



Введение в теорию компиляторов

Введение в теорию компиляторов



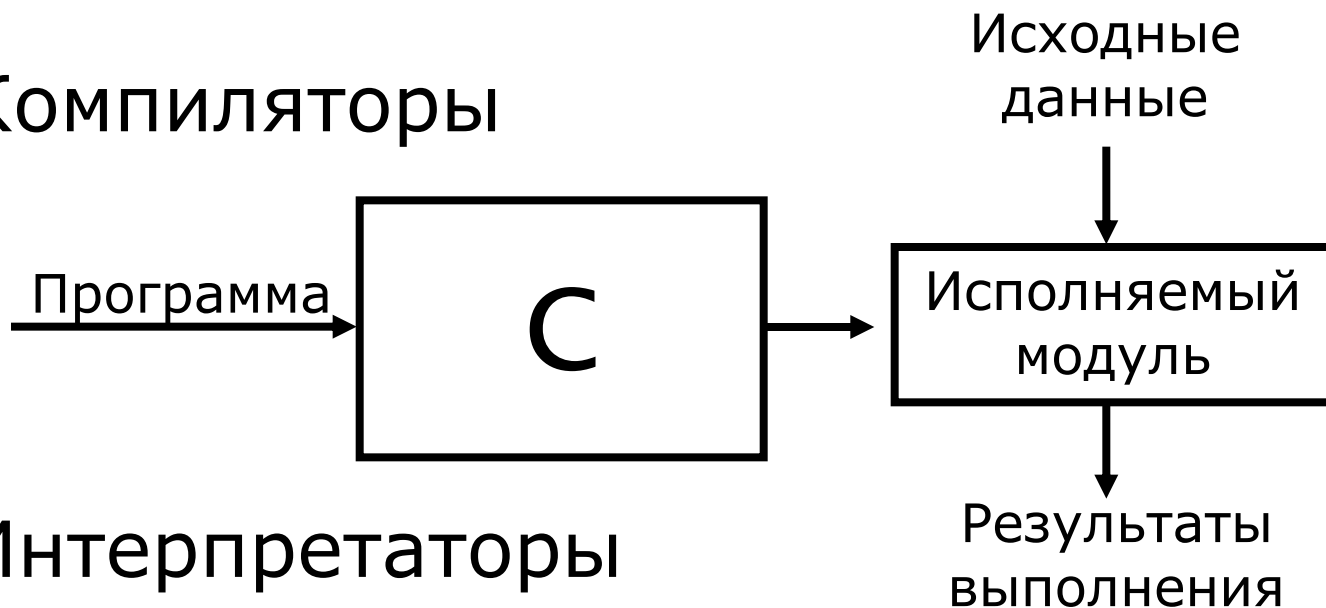


Трансляторы

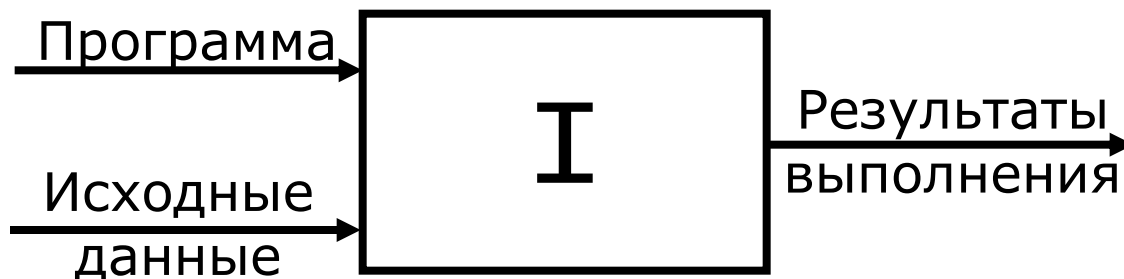
- Транслятор — программа или техническое средство, выполняющее трансляцию программы.
- Трансляция программы — преобразование программы, представленной на одном из языков программирования, в программу на другом языке и, в определённом смысле, равносильную первой.

Трансляторы

- Компиляторы



- Интерпретаторы



История трансляторов

1954 г.



Программное обеспечение



Аппаратное обеспечение

IBM 704

История трансляторов

Speedcode (Speedcoding)

- ✓ Первый интерпретатор
- ✓ Первый язык высокого уровня для компьютеров IBM

**В 10–20 раз
медленнее**

**310
машинных
слов
(30%)**



Джон Бэкус

История трансляторов

FORTRAN I

Formulas

Translated

1954–1957 гг.

1958 г. 50% программ



Джон Бэкус



FORTRAN I

- Первый компилятор.
- Привёл к активному развитию теории компиляторов.
- Современные компиляторы имеют структуру, аналогичную FORTRAN I.



