Министерство науки и высшего образования Российской Федерации

федеральное государственное автономное образовательное учреждение высшего образования

«Национальный исследовательский университет ИТМО»

**Лабораторная работа №1**

По дисциплине «Экономика программной инженерии»

Вариант: https://www.twitch.tv/

Группа: P3412

Выполнили: Балин А. А., Кобелев Р. П.

Проверил: преподаватель Гаврилов А. В.

# Оглавление

[Оглавление 2](#_Toc209704364)

[Задание 4](#_Toc209704365)

[Выполнение 5](#_Toc209704366)

[Функциональные требования 5](#_Toc209704367)

[Наивный метод 7](#_Toc209704368)

[Метод PERT 10](#_Toc209704369)

[Метод функциональных точек и COCOMO II 12](#_Toc209704370)

[Границы допущения 12](#_Toc209704371)

[ILF/EIF 12](#_Toc209704372)

[Транзакции (EI/EO/EQ) 13](#_Toc209704373)

[Подсчет UFP 14](#_Toc209704374)

[Определение значения фактора выравнивания (VAF) 14](#_Toc209704375)

[Расчет количества выровненных функциональных точек (AFP) 15](#_Toc209704376)

[Размер продукта (SIZE) 15](#_Toc209704377)

[Факторы масштаба - SF 16](#_Toc209704378)

[Множители трудоемкости - EM 16](#_Toc209704379)

[Оценка трудоемкости 16](#_Toc209704380)

[UCP 18](#_Toc209704381)

[Акторы и UAW 18](#_Toc209704382)

[Use Cases 18](#_Toc209704383)

[Unadjusted Use Case Points (UUCP) 19](#_Toc209704384)

[Technical Complexity Factor (TCF) 19](#_Toc209704385)

[Environment Complexity Factor (ECP) 20](#_Toc209704386)

[Productivity Factor (PF) 21](#_Toc209704387)

[UCP 21](#_Toc209704388)

[Заключение 22](#_Toc209704389)

[Список литературы 23](#_Toc209704390)

# Задание

Для выданного веб-проекта:

1. Сформировать набор функциональных требований для разработки проекта.
2. Оценить трудоемкость разработки проекта наивным методом.
3. Оценить трудоемкость разработки проекта методом PERT (Project Evaluation and Review Technique). Нарисовать сетевую диаграмму взаимосвязи работ и методом критического пути рассчитать минимальную продолжительность разработки. Предложить оптимальное количество разработчиков и оценить срок выполнения проекта.
4. Оценить размер проекта методом функциональных точек, затем, исходя из предположения, что собранной статистики по завершенным проектам нет, рассчитать трудоемкость методом COCOMO II (Обновленная таблица количества строк на точку для разных языков программирования)
5. Оценить размер проекта методом оценки вариантов использования (Use Case Points). Для расчета фактора продуктивности PF использовать любой свой завершенный проект с известными временными трудозатратами, оценив его размер методом UCP.
6. Сравнить полученные результаты и сделать выводы.

# Выполнение

## Функциональные требования

1. Система должна давать возможность авторизованным пользователям проводить трансляции.
   1. Система должна давать всем пользователям искать трансляции по её описанию или названию.
   2. Система должна давать всем пользователям возможность заходить на любые трансляции и смотреть их.
   3. Система должна обеспечивать возможность авторизованным пользователям запускать свои трансляции.
   4. Система должна иметь возможность сохранения проведённой трансляции в профиле пользователя.
2. Система должна обеспечивать возможность подписки одних пользователей на других.
   1. Система должна обеспечивать отображение количества подписчиков у конкретного пользователя.
   2. Система должна обеспечивать возможность отписки от пользователя, если подписка уже существует.
3. Система должна обеспечивать возможности онлайн-чата во время проведения трансляций.
   1. Система должна обеспечивать возможность писать в чат.
   2. Система должна обеспечивать возможность читать сообщения других пользователей в чате, включая эмодзи, гиперссылки.
4. Система должна обеспечивать возможность настройки трансляции для пользователей.
   1. Система должна обеспечивать возможность настройки трансляции перед её запуском.
      1. Система должна обеспечивать возможность настройки битрейта, разрешения, названия и описания трансляции.
   2. Система должна обеспечивать возможность настройки трансляции для зрителя.
      1. Система должна обеспечивать возможность настройки скорости трансляции, её разрешения.
      2. Система должна обеспечивать возможность перемотки трансляции.
5. Система должна обеспечивать возможность регистрации и авторизации.
   1. Система должна обеспечивать возможность авторизации через логин и пароль Twitch.
   2. Система должна обеспечивать возможность сменить пароль.
6. Система должна обеспечивать возможность настройки профиля пользователя.
   1. Система должна обеспечивать возможность смены имени пользователя.
   2. Система должна обеспечивать возможность смены описания профиля.
   3. Система должна обеспечивать возможность смены фотографии профиля.
7. Система должна обеспечивать отображение группировок по разным фильтрам.
   1. Система должна обеспечивать группировки по играм.
   2. Система должна обеспечивать группировки по жанрам трансляций или действиям, которые происходят на трансляции.
8. Система должна обеспечивать возможность создавать клипы.
   1. Система должна обеспечивать возможность создавать клипы, указывая начало и конец клипа из определённой трансляции.
   2. Система должна обеспечивать возможность делиться созданными клипами с другими пользователями.

