Лекция 1. Введение. Основы языка Java. Принципы ООП. Основные понятия. Договоренности.

NetCracker®

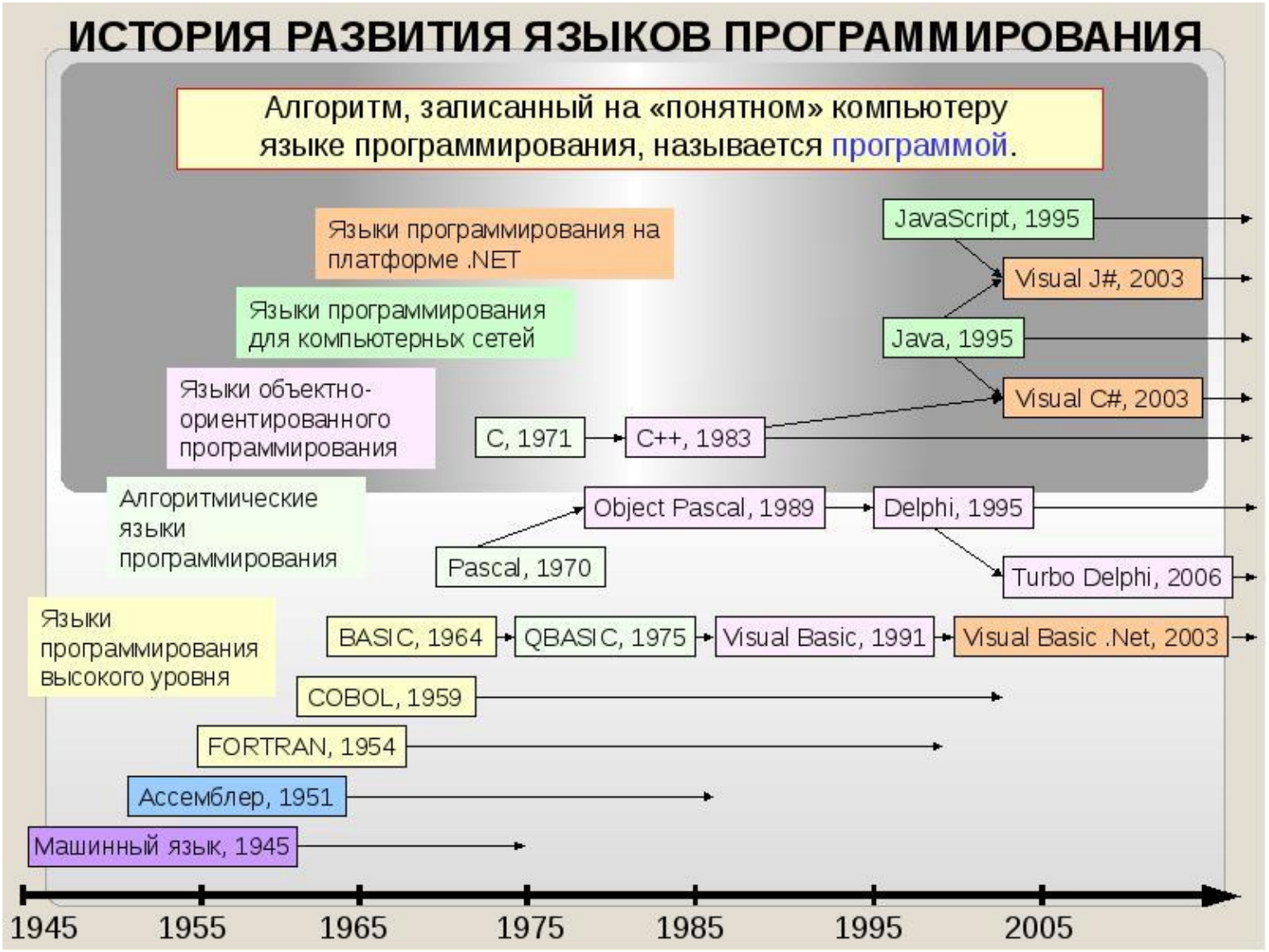
Сетевые Java-технологии

Дисц. “Сетевые Java-технологии “

Семестр	Всего, ч	Аудиторные занятия, ч			Самост. работа студента, ч		Форма контр.
		Всего	Лекции	Практ.	Всего	Инд. задан.	
7	*162*	*60*	*30*	*30*	*102*	*Кр*	*ДСК*