Структура компилятора

1.

Структура компилятора

- Шаг 1: распознавание слов
 - наименьшая единица программы после букв


Это предложение на русском языке.

пре длож е ниел иэ то

Структура компилятора

- На этапе лексического анализа программа разбивается на «слова» — **токены**.

```
if x == y then z = 1; else z = 2;
```

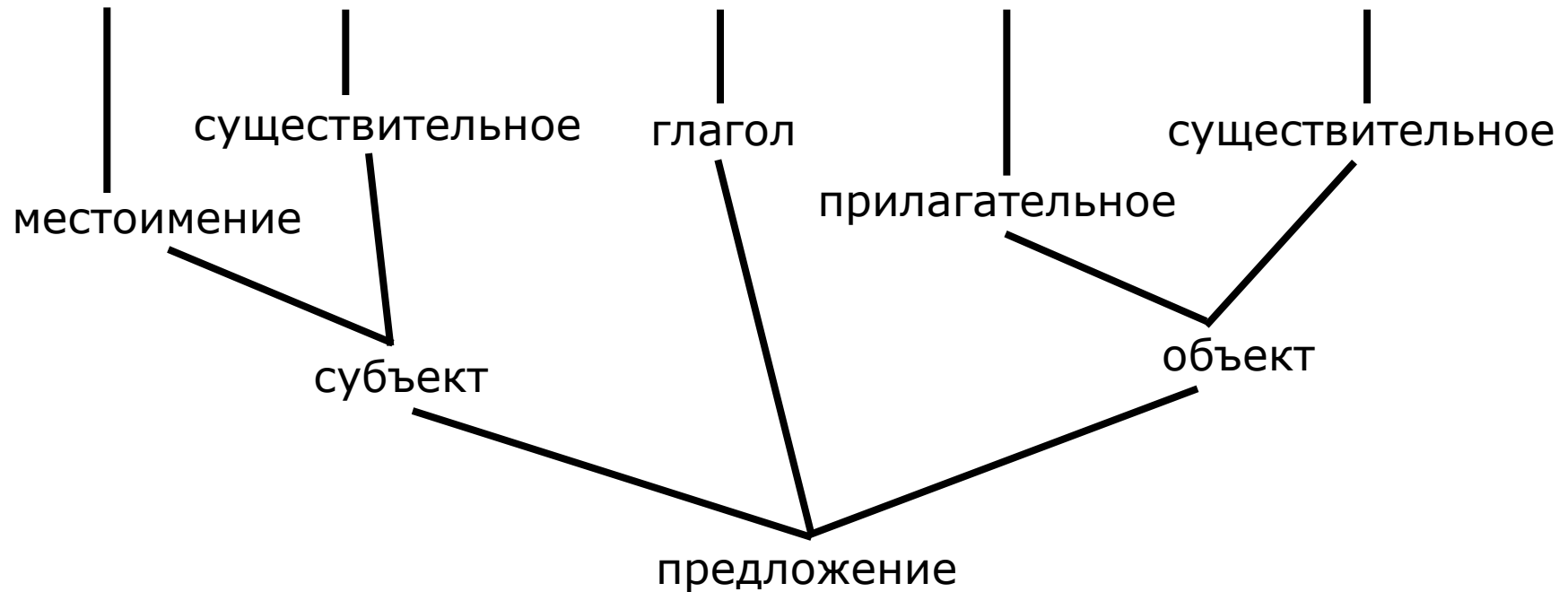




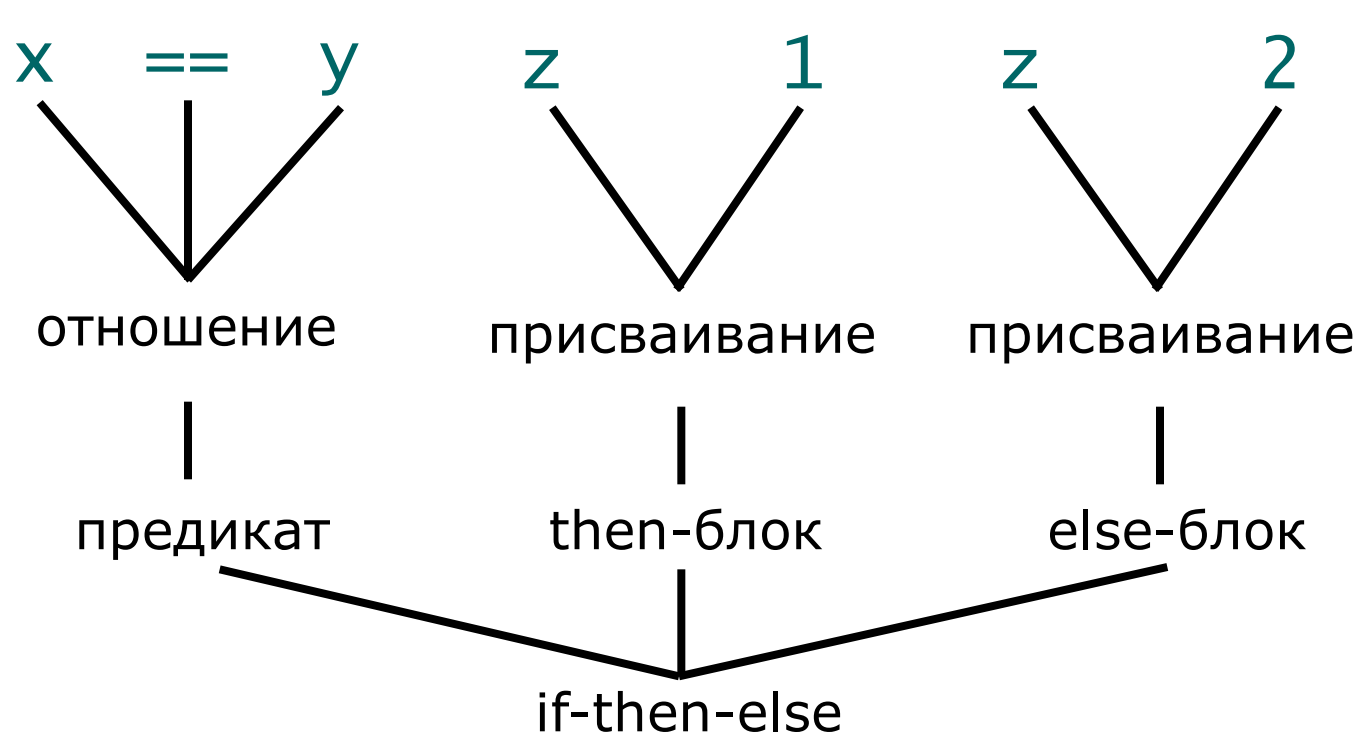
Структура компилятора

- Шаг 2:
понимание структуры предложения
- Парсинг (синтаксический анализ)

Этот текст является простым предложением



if x == y then z = 1; else z = 2;





Структура компилятора

- Шаг 3:
понимание смысла предложения
 - Это сложно!
- Компиляторы выполняют ограниченный семантический анализ:
 - обнаружение несоответствий

Алиса сошла со сцены,
и на неё посыпалось конфетти.

```
{  
    int Alice = 3;  
    {  
        int Alice = 4;  
        printf("%d", Alice);  
    }  
}
```



Структура компилятора

- Компиляторы выполняют различные проверки.
- Пример:

Алиса забыл свою книгу дома.

- «Type mismatch» между словами Алиса и забыл.

Структура компилятора

- Шаг 4: оптимизация.

- Не имеет аналога в естественных языках.
- Похожая операция — правка текста.

В четверг

~~На следующий день после среды~~ пошёл дождь.

- Автоматическое изменение программ таким образом, чтобы они:
 - выполнялись быстрее;
 - использовали меньше памяти.

$$x = x * 0.$$

**Такое правило
верно не всегда!**

$$x = 0,$$

- ✓ Верно для целочисленных типов.
