

# ЛЕКЦИЯ 02

## ПЕРВАЯ ПРОГРАММА

ОСНОВЫ АЛГОРИТМИЗАЦИИ И  
ПРОГРАММИРОВАНИЯ



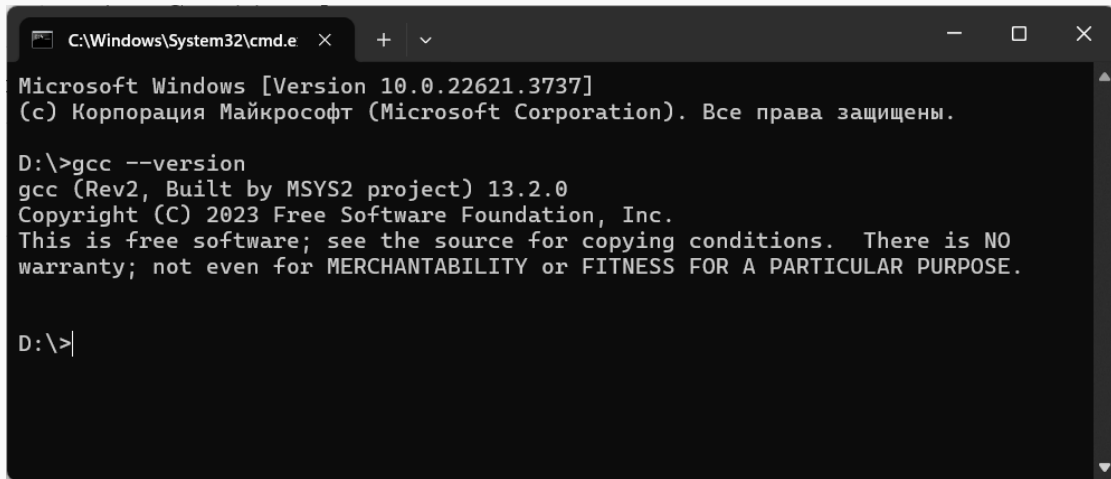
# ОСНОВНЫЕ ПОНЯТИЯ

**Программный продукт**— комплекс взаимосвязанных программ для решения определенной проблемы (задачи) массового спроса, подготовленный к реализации как любой вид промышленной продукции.

**Сопровождение программного продукта** — поддержка работоспособности программного продукта, переход на его новые версии, внесение изменений, исправление обнаруженных ошибок.

# УСТАНОВКА КОМПИЛЯТОРА

- Скачать компилятор gcc (<https://www.mingw-w64.org/downloads/>) или clang (<https://releases.llvm.org/download.html>)
- Установка, распаковка компилятора
- Добавление переменных сред для корректного запуска компилятора из любого доступного каталога
- Проверка корректности установки. Для gcc это проверяется командой в терминале `gcc --version`



```
C:\Windows\System32\cmd.e  X  +  v  -  □  X
Microsoft Windows [Version 10.0.22621.3737]
(c) Корпорация Майкрософт (Microsoft Corporation). Все права защищены.

D:\>gcc --version
gcc (Rev2, Built by MSYS2 project) 13.2.0
Copyright (C) 2023 Free Software Foundation, Inc.
This is free software; see the source for copying conditions. There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

D:\>|
```

# HELLO, WORLD

- `#include <iostream>`
- `int main() {`
- `std::cout << "Hello World!" << std::endl;`
- `return 0;`
- `}`

# HELLO, WORLD

- `#include <iostream> //Подключение библиотеки ввода-вывода`
- `int main() { //точка входа в программу`
- `std::cout << "Hello World!" << std::endl;`
- `return 0; //успешное завершение работы функции`
- `}`

# КОМПИЛЯЦИЯ ПРОГРАММЫ

g++ <имя файла>

Программа скомпилируется в файл a.exe (a.out) в том же каталоге, в котором файл с программным кодом находится

# ОСНОВНЫЕ ФЛАГИ КОМПИЛЯТОРА g++

- c Компилировать или ассемблировать исходные файлы, но не линковать.
- S Остановиться после собственно компиляции; не ассемблировать.
- E Остановиться после стадии препроцессирования.
- o <имя\_файла> Поместить вывод в файл 'файл'.
- pedantic Выдаются все предупреждения, требуемые строгим ANSI стандартом C, отбрасываются все программы, которые используют запрещенные расширения.
- Wall Все виды предупреждений отображаются.

