Информатика

InformatiCS

Информатика

(Информация и авто**матика)**

- Теоретические основы информации и вычислений
  - Computer Science: алгоритмы и структуры данных, теория информации и кодирования и т.д.
- Практические методы для реализации и применения теоретических основ
  - IT и программная инженерия: базы данных, сети, параллельные вычисления, криптография и инфобез, компьютерная графика и визуализация, ИИ.

Информация

Информация — это не материя
и не энергия, информация — это
информация (Н. Винер)

- Сведения, независимо от формы их представления, воспринимаемые **человеком или специальными устройствами** как отражение фактов материального мира в **процессе коммуникации** (ГОСТ 7.0-99).

- Знания о предметах, фактах, идеях и т. д., которыми **могут обмениваться люди** в рамках конкретного контекста (ISO/IEC 10746-2:1996);

Информация

- Сведения, передаваемые **объектами** в процессе **коммуникации**
  - между людьми
  - в животном и растительном мире
  - между человеком и автоматом, автоматом и автоматом

Действия с информацией

- Поиск
- Сбор (считывание)
- Преобразование (кодирование)
- Передача
- Обработка
- Хранение
- Отображение (воспроизведение)

- **Информационный процесс** – набор действий с информацией

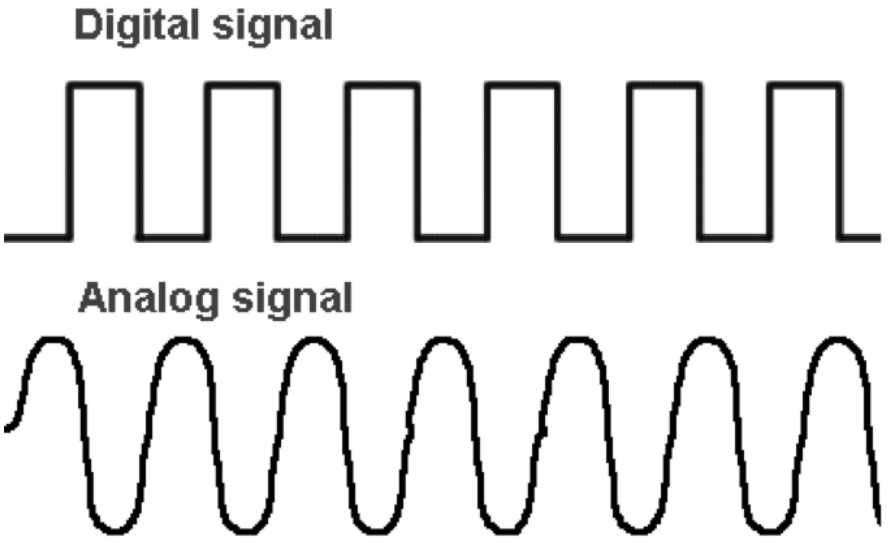
- **Информационная система** – система, в которой реализованы информационные процессы

Примеры

Информационный процесс	**Информационная система**
Регистрация на сайте (сбор, преобразование, передача, хранение)	Сайт (приложение)
Слушаем музыку (сбор, преобразование, передача, обработка, хранение, воспроизведение)	Интернет-радио (ITunes, Google Play Music)
Программирование на C# (сбор, преобразование, хранение, воспроизведение)	IDE (среда разработки)

Носители информации

- Сигналы
- **Цифровой**
  - Дискретный, представим числом
- Аналоговый
  - Непрерывный любые значения



Формы представления информации

- Текстовая (символы языка)
- Числовая (выражения, формулы)
- Графическая (изображения, графики)
- Звуковая (устная, запись)
- Видео (видеозапись)

- **Данные (data)** – информация, представленная в формализованном (закодированном/цифровом) виде

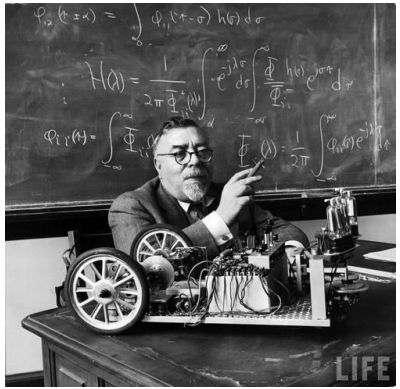
Информационные технологии (IT)

- Технологии работы с информацией
- Создание, развитие и эксплуатация **информационных систем**

- Современные IT – **цифровые**

Кибернетика

- Управление информационными процессами в информационных системах
  - Норберт Винер (1894 – 1964)
    закономерности процессов
    управления и передачи информации
    в машинах, живых организмах, обществе
- Описать работу ИС, указать как и что должно работать…



Алгоритм

- Абу Абдуллах Мухаммед ибн Муса Аль-Хорезми (хорезмский математик IX века)
  - алгоритм, цифра, шифр, алгебра

- **Алгоритм** – набор инструкций, описывающий порядок действий исполнителя для решения задачи (достижения результата)

Математика на рубеже XIX – XX вв.