- Эволюция языков программирования, классификация языков программирования, перспективы дальнейшего развития.
- История возникновения языка Java.
- Объектно-ориентированный язык Java и ее особенности.
- Понятие классов и объектов, абстракция, инкапсуляция, наследование, полиморфизм.
- Преимущества и недостатки ООП.

- В 20-х г. XIX ст. Ч.Беббидж подал идею предварительной записи **порядка действий машины** для последующей автоматической реализации расчетов.
- Ада Лавлейс теоретически разработала **методы управления последовательностью расчетов**, описала одну из конструкций языков программирования - **цикл**.
- **Перфокарты** Жозефа Мари Жаккара применялись в аналитической машине Ч. Беббиджа для хранения чисел.
- Революционным моментом в истории языков программирования была разработка Джоном Моучли (Пенсильванский университет) **системы кодирования машинных команд** с помощью специальных символов.
- 1951 г. Джейн Мюррей Хоппер предложила первый в мире **компилятор**, который осуществлял функцию объединения команд и в ходе трансляции, проводил организацию подпрограмм, выделение памяти компьютера, преобразование команд высокого уровня в машинные команды.
- 50-е годы - прогресс развития в области языков программирования, разработанный одна из распространенных в то время алгоритмических языков программирования FORTRAN.


ИСТОРИЯ РАЗВИТИЯ ЯЗЫКОВ ПРОГРАММИРОВАНИЯ
Алгоритм, записанный на «понятном» компьютеру языке программирования, называется программой.
Языки программирования на платформе .NET
Языки программирования для компьютерных сетей
Языки объектно-ориентированного программирования
Алгоритмические языки программирования
Языки программирования высокого уровня
JavaScript, 1995
Visual J#, 2003
Java, 1995
Visual C#, 2003
C, 1971
C++, 1983
Object Pascal, 1989
Delphi, 1995
Pascal, 1970
Turbo Delphi, 2006
BASIC, 1964
QBASIC, 1975
Visual Basic, 1991
Visual Basic .Net, 2003
COBOL, 1959
FORTRAN, 1954
Ассемблер, 1951
Машинный язык, 1945
1945
1955
1965
1975
1985
1995
2005

Создатель Java – Джеймс Гослинг (США)

- Первое применение – бытовая электроника (микроволновые печи, стиральные машины, пульты управления)

Первое название языка – Oak («Дуб»)

- В честь дуба, стоявшего напротив офиса Джеймса Гослинга
- К тому времени уже был ещё один язык Oak

Название Java произошло от сорта кофе

- Это кофе производится на о. Ява (Индонезия)
- Его очень часто употреблял и первые разработчики языка

