Практические и домашние задания

## Вводное. Установка окружения (Java, Intellij IDEA). Запуск первой программы

## Типы данных. Переменные. Операторы

* Вывести последнюю цифру числа

**int number = 952;**

**int lastDigit = number % 10;**

**System.out.println("Last digit: " + lastDigit);**

* Найти сумму цифр ab + cd числа abcd

**int ab = abcd / 100; // Extracts the digits 'ab'**

**int cd = abcd % 100;**

* Вывести 3юю цифру 5ти-значного числа

**int thirdDigit = (number / 100) % 10;**

* Вывести длину отрезка, проходящей через две точки
* A screenshot of a computer program

  Description automatically generated with medium confidence
* Имеется прямоугольное отверстие размера a на b. Определить можно ли полностью закрыть отверстие круглой картонкой радиусом r  
  Пример:  
  6

8

5

Картонка с радиусом 5 закрывает полностью отверстие размера 6 \* 8

A picture containing text, electronics, screenshot, display

Description automatically generated

* Вывести уравнение прямой, проходящей через две точки
* A picture containing text, screenshot, font

  Description automatically generated

## Операторы if/switch

* Напишите программу, которая будет переводить секунды в форматированный вид - часы минуты секунды учитывая окончания слов  
  Пример:   
  1249  
  20 минут 49 секунд  
  648958  
  180 часов 15 минут 58 секунд
  + Добавьте 3 строчку, которая будет выводить недели, сутки, часы минуты и секунды

**Solution: TimeTransformer\_1**

* Напишите программу расчета даты следующего дня введя 3 числа - день месяц и год

Пример:  
15

3

2000

16 3 2000

**Solution:** NextDay\_2

* На некотором предприятии инженер Петров создал устройство, на табло которого показывается количество секунд, оставшихся до конца рабочего дня. Когда рабочий день начинается ровно в 9 часов утра — табло отображает «28800» (т.е. остаётся 8 часов), когда времени 14:30 — на табло «9000» (т.е. остаётся два с половиной часа), а когда наступает 17 часов — на табло отображается «0» (т.е. рабочий день закончился).  
  Программист Иванов заметил, как страдают офисные сотрудницы — им неудобно оценивать остаток рабочего дня в секундах. Иванов вызвался помочь сотрудницам и написать программу, которая вместо секунд будет выводить на табло понятные фразы с информацией о том, сколько полных часов осталось до конца рабочего дня. Например: «осталось 7 часов», «осталось 4 часа», «остался 1 час», «осталось менее часа».  
  Итак, в переменную n должно записываться случайное (на время тестирования программы) целое число из [0;28800], далее оно должно выводиться на экран (для Петрова) и на следующей строке (для сотрудниц) должна выводиться фраза о количестве полных часов, содержащихся в n секундах.  
  Пример:  
  23466  
  Осталось 6 часов

**Solution:** **TimeSimplifier\_3**

* Найти среднее число из трех, введенных с клавиатуры

**Solution: MiddleValueCalculator\_4**

* Создать программу, которая будет находить вещественные корни [квадратного уравнения](http://ru.wikipedia.org/wiki/%D0%9A%D0%B2%D0%B0%D0%B4%D1%80%D0%B0%D1%82%D0%BD%D0%BE%D0%B5_%D1%83%D1%80%D0%B0%D0%B2%D0%BD%D0%B5%D0%BD%D0%B8%D0%B5) ax²+bx+c=0, либо сообщать, что корней нет.

**Solution: QuadraticEquationSolver\_5**

* Определить принадлежность точки кругу
* Определить является ли целое число чётным либо нечётным
* По порядковому номеру пальца руки необходимо вывести его название на экран.
* По порядковому номеру дня недели необходимо вывести его название на экран.