- Кризис оснований математики
  - Парадоксы (противоречия):
    пример: множество всех подмножеств
- Формалисты, интуиционисты
- Формалист Давид Гильберт
  - 23 проблемы (1900)
  - Проблема разрешения (Entscheidungsproblem)
  - Проверка противоречивости аксиом арифметики(2 проблема)
- Теорема о неполноте - Курт Гёдель (1931)

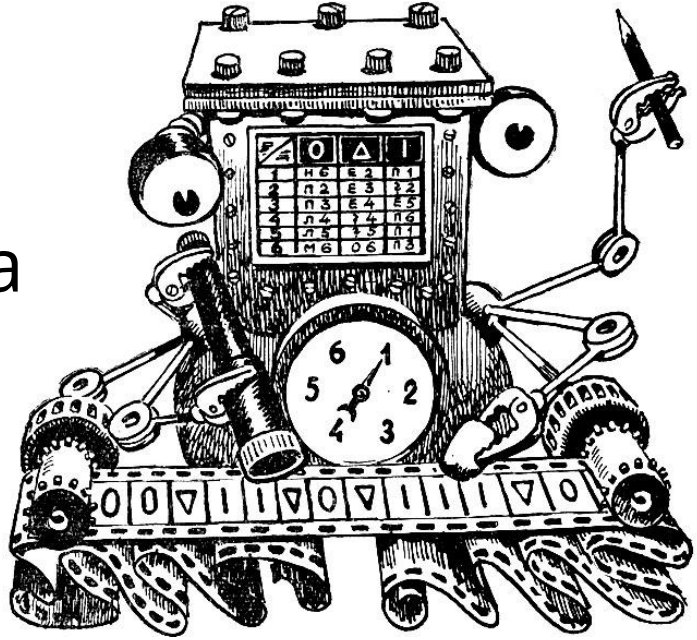
Формализация понятия алгоритма

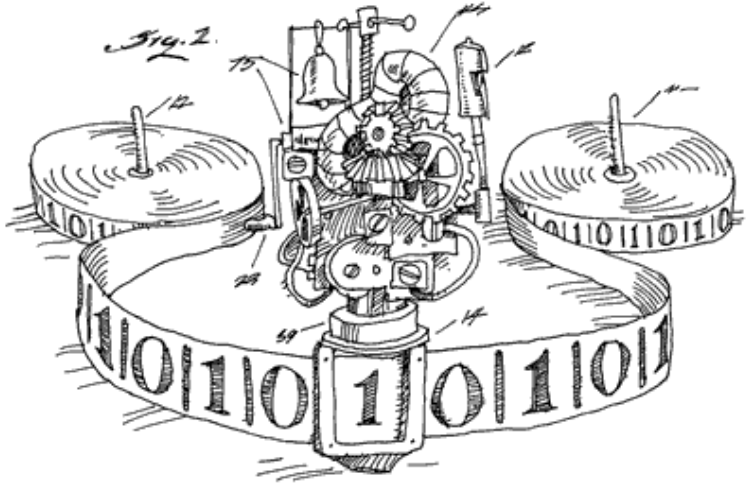
- Формализация понятия вычислений (функции) и алгоритма
- Несколько моделей:
  - Рекурсивные функции Клини
  - λ-исчисление Чёрча
  - Машина Поста
  - Машина Тьюринга
- Современное определение алгоритма сформулировал Алан Тьюринг (1936)

Машина Тьюринга

Алан Тьюринг (1936)