# ОСНОВНЫЕ ХАРАКТЕРИСТИКИ ПРОГРАММ

- алгоритмическая сложность;
- состав и глубина проработки реализованных функций обработки;
- полнота и системность функций обработки;
- объем файлов программ;
- требования к операционной системе и техническим средствам обработки со стороны программного средства;
- объем дисковой памяти;
- размер оперативной памяти для запуска программ;
- разрядность;
- версия операционной системы;
- наличие вычислительной сети



# ПОКАЗАТЕЛИ КАЧЕСТВА ПРОГРАММ

**Мобильность** программных продуктов означает их независимость от технического комплекса системы обработки данных, операционной среды, сетевой технологии обработки данных, специфики предметной области и т.п. Мобильный (многоплатформный) программный продукт может быть установлен на различных моделях компьютеров и операционных систем, без ограничений на его эксплуатацию в условиях вычислительной сети. Функции обработки такого программного продукта пригодны для массового использования без каких-либо изменений.

# ПОКАЗАТЕЛИ КАЧЕСТВА ПРОГРАММ

**Надежность** работы программного продукта определяется бесперебойностью и устойчивостью в работе программ, точностью выполнения предписанных функций обработки, возможностью диагностики возникающих в процессе работы программ ошибок.

# ПОКАЗАТЕЛИ КАЧЕСТВА ПРОГРАММ

**Эффективность** программного продукта оценивается как с позиций прямого его назначения — требований пользователя, так и с точки зрения расхода вычислительных ресурсов, необходимых для его эксплуатации.

Расход вычислительных ресурсов оценивается через объем внешней памяти для размещения программ и объем оперативной памяти для запуска программ.

# ПОКАЗАТЕЛИ КАЧЕСТВА ПРОГРАММ

**Учет человеческого фактора** означает обеспечение дружественного интерфейса для работы конечного пользователя, наличие контекстно-зависимой подсказки или обучающей системы в составе программного средства, хорошей документации для освоения и использования заложенных в программном средстве функциональных возможностей, анализ и диагностику возникших ошибок и др.

# ПОКАЗАТЕЛИ КАЧЕСТВА ПРОГРАММ

**Модифицируемость** программных продуктов означает способность к внесению изменений, например, расширение функций обработки, переход на другую техническую базу обработки и т.п.

# ПОКАЗАТЕЛИ КАЧЕСТВА ПРОГРАММ

**Коммуникативность** программных продуктов основана на максимально возможной их интеграции с другими программами, обеспечении обмена данными в общих форматах представления (экспорт/импорт баз данных, внедрение или связывание объектов обработки и др.).

# ЖИЗНЕННЫЙ ЦИКЛ ПРОГРАММНОГО ПРОДУКТА

**Жизненным циклом ПП** называют период от момента появления идей создания некоторого ПО до момента завершения его поддержки фирмой разработчиков или фирмой выполнявших сопровождение.

# ЖИЗНЕННЫЙ ЦИКЛ ПРОГРАММНОГО ПРОДУКТА

**Модель жизненного цикла ПО** — структура, определяющая последовательность выполнения и взаимосвязи процессов, действий и задач на протяжении жизненного цикла. Модель жизненного цикла зависит от специфики, масштаба и сложности проекта и специфики условий, в которых система создается и функционирует



# ЖИЗНЕННЫЙ ЦИКЛ ПРОГРАММНОГО ПРОДУКТА

Модель жизненного цикла ПО включает в себя:

- Стадии;
- Результаты выполнения работ на каждой стадии;
- Ключевые события — точки завершения работ и принятия решений.

# ЖИЗНЕННЫЙ ЦИКЛ ПРОГРАММНОГО ПРОДУКТА

**Стадия** — часть процесса создания ПО, ограниченная определенными временными рамками и заканчивающаяся выпуском конкретного продукта (моделей, программных компонентов, документации), определяемого заданными для данной стадии требованиями.

На каждой стадии могут выполняться несколько процессов, определенных в стандарте ГОСТ Р ИСО/МЭК 12207-99, и наоборот, один и тот же процесс может выполняться на различных стадиях.

# ВОДОПАДНАЯ МОДЕЛЬ

Предусматривает последовательное выполнение всех этапов проекта в строго фиксированном порядке. Переход на следующий этап означает полное завершение работ на предыдущем этапе. Требования, определенные на стадии формирования требований, строго документируются в виде технического задания и фиксируются на все время разработки проекта. Каждая стадия завершается выпуском полного комплекта документации, достаточной для того, чтобы разработка могла быть продолжена другой командой разработчиков.