## Циклы

* Необходимо вывести на экран таблицу умножения на 5
* Необходимо вывести на консоль такую последовательность чисел:
  + 1 2 4 8 16 32 64 128 256 512
* Необходимо написать программу, которая бы вывела в консоль звездочки вот таким образом, как на картинке ниже.  
  \* \* \* \* \*   
  \* \* \* \* \*   
  \* \* \* \* \*
* Необходимо написать программу, которая бы вывела в консоль звездочки вот таким образом, как на картинке ниже.  
  \*  
  \* \*  
  \* \* \*  
  \* \* \* \*  
  \* \* \* \* \*
* Найдите сумму и произведение цифр числа, введенного с клавиатуры
* Найдите самую большую цифру числа, введенного с клавиатуры, а также ее индекс
* Создайте программу, вычисляющую [факториал](https://ru.wikipedia.org/wiki/%D0%A4%D0%B0%D0%BA%D1%82%D0%BE%D1%80%D0%B8%D0%B0%D0%BB) натурального числа n, которое пользователь введёт с клавиатуры
* Выведите на экран первые 10 членов последовательности [Фибоначчи](https://ru.wikipedia.org/wiki/%D0%A7%D0%B8%D1%81%D0%BB%D0%B0_%D0%A4%D0%B8%D0%B1%D0%BE%D0%BD%D0%B0%D1%87%D1%87%D0%B8)
* В американской армии считается несчастливым число 13, а в японской — 4. Перед международными учениями штаб российской армии решил исключить номера боевой техники, содержащие числа 4 или 13 (например, 40123, 13313, 12345 или 13040), чтобы не смущать иностранных коллег. Если в распоряжении армии имеется 100 тыс. единиц боевой техники и каждая боевая машина имеет номер от 00001 до 99999, то сколько всего номеров придётся исключить?
* В городе N проезд в трамвае осуществляется по бумажным отрывным билетам. Каждую неделю трамвайное депо заказывает в местной типографии рулон билетов с номерами от 000001 до 999999. «Счастливым» считается билетик у которого сумма первых трёх цифр номера равна сумме последних трёх цифр, как, например, в билетах с номерами 003102 или 567576. Трамвайное депо решило подарить сувенир обладателю каждого счастливого билета и теперь раздумывает, как много сувениров потребуется. С помощью программы подсчитайте сколько счастливых билетов в одном рулоне?
* В городе N есть большой склад на котором существует 50000 различных полок. Для удобства работников руководство склада решило заказать для каждой полки табличку с номером от 00001 до 50000 в местной типографии, но когда таблички напечатали, оказалось что печатный станок из-за неисправности не печатал цифру 2, поэтому все таблички, в номерах которых содержалась одна или более двойка (например, 00002 или 20202) — надо перепечатывать. Напишите программу, которая подсчитает сколько всего таких ошибочных табличек оказалось в бракованной партии.
* Для введенного целого числа определить является ли это число простым
* Электронные часы показывают время в формате от 00:00 до 23:59. Подсчитать сколько раз за сутки случается так, что слева от двоеточия показывается симметричная комбинация для той, что справа от двоеточия (например, 02:20, 11:11 или 15:51)

## Массивы

* Найти максимальное значение из массива
* Найти минимальное значение из массива
* Найти сумму цифр массива
* Найти максимальный из элементов массива с четными индексами
* Найти минимальный из элементов массива с нечетными индексами
* Разложить положительные и отрицательные числа по разным массивам
* Исключить одинаковые элементы массива (каждое значение должно присутствовать в единственном экземпляре)
* Найти максимальный и минимальные элементы массива и вывести их индексы на экран
* Выполнить слияние 2 упорядоченных массивов
* Найти строку матрицы с максимальной суммой элементов
* Сделать транспонирование матрицы (замена строк на столбцы)
* Создать массив заполнить его случайными элементами, распечатать, перевернуть, и снова распечатать
  + При перевороте элементов не желательно создавать еще один массив
* Создать двумерный квадратный массив размера n. Заполнить его случайными числами таким образом:
  + числа по диагонали равны 0;
  + вверху и снизу от пересечения диагоналей находятся только положительные числа;
  + слева и справа от пересечения диагоналей находятся только отрицательные числа;

Вывести массив на экран.

* + Найти сумму всех элементов
  + Найти среднее арифметическое всех элементов, которые больше *суммы всех элементов  
    Пример:  
    0 4 5 5 3 8 0*

