаточно выполнить 2 – 4 пункта. Сначала было выполнено первое задание. Для решения первого задания сразу было предложено несколько идей, но последующий их анализ позволил выбрать самую эффективную (рис. 3.1).

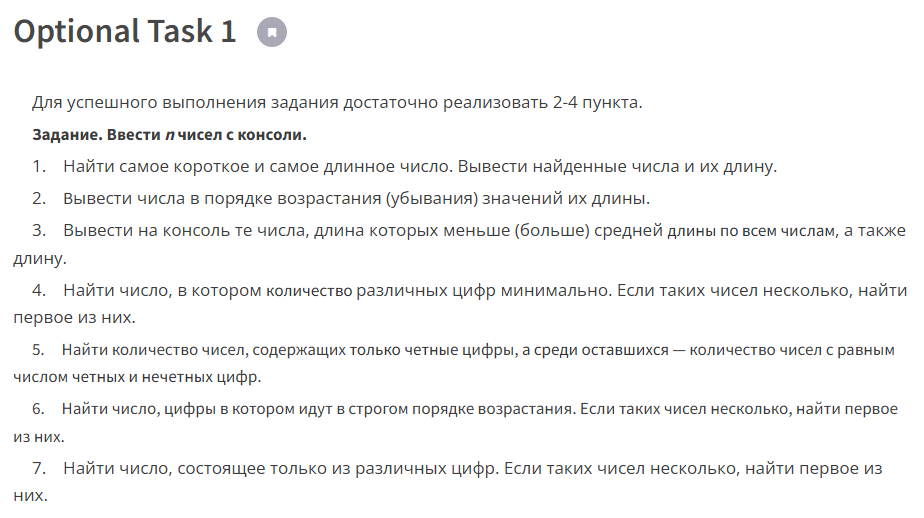


Рисунок 3.1 – Дополнительные задания

Чтобы решить поставленную задачу был использован метод привидения типов данных. Каждое вводимое число рассматривалось как строка символов и длина определялась с помощью определения длины строки. Для этого в Java есть специальная функция .lenght() (рис. 3.2)



Рисунок 3.2 – Код программы

Результатом выполнения является вывод в консоль самого длинного числа и его длины (рис. 3.3), что является полным решением поставленной задачи.

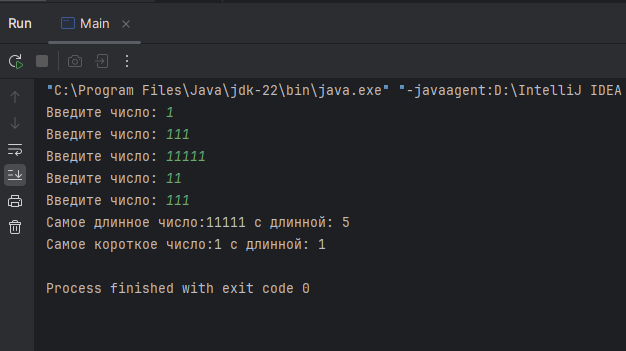


Рисунок 3.3 – Результат выполнения

После первого задания была начата работа над третьим, которое по своей сути сильно похоже на первое, однако сам функциональный блок пришлось полностью переделывать. По условию требовалось определить среднюю длину вводимых чисел и вывести в консоль только те, длина которых меньше средней длины всех чисел. Поставленная задача была реализована за короткий срок (рис. 3.4).



Рисунок 3.4 – Код программы

Каждый программист знает, что любую программу после написания следуют тестировать. Результат тестирования представлен на рисунке 3.5.

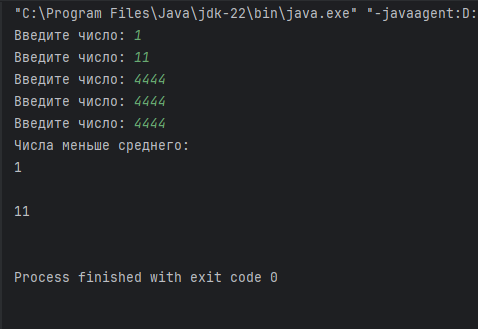


Рисунок 3.5 – Результат тестирования

По итогам дня можно сказать, что данные опциональные задания являются не сложными. В этот день было изучено и применено на практике приведение типов данных, что в значительной степени облегчает решение задач. Так же были повторены циклы. Основные задания было интереснее делать.