# ВОДОПАДНАЯ МОДЕЛЬ

Этапы проекта в соответствии с каскадной моделью:

- Формирование требований;
- Проектирование;
- Реализация;
- Тестирование;
- Внедрение;
- Эксплуатация и сопровождение.



# ВОДОПАДНАЯ МОДЕЛЬ

## Преимущества:

- Полная и согласованная документация на каждом этапе;
- Легко определить сроки и затраты на проект.

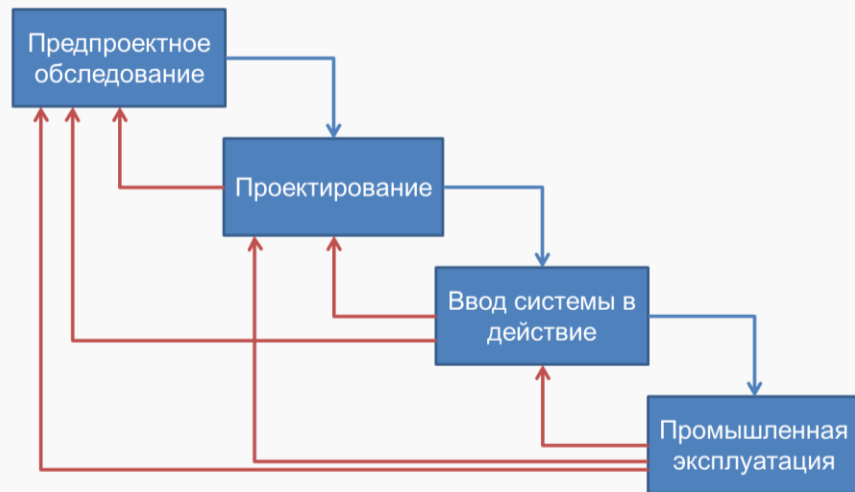
## Недостатки:

- Переход от одной фазы проекта к другой предполагает полную корректность результата (выхода) предыдущей фазы.



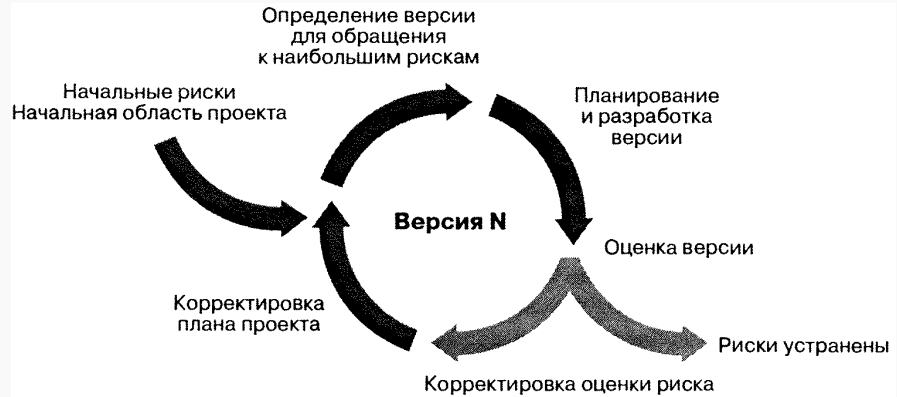
# ПОЭТАПНАЯ МОДЕЛЬ С ПРОМЕЖУТОЧНЫМ КОНТРОЛЕМ

Разработка ИС ведется итерациями с циклами обратной связи между этапами. Межэтапные корректировки позволяют учитывать реально существующее взаимовлияние результатов разработки на различных этапах; время жизни каждого из этапов растягивается на весь период разработки.