*-2 0 5 6 3 0 -4*

*-2 -8 0 6 0 -1 -2*

*-5 -9 -2 0 -9 -7 -3*

*-6 -2 0 1 0 -1 -5*

*-2 0 3 5 2 0 -8*

*0 4 2 9 1 3 0*

*Суммa всех элементов: -3*

*Cреднее арифметическое всех элементов больше -3: 1.6579*

## Классы и объекты

* Создайте класс с именем fraction, содержащий два поля типа int - числитель и знаменатель обыкновенной дроби.
  + Конструктор класса должен инициализировать их заданным набором значений.
  + Создайте метод класса, который будет выводить дробь в текстовую строку в формате x / y;
  + метод, добавляющий (сложение) к текущей дроби дробь, переданную ему в параметре и возвращающий дробь - результат сложения;
  + метод, умножающий (произведение) текущую дробь на число типа double, переданное ему в параметре и возвращающий дробь - результат умножения;
  + метод, делящий (деление) текущую дробь на число типа double, переданное ему в параметре и возвращающий дробь - результат деления.
* Создайте класс Phone, который содержит поля number, model и weight.
  + Создайте три экземпляра (instance) этого класса.
  + Выведите в консоль их значения.
  + Добавить в класс Phone методы: receiveCall (один параметр – имя звонящего). Выводит на консоль сообщение “Звонит {name}”. getNumber – возвращает номер телефона. Вызвать эти методы для каждого из объектов
  + Добавить второй метод receiveCall, который принимает два параметра - имя звонящего и номер телефона звонящего. Вызвать этот метод.
  + Создать метод sendMessage с аргументами переменной длины. Данный метод принимает на вход номера телефонов, которым будет отправлено сообщение. Метод выводит на консоль номера этих телефонов.

## Строки. Регулярные выражения

* Введите строку. Если она начинается на 'abc', то заменить их на 'www', иначе добавить в конец строки 'zzz'.
* Вводится ненормированная строка, у которой могут быть пробелы в начале, в конце и между словами более одного пробела. Привести ее к нормированному виду, т.е. удалить все пробелы в начале и конце, а между словами оставить только один пробел.
* Введите строку (текст). Найдите наибольшее количество идущих подряд цифр.
* Поменяйте местами первое самое длинное слово с последним самым коротким
* Найти в строке все знаки препинания. Подсчитать общее количество их.
* Имеется строка с текстом. Подсчитайте уникальное количество слов в тексте. Учитывайте, что слова могут разделяться несколькими пробелами, знаками препинания. Пробелы могут присутствовать в начале и конце текста. Примеры уникальности:
  + *dog* и *Dog* - одинаковые
  + *dog* и *dogs* - разные

Пример:  
Listen to the news from today and read the text at the same time. Listen to the news from today without reading the text.  
14

* RegEx. Шестнадцатеричный цвет. Символ # (необязательно), затем слово, состоящее из букв от a до f или цифр, длиной 3 или 6:   
  Пример: FFFFFF #000 101011

**Solution: String regex = "#?[A-Fa-f0-9]{3}(?:[A-Fa-f0-9]{3})?";**

* RegEx. Email. Общий вид — логин@поддомен.домен. Логин, как и поддомен — слова из букв, цифр, подчеркиваний, дефисов и точек. А домен (имеется в виду 1го уровня) — это от 2 до 6 букв и точек.   
  Пример: myemail@gmail.com a.petrov@gmail.com.au [coder4575@yandex.ru](mailto:coder4575@yandex.ru)
* RegEx. IP адрес. 4 группы цифр (от 1 до 3 цифр в каждой) разделены точками. Группа содержит число в диапазоне от 1 до 255  
  Пример: 127.0.0.1 255.255.255.253 100.99.44.2
* RegEx. URL. Первым делом — необязательный протокол (http:// или https://), затем последовательность букв, цифр, дефисов, подчеркиваний и точек (домены уровня > 1), потом домен нулевого уровня (от 2 до 6 букв и точек) и, наконец, файловая структура — набор слов из букв, цифр, дефисов, подчеркиваний и точек со слэшем в конце. Всё это может завершаться опять-таки слэшем.   
  Примеры:  
  https://www.it-academy.by/course/java-developer/  
  https://catalog.onliner.by/bike?is\_prime=1

## Наследование и полиморфизм. Внутренние, анонимные классы

* Создать класс Man (человек), с полями: имя, возраст, пол и вес. Определить методы задания имени, возраста и веса. Создать производный класс Student, имеющий поле года обучения. Определить методы задания и увеличения года обучения.
* Реализовать иерархию классов, описывающую трехмерные фигуры
* Определить иерархию домашней техники. Включить некоторые в розетку. Рассчитать потребляемую мощность. Провести сортировку приборов в квартире на основе одного из параметров. Найти кухонный прибор в квартире, соответствующий заданному диапазону параметров.
* Авиакомпания. Определить иерархию самолетов, вертолетов, квадрокоптеров и т.д. Создать авиакомпанию. Посчитать общую вместимость и грузоподъемность. Провести сортировку летных средств компании по дальности полета. Найти самолет в компании, соответствующий заданному диапазону параметров.
* Парк общественного транспорта. Определить иерархию различных видов ОТ. Создать парк ОТ. Рассчитать стоимость автопарка. Провести сортировку машин парка по расходу топлива. Найти автобус в компании, соответствующий заданному диапазону параметров.
* Реализовать иерархию цветов (розы, гвоздики, тюльпаны, лилии и... что-то на свой вкус).
  + Создать несколько объектов-цветов.
  + Собрать букет и определить его стоимость.
  + Определить все цвета, используемые в букете.
  + Определить когда весь букет завянет.