- ✓ Неверно для вещественных типов.



Структура компилятора

- Шаг 5: генерация кода
- Перевод на другой язык.



Языки программирования

- Почему создано так много языков программирования?
- Почему создаются новые языки программирования?
- Какой язык программирования — хороший?



Языки программирования

- **Почему создано так много языков программирования?**
- Почему создаются новые языки программирования?
- Какой язык программирования — хороший?



Языки программирования

○ Разнообразие предметных областей.

Научные
вычисления

- ✓ Хорошая поддержка вещественных типов.
- ✓ Хорошая поддержка массивов.
- ✓ Параллелизм.

Бизнес-задачи

- ✓ Устойчивость к ошибкам.
- ✓ Генерация отчётов.
- ✓ Анализ данных.

Системное
программирование

- ✓ Управление ресурсами.
- ✓ Ограничения по скорости.



Языки программирования

- Почему создано так много языков программирования?
- **Почему создаются новые языки программирования?**
- Какой язык программирования — хороший?



Языки программирования

- Обучение программистов — самый большой вклад в стоимость языка.
 1. Широко используемые языки изменяются медленно.
 2. Проще создать новый язык.
 3. Новые языки обеспечивают решение новых задач.
 4. Новые языки похожи на старые.



Языки программирования

- Почему создано так много языков программирования?
- Почему создаются новые языки программирования?
- **Какой язык программирования — хороший?**



Языки программирования

- Нет общепринятых способов оценки языков программирования.



Вопросы?