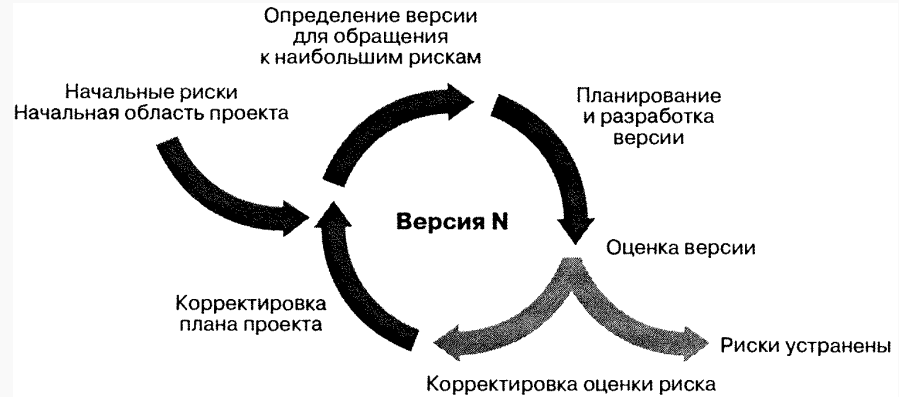
# ИТЕРАЦИОННАЯ МОДЕЛЬ

Альтернативой последовательной модели является так называемая модель итеративной и инкрементальной разработки (англ. iterative and incremental development, IID), получившей также от Т. Гилба в 70-е гг. название эволюционной модели. Также эту модель называют итеративной моделью и инкрементальной моделью.



# ИТЕРАЦИОННАЯ МОДЕЛЬ

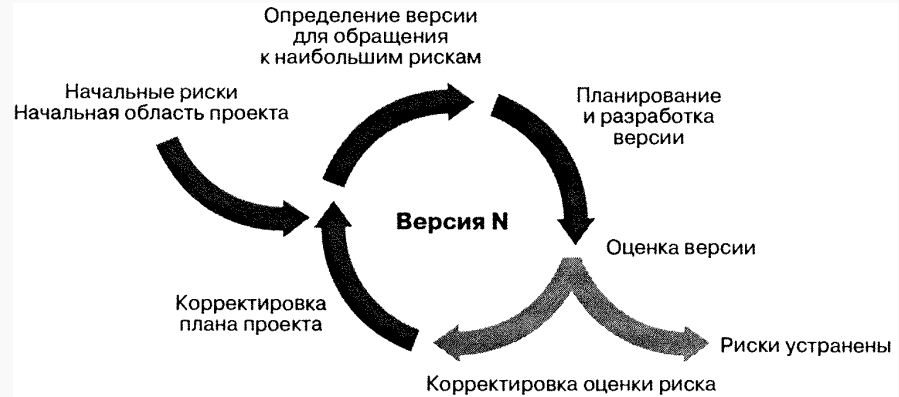
Модель IID предполагает разбиение жизненного цикла проекта на последовательность итераций, каждая из которых напоминает «мини-проект», включая все процессы разработки в применении к созданию меньших фрагментов функциональности, по сравнению с проектом в целом.





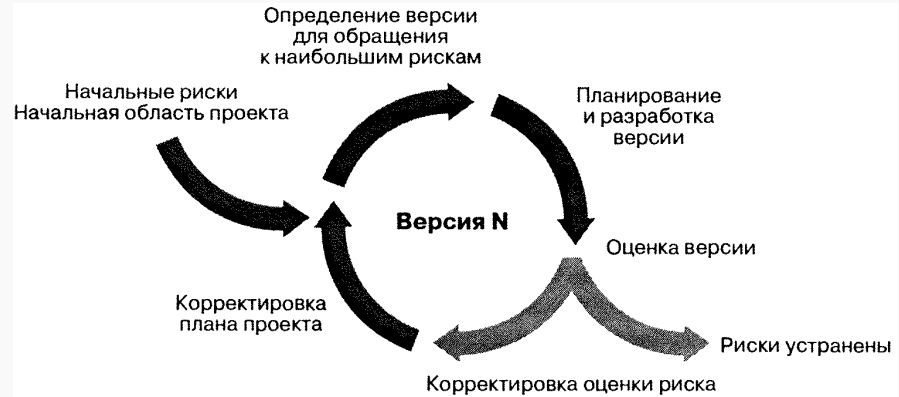
# ИТЕРАЦИОННАЯ МОДЕЛЬ

Цель каждой итерации — получение работающей версии программной системы, включающей функциональность, определённую интегрированным содержанием всех предыдущих и текущей итерации. Результат финальной итерации содержит всю требуемую функциональность продукта.



