Лекция: Объектно-Ориентированное Программирование

Учебный курс

Кензин Максим Юрьевич
Кафедра Информационных технологий
ИМИТ ИГУ / Институт динамики
систем и теории управления СО РАН

ИМИТ ИГУ, 2022

Технологии ПП

Технологии параллельного программирования:

1) Ориентация на кластеры/суперкомпьютеры (MPI);

2) Ориентация на многоядерные CPU системы (POSIX threads, Boost threads, Java threads, java.util.concurrent, OpenMP, Intell TBB);

3) Ориентация на многоядерные GPU системы (NVIDIA CUDA).

OpenMP

OpenMP [omp] - это API-интерфейс, который является отраслевым стандартом для создания параллельных приложений для компьютеров с совместным использованием памяти.

Главная задача OpenMP - облегчить написание программ, ориентированных на циклы. OpenMP основывается на модели программирования "разветвление-объединение" (fork-join).

OpenMP

Языковые конструкции в OpenMP определены как директивы компилятора, которые сообщают компилятору, что он должен делать, чтобы реализовать требуемый параллелизм.

В C и C++ такие директивы называются "прагмы".

Прагма OpenMP всегда имеет один и тот же вид:

```
#pragma omp construct_name one_or_more_clauses
```

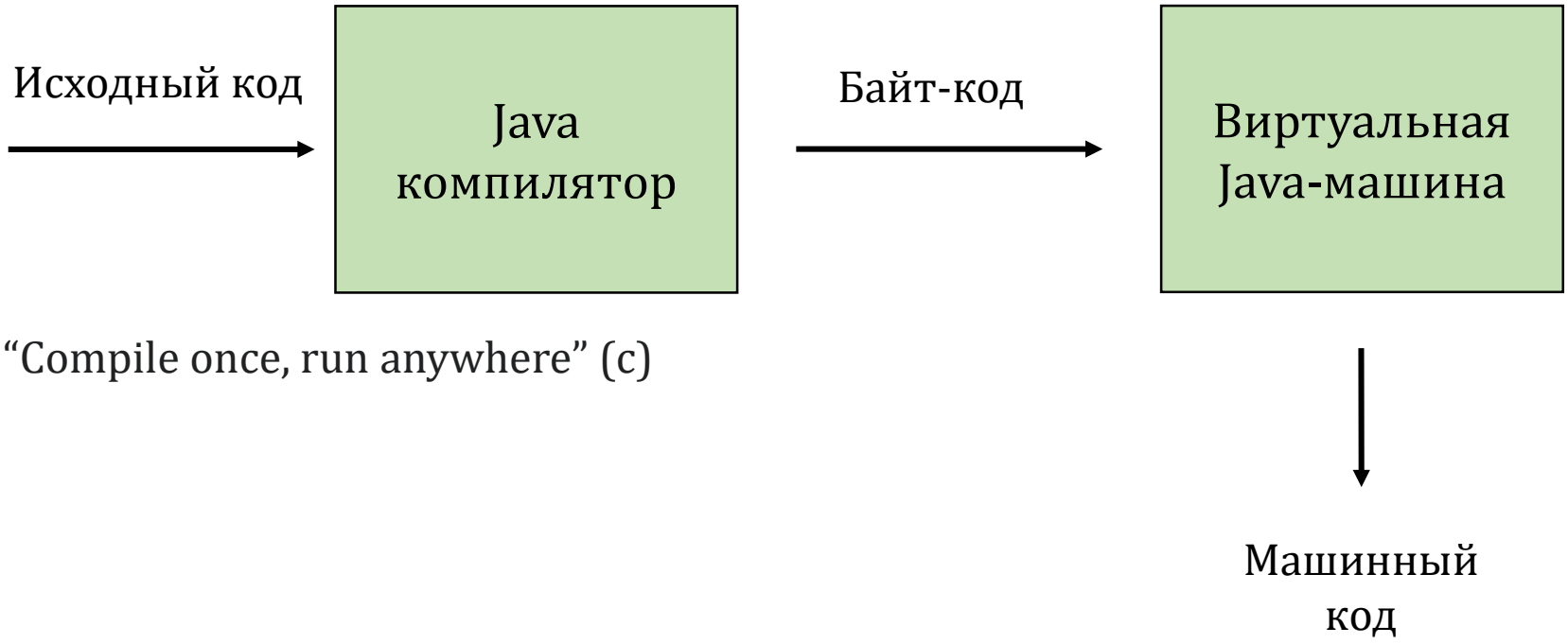
OpenMP

```
#pragma omp parallel for private(x) reduction(+:sum)

for (i=0;i<= num_steps; i++){
       x = (i+0.5)*step;
       sum = sum + 4.0/(1.0+x*x);
       }
```

Java

– строго типизированный объектно-ориентированный язык программирования общего назначения