# Наивный метод

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| # | Название | Описание | Optimist. | Pess. | Optimal |
| 1 | Проведение трансляций | | 1300 | 2600 | 1258 |
| 1.1 | Поиск трансляции | Система должна давать всем пользователям искать трансляции по её описанию или названию. | 20 | 80 | 48 |
| 1.2 | Просмотр трансляции | Система должна давать всем пользователям возможность заходить на любые трансляции и смотреть их. | 400 | 1000 | 600 |
| 1.3 | Запуск трансляции | Система должна обеспечивать возможность авторизованным пользователям запускать свои трансляции. | 320 | 800 | 130 |
| 1.4 | Сохранение трансляции | Система должна иметь возможность сохранения проведённой трансляции в профиле пользователя. | 560 | 720 | 480 |
| 2 | Подписки | | 60 | 112 | 84 |
| 2.1 | Отображение количества подписчиков | Система должна обеспечивать отображение количества подписчиков у конкретного пользователя. | 40 | 80 | 60 |
| 2.2 | Отписка | Система должна обеспечивать возможность отписки от пользователя, если подписка уже существует. | 20 | 32 | 24 |
| 3 | Онлайн-чат | | 35 | 60 | 44 |
| 3.1 | Писать в чат | Система должна обеспечивать возможность писать в чат. | 20 | 30 | 24 |
| 3.2 | Отображение сообщений других пользователей в чате | Система должна обеспечивать возможность читать сообщения других пользователей в чате, включая эмодзи, гиперссылки. | 15 | 30 | 20 |
| 4 | Настройка трансляции | | 520 | 1000 | 754 |
| 4.1.1 | Настройка перед запуском трансляции | Система должна обеспечивать возможность настройки битрейта, разрешения, названия и описания трансляции. | 240 | 400 | 320 |
| 4.2.1 | Настройка параметров трансляции для зрителя | Система должна обеспечивать возможность настройки скорости трансляции, её разрешения. | 200 | 400 | 280 |
| 4.2.2 | Перемотка трансляции | Система должна обеспечивать возможность перемотки трансляции. | 80 | 200 | 154 |
| 5 | Регистрация и авторизация | | 120 | 260 | 144 |
| 5.1 | Авторизация по логину и паролю Twitch | Система должна обеспечивать возможность авторизации и регистрации через логин и пароль Twitch. | 80 | 200 | 96 |
| 5.2 | Смена пароля | Система должна обеспечивать возможность сменить пароль. | 40 | 60 | 48 |
| 6 | Найстройка профиля | | 72 | 144 | 88 |
| 6.1 | Смена имени пользователя | Система должна обеспечивать возможность смены имени пользователя. | 16 | 32 | 20 |
| 6.2 | Смена описания профиля | Система должна обеспечивать возможность смены описания профиля. | 16 | 32 | 20 |
| 6.3 | Смена фотографии пользователя | Система должна обеспечивать возможность смены фотографии профиля. | 40 | 80 | 48 |
| 7 | Группировка трансляций по категориям | | 160 | 240 | 192 |
| 7.1 | Группировка по играм | Система должна обеспечивать группировки по играм. | 80 | 120 | 96 |
| 7.2 | Группировка по жанрам и действиям | Система должна обеспечивать группировки по жанрам трансляций или действиям, которые происходят на трансляции. | 80 | 120 | 96 |
| 8 | Создание клипов | | 280 | 400 | 344 |
| 8.1 | Создание клипа | Система должна обеспечивать возможность создавать клипы, указывая начало и конец клипа из определённой трансляции. | 160 | 240 | 200 |
| 8.2 | Возможность поделиться клипом | Система должна обеспечивать возможность делиться созданными клипами с другими пользователями. | 120 | 160 | 144 |
| Дополнительная разработка красивого UI для всех функций | | | 600 | 1000 | 700 |
| Тестирование | | | 400 | 800 | 500 |
| ИТОГО (человек/час) | | | 3547 | 6616 | 4108 |
| ИТОГО (человек/рабочий день) | | | 443.375 | 827 | 513.5 |
| ИТОГО (человек/месяц) | | | 21.6280488 | 40.34146 | 25.04878 |
| ИТОГО (человек/месяц) \* (π+e) | | | 126.558531 | 236.0618 | 146.5753 |