**24.06.2024.**

В этот день продолжилась работа с опциональными заданиями. Следующее опциональное задание №5 подразумевало определение количества чётных и нечётных цифр в числе и последующий их подсчёт. Код программы, реализующий данную задачу, представлен на рисунке 4.1.



Рисунок 4.1 – Код программы

Для решения данной задачи снова пришлось рассматривать каждое число как строку символов. Для определения четности или нечётность числа необходимо рассматривать каждый символ как число. Для этого необходимо использовать привидение типов данных.

Тестирование программы доказало правильность написания кода и подтвердило соответствие условия и результата (рис. 4.2).

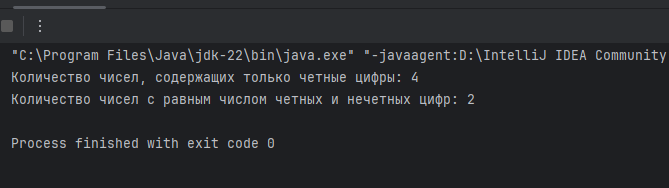


Рисунок 4.2 – Результат выполнения программы

Для решения шестого задания было решено создать новую функцию, которую вызовет main. Это связано с тем, что все программы за последние несколько дней являлись однотипными и скучными. Пример кода представлен на рисунке 4.3.

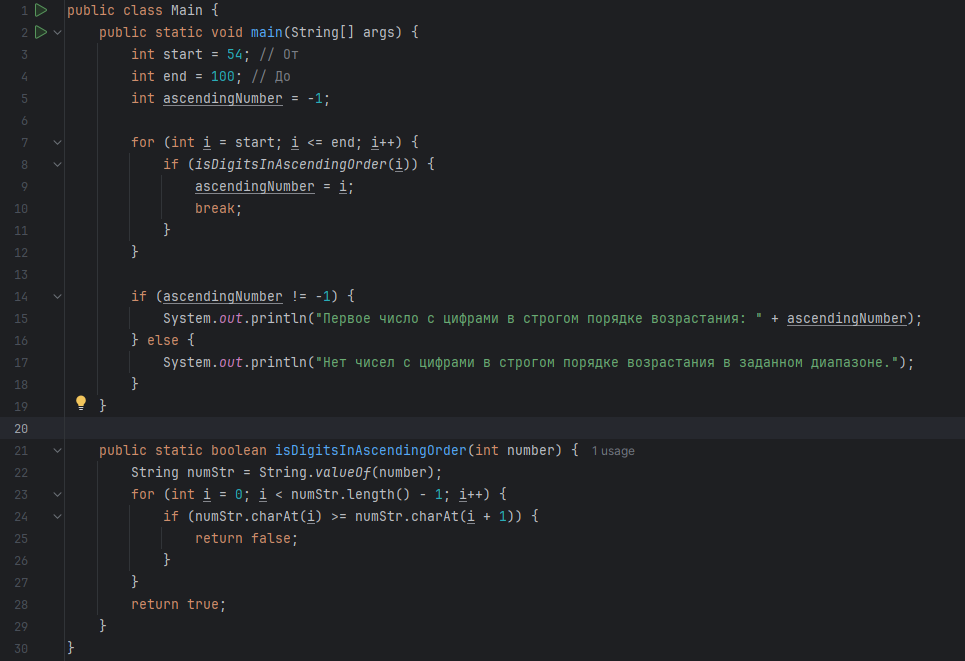


Рисунок 4.3 – Код программы

Главная функция main перебирает все числа диапазона и вызывает дополнительную функцию isDigitsInAscendingOrder(), которая уже проверяет, следуют ли цифры в числе в порядке возрастания.

В дополнительной функции, так же, как и в прошлом пятом задании, каждое число рассматривалось как строка символов и после каждый символ преобразовывался в целочисленный тип (рис. 4.4).

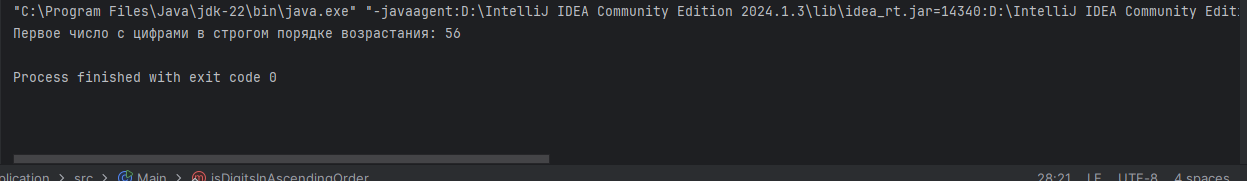


Рисунок 4.4 – Результат выполнения программы

Результатом выполнения программы в диапазоне от 54 до 100 является вывод числа 56.