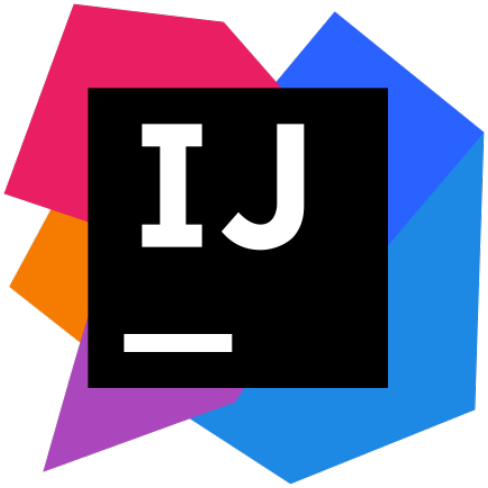
“Compile once, run anywhere” (c)

Java

– строго типизированный объектно-ориентированный язык программирования общего назначения.

JDK + IDE

IntelliJ IDEA



Eclipse

NetBeans

ООП vs СтруктурноеП

Объектно-ориентированная парадигма:

- Данные структурируются в виде объектов, каждый из которых имеет определенный тип, т.е. принадлежит к какому-то классу.
- Классы – результат формализации решаемой задачи, выделения главных и второстепенных ее аспектов (абстракция данных).
- Внутри объекта инкапсулируется вся логика работы с относящейся к нему информацией.
- Объекты в программе взаимодействуют друг с другом, обмениваются запросами и ответами.
- При этом объекты одного типа сходным образом отвечают на одни и те же запросы.
- Объекты могут организовываться в более сложные структуры (включать другие объекты или наследовать от других объектов).

Java

Особенности языка:

- Все делается через классы-объекты (ООП). Нет процедурного программирования.

Базовые принципы ООП:
Инкапсуляция — это свойство системы, позволяющее объединить данные и методы в классе, скрыв детали от пользователя.
Наследование — это свойство системы, позволяющее описать новый класс на основе уже существующего с частично или полностью заимствующейся функциональностью.
Полиморфизм —свойство системы использовать объекты с одинаковым интерфейсом без информации о типе и внутренней структуре объекта. “Один интерфейс, множество методов”.
Абстракция данных — это способ выделить набор значимых характеристик объекта, исключая из рассмотрения не значимые.

Java

Особенности языка:

- Все делается через классы-объекты (ООП). Нет процедурного программирования.

Базовые принципы ООП:
Инкапсуляция — это свойство системы, позволяющее объединить данные и методы в классе, скрыв детали от пользователя.
Наследование — это свойство системы, позволяющее описать новый класс на основе уже существующего с частично или полностью заимствующейся функциональностью.
Полиморфизм —свойство системы использовать объекты с одинаковым интерфейсом без информации о типе и внутренней структуре объекта. “Один интерфейс, множество методов”.
Абстракция данных — это способ выделить набор значимых характеристик объекта, исключая из рассмотрения не значимые.

Java

Особенности языка:

- Все делается через классы-объекты (ООП). Нет процедурного программирования.

Базовые принципы ООП:
Инкапсуляция — это свойство системы, позволяющее объединить данные и методы в классе, скрыв детали от пользователя.
Наследование — это свойство системы, позволяющее описать новый класс на основе уже существующего с частично или полностью заимствующейся функциональностью.
Полиморфизм —свойство системы использовать объекты с одинаковым интерфейсом без информации о типе и внутренней структуре объекта. “Один интерфейс, множество методов”.
Абстракция данных — это способ выделить набор значимых характеристик объекта, исключая из рассмотрения не значимые.

Java

Особенности языка:

- Все делается через классы-объекты (ООП). Нет процедурного программирования.
- Переменные объектного типа* являются ссылками (аналоги указателей на динамически создаваемые объекты).

C++
```
double a[10][20]; Foo b(30);
```

Java
```
double[][] a = new double[10][20]; Foo b = new Foo(30);
```

Присваивания, передача и сравнения работают с ссылками на объект, но прямого манипулирования памятью нет.

*Все, за исключением: boolean, byte, char, short, int, long, float, double

Java

Особенности языка:

- Все делается через классы-объекты (ООП). Нет процедурного программирования.
- Переменные объектного типа являются ссылками (аналоги указателей на динамически создаваемые объекты).
- Из-за того, что объектные переменные являются ссылочными, при присваивании не происходит копирования объекта.

```
Foo foo, bar;
…
bar = foo;
```

В результате адрес из foo скопируется в bar,
обе переменные указывают на один объект:
Изменили поле объекта в одном ⟹ в другом.
Для копии объектов используется метод clone().

Java

Особенности языка:

- Все делается через классы-объекты (ООП). Нет процедурного программирования.
- Переменные объектного типа являются ссылками (аналоги указателей на динамически создаваемые объекты).
- Из-за того, что объектные переменные являются ссылочными, при присваивании не происходит копирования объекта.

```
Foo foo, bar;
…
bar = foo;
```

В результате адрес из foo скопируется в bar,
обе переменные указывают на один объект:
Изменили поле объекта в одном ⟹ в другом.
Для копии объектов используется метод clone().