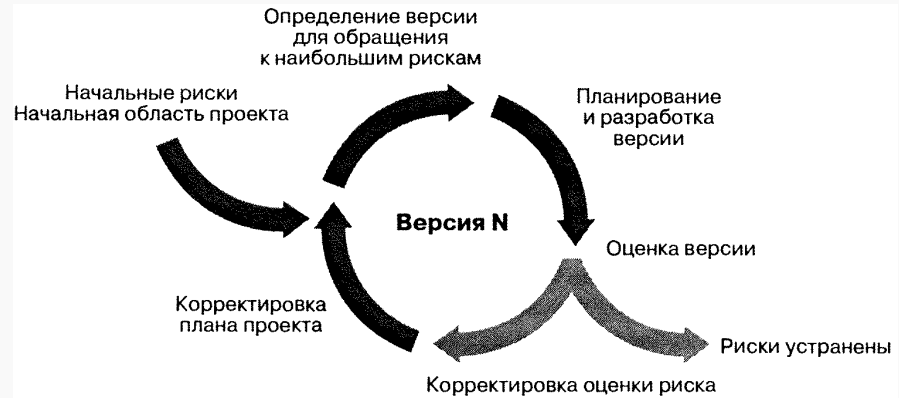
# ИТЕРАЦИОННАЯ МОДЕЛЬ

Таким образом, с завершением каждой итерации продукт получает приращение — инкремент — к его возможностям, которые, следовательно, развиваются эволюционно. Итеративность, инкрементальность и эволюционность в данном случае есть выражение одного и то же смысла разными словами со слегка разных точек зрения.



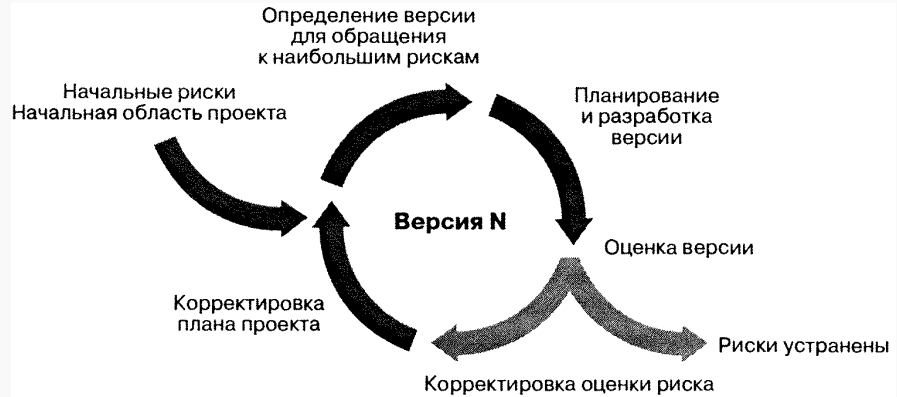
# ИТЕРАЦИОННАЯ МОДЕЛЬ

Шансы успешного создания сложной системы будут максимальными, если она реализуется в серии небольших шагов и, если каждый шаг включает в себе четко определённый успех, а также возможность «отката» к предыдущему успешному этапу в случае неудачи. Перед тем, как пустить в дело все ресурсы, предназначенные для создания системы, разработчик имеет возможность получать из реального мира сигналы обратной связи и исправлять возможные ошибки в проекте».



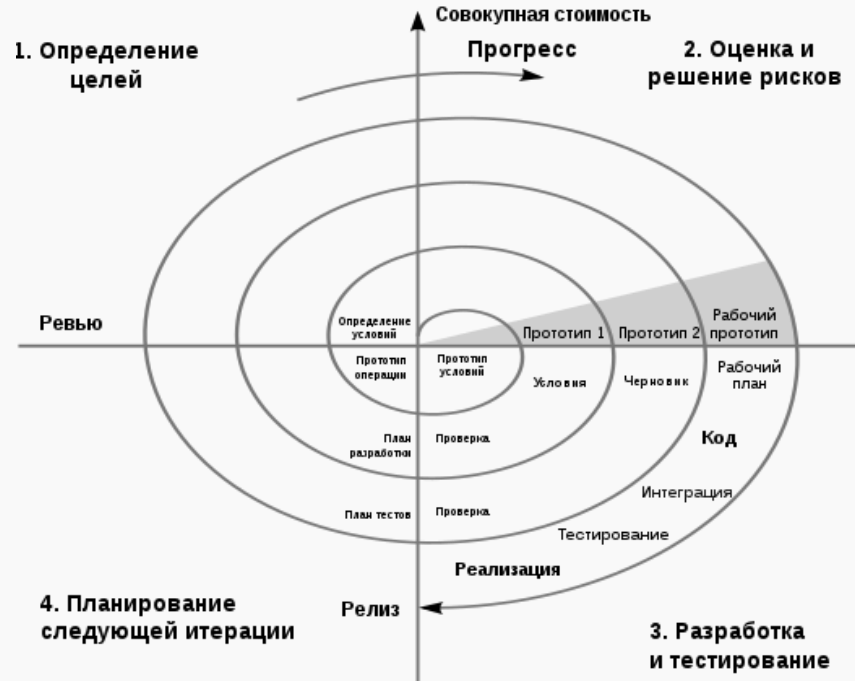
# ИТЕРАЦИОННАЯ МОДЕЛЬ

Подход IID имеет и свои отрицательные стороны, которые, по сути, — обратная сторона достоинств. Во-первых, целостное понимание возможностей и ограничений проекта очень долгое время отсутствует. Во-вторых, при итерациях приходится отбрасывать часть сделанной ранее работы. В-третьих, добросовестность специалистов при выполнении работ всё же снижается, что психологически объяснимо, ведь над ними постоянно довлечет ощущение, что «всё равно всё можно будет переделать и улучшить позже».