В букет может входить несколько цветов одного типа.

## Дженерики. Перечисляемые типы

* Написать класс, который умеет хранить в себе массив любых типов данных (int, long etc.). Реализовать метод, который возвращает любой элемент массива по индексу.
* Реализовать класс **Pair**, который будет содержать 2 значения любого типа. Класс умеет выводить:
  + first() - возвращает 1ый элемент
  + last() - возвращает 2ой элемент
  + swap() - меняет элементы местами
  + replaceFirst() - заменяет 1ый элемент на новый
  + replaceLast() - заменяет 2ой элемент на новый
* Написать итератор по массиву (посмотрите interface Iterator). Ваш класс должен содержать 2 метода - **hasNext()** и **next()**
* Написать итератор по двумерному массиву
* Создать enum - FoodType (MEAT, FISH, FRUIT, VEGETABLE, MILK, UNKNOWN)
  + Создать базовый абстрактный класс Food и в нем абстрактный метод public FoodType getFoodType();
  + Создать 2-3 класса наследника Food и реализовать методы get/set name, a также переопределить getFoodType для каждого.
  + Создать отдельный класс с методом проверк boolean isVegetarian(Food[] foods) в метод передается набор ингредиентов, на выходе true если все они вегетарианские.

## Коллекции. List, Set, Map

* Напишите метод, который получает на вход массив элементов типа К (Generic) и возвращает Map<K, Integer>, где K — значение из массива, а Integer — количество вхождений в массив.

Сигнатура метода: <K> Map<K, Integer> arrayToMap(K[] ks)

* Реализуйте свой собственный стек (очередь) со стандартными функциями push() и pop(), а так же с дополнительной функцией max() за O(1)
* Реализовать функцию подсчета количества разных символов в строке - public Map<Character, Integer> getCharsCount(String s)  
  Пример: строка “baaccc” и результат: ‘b’ -1, ‘a’ - 2, ‘c’ - 3
* Создать класс User с полями firstName lastName.
  + Добавить getters/setters
  + Реализовать методы equals & hashCode
  + Создавать два экземпляра User, оба с именем John Smith
  + Добавить оба экземпляра в HashSet

## Исключения

* Реализуйте игру “Угадай число” от 1 до 100 с подсказкой - “искомое число больше/меньше”  
  При неправильном вводе, обработайте ошибку и сообщите пользователю

## Работа с файлами

* Даны 2 файла - in1.txt и in2.txt в каждом файле по 1000 чисел от 1 до 100000. Создайте файл out.txt, который будет содержать все отсортированные числа из файлов in1.txt и in2.txt. Файлы in1.txt и in2.txt можно создать с помощью кода.
* Создать файл с текстом, прочитать, подсчитать в тексте количество знаков препинания и слов.
* Записать в двоичный файл 30 случайных чисел от 1 до 100. Прочитать записанный файл распечатать числа и их среднее арифметическое.
* Прочитать файл по строкам. Подсчитать количество символов 'a' в файле. Заменить содержимое файла на число символов 'a'

## Lambda, Streams API

* Дана коллекция Collection<String> col. С помощью stream api:
  + узнать, содержит ли какая-нибудь из строк слово login
  + найти самую длинную строку
  + найти самую короткую строку
  + найти строки-слова (не содержат пробелов и знаков препинания)
  + получить все слова используемые в строках
* Дана коллекция Collection<Integer> col. С помощью stream API:
  + найти минимальное число
  + найти максимальное число
  + найти среднее арифметическое чисел
  + найти произведение всех чисел
  + найти сумму всех чисел
  + найти сумму всех цифр
* Создать список строк List<String> myList = Arrays.asList("a1", "a2", "a3", "b1", "b3", "c2", "c1", "c5"); Используя stream и lambda выполнить следующие действия:
  + Удалить все содержащие “3”
  + Отсортировать сперва по числу (по возрастанию), потом по букве (по убыванию)
  + Отбросить первый и последний элемент
  + Привести в uppercase
  + Выдать на печать результат
  + Напечатать количество оставшихся элементов