Подводя итоги дня хочется отметить то, что весь день происходило повторение изученного ранее. Из нового были изучены функции пользователя, что позволит в будущем писать различные модули к уже существующему коду и сделать этот код понятным для других программистов.

**25.06.2024.**

Проведя анализ оставшихся заданий курса, принято решение доделывать все задания из Optional Task 1. Осталось выполнить четвёртое и седьмое задания.

Первым заданием на этот день будет четвёртое. В нём необходимо найти число, в котором количество различных цифр минимально. Если таких чисел несколько, найти первое из них.

Для реализации программа была разделена на две функции: главная функция main() (рис. 5.1), функция пользователя countUniqueDigits() (рис. 5.2).



Рисунок 5.1 - Код функции main

В главной функции main() подключён объект scanner для чтения пяти чисел, введённых пользователем. Пять чисел сразу записываются в одномерный массив numbers[]. После вызывается функция пользователя countUniqueDigits().

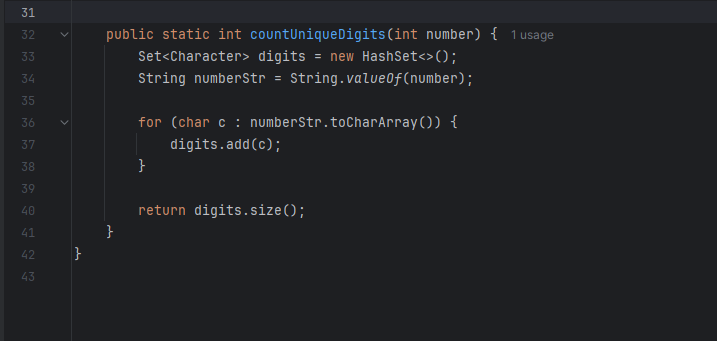


Рисунок 5.2 – Функция пользователя countUniqueDigits()

В функции пользователя создаётся такой объект, как множество, предназначенный для хранения элементов. После идёт преобразование числа в символьную строку. Метод toCharArray() преобразует строку в массив символов.

После написания программы обязательно следует тестирование и проверка на соответствие с поставленной задачей (рис. 5.3).

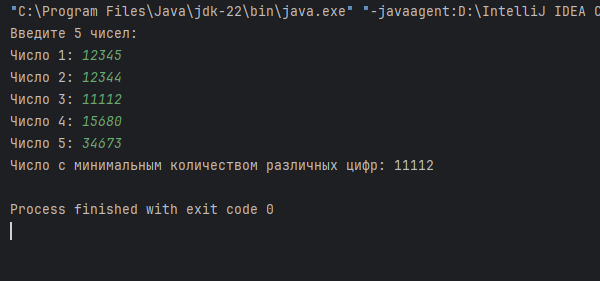


Рисунок 5.3 – Проверка работоспособности кода

Подводя итоги дня, следует отметить, что сегодня в основном повторялся ранее изученный материал, что помогло закрепить полученные знания и повысить уверенность в их применении. Новым было изучение пользовательских функций, что в будущем поможет создавать модули для существующего кода и сделать его более понятным для других программистов.

Это важный шаг к написанию более структурированного и читаемого кода, что значительно упростит совместную работу и поддержку проектов. В целом, день был продуктивным и полезным для дальнейшего развития навыков программирования.

**26.06.2024.**

Для этого дня осталось последнее, седьмое задание. Оно в некоторой степени похоже на четвёртое, однако имеет свои особенности. Для решения данной задачи, аналогично четвёртому заданию, было решено разбить код на две функции: главную функцию main() (рис. 6.1) и пользовательскую функцию hasAllUniqueDigits() (рис. 6.2).

Такое разделение кода на модули позволяет улучшить его читаемость и упрощает процесс тестирования и отладки. Главная функция отвечает за ввод данных и вызов пользовательской функции, а функция hasAllUniqueDigits() — за проверку уникальности цифр в числе. Этот подход помогает писать более структурированный и понятный код, что облегчает его сопровождение и развитие в будущем.