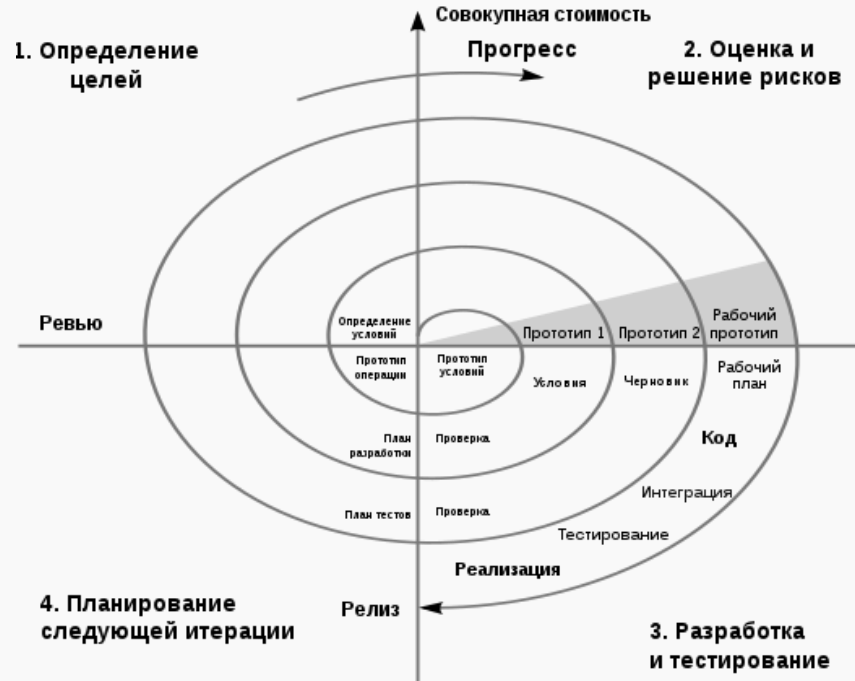
# СПИРАЛЬНАЯ МОДЕЛЬ

Спиральная модель (англ. spiral model) была разработана в середине 1980-х годов Барри Бозмом (Рисунок 3). Она основана на классическом цикле Деминга PDCA (plan-do-check-act). При использовании этой модели ПО создается в несколько итераций (витков спирали) методом прототипирования.