## Класс Thread и интерфейс Runnable

* Пользователь вводит с клавиатуры значение в массив. После чего запускаются два потока. Первый поток находит максимум в массиве, второй — минимум. Результаты вычислений возвращаются в метод main()
* Заданы три целочисленных массива. Записать эти массивы в файл в параллельных потоках. Создать класс SaveAsThread для представления потока, который записывает целочисленный массив в файл.
* Создать класс Generator extends Thread
  + В конструктор класса передается List<Integer> ссылка на который сохраняется в классе.
  + В методе run() в list по одному добавляются случайные числа и после каждого добавления поток “засыпает” (sleep) на 200 msec
  + В лист должно добавится 100 чисел

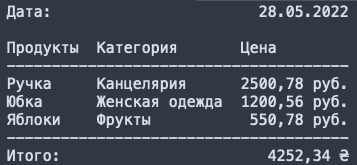
## Пулы потоков. Асинхронные вычисления

* Создайте матрицу размера n\*m. Создайте два потока, один из которых перемножает все значения в столбцах и потом находит сумму, а другой поток перемножает все значения в строках и тоже потом находит сумму. Определите какое число больше.

## Взаимодействие потоков. Producer - Сonsumer

* **Producer/Consumer**. Реализуйте классы **Store**, **Producer**, **Consumer** c помощью *java.util.concurrent.Semaphore*
* *Задача о читателях и писателях. К базе данных имеют доступ два типа процессов - читатели и писатели. Первые могут выполнять транзакции, которые могут только читать данные, для вторых добавляется возможность их читать. Необходимо обеспечить такой доступ к базе данных, что в любой момент времени в ней находится либо один писатель и ни одного читателя, либо сколь угодно читателей, но ни одного писателя.*
* *Задача критической секции. N процессов многократно повторяют сначала критическую а затем не критическую секции. Под критической секцией понимается последовательность операторов, содержащих работу с некоторым общим ресурсом, то есть работу, которую в один момент времени должен выполнять только один процесс. Задача состоит в том, чтобы разработать такие вход и выход из критической секции, которые удовлетворяют следующим свойствам:*
  + *Взаимное исключение. Очевидное свойство, требующее того, что в каждый момент времени в секции находится только один процесс.*
  + *Отсутствие взаимной блокировки. При желании входа в секцию несколькими потоками, хотя бы один это сделает.*
  + *Отсутствие излишних задержек. Если процесс хочет войти в секцию и там нет других процессов, он это осуществит.*
  + *Возможность входа. Процесс, который хочет войти в секцию, когда-нибудь сделает это.*

## Date Time API

* Реализовать метод isDateOdd(String date) так, чтобы он возвращал true, если количество дней с начала года - нечетное число, иначе false  
  Пример: “JANUARY 1 2000” => true “JANUARY 2 2020” => false
* Распечатать график праздников на текущий год используя форматирование.
* Интернет-магазин
  + Создать класс Товар, имеющий переменные имя, цена, рейтинг.
  + Создать класс Категория, имеющий переменные имя и массив товаров. Создать несколько объектов класса Категория.
  + Создать класс Basket, содержащий массив купленных товаров.  
    

## Принципы дизайна ПО. SOLID. Паттерны

* Планеты солнечной системы. Создайте три Singleton-класса: Sun, Moon и Earth. Затем реализуйте в них интерфейс Planet с некоторыми операциями.
* Фигругы. Создайте класс ShapeType с типами фигур, интерфейс Shape с методом подсчета площади фигуры и классы фигур которые наследуют его, класс ShapeFactory, который будет создавать объекты фигур

## Reflection API. Аннотации

* Изучение класса. загрузите класс "java.util.LinkedList". Выведите все поля класса, родительский класс, методы класса, конструкторы. Выведите информацию о внутренних классах, если они есть.
* Изучение класса. загрузите класс "java.util.HashMap". Выведите все поля класса, родительский класс, методы класса, конструкторы. Выведите информацию о внутренних классах, если они есть.
* Создайте класс TestAnnotation и в нем метод public String old() возвращающий любой текст
  + Создайте вызывающий класс CallOld с методом call() вызывающим и печатающим результат old()
  + Скомпилируйте и убедитесь в отсутствии warning
  + Пометьте old() аннотацией @Deprecated
  + Скомпилируйте и убедитесь что появится warning
  + Пометьте вызов аннотацией @SuppressWarning чтобы не было warning