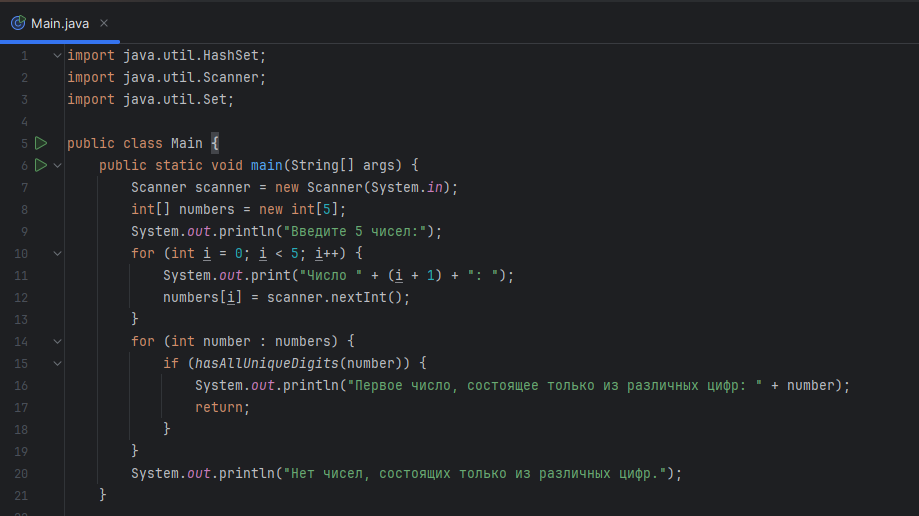


Рисунок 6.1 – Функция main()

В главной функции main() подключён объект scanner для чтения пяти чисел, введённых пользователем. Пять чисел сразу записываются в одномерный массив numbers[]. После вызывается функция пользователя hasAllUniqueDigits().

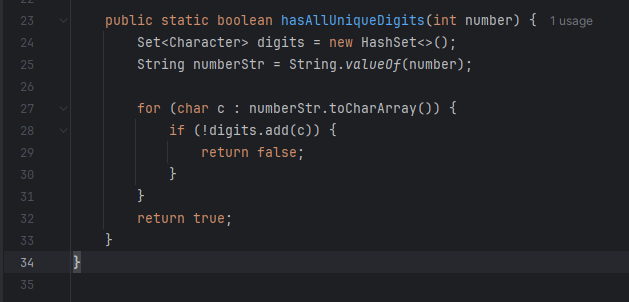


Рисунок 6.2 – Функция пользователя hasAllUniqueDigits()

В пользовательской функции создается объект типа множество для хранения элементов. Затем введённые числа преобразуется в строку символов. Метод toCharArray() преобразует эту строку в массив символов.

После написания программы необходимо провести тестирование и проверить её соответствие поставленной задаче (рис. 6.3). В процессе тестирования удалось убедился, что программа корректно обрабатывает входные данные и дает ожидаемые выходные результаты.

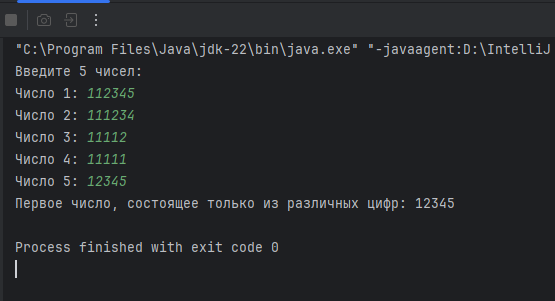
\

Рисунок 6.3 – Тестирование программы

В процессе работы удалось улучшить свои навыки работы с Java, что помогло более глубоко понять принципы программирования. Этот опыт был очень полезен для дальнейшего изучения языка.

**27.06.2024.**

В этот день была начата работа над Optional Task 2 (рис. 7.1) из курса Java, которое состоит из основного задания по созданию матрицы и ещё четырёх дополнительных. Работа с матрицами в программировании является одним из важнейших элементов, т.к. она проста и функциональна. Для её создания и полезного практического применения достаточно знать простейшие элементы языка программирования.