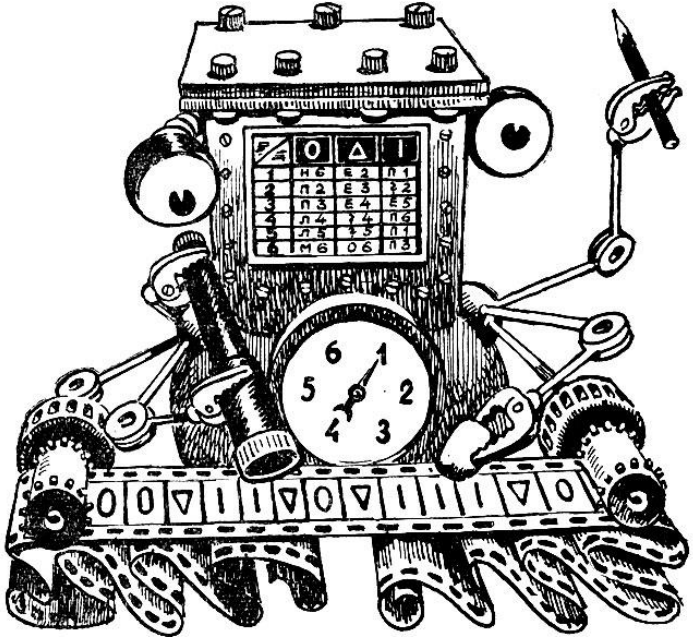
- Абстрактная модель вычислительного устройства





Устройство машины Тьюринга

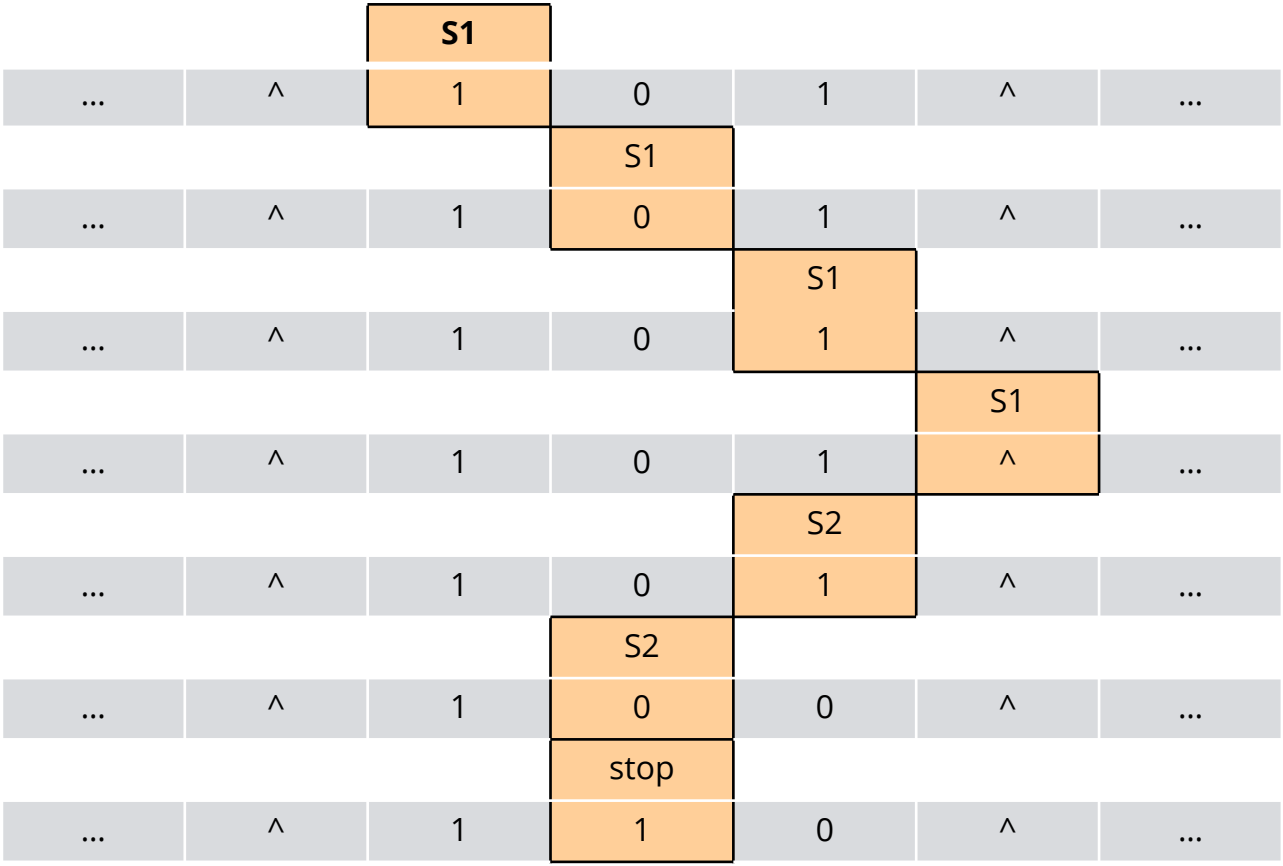
- Алфавит
- Состояния (память)
- Лента
- Считывающая головка
- Программа