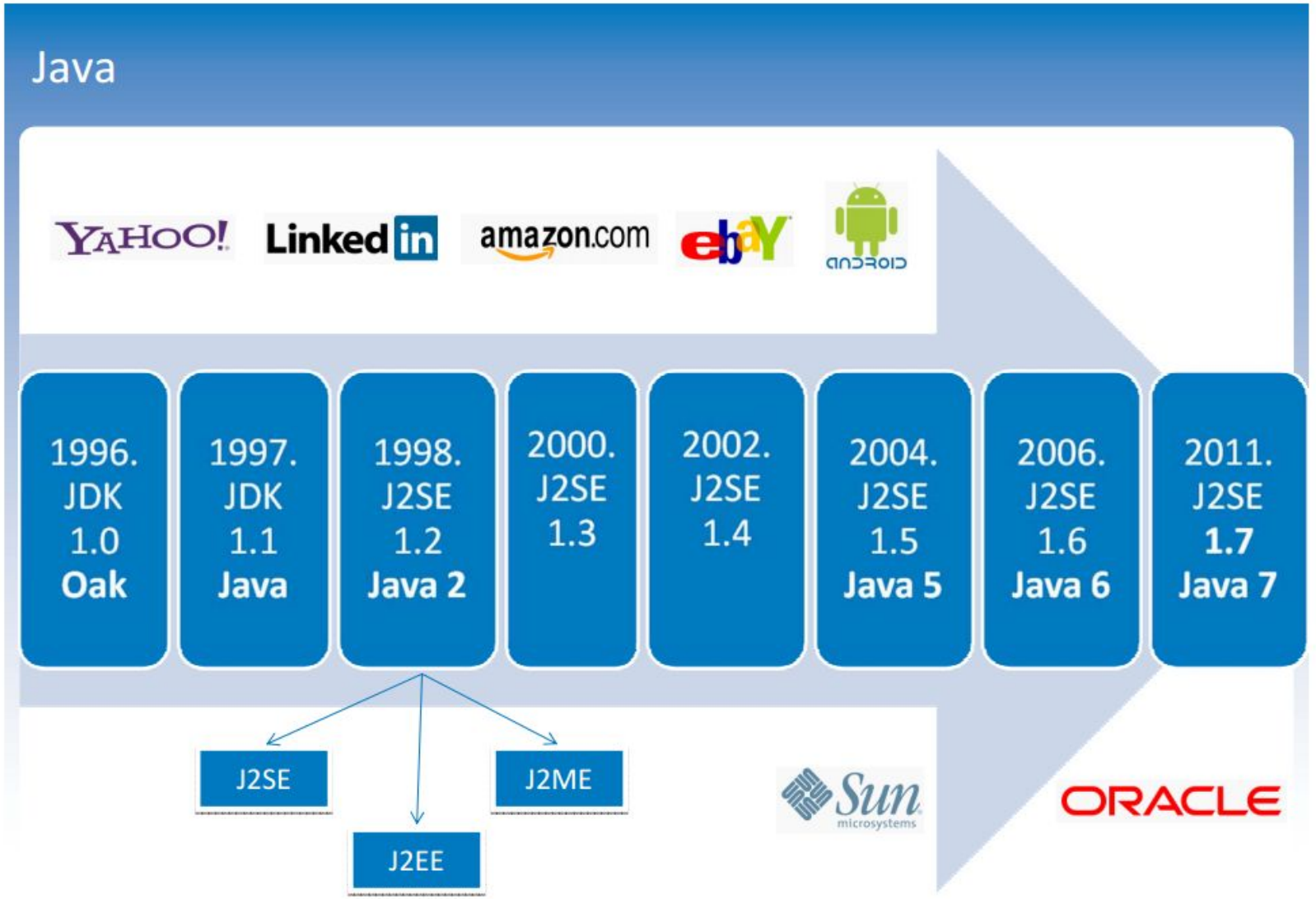
Duke - талисман языка Java

- Ежегодно проводится конкурс Duke Choice Awards
- В 2011 году Duke изменил свой внешний вид

http://www.tiobe.com

Position Aug 2013	Position Aug 2012	Delta in Position	Programming Language
1	2	↑	Java
2	1	↓	C
3	4	↑	C++
4	3	↓	Objective-C
5	6	↑	PHP
6	5	↓	C#
7	7	=	(Visual) Basic
8	8	=	Python
9	11	↑↑	JavaScript
10	10	=	Ruby

Java



- кроссплатформенность
- объектная ориентированность
- привычный синтаксис C/C++
- безопасность
- ориентация на Internet
- динамичность
- простота освоения

- **Java Runtime Environment, JRE** – это исполнительная среда Java, в которой выполняются программы, написанные на этом языке. Среда состоит из виртуальной машины – **Java Virtual Machine(JVM)** и библиотеки Java-классов. JRE является частью JDK.