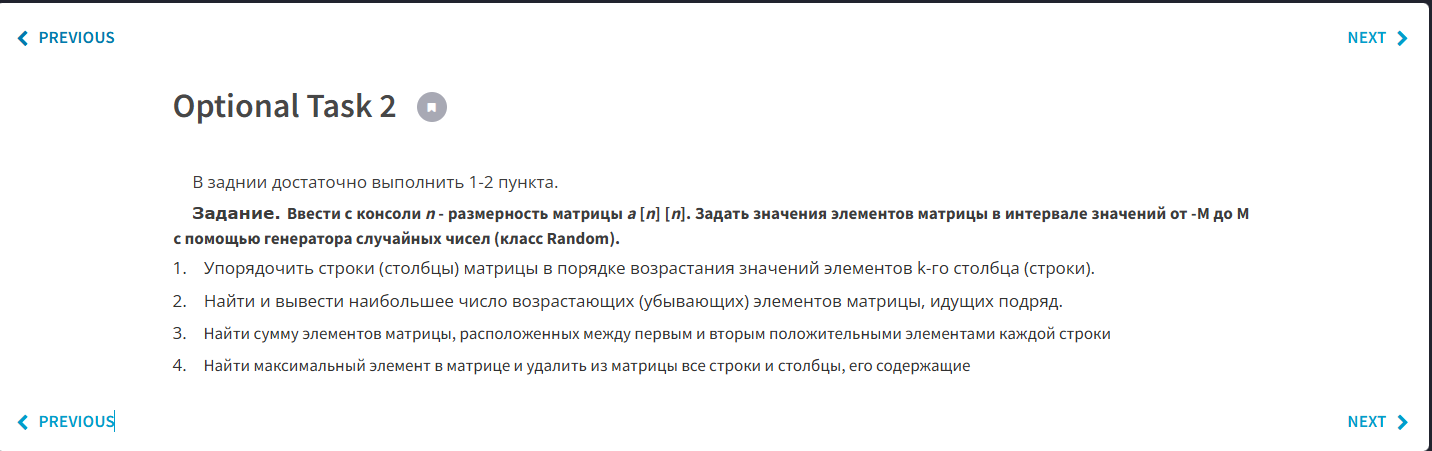


Рисунок 7.1 – Optional Task 2

Для решения поставленной задачи понадобилось использовать объект scanner, для считывания данных из консоли. После считывания данных программист задает значение переменной М, которая является неизменной в процессе выполнения программы и отвечает за диапазон значений матрицы. После выделяется динамическая память для двумерного массива и массив с помощью класса random заполняется случайны числами. В конце программы матрица с помощью вложенного цикла for выводится в консоль (рис. 7.2).

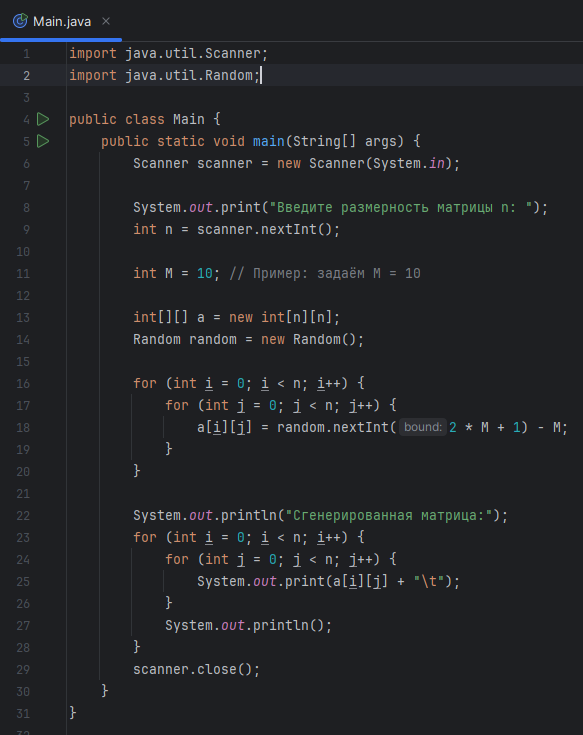


Рисунок 7.2 – Код программы

После написания программы обязательно проверяем результат в консоли. Изначально в программе была ошибка и результат не удовлетворял условию задачи. После исправления в консоли появился верный результат (рис. 7.3).