	S1	S2
0	S1 ->	1 stop
1	S1 ->	0 S2 <-
^	S2 <-	1 stop

Работа машины Тьюринга

	S1	S2
0	S1 ->	1 stop
1	S1 ->	0 S2 <-
^	S2 <-	1 stop



Сложность алгоритмов

- С моделью МТ можно формально оценивать сложность алгоритмов
- Алгоритмы, реализованные на МТ можно оценивать по:
  - Времени (количеству шагов – переходов состояний)
  - Памяти (количеству состояний)
- Классы сложности по скорости роста функции зависимости сложности от размера входных данных
  - Логарифмический $\mathrm{C(n)} \in \mathrm{O(log(n))}$ [$f(n) \leq c * \log(n)$]
  - Линейный $\mathrm{C(n)} \in \mathrm{O(n)}$
  - Полиномиальный $\mathrm{C(n)} \in \mathrm{O}\left(\mathrm{n^k}\right)$
  - Экспоненциальный $\mathrm{C(n)} \in \mathrm{O}(k^{\mathrm{n}})$

Программа МТ - данные

- Написание программ для МТ - **программирование**
- Программу МТ можно выписать в текстовом виде и перевести в числа – преобразовать в цифровую информацию (код МТ – текст в двоичном алфавите)
- Таким образом, количество всевозможных МТ счётно *(можем установить соответствие с натуральными числами)*

Алгоритмическая разрешимость

- Машин Тьюринга счётное количество
- **Задач – несчётное количество**
  - Пример: по вещественному числу определить,
    поступало ли оно на вход раньше
    (или просто вывести все вещественные числа в интервале [0,1])
    (Следствие теоремы Кантора – множество вещественных чисел несчётно)
- Существует ∞ много задач, которые нельзя решить МТ
- Реальный пример:
  - Задача об останове. По программе МТ и входным данным определить остановится машина на этом входе или нет
- Семантические задачи по статическому анализу кода часто являются неразрешимой задачей

Практически неразрешимые задачи

- Среди алгоритмически разрешимых задач есть задачи, **не разрешимые за приемлемое время**
- Задача, имеющая полиномиальный алгоритм решения – практически разрешима (в классе P)
- Также есть интересный класс задач NP
  - Нет эффективного алгоритма поиска решения, но есть эффективный алгоритм проверки корректности решения
- Ответ на вопрос *P=NP? – самая большая загадка CS*
  - Большинство ученых полагают, что они не равны

Тезис Чёрча – Тьюринга

- Любой интуитивно-вычислимый алгоритм может быть реализован на машине Тьюринга
  - Любая алгоритмически разрешимая задача может быть решена на МТ
- Другие формальные модели, удовлетворяющие этому тезису – **Тьюринг-полные**
- + На МТ можно за полиномиальное время смоделировать работу современного компьютера

Универсальный вычислитель

- Моделирование работы других МТ
  - На вход подают код другой МТ и входные данные,
    универсальная МТ выдает ответ, как если бы работала другая МТ
    (аналог универсальной функции)
- Теорема о существовании универсальной машины Тьюринга
  - То, без чего не было бы современного цифрового мира

