

Домашнее задание

Структуры и `enum`-ы

Всего баллов:

10

1. Дни недели

Баллы:

2

Билли очень ленивый, и не хочет перерабатывать, поэтому он хочет точно знать, что сейчас рабочий день: только так он может заставить себя что-то делать. Конечно же, Билли постоянно ждёт пятницы и часто задумывается, «какой же завтра день?»

Кроме того, у Билли плохая память, и он не может обойтись без `struct Day`, который поможет ему интерпретировать информацию о том, какой сегодня день, полученную от коллег.

Реализовать функции:

- `fn next(self) → Day` – следующий день недели
- `fn stringify(self) → &'static str` – строковое представление с большой буквы
- `fn is_workday(self) → bool`
- `fn is_weekend(self) → bool`

Конечно, это можно сделать с меньшим объёмом дубликации кода (и, возможно, более эффективно) — через `unsafe`, библиотеки или макросы, но здесь не надо таким париться.

2. To Ban or Not to Ban?

Баллы:

2

Вы испокон веков работаете в Коммуникационном Центре. Но недавно всё изменилось: к власти пришёл Император Палпатин, и у него большие планы на КЦ. Вас перекинули на новый, перспективный проект по *инспектированию* сообщений, которые пользователи посылают друг другу.

У Императора много идей, за что стоит банить пользователей:

- Номер школы
- Слишком часто делает запросы
- Не отвечает
- Длинное имя
- Не смог ввести капчу
- Предлагает нюдсы
- Оскобляет/шутит на «чувствительные» темы
- Нарушение правил сообщества
- Авторские права
- Произнёс слово негр/суицид/роскомнадзор (заменять всё на РКН/Twitch?)
- `random() > 0.1`: Рандом превысил норму, что-то тут нечисто.
- Such a lonely day (should be banned)
- Пытается сделать DROP TABLE USERS
- Подрыв государственных устоев
- Белый гетеросексеальный мужчина
- Получил двойку
- Выделяется адреналин → чего-то боится → виновен

- Выложил фотографию Винни-Пуха
- Sentiment analysis: если унывает, надо с этим что-то сделать!

Но в качестве MVP было решено *банить исключительно за использования N-word (слова «негр»)* (Император недолюбливает людей с тёмной кожей).

От вас требуется реализовать

```
1 struct Messenger {
2     users: HashMap<usize, UserData>
3 }
4
5 struct UserData {
6     user: User,
7     communicated_with: Vec<usize>
8 }
```

со следующими методами:

```
1 fn register(&mut self, newbie: User) { ... }
2 fn inspect_message(
3     &mut self,
4     message: &str,
5     from_user_id: usize,
6     to_user_id: usize
7 ) → Option<User> { ... } // Returns ownership of banned
    if there was one
```

- В метод `register` не могут передать несколько `id`
- Метод `inspect_message` проверяет сообщение и банит отправителя (удаляя его из списка и возвращая владение над ним), если в нём содержится `"негр"` как подстрока в любом регистре (то есть попытка обойти ограничения через `"нЕгРа"` не выйдет).

Последний метод (см. ниже) нужен Императору на случай смуты — чтобы забиндить его на красную кнопку «совершить чистку».

2.1. Relaxation

Надо бы избавиться от знакомых подозрительных личностей... Да что там, и от знакомых их знакомых тоже. И знакомых знакомых знакомых. Ну, вы поняли.

Надо реализовать метод, который позволит провести большую чистку — удалить всех людей, *косвенно знакомыми* с забаненными:

```
1 /// Bans indirect acquaintances of all banned users
2 /// Returns ownership over banned users
3 fn do_ban_relaxation(&mut self) → Vec<User> { ... }
```

- Метод `do_ban_relaxation` возвращает список всех *косвенно знакомых* с кем-либо из ранее забаненных.

2 человека считаются

- *непосредственно знакомыми*, если кто-нибудь из них послал сообщение другому
- *косвенно знакомыми*, если существует цепочка из *непосредственных знакомств*, которая приводит от одного к другому

Баллы: 2

Баллы: 4

3. Коллега










Здесь придётся работать с *некомпилирующим* кодом. Чтобы он не портил остальные тесты, мы его отделили в отдельный *модуль*. Что такое модуль, пока задумываться не надо — просто когда будете готовы делать эту задачу, нужно *раскомментировать* 26 строчку в `Cargo.toml` (там, где `default = [colleague]`, оно включит задание с коллегой как «фичу») и перейти в `colleague.rs`.

POV: Вы нашли работу мечты — вы будете программировать на Rust (правда, пока неизвестно, что именно). Несколько подозрительным вам показалось, что ваш предшественник на этом месте уволился по собственному желанию. Поспрашивав у коллег, вы накопили некоторую информацию:

- Он был талантливым программистом, и оставил после себя очень важный код, с которым вам предстоит работать
- сейчас он уже «в возрасте», и ему тяжело усваивать новую информацию. В частности, он не разбирается в правилах `ownership` в Rust.
- Он был не в восторге от эзотерических практик вашей компании (при ошибке компиляции программиста бьёт *шокером*).

Ваша задача — *заставить код компилироваться*, внося минимальные изменения сохраняя алгоритм и структуру кода (тесты проверяют, что вы не убрали какие-то структурные блоки совсем). В идеале — ещё объяснить в комментариях, что было не так.

К счастью, вам выдали буклет о стратегиях удовлетворения компилятора, в особенности — Borrow checker-а — части компилятора, отвечающей за анализ заимствований (`borrow`) и владения (`ownership`):

- Принимать в функции правильно по значению, ссылке или мутабельной ссылке
- Оборачивать или разворачивать ошибки (`Option`, `Result`)
- [в крайнем случае, если вы осознанно решили потратить память и время на глубокое копирование] Использовать `.clone()`, чтобы компилятор не ругался, что вы используете потреблённое значение ещё раз или зоны `&mut`-заимствований пересекаются (для этого должен быть реализован `Clone` ← можно сделать с помощью `#derive`-макросов)
- Использовать более специфичные функции, которые больше знают о ситуации, и специфично удовлетворить borrow-checker (например, `[T]::swap` вместо `std::mem::swap`)
-  Эта стратегия вас недоступна: **перейдите на следующий уровень!**
-  Эта стратегия вас недоступна: **перейдите на следующий уровень!**
-  Эта стратегия вас недоступна: **перейдите на следующий уровень!**
-  Эта стратегия вас недоступна: **перейдите на следующий уровень!**
-  Эта стратегия вас недоступна: **перейдите на следующий уровень!**
-  Эта стратегия вас недоступна: **перейдите на следующий уровень!**
-  Эта стратегия вас недоступна: **перейдите на следующий уровень!**
-  Эта стратегия вас недоступна: **перейдите на следующий уровень!**
-  Эта стратегия вас недоступна: **перейдите на следующий уровень!**

-  Эта стратегия вас недоступна: **перейдите на следующий уровень!**