# Метод PERT

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| # | Название | Optimist. | Pess. | Optimal | E\_i | (СКО\_i)^2 |
| 1 | Проведение трансляций | 1300 | 2600 | 1258 |  |  |
| 1.1 | Поиск трансляции | 20 | 80 | 48 | 48.6666667 | 100 |
| 1.2 | Просмотр трансляции | 400 | 1000 | 600 | 633.333333 | 10000 |
| 1.3 | Запуск трансляции | 320 | 800 | 130 | 273.333333 | 6400 |
| 1.4 | Сохранение трансляции | 560 | 720 | 480 | 533.333333 | 711.1111111 |
| 2 | Подписки | 60 | 112 | 84 |  |  |
| 2.1 | Отображение количества подписчиков | 40 | 80 | 60 | 60 | 44.44444444 |
| 2.2 | Отписка | 20 | 32 | 24 | 24.6666667 | 4 |
| 3 | Онлайн-чат | 35 | 60 | 44 |  |  |
| 3.1 | Писать в чат | 20 | 30 | 24 | 24.3333333 | 2.777777778 |
| 3.2 | Отображение сообщений других пользователей в чате | 15 | 30 | 20 | 20.8333333 | 6.25 |
| 4 | Настройка трансляции | 520 | 1000 | 754 |  |  |
| 4.1.1 | Настройка перед запуском трансляции | 240 | 400 | 320 | 320 | 711.1111111 |
| 4.2.1 | Настройка параметров трансляции для зрителя | 200 | 400 | 280 | 286.666667 | 1111.111111 |
| 4.2.2 | Перемотка трансляции | 80 | 200 | 154 | 149.333333 | 400 |
| 5 | Регистрация и авторизация | 120 | 260 | 144 |  |  |
| 5.1 | Авторизация по логину и паролю Twitch | 80 | 200 | 96 | 110.666667 | 400 |
| 5.2 | Смена пароля | 40 | 60 | 48 | 48.6666667 | 11.11111111 |
| 6 | Найстройка профиля | 72 | 144 | 88 |  |  |
| 6.1 | Смена имени пользователя | 16 | 32 | 20 | 21.3333333 | 7.111111111 |
| 6.2 | Смена описания профиля | 16 | 32 | 20 | 21.3333333 | 7.111111111 |
| 6.3 | Смена фотографии пользователя | 40 | 80 | 48 | 52 | 44.44444444 |
| 7 | Группировка трансляций по категориям | 160 | 240 | 192 |  |  |
| 7.1 | Группировка по играм | 80 | 120 | 96 | 97.3333333 | 44.44444444 |
| 7.2 | Группировка по жанрам и действиям | 80 | 120 | 96 | 97.3333333 | 44.44444444 |
| 8 | Создание клипов | 280 | 400 | 344 |  |  |
| 8.1 | Создание клипа | 160 | 240 | 200 | 200 | 177.7777778 |
| 8.2 | Возможность поделиться клипом | 120 | 160 | 144 | 142.666667 | 44.44444444 |
| Дополнительная разработка красивого UI для всех функций | | 600 | 1000 | 700 | 733.333333 | 4444.444444 |
| Тестирование | | 400 | 800 | 500 | 533.333333 | 4444.444444 |
|  | | E | СКО | E\_95% |  | |
| ИТОГО (человек/час) | | 4432.5 | 170.7647 | 4774.0294 |
| ИТОГО (человек/рабочий день) | |  | | 596.753675 |
| ИТОГО (человек/месяц) | | 29.1099354 |
| ИТОГО (человек/месяц) \* (π+e) | | 170.339484 |

# Сетевая диаграмма и метод критического пути

Изображение выглядит как текст, диаграмма, линия, План

Содержимое, созданное искусственным интеллектом, может быть неверным.