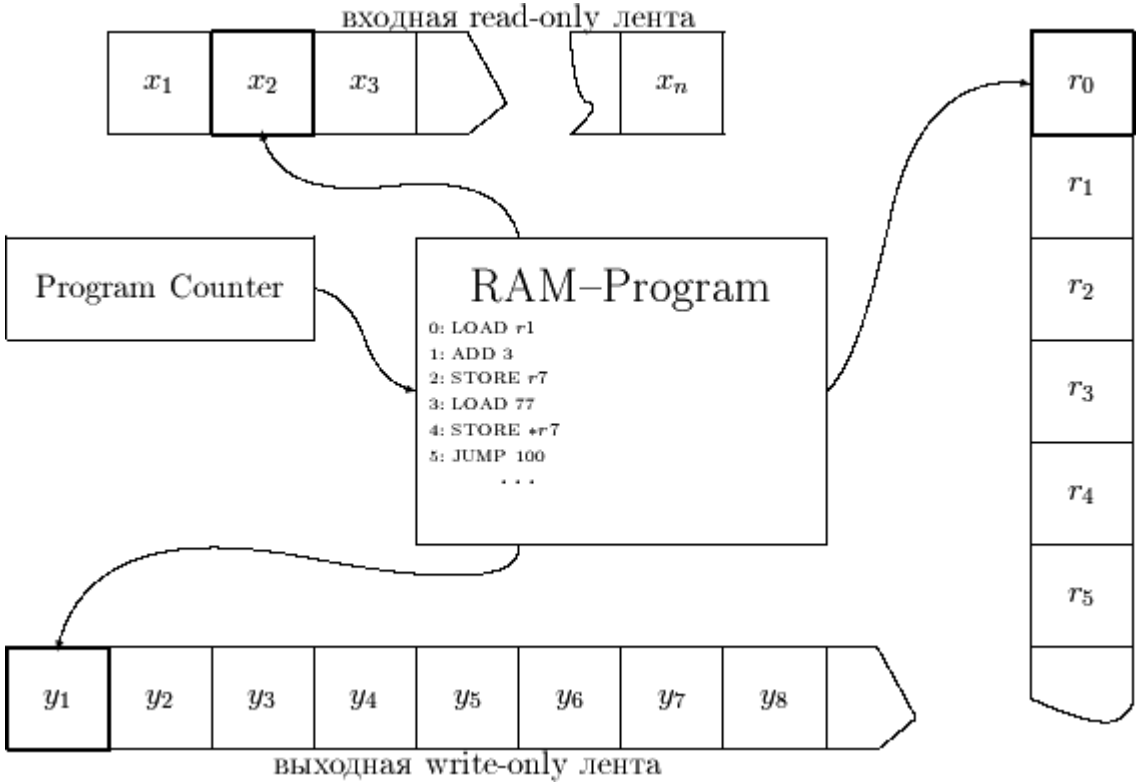
Объяснение

- МТ – модель вычислительного устройства, решающего конкретную задачу (вычисляющую конкретную функцию)
- Если взять универсальную МТ, подавать на вход код программ других МТ – сможем выполнять на одном устройстве все возможные алгоритмы
  - главное – уметь писать программы
- Одно устройство, много алгоритмов, программный код… Ничего не напоминает?

Программирование

- Теорема о существовании универсальной машины Тьюринга – обоснование наличия программирования как деятельности
  - Не нужно строить кучу разных устройств для каждого алгоритма
  - Нужен только один универсальный вычислитель (computer), на котором мы будем выполнять программы, записанные на определенном языке (код программы)

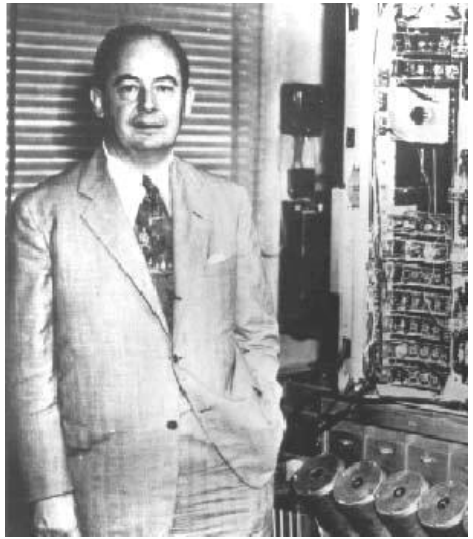
Random Access Machine



- Тьюринг полна
- Эквивалентна УМТ
- RA Регистры
- Операции
  - HALT
  - READ | WRITE
  - LOAD
  - STORE
  - NEG
  - LSHIFT | RSHIFT
  - JUMP | JG

Архитектура фон Неймана

Джон фон Нейман (1945)