- **Java Virtual Machine, JVM** – это виртуальная машина Java — основная часть исполняющей среды **JRE**. Виртуальная машина Java интерпретирует и исполняет байт-код Java. Байт-код получают посредством компиляции исходного кода программы с помощью компилятора Java (стандартный - javac).

- **Java Development Kit, JDK** – это бесплатно распространяемый корпорацией Sun комплект разработчика приложений на языке Java, включающий в себя компилятор Java (javac), стандартные библиотеки классов Java, примеры, документацию, различные утилиты и исполнительную систему Java (JRE). В состав JDK не входит интегрированная среда разработки на Java (IDE), поэтому разработчик, использующий только JDK, вынужден использовать внешний текстовый редактор и компилировать свои программы, используя утилиты командной строки.

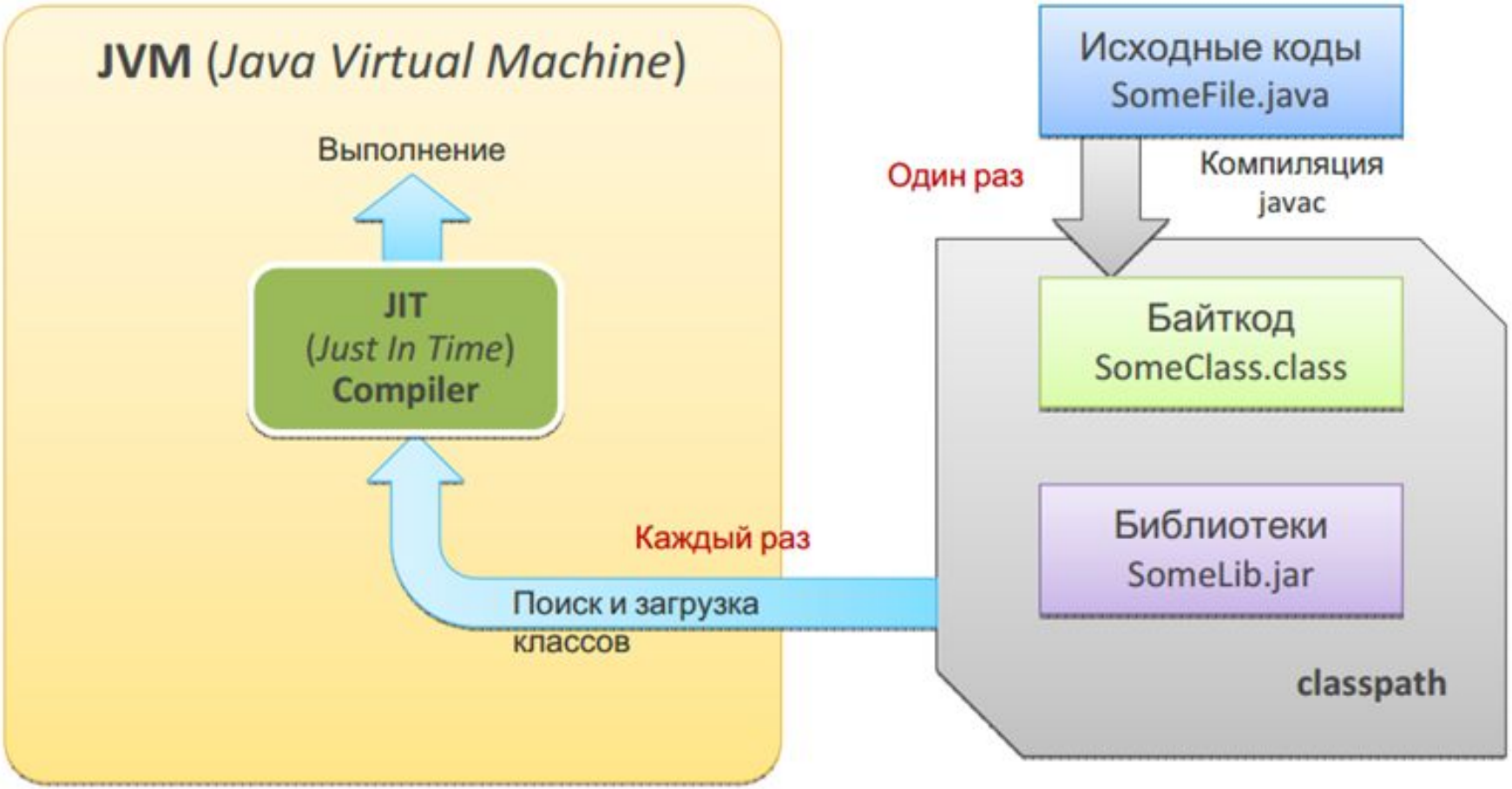
- **Java 2 Standart Edition, J2SE** – это стандартная редакция языка Java, используемая для разработки простых Java-приложений. Используя данную редакцию можно создавать апплеты, консольные приложения, приложения с графическим интерфейсом пользователя.