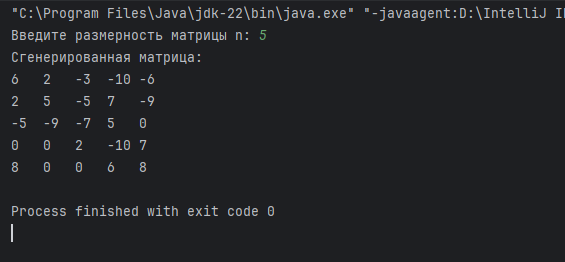


Рисунок 7.3 – Результат выполнения программы

Подводя итоги дня хочется отметить, что это задание познакомило с таким классом, как random. Так же повторялась работа с двумерными массива, на базе которых и была реализована квадратная матрица.

**28.06.2024.**

На этот день целью являлось решение следующего задания: упорядочить строки (столбцы) матрицы в порядке возрастания значений элементов k-го столбца (строки).

Для решения данной задачи пришлось прибегнуть к сторонним источникам в виде Google. Благодаря ему удалось узнать про такой объект в языке программирования Java, как компаратор (рис. 8.1).



Рисунок 8.1 – Пример использования компаратора

Компаратор в Java — это интерфейс, который позволяет определить порядок элементов, что особенно полезно при сортировке. Он предоставляет метод comparator, который используется для сравнения двух объектов.

После знакомства с таким методом, как компаратор, программа всё же была дописана (рис. 8.2).

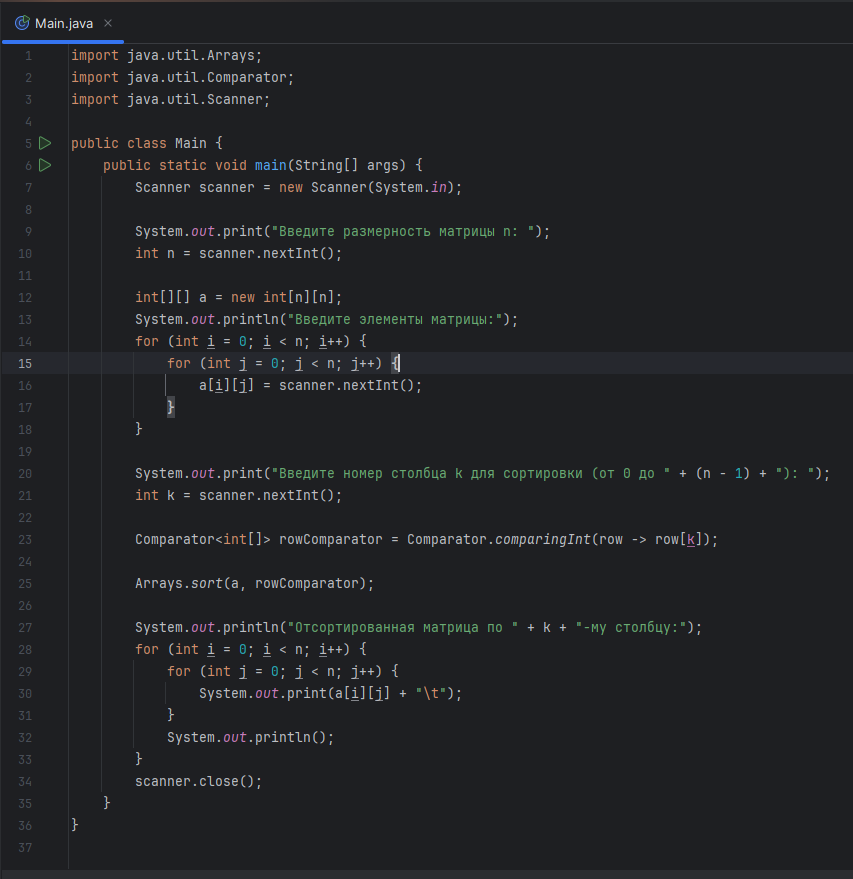


Рисунок 8.2 – Код программы

Изначально пользователь вводит в консоль размерность матрицы и элементы, тем самым заполняя её числами. Затем вводит номер столбца для сортировки. Всё это реализовано через объект scanner. Затем с помощью метода Comparator сравниваются строки матрицы по значениям элементов в kом столбце. С помощью Arrays.sort(a, rowComparator) происходит сортировка массива по заданному компаратору. И после вех этих действий с помощью цикла for происходит вывод матрицы на консоль.

Результат выполнения программы представлен на рисунке 8.3.