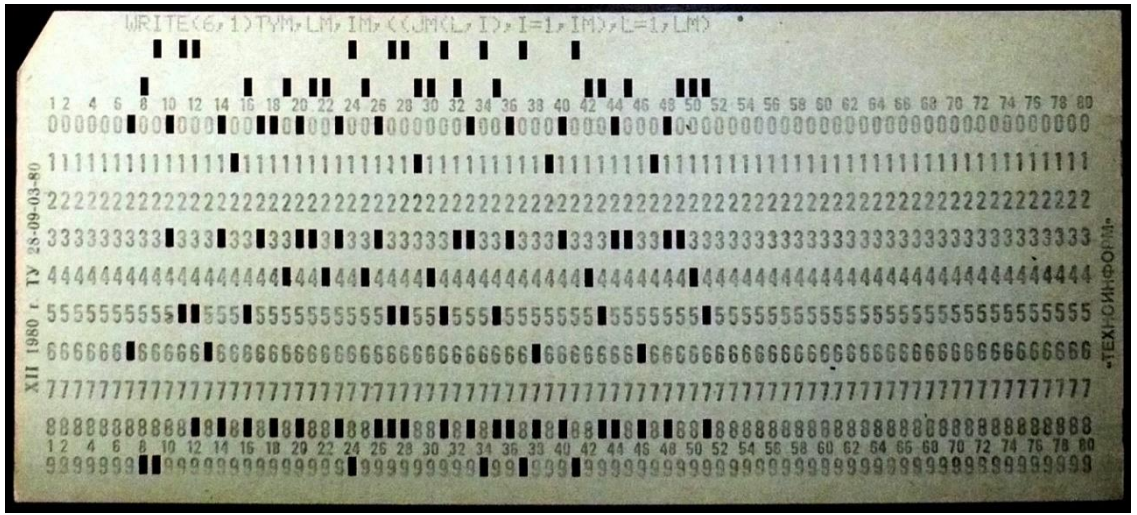
- Принципы:
  - Однородность памяти
    - Команды и данные в общей памяти (нет привязки к устройству)
  - Адресность
    - Память – пронумерованные ячейки
  - Программное управление
    - Программа – последовательность команд
  - Двоичное кодирование

Архитектура фон Неймана

Машинные коды и ассемблер

- Ассемблер – язык низкого уровня (детализация на уровне процессора)
- Пример: вывести 10 букв A

```
_main:
mov $10, %cl
loopy:
push $65
call _putchar
dec %cl
cmp $0, %cl
jne loopy
```



Языки высокого уровня

- Уход от адресов, регистров и операций «переноса» (низкоуровневых операций)
- Понятие переменная
- Условные и циклические структуры управления
- Акцент на обработке данных
  - Обработка цифровых данных (цифры и числа), это - математические операции
- Первый язык высокого уровня – Fortran (1957)

Язык программирования

- Синтаксис (*грамматика*) – правила написания корректных программ

- Программа, переводящая корректные *программы* на языке программирования (текстовые файлы .pas, .cpp, .py, .java, .cs) *в машинный код*

Компилятор / интерпретатор

- Компилятор – трансляция всей программы на ЯП высокого уровня в машинный код
  - Fortran, C \ C++, Pascal \ Delphi, Go, Rust
- Интерпретатор – построчная трансляция и выполнение
  - Python, PHP, Ruby
- Компиляция в байт-код с трансляцией в машинный код во время выполнения
  - Байт-код Java + JVM
  - Стековый байт-код .NET (CIL) + CLR

Первые языки высокого уровня

60-е годы

- Языки тесно связаны с математическим аппаратом
  - Программа – алгоритм,
    алгоритм – вычисление функции
  - Программа – функция
    (преобразует аргумент в значение)
  - Проверка правильности программы через свойства реализуемой ей функции
    (область определения, область значения …)

Программа – сложная функция

- Каждая команда – тоже функция
- Программа – композиция функций (сложная функция)

```
begin
read(x);
x := x * x; // отработала функция f(x) = x * x
x := x + 100500; // отработала функция g(x) = x + 100500
write(x)
end.
```

- $F(x) = g(f(x))$ или $g \circ f$
- ; - оператор сложной функции
  - Поэтому нет перед end или else – значение никуда не подставляется
  - После end – обязательно (весь блок можно подставить в другой)

Появление ПК

- 1970-е – появление персональных компьютеров
- Рост популярности компьютеров
- Проникновение в бизнес и общество
- Рост задач и программных проектов
- Отдаление от математических основ
  Проблемы формальной верификации
  - Как верифицировать Word? Quake? Linux?

Язык C

Деннис Ритчи, Кен Томпсон (1972)

- Премия Тьюринга (1983)
- Медаль Хэмминга (1990)
- Медаль «Пионер компьютерной техники» (1994)
- и многие другие…

- Разработан программистами для программистов
  - Реализации ОС Unix
- Позволяет эффективно (легко и быстро) разрабатывать новые приложения
- Оказал колоссальное влияние на разработку ПО

Что принёс C

- Работу с памятью через указатели
- Ассемблер «высокого уровня»
- Библиотека языка
- Пространства имён (namespace)
- Структуры, объединения (struct, union)
- Активное использование препроцессора