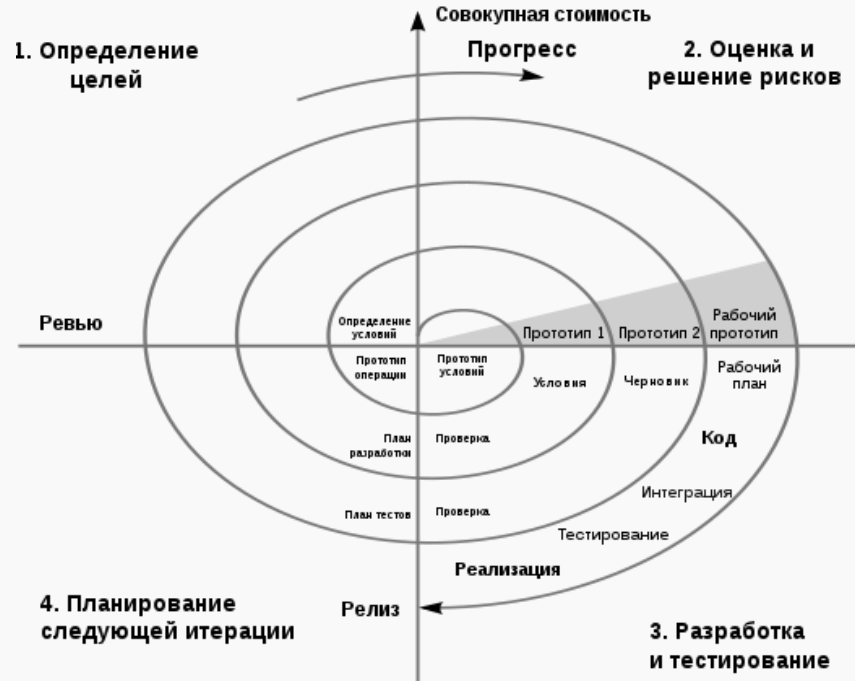
# СПИРАЛЬНАЯ МОДЕЛЬ

Спиральная модель является не альтернативой эволюционной модели (модели IID), а специально проработанным вариантом. К сожалению, нередко спиральную модель либо ошибочно используют как синоним эволюционной модели вообще, либо (не менее ошибочно) упоминают как совершенно самостоятельную модель наряду с IID.



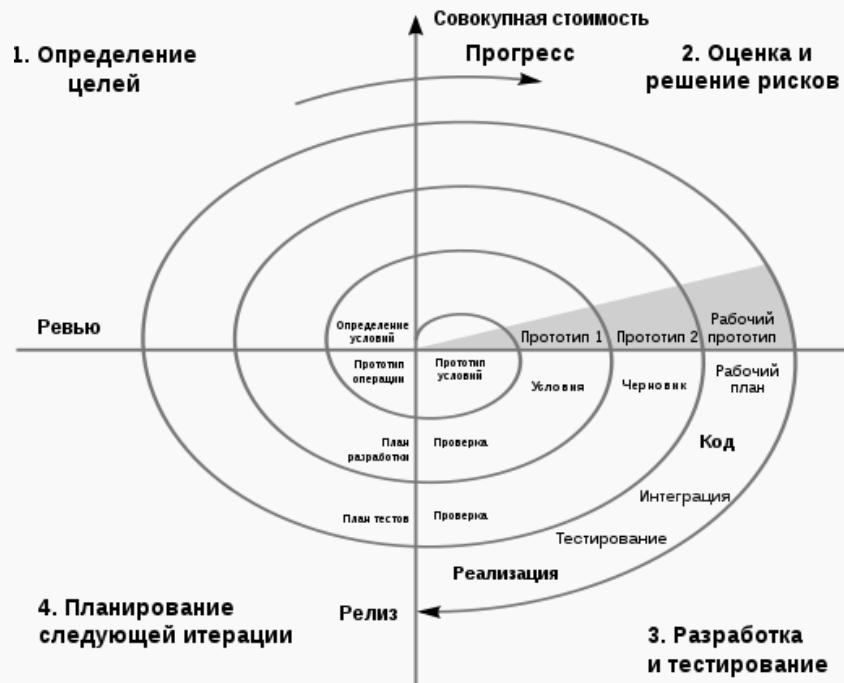
# СПИРАЛЬНАЯ МОДЕЛЬ

Отличительной особенностью спиральной модели является специальное внимание, уделяемое рискам, влияющим на организацию жизненного цикла, и контрольным точкам. Бозм формулирует 10 наиболее распространённых (по приоритетам) рисков



# СПИРАЛЬНАЯ МОДЕЛЬ. РИСКИ

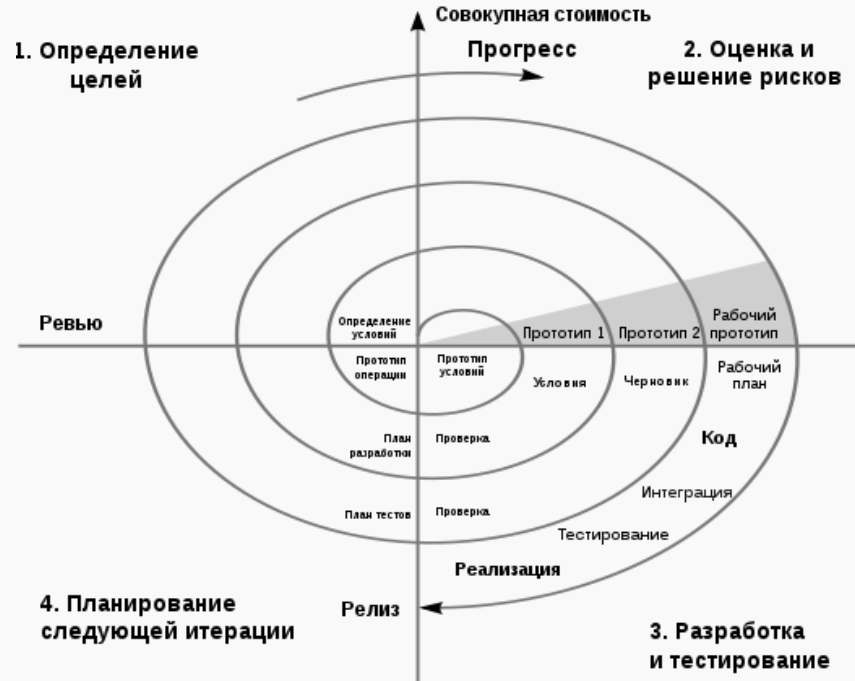
- Дефицит специалистов;
- Нереалистичные сроки и бюджет;
- Реализация несоответствующей функциональности;
- Разработка неправильного пользовательского интерфейса;
- Ненужная оптимизация и оттачивание деталей;
- Непрерывающийся поток изменений;