- **Java 2 Enterprise Edition, J2EE** – это редакция языка Java для разработки распределенных приложений масштаба предприятия. Включает в себя технологию Enterprise Java Beans (EJB), Java Server Pages (JSP) и сервлеты(Servlets). Каждая из этих технологии, в свою очередь также имеет свой отдельный номер версии..

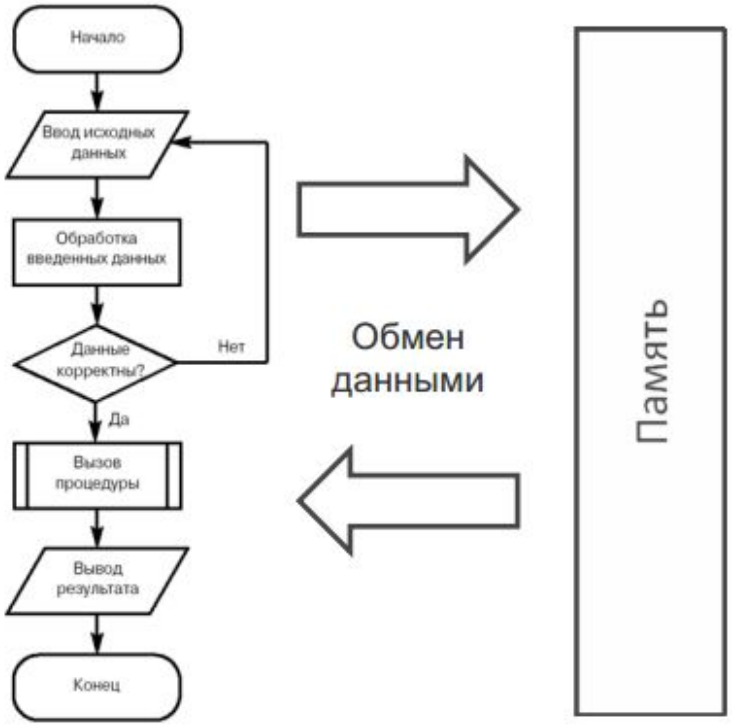
- **Java 2 Micro Edition, J2ME** – это редакция языка Java для разработки приложений для микрокомпьютеров (мобильных устройств). В нее входят "облегченные" стандартные классы и классы для написания мидлетов (Midlets). Мидлеты – это аналоги аплетов, но только приспособленные специально для небольших устройств. В них также поддерживается графика, звук, реакция на события (нажатие кнопок и т.д.). Java ME наиболее полно соответствует начальному предназначению Java – платформы для написания программ для бытовых устройств.