C++

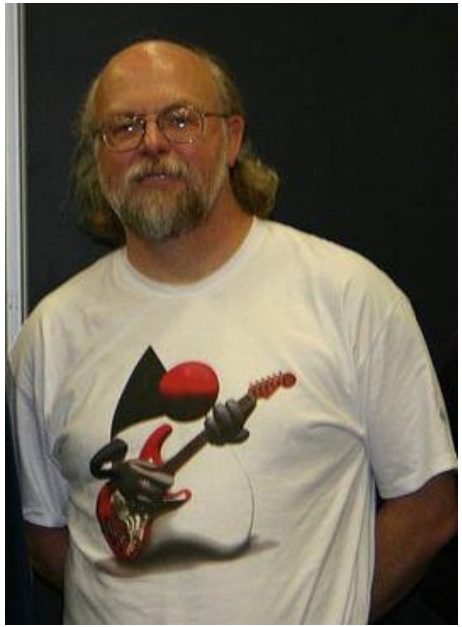
Бьёрн Страуструп (1983)

- Си с объектно-ориентированными возможностями и многим другим…
- *Р. Керниган: «Си – инструмент, острый, как бритва: с его помощью можно создать и элегантную программу, и кровавое месиво»*
- Для C++ это справедливо еще в большей степени

Java

James Gosling, проект «Oak» (1995)

- Java 1.0 (1996)
- Java 8 (1.8) (2014)
- Объектно-ориентированный язык программирования
- Си-образный синтаксис
- Кроссплатформенный
- Применение
  - Бизнес-решинея
  - Клиент-серверные приложения
  - Мобильные устройства



JVM

- Java Virtual Machine – реализует кросс-платформенность для любого приложения на Java
- Java-приложения компилируются в байт-код, выполняемый JVM
- Компилятор интерпретируемого типа с Just-In-Time трансляцией в машинный код

C#

Андерс Хейлсберг (2000)

- C# 1.0 (2002)
- C# 6.0 (2015)

  C → C++ → C++++ (C#)
- **Язык для платформы .NET**
- Постарался взять лучшее у других и решить их проблемы
  - Синтаксис и ОО модель Java
  - Структуры C, перечисления, перегрузка операторов C++
  - Свойства Visual Basic, Delphi

Использование C#

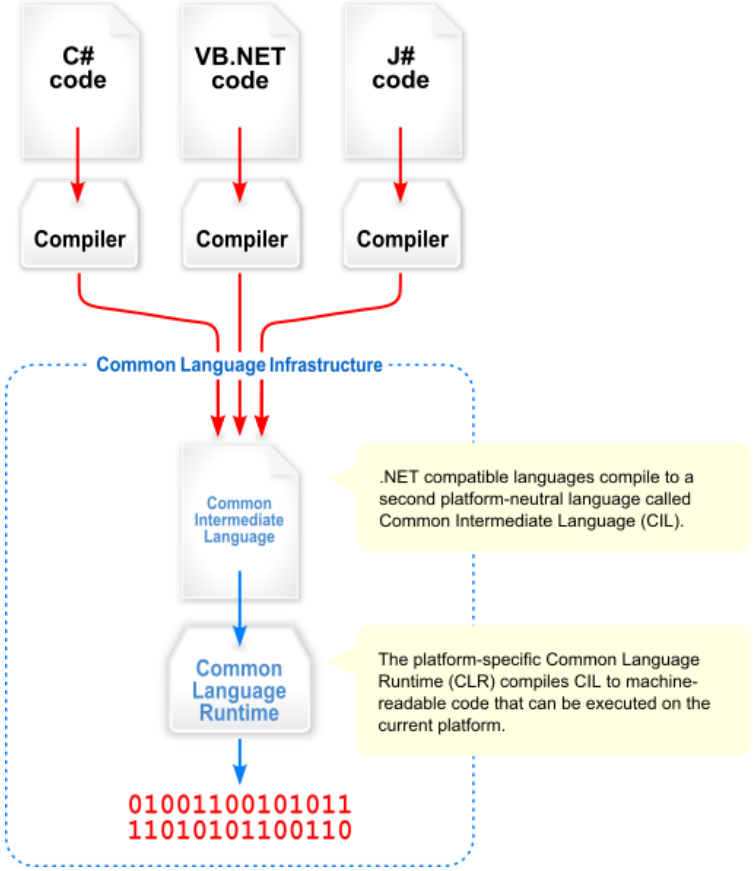
- Разработка приложений универсальной платформы Windows (UWP)
- Кроссплатформенная разработка (.NET Core, Xamarin)
- Разработка веб-приложений ASP.NET MVC
- Работа с базами данных
- Разработка сервисов (WPF)