По сетевой диаграмме (PERT) критический путь: 4.1.1 Настройка перед запуском (320.00) → 1.3 Запуск трансляции (273.33) → 1.4 Сохранение трансляции (533.33) → 8.1 Создание клипа (200.00) → 8.2 Поделиться клипом (142.67) → Альфа‑тестирование (150.00) → Бета‑тестирование (250.00) → Релизные испытания и регресс (100.00).

Минимальная продолжительность разработки (сумма по критическому пути):

До тестов: 320.00 + 273.33 + 533.33 + 200.00 + 142.67 = 1469.33 ч

Тесты: 150.00 + 250.00 + 100.00 = 500.00 ч

Итого: 1969.33 ч ≈ 246 рабочих дней ≈ 49 недель ≈ 12.3 мес (при 160 ч/мес).

Ветки UI (733.33 ч) и вспомогательные подсистемы укладываются в этот срок и не удлиняют критическую цепочку.

Предложение по команде и сроку

Оптимально: 3 разработчика + QA на финальные испытания.

2 backend: один ведёт непрерывно критическую ветку (4.1.1 → 1.3 → 1.4 → 8.1 → 8.2), второй закрывает параллельные серверные задачи (просмотр, чат, подписки, профили, категории).

1 frontend: весь UI (733.33 ч), синхронизирован к началу тестов.

QA: подключается на этапах Альфа/Бета/Релиз, помогает удержать суммарно 500 ч тестовой фазы в срок.

Оценка календарного срока: нижняя граница — 1969.33 ч ≈ 12.3 мес; с управленческим резервом 10–15% на риски и стабилизацию — 13–14 мес.

Если нужен дополнительный запас по параллелизму/рискам — 4‑й разработчик (3 backend + 1 frontend) позволит разгрузить не критические ветки и снизить риск их перерастания в критические, но сам нижнюю границу 1969.33 ч не уменьшит (она определяется последовательностью работ на критическом пути).

# Метод функциональных точек и COCOMO II

## Границы допущения

Учитываем только пользовательски распознаваемые процессы и персистентные данные.

Настройки зрителя (скорость/качество проигрывателя) считаю неперсистентными — не учитываю как EI/ILF.

Сохранение трансляции происходит по действию «Stop/End Stream» (учтено как EI).

## ILF/EIF

* Пользователи/Профиль — ILF, RET≈2, DET≈25 -> Avg. = 10
* Трансляции + настройки трансляции — ILF, RET≈2, DET≈25 -> Avg. = 10
* Видео — ILF, RET=1, DET≈15 -> Low = 7
* Клип — ILF, RET=1, DET≈10 -> Low = 7
* Подписки (follow) — ILF, RET=1, DET≈4 -> Low = 7
* Сообщения чата — ILF, RET=1, DET≈6 -> Low = 7
* Категории/Игры — ILF, RET=1, DET≈5 -> Low = 7
* Жанры — ILF, RET=1, DET≈3 -> Low = 7
* Аккаунты внешней авторизации — ILF, RET=1, DET≈7 -> Low = 7

Итого ILF = 69 UFP

EIF нешний профиль IdP — 1×Low = 5 UFP

## Транзакции (EI/EO/EQ)

* Регистрация (EI, FTR=1, DET≈5) -> Low = 3
* Логин (EI, FTR=1, DET≈3) -> Low = 3
* OAuth логин/регистрация (EI, FTR=2, DET≈5) -> Avg. = 4
* Обновить/задать настройки трансляции (EI, FTR=3, DET≈8) -> Высок. = 6
* Войти на страницу трансляции/получить метаданные (EQ, FTR=3, DET≈10) -> Avg. = 4
* Читать чат (EQ, FTR=2, DET≈6) -> Avg. = 4
* Отправить сообщение в чат (EI, FTR=3, DET≈2) -> Avg. = 4
* Создать клип (EI, FTR=3, DET≈4) -> Avg. = 4
* Лента рекомендаций/живых каналов (EQ, FTR=2, DET≈8) -> Avg. = 4
* Список «Подписки/Following» (EQ, FTR=3, DET≈6) -> Avg. = 4
* Подписаться (EI, FTR=2, DET≈2) -> Low = 3
* Отписаться (EI, FTR=2, DET≈2) -> Low = 3
* Категории (EQ, FTR=1, DET≈6) -> Low = 3
* Просмотр по жанру (EQ, FTR=2, DET≈6) -> Avg. = 4
* Список клипов пользователя (EQ, FTR=2, DET≈6) -> Avg. = 4
* Поделиться клипом (EO, FTR=2, DET≈3) -> Low = 4
* Старт трансляции (EI, FTR=3, DET≈6–10) -> Высок. = 6
* Завершить и сохранить трансляцию (EI, FTR=2, DET≈5) -> Avg. = 4
* Показ количества подписчиков (EO, FTR=2, DET≈2) -> Low = 4