JRE (*Java Runtime Environment*)
программное обеспечение, необходимое для запуска приложений, созданных с помощью Java. Состоит из виртуальной машины — Java Virtual Machine и библиотеки Java-классов

JDK (*Java Development Kit*)
комплект разработчика приложений на языке Java, включающий в себя компилятор Java (javac), стандартные библиотеки классов Java, примеры, документацию, различные утилиты и исполнительную систему Java (JRE).

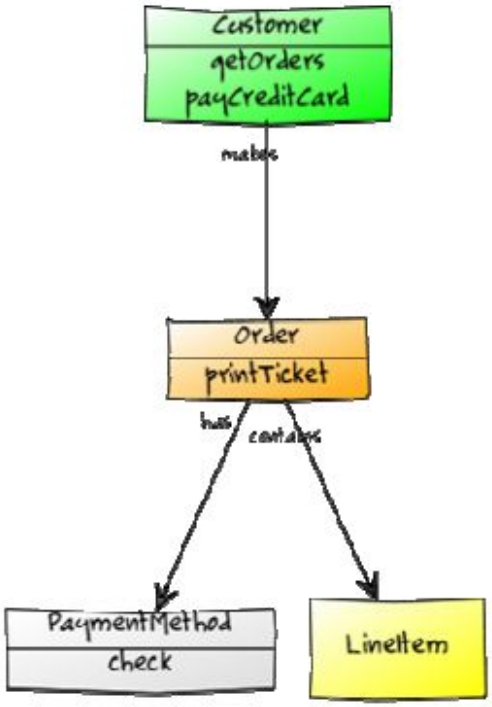


Процедурный подход



Программа – алгоритм последовательного вызова проедур изменения данных в памяти.

Объектно-ориентированный



Программа – взаимодействие объектов, компонентов, отсылка и обработка событий.

Достоинства

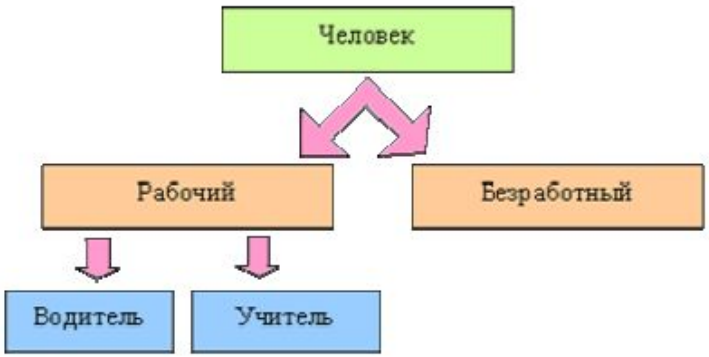
- Конструирование из простых компонент (абстракция).
- Данные связаны с операциями обработки.
- Инкапсуляция делает код более безопасным.
- Повторное использование компонент.
- Обобщенные алгоритмы.
- Изменение поведения во время выполнения (полиморфизм).
- Создание полуфабрикатов или фреймворков.

Недостатки

- Необходимость изучения концепций ООП
- Обилие библиотек компонентов повторного использования.
- Проектирование классов сложный процесс.
- Меньшее быстродействие
- Больший расход памяти
- Излишняя универсальность

Абстракция позволяет акцентировать внимание на способах использования объекта и не вдаваться в подробности его реализации.

- **Инкапсуляция** — свойство языка программирования, позволяющее объединить и защитить данные и код в объекте и скрыть реализацию объекта от пользователя (прикладного программиста). При этом пользователю предоставляется только спецификация (интерфейс) объекта.