# СПИРАЛЬНАЯ МОДЕЛЬ. РИСКИ

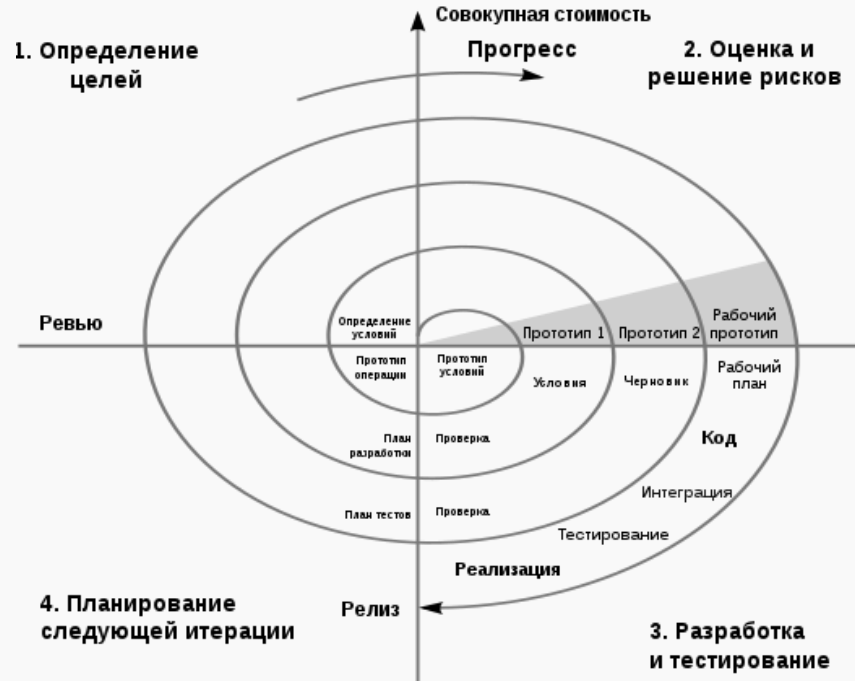
- Нехватка информации о внешних компонентах, определяющих окружение системы или вовлеченных в интеграцию;
- Недостатки в работах, выполняемых внешними (по отношению к проекту) ресурсами;
- Недостаточная производительность получаемой системы;
- Разрыв в квалификации специалистов разных областей.



# СПИРАЛЬНАЯ МОДЕЛЬ

На каждой итерации оцениваются:

- риск превышения сроков и стоимости проекта;
- необходимость выполнения ещё одной итерации;
- степень полноты и точности понимания требований к системе;
- целесообразность прекращения проекта.



# ДОМАШНЕЕ ЗАДАНИЕ

1. Установить и настроить компилятор gcc или clang
2. Написать, скомпилировать и запустить программу, которая будет выводить в консоль "Hello, World!"
3. Написать, скомпилировать и запустить программу, которая будет выводить в консоль номер вашей группы, ваши фамилию и инициалы.
4. Выучить историю языков программирования и виды жизненного цикла программ, а также соответствующие этим темам определения

# РЕКОМЕНДУЕМАЯ ЛИТЕРАТУРА

1. Дорохова Т.Ю., Основы алгоритмизации и программирования : учебное пособие для СПО / Т.Ю. Дорохова, И.Е. Ильина. – Саратов, Москва : Профобразование, Ай, Пи Ар Медиа, 2022. – 139 с.
2. Кудинов Ю.И., Основы алгоритмизации и программирования : учебное пособие для СПО / Ю.И. Кудинов, А.Ю. Келина. – 2-е изд. – Липецк, Саратов: Липецкий государственный технический университет, Профообразование, 2020. – 71 с.