#### Цель практической работы

Научиться использовать параметризованные классы в контексте работы с уведомлениями.

#### Что нужно сделать

##### Задание 1. Работа с уведомлениями в приложении

Выполните задание в репозитории java\_basics в проекте **Generics**:

1. Ваше приложение должно уметь отправлять уведомления, то есть выводить сообщение в соответствующем формате в консоль.
2. Каждое уведомление должно имплементировать интерфейс Notification.
3. Уведомления отправляются клиентам по разным каналам через свой класс-сервис. Каждый класс для отправки уведомлений должен имплементировать интерфейс NotificationSender.

В приложении должны быть реализованы три вида уведомлений:

* EmailNotification — письмо на электронную почту.
* SmsNotification — СМС-сообщение.
* PushNotification — уведомление для приложений мобильных устройств.

**Особенности каждого класса уведомлений:**

1. EmailNotification:
   * Текст для электронной почты должен быть обозначен тегами <p>Текст сообщения</p>.
   * Например, если надо отправить сообщение для пользователя: «Спасибо за регистрацию на сервисе!», то при отправке через имейл текст преобразуется в: <p>Спасибо за регистрацию на сервисе!</p>.
   * Уведомление должно содержать дополнительные поля: тема сообщения, список адресов получателей.
2. SmsNotification:
   * Текст для СМС не меняется. Отправлять сообщение необходимо в оригинальном формате.
   * Например, если надо отправить сообщение для пользователя: «Спасибо за регистрацию на сервисе!», то при отправке через СМС текст преобразуется в: Спасибо за регистрацию на сервисе!
   * Уведомление должно содержать дополнительное поле — список телефонных номеров получателей.
3. PushNotification:
   * Пуш перед основным текстом сообщения должен содержать эмодзи 👋 для привлечения внимания.
   * Например, если надо отправить сообщение для пользователя: «Спасибо за регистрацию на сервисе!», то при его отправке через СМС текст преобразуется в: 👋 Спасибо за регистрацию на сервисе!
   * Пуш-уведомление должно содержать дополнительные поля: заголовок уведомления, аккаунт пользователя (строка).

##### Задание 2. Сервисы по отправке уведомлений

1. Создайте классы, имплементирующие интерфейс NotificationSender для обработки каждого типа уведомлений. У вас должно получиться три класса: SmsNotificationSender, EmailNotificationSender и PushNotificationSender, каждый из которых содержит следующие методы:
   * void send() с одним аргументом, в который передаётся одно уведомление. Метод получает форматированное сообщение из объекта уведомления, дополнительные поля и выводит его в консоль.
   * void send() с одним аргументом, который принимает список уведомлений и распечатывает сообщения и дополнительные поля каждого из них.

Класс NotificationSender должен работать только с наследниками Notification:

public interface NotificationSender<T extends Notification> {

void send(T notification);

void send(List<T> notifications);

}

1. Создайте по три уведомления каждого типа и выполните имитацию отправки уведомлений через сервис для отправки.

В результате в консоли должны быть распечатаны все данные каждого уведомления. Например, возможен такой вывод:

EMAIL

subject: Успешная регистрация!

receivers: oleg@java.skillbox.ru, masha@java.skillbox.ru, yan@java.skillbox.ru

message: <p>Спасибо за регистрацию на сервисе!</p>

SMS

receivers: +70001234567

message: Спасибо за регистрацию на сервисе!

PUSH

title: Успешная регистрация!

receiver: o.yanovich

message: 👋 Спасибо за регистрацию на сервисе!

В итоге в вашем проекте должно появиться следующее:

* три класса наследника Notification;
* три класса сервиса отправки уведомлений, наследники NotificationSender;
* создание в Main-классе метода main и работа с уведомлениями и сервисами.

##### Задание 3. Использование библиотеки Lombok

Замените весь шаблонный код классов на аннотации Lombok:

1. Подключите библиотеку Lombok к вашему проекту, добавив зависимость в pom.xml-файл в блок <dependencies>.

<dependency>

<groupId>org.projectlombok</groupId>

<artifactId>lombok</artifactId>

<version>1.18.28</version>

<scope>provided</scope>

</dependency>

1. Замените геттеры, сеттеры и конструкторы на аннотации Lombok. Для этого выберите необходимые аннотации [в документации Lombok](https://projectlombok.org/features/).

В результате в вашем проекте не должно остаться шаблонного кода.