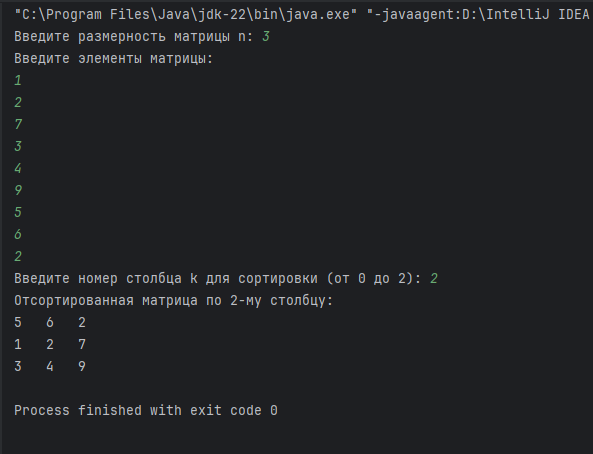


Рисунок 8.3 – Результат выполнения программы

Итогом дня является изучение новых возможностей языка программирования Java. Был изучен и применён в программе такой метод, как компаратор. С помощью компаратора можно сравнивать значения переменных. Это может быть очень полезно для сортировки данных, что очень важно для программиста.

**29.06.2024.**

Целью на этот день было решить финальное задание из Optional Task 2 курса Java.Fundamentals. Задача звучит так: найти сумму элементов матрицы, расположенных между первым и вторым положительными элементами каждой строки. Задача оказалась не самой простой. Весь код программы можно разбить на 2 основные части. Первая часть будет отвечать за создание матрицы размера М, заполненную случайными числами, и последующий вывод её на экран (рис. 9.1).

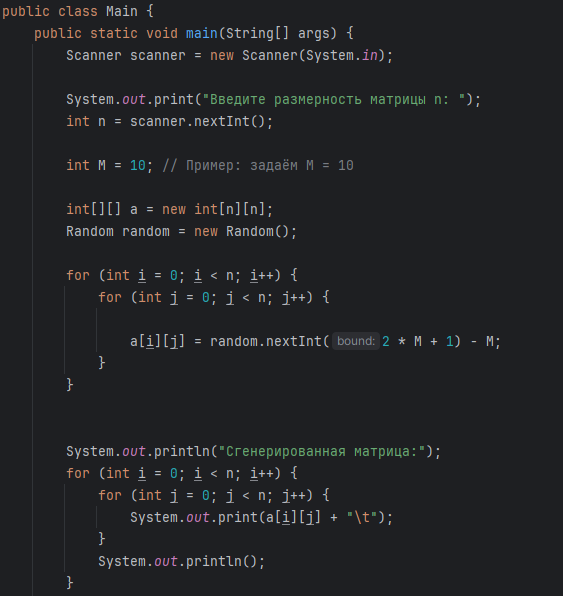


Рисунок 9.1 – Код первой части программы

В этом отрезке пользователь задаёт размер матрицы. После выделяется память для двумерного массива (основы матрицы). Далее этот массив заполняется случайными числами с помощью класса random. И уже после всех этих манипуляций с помощью вложенных циклов for в консоль выводится исходная матрица.

После данного участка начинаются арифметические вычисления (рис. 9.2).



Рисунок 9.2 – Вторая часть кода

На данном рисунке представлена часть кода, отвечающая за вычисление суммы значений элементов между первым и вторым положительными элементами каждой строки.

Для реализации данной задачи используются циклы, которые выполняются при истинности или же ложности выражений.

Для проверки истинности программы проводится тестирование (рис. 9.3).

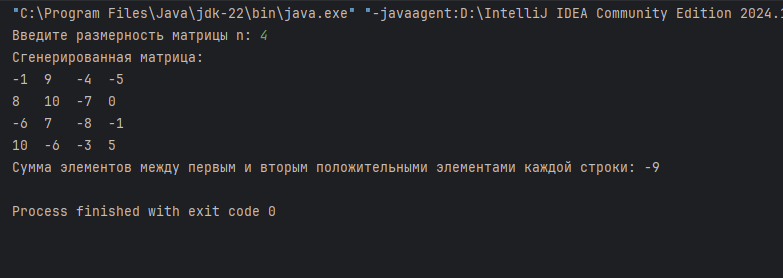


Рисунок 9.3 – Результат выполнения программы

После выполнения всех вышеописанных задач курс можно было бы считать оконченным. Однако в разделе лекций остались недоделанные тесты. Поэтому начинаем доделывать тесты (рис. 9.4).