Итого:

EI = 43 UFP

EO = 8 UFP

EQ = 31 UFP

## Подсчет UFP

ILF: 69

EIF: 5

EI: 43

EO: 8

EQ: 31

Итого UFP = 69 + 43 + 8 + 31 + 5 = 156

## Определение значения фактора выравнивания (VAF)

* Обмен данными = 4 - Продукт обменивается данными по нескольким протоколам (WebSocket, HTTP)
* Распределенная обработка данных = 3 - Обработка данных распределена между клиентом и сервером
* Производительность = 5 - Критичны задержки стриминга и чата, требуются специальные решения
* Ограничения по аппаратным ресурсам = 4 - Есть требования к процессору для стриминга
* Транзакционная нагрузка = 4 - Высокая нагрузка при большом количестве зрителей
* Интенсивность взаимодействия с пользователем = 4 - Большинство операций интерактивные
* Эргономика = 4 - Важна эффективность работы стримера и зрителей
* Интенсивность изменения данных = 3 - Умеренная интенсивность обновления контента
* Сложность обработки = 2 - Средняя сложность бизнес-логики
* Повторное использование = 1 - Минимальное повторное использование компонентов
* Удобство инсталляции = 3 - Автоматическое обновление клиентской части
* Удобство администрирования = 3 - Базовый мониторинг и восстановление
* Портируемость = 3 - Веб-приложение с поддержкой основных браузеров
* Гибкость = 4 - Настраиваемые параметры трансляции и профиля

DI = 47

VAF = 0.65 + 0.01×DI = 0.65 + 0.47 = 1.12

## Расчет количества выровненных функциональных точек (AFP)

AFP = 156 × 1.12 ≈ 175

## Размер продукта (SIZE)

Так как в прошлом методе мы посчитали количество функциональных точек (175), распределим их между компонентами системы:

* Бэкенд (C++) - 105 FP (60%) - основная бизнес-логика, обработка стримов
* База данных (SQL) - 35 FP (20%) - хранение данных пользователей, стримов, подписок
* Фронтенд (JavaScript) - 35 FP (20%) - клиентский интерфейс

По [таблице](https://www.qsm.com/resources/function-point-languages-table) определим количество строк кода:

* C++: 105 FP \* 50 = 5250 LOC
* SQL: 35 FP \* 21 = 735 LOC
* JavaScript: 35 FP \* 47 = 1645 LOC

Итого: 7630 SLOC (7.63 KSLOC)

## Факторы масштаба - SF

* PREC (наличие опыт аналогичных разработок) = 3.72 (Nominal)
* FLEX (гибкость процесса разработки) = 2.03 (High)
* RESL (архитектура и разрешение рисков) = 7.07 (Very Low)
* TEAM (сработанность команды) = 1.1 (Very High)
* PMAT (зрелость процессов) = 6.24 (Low)

## Множители трудоемкости - EM

* PERS (квалификация персонала) = 1.62 (Low)
* RCPX (сложность и надежность продукта) = 1.33 (High)
* RUSE (разработка для повторного использования) = 0.95 (Low)
* PDIF (сложность платформы разработки) = 1.29 (High)
* PREX (опыт персонала) = 1.33 (Low)
* FCIL (оборудование) = 1.00 (Nominal)
* SCED (сжатие расписания) = 1.00 (Nominal)

## Оценка трудоемкости

Имеем формулу:

Где:

Подставляем и получаем результат:

Для двух разработчиков получаем 120 месяцев на разработчика

# UCP

## Акторы и UAW

* Анонимный посетитель (GUI) — сложный -> 3
* Авторизованный зритель (GUI) — сложный -> 3
* Стример (GUI) — сложный -> 3
* Провайдер внешней аутентификации (OAuth 2.0/OIDC, HTTP redirect) — средний -> 2

Итого: UAW = 3 + 3 + 3 + 2 = 11

## Use Cases

Ниже указано приблизительное число транзакций для классификации.

