




# Почему вы не можете не использовать API

Семинар 2





## Что будет на уроке сегодня

-  повторим теорию о String, StringBuilder, Exception Handling, Logging, Import
-  получим практические навыки в работе с файловой системой и файлами, обработке исключений и логировании, работой со строками
-  научимся составлять программы с логированием и корректной обработкой исключений



Повторим основные теоретические  
моменты с лекции



## Снова о строках - String

Следует помнить, что объекты класса String являются **неизменяемыми** (immutable).

String – это объект и он может быть **null**!

length()	Возвращает длину строки
isEmpty()	Проверяет пустая ли строка
charAt()	Возвращает символ на i-ой позиции
toCharArray()	Возвращает представление строки в виде массива символов
equals()	Сравнивает строки с учетом регистра. Возвращает boolean.
equalsIgnoreCase()	Сравнивает строки без учета регистра. Возвращает boolean.
compareTo()	Сравнивает строки с учетом регистра. Возвращает int.
compareToIgnoreCase()	Сравнивает строки без учета регистра. Возвращает int.
split()	Возвращает представление строки в виде массива подстрок. Разбиение строки происходит по входному правилу
replace()	Заменяет один символ (последовательность символов) на другой (другую последовательность)



## String vs StringBuffer vs StringBuilder

	String	StringBuffer	StringBuilder
Когда использовать	При работе со строками, которые редко будут модифицироваться	При работе со строками, которые часто будут модифицироваться в многопоточной среде	При работе со строками, которые часто будут модифицироваться, в однопоточной среде



# Обработка исключений и логгирование

За минимально “базовое” логгирование отвечает класс `java.util.logging.Logger`.

```
public static void main(String[] args) {
    Logger logger = Logger.getLogger( name: "SimpleClass");
    try{
        int i = 0 / 0;
    } catch (Exception e){
        logger.severe(e.getMessage());
    }
}
```

fatal	Level.SEVERE	severe()
error	Level.SEVERE	severe()
warning	Level.WARNING	warning()
info	Level.INFO	info()
debug	Level.FINE	fine()
trace	Level.FINEST	finest()

```
public static void main(String[] args) {
    Logger logger = Logger.getLogger( name: "SimpleClass");
    try(FileOutputStream stream = new FileOutputStream(new File( pathname: "example.txt"))) {
        stream.write("data".getBytes());
        stream.flush();
    } catch (Exception e) {
        logger.severe(e.getMessage());
    }
}
```




А теперь практика!



## Задание №1

 Дано четное число  $N (>0)$  и символы  $s1$  и  $s2$ .

 Написать метод, который вернет строку длины  $N$ , которая состоит из чередующихся символов  $s1$  и  $s2$ , начиная с  $s1$ .





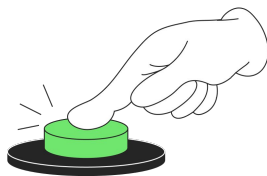
## Задание №1



Дано четное число  $N (>0)$  и символы  $s1$  и  $s2$ .



Написать метод, который вернет строку длины  $N$ , которая состоит из чередующихся символов  $s1$  и  $s2$ , начиная с  $s1$ .



Поставьте видео на паузу и  
выполните задание



## Задание №2

 Напишите метод, который сжимает строку.

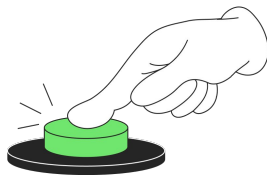
 Пример: вход aaaabbbbcdd.



## Задание №2

 Напишите метод, который сжимает строку.

 Пример: вход aaaabbbbcdd.



Поставьте видео на паузу и  
выполните задание



## Задание №3



Посчитайте сколько ”драгоценных камней” в куче ”обычных камней”



Пример: jewels = “aB”, stones = “aaaAbbbbB”



Результат: ”a3B1”



## Задание №3



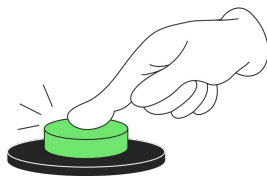
Посчитайте сколько ”драгоценных камней” в куче ”обычных камней”



Пример: jewels = “aB”, stones = “aaaAbbbbB”



Результат: ”a3B1”



Поставьте видео на паузу и  
выполните задание



## Задание №4



Вам дается строка  $S$  и целочисленный массив индексов  $\text{int index}[s.length]$ . Напишите программу, которая перетасует символы в  $S$  таким образом, что символ с  $i$ -й позиции переместится на индекс  $\text{index}[i]$  в результирующей строке.



Пример:  $s = \text{"cba"}$ ,  $\text{index} = [3, 2, 1]$



Результат  $\text{"abc"}$



## Задание №4



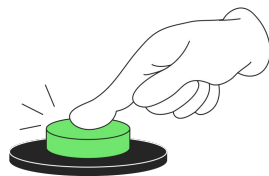
Вам дается строка  $S$  и целочисленный массив индексов  $\text{int index}[s.length]$ . Напишите программу, которая перетасует символы в  $S$  таким образом, что символ с  $i$ -й позиции переместится на индекс  $\text{index}[i]$  в результирующей строке.



Пример:  $s = \text{"cba"}$ ,  $\text{index} = [3, 2, 1]$



Результат  $\text{"abc"}$



Поставьте видео на паузу и  
выполните задание



# Домашнее задание





## ДЗ

1. Дана строка sql-запроса "select \* from students where ". Сформируйте часть WHERE этого запроса, используя StringBuilder. Данные для фильтрации приведены ниже в виде json строки.

Если значение null, то параметр не должен попадать в запрос.

Параметры для фильтрации: {"name":"Ivanov", "country":"Russia", "city":"Moscow", "age":"null"}

2. Реализуйте алгоритм сортировки пузырьком числового массива, результат после каждой итерации запишите в лог-файл.



## ДЗ (дополнительное)

3.\*\* Дана json строка (можно сохранить в файл и читать из файла)

```
[{"фамилия":"Иванов","оценка":"5","предмет":"Математика"}, {"фамилия":"Петрова","оценка":"4","предмет":"Информатика"}, {"фамилия":"Краснов","оценка":"5","предмет":"Физика"}]
```

Написать метод(ы), который распарсит json и, используя StringBuilder, создаст строки вида: Студент [фамилия] получил [оценка] по предмету [предмет].

Пример вывода:

Студент Иванов получил 5 по предмету Математика.

Студент Петрова получил 4 по предмету Информатика.

Студент Краснов получил 5 по предмету Физика.

4\*. К калькулятору из предыдущего ДЗ добавить логирование.



Подведем итоги



Напишите 3 вещи в  
комментариях, которым  
вы научились сегодня.





Спасибо за работу!