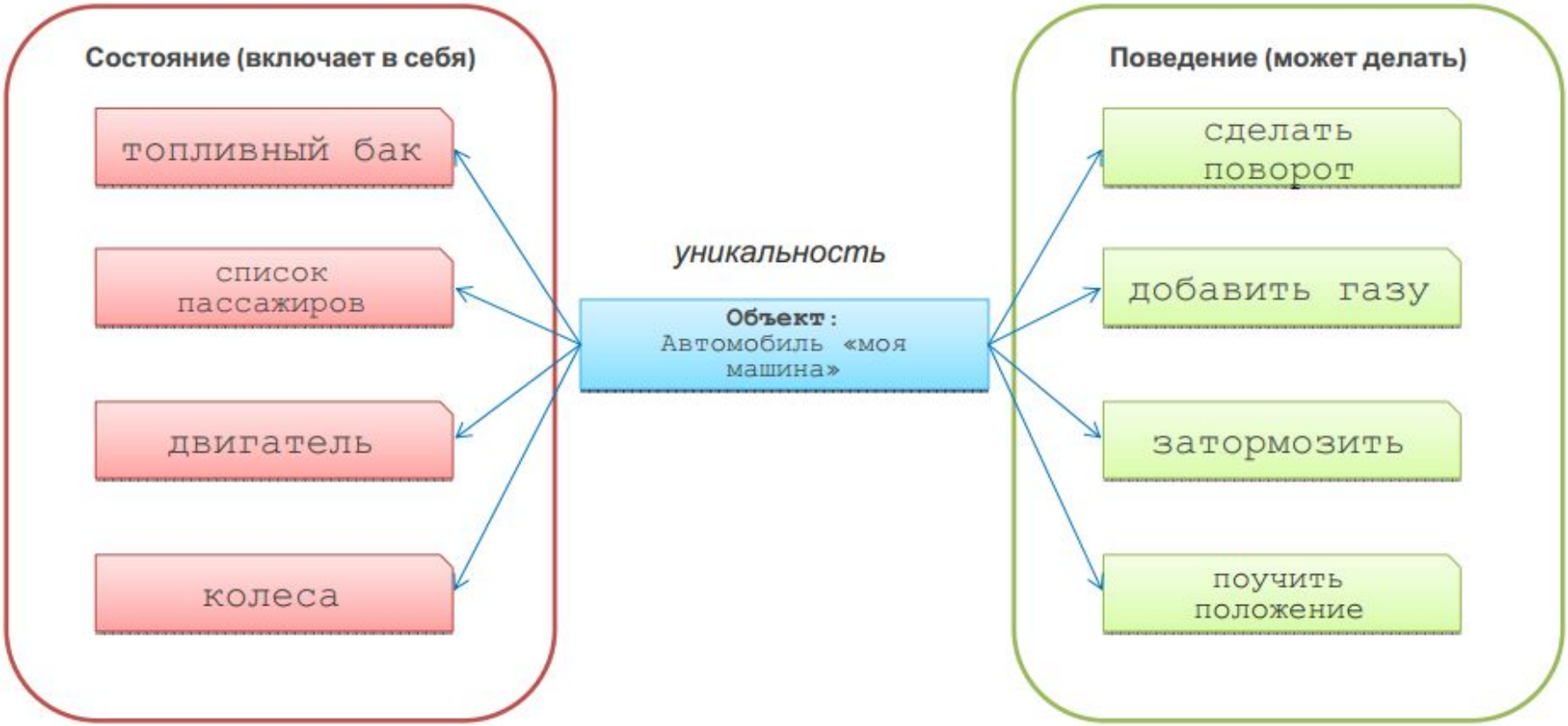
- **Наследование** – описание нового класса на основе уже существующего (родительского), при этом свойства и функциональность родительского класса заимствуются новым классом.
  - Позволяет избавиться от дублирования кода.
  - Позволяет добавить новую функциональность в класс.
  - Позволяет описать отношения обобщения

```
class Грузовик extends Автомобиль {
    Кузов кузов;
    процедура загрузить(груз) {
        кузов.загрузить(груз);
        кузов.проверитьПерегруз();
    }
}

Грузовик мойГрузовик = new Грузовик();
мойГрузовик.загрузить(помидоры);
мойГрузовик.добавитьГазу(немного);
```