Простые (5 баллов):

* Регистрация по форме (открыть форму; ввести/валидировать; создать учётную запись)
* Вход по паролю (форма; аутентификация; редирект)
* Редактирование профиля (имя/описание/фото) (ввод; загрузка/обработка; сохранение)

Средние (10 баллов):

* Вход/регистрация через OAuth (редирект к IdP; авторизация; callback; обмен код/токен/линковка)
* Обнаружение контента (ввод запроса; выдача; выбор категории; фильтры; просмотр списка каналов)
* Просмотр трансляции (загрузка метаданных; запуск плеера; подключение к чату; получение сообщений; показ сайд‑списков)
* Управление клипами (отметить начало/конец; сохранить; получить ссылку/поделиться; просмотр списка)
* Отправка сообщения в чат (ввод; отправка/подтверждение)
* Управление подпиской (действие; подтверждение/обновление состояния)

Сложные (15 баллов):

* Управление сессией трансляции (задать параметры; валидация; привязка категории/тегов; старт; мониторинг; стоп; сохранение VOD; публикация)

Итого UUCW: простые 3×5=15; средние 6×10=60; сложные 1×15=15 -> UUCW = 90

## Unadjusted Use Case Points (UUCP)

UUCP = UAW + UUCW = 11 + 90 = 101

## Technical Complexity Factor (TCF)

Оценки факторов и веса Wi:

* Распределённость системы (Wi=2): F=4 -> 8 — сервис распределён: CDN/HLS, WebSocket‑чат, API.
* Производительность (Wi=1): F=5 -> 5 — критичны задержки стрима и чата.
* Эффективность для пользователя (Wi=1): F=4 -> 4 — UX важен (стример/зритель).
* Сложная внутренняя обработка (Wi=1): F=3 -> 3 — медиапайплайн, модерация, рекомендации.
* Повторное использование кода (Wi=1): F=2 -> 2 — умеренное переиспользование модулей.
* Простота установки (Wi=0.5): F=2 -> 1 — веб‑сервис, требований немного.
* Простота использования (Wi=0.5): F=4 -> 2 — высокие требования к простоте UI.
* Переносимость (Wi=2): F=3 -> 6 — кросс‑браузерность, разные устройства.
* Простота изменений (Wi=1): F=3 -> 3 — частые изменения контента/настроек.
* Многопоточность (Wi=1): F=4 -> 4 — конкурентные стримы/чат/фоновая обработка.
* Доп. возможности безопасности (Wi=1): F=4 -> 4 — OAuth, защита от спама.
* Доступ к другим системам (Wi=1): F=3 -> 3 — интеграции с IdP.
* Необходимы тренажёры для пользователей (Wi=1): F=0 -> 0 — обучение не требуется.

## Environment Complexity Factor (ECP)

* Уверенное использоввание UML/RUP (Wi=1.5): F=3 -> 4.5 — средний навык
* Кол-во работников на неполный рабочий день (Wi=-1): F=2 -> -2 — все разработчики на неполный рабочий день
* Опытность аналитика (Wi=0.5): F=2 -> 1 — аналитик не очень опытный
* Опыт работы с приложениями (Wi=0.5): F=5 -> 4.5 — у каждого разработчика имеется опыт с приложениями
* Опыт ОО разработки (Wi=0.5): F=5 -> 4.5 — у каждого разработчика имеется опыт с ОО разработкой
* Мотивация (Wi=1): F=3 -> 3 — средняя мотивация
* Сложный язык разработки (Wi=-1): F=2 -> -2 — у каждого разработчика небольшой опыт с C++
* Неизменность требований (Wi=2): F=5 -> 10 — требования неизменяемы, разработчикам удобнее строить планы и разрабатывать

## Productivity Factor (PF)

По прошлым разработкам возьмем 8 часов/UC

## UCP

При **160** рабочих часах в месяц имеем **1.84** месяцев работы для каждого разработчика

# Заключение

Как мы видим, наивный метод, метод PERT и метод функциональных точек + COCOMO II получились примерно похожие значения, в отличие от метода Use-Case. Наверное, дело в том, что первые три метода основаны на среднестатистическом опыте разработчика, а последний метод сильно зависит от выбранного проекта для сравнения. Лабораторная работа не может нормально сравниваться с лучшим сервисом для стриминга.