.NET Framework

- Программная платформа от Microsoft (2002)
- Кроссплатформенная (Windows, ReactOS, OS X, Linux)
- Создавался для разработки современных приложений и корпоративных решений, а также «Интернета следующего поколения» (Б. Гейтс, 2000)
- **В основе – CLR** – среда исполнения CIL кода, сгенерированного .NET языками (не только C#)

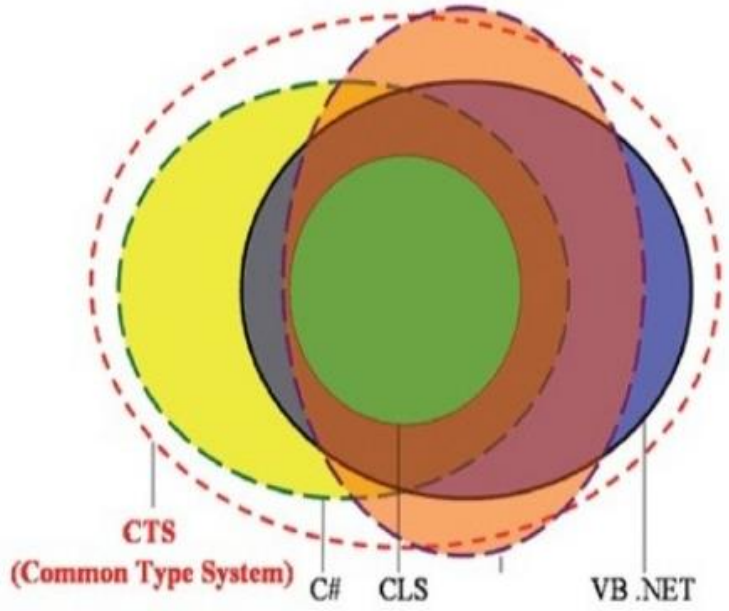
CLR

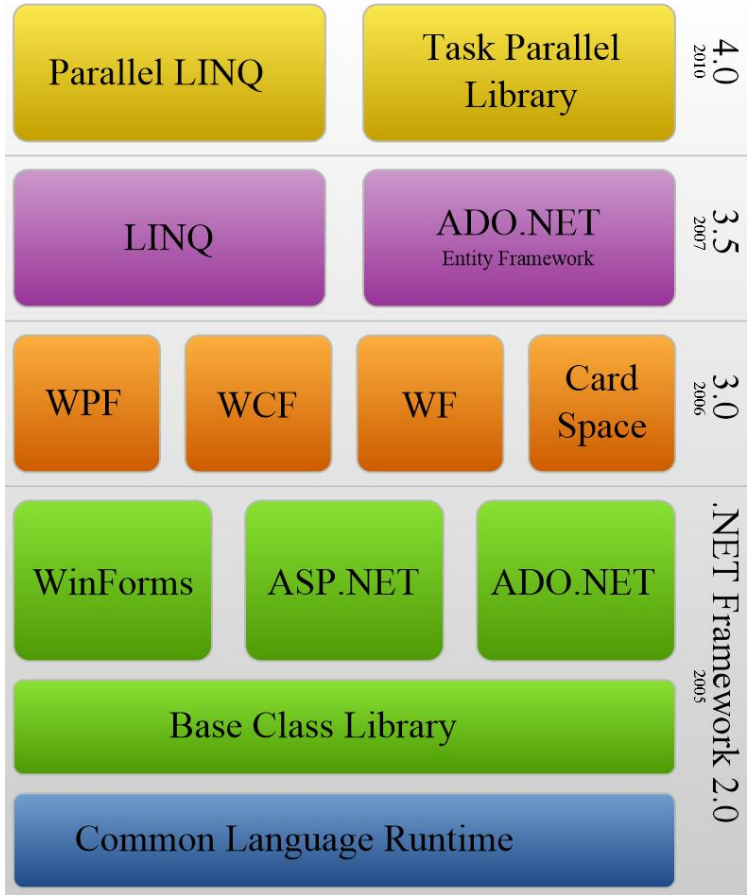
- Загружает, исполняет код
- Транслирует код CIL в машинный код
- Управляет памятью
- Следит за безопасностью кода и доступа
- Связывает модули



Архитектура .NET Framework

Компоненты .NET Framework





Вопросы?

e-mail: marchenko@it.kfu.ru