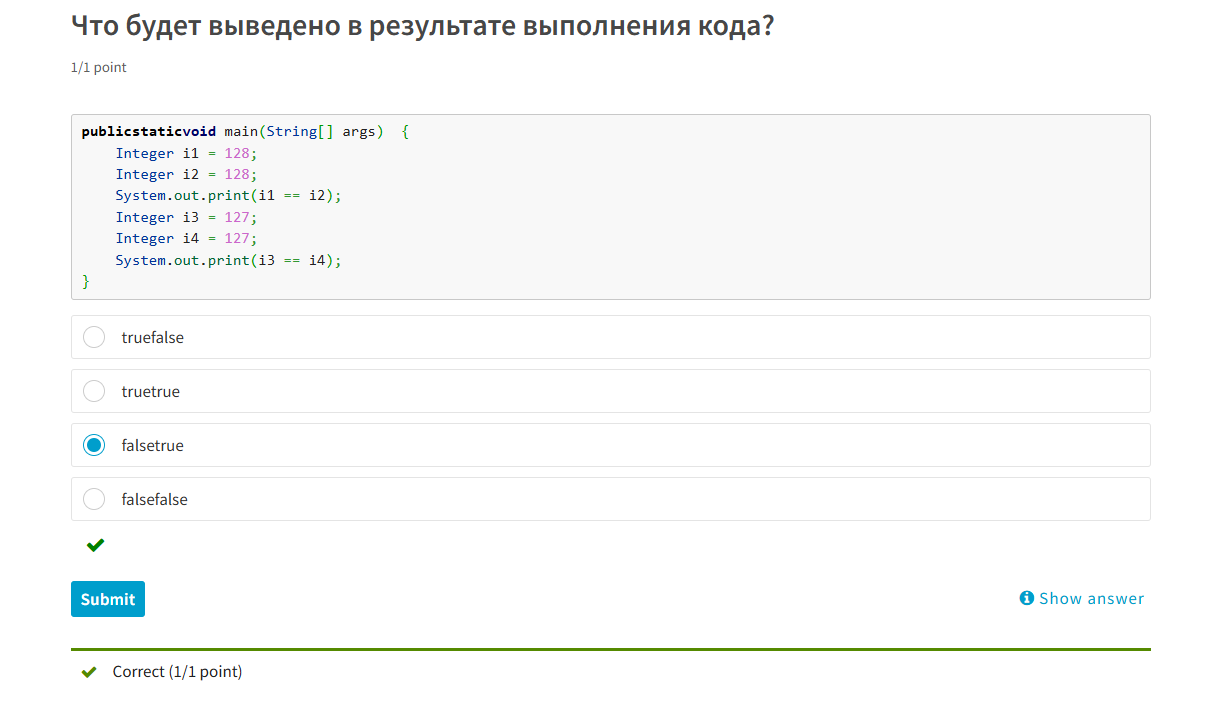


Рисунок 9.4 – Тест

Тесты были пройдены успешно, несмотря на то что вопросы в тесте были достаточно хитрыми и требовали внимательного изучения как вопроса, так и вариантов ответа.

Подводя итоги дня хочется сказать, что за день произвелась систематизация знаний, полученных за курс.

**Вывод**

По завершении курса Java.Fundamentals, который я проходил в рамках своего образовательного процесса, хочется сказать следующее: курс оказался ценным и полезным источником знаний в области языка программирования Java. Хочется отметить значимость данного курса для начинающих программистов, т.к. в неё описываются фундаментальные понятия, как типы данных, массивы, циклы. Все изложено очень лаконично и понятно.

Этот курс так же хорошо подойдёт и тем, кто знает много языков и захочет выучить новый. Каждая тема и концепция были представлены в понятной и систематизированной форме, что помогло мне легко усваивать информацию и применять ее на практике. Я был приятно удивлен тем, как хорошо структурированы лекции и модули курса, что способствовало моему углубленному пониманию курса.

Из минусов хочется отметить, что для выполнения практической части не всегда хватало теоретической базы. Не все достаточно важные моменты были описаны в курсе лекций.

В заключение, курс Java. Fundamentals является ценным ресурсом для студентов, желающих овладеть навыками работы с Java. Хорошая структура и подача материала обеспечивают эффективное усвоение информации.