- **Полиморфизм** — возможность объектов с одинаковой спецификацией иметь различную реализацию. При этом различные объекты могут быть использованы одинаковым образом.
  - Позволяет писать более абстрактные программы.
  - Позволяет усилить повторное использование кода.
- Реализуется с помощью наследования и интерфейсов.

```
class Автостоянка {
    СписокАвтомобилей автомобили;
    процедура добавить(автомобиль) {
        автомобиль.закрыть();
        автомобиль.включитьСигнализацию();
        автомобили.добавить(автомобиль);
    }
}
Автостоянка стоянка = дом.гдеБлижайшаяАвтостоянка()
стоянка.добавить(мойАвтомобиль);
стоянка.добавить(мойГрузовик);
```


```
class Автомобиль {
    ТопливныйБак бак;
    СписокПассажиров пассажиры;
    Колеса колеса;
    Двигатель двигатель;

    процедура сделатьПоворот(угол);
    процедура добавитьГазу(уровень);
    процедура затормозить();
    Координаты получитьПоложение();
}
```

```
class Двигатель {
    Свечи свечи;
    Цилиндры цилиндры;
    Карбюратор карбюратор;

    процедура увеличитьОбороты();
    процедура переключитьРежим(режим);
}
```

```
Автомобиль мояМашина = я.купитьМашину();
Двигатель двигательМоейМашины = мояМашина.двигатель;
двигательМоейМашины.переключитьРежим(форсированный);
```


Объект –
экземпляр класса.

- состояние
- поведение
- уникальность

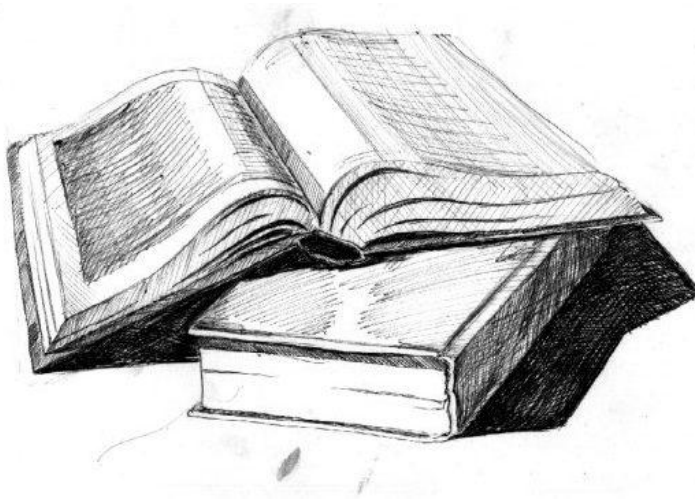
Класс –
тип объекта.

- определяет набор свойств и интерфейс взаимодействия.
- определяет поведение (реализацию)

```
Автомобиль мой = new Автомобиль();
Автомобиль жены = new Автомобиль();
Топливо топливо = жены.слитьТопливо();
мой.заправить(топливо);
мой.двигаться();
```

```
class Автомобиль {
    Двигатель двигатель;
    процедура двигаться() {
        двигатель.завести();
        двигатель.добавитьГазу();
    }
}
```


- Эккель Б. Философия Java. –Эккель Б. Философия Java. – СПб.: Питер, 2009. – 640 с.
- http://www.intuit.ru/studies/courses/16/16/info
- Шилдт Г. Java. Полное руководство. – СПб.: Вильямс, 2012. – 1104 с.
- Шилдт Г. Полный справочник по Java. Java SE 6 Edition. – СПб.: Вильямс, 2007. – 1040 с.
- Шилдт Г., Холмс Д. Искусство программирования на Java. – СПб.: Вильямс, 2005. – 333 с.
- Шилдт Г. Java. для начинающих. – СПб.: Ви



Q&A

Thank you!