Java

Особенности языка:

- Все делается через классы-объекты (ООП). Нет процедурного программирования.
- Переменные объектного типа являются ссылками (аналоги указателей на динамически создаваемые объекты).
- Из-за того, что объектные переменные являются ссылочными, при присваивании не происходит копирования объекта.
- По той же причине не работает сравнение двух одинаковых по свойствам объектов.

```
Car car1 = new Car();                    Car car2 = new Car();
car1.model = "Ferrari";                  car2.model = "Ferrari";
car1.maxSpeed = 300;                     car2.maxSpeed = 300;
                    System.out.println(car1 == car2);
```

Сравнивается адрес: надо использовать метод equals()*.

Java

Особенности языка:

- Все делается через классы-объекты (ООП). Нет процедурного программирования.
- Переменные объектного типа являются ссылками (аналоги указателей на динамически создаваемые объекты).
- Из-за того, что объектные переменные являются ссылочными, при присваивании не происходит копирования объекта.
- По той же причине не работает сравнение двух одинаковых по свойствам объектов.
- Сравнение строк работает иначе, чем остальных переменных.

Java

Особенности языка:

- Сравнения строк.

```
String s1 = “ABC";
String s2 = new String(“ABC");
System.out.println(s1 == s2);
```

Java

Особенности языка:

- Сравнения строк.

```
String s1 = “ABC";
String s2 = new String(“ABC");
System.out.println(s1 == s2);
```

```
String s1 = “ABC";
String s2 = “ABC";
System.out.println(s1 == s2);
```

Java

Особенности языка:

- Сравнения строк.

```
String s1 = “ABC";
String s2 = new String(“ABC");
System.out.println(s1 == s2);
```

```
String s1 = “ABC";
String s2 = “ABC";
System.out.println(s1 == s2);
```

STRING POOL
“Hello World”
“142”
“ABC”

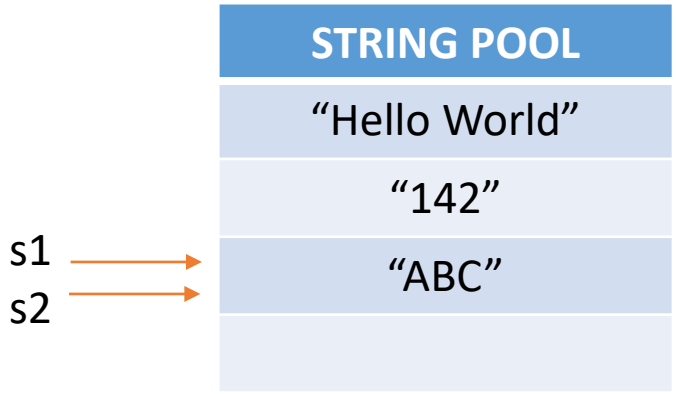
Java

Особенности языка:

- Сравнения строк.

```
String s1 = “ABC";
String s2 = new String(“ABC");
System.out.println(s1 == s2);
```

```
String s1 = “ABC";
String s2 = “ABC";
System.out.println(s1 == s2);
```



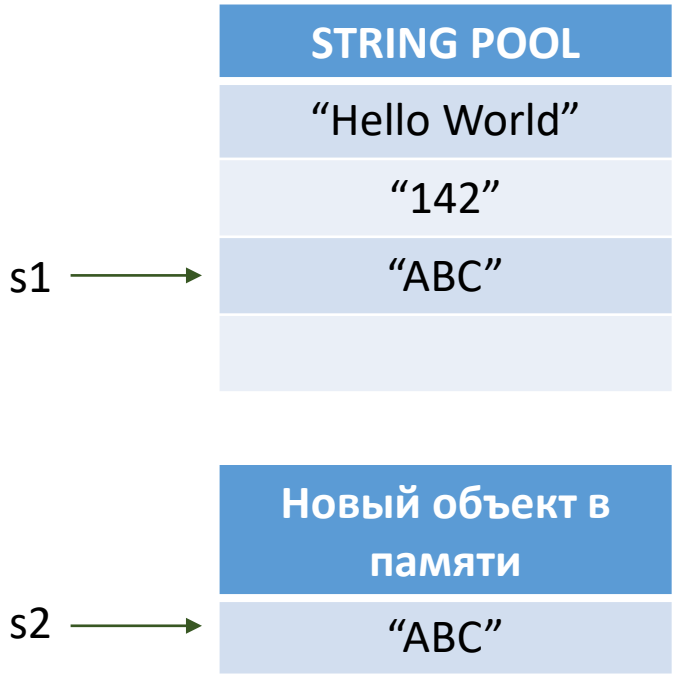
Java

Особенности языка:

- Сравнения строк (equals() переопределять не надо).

```
String s1 = “ABC";
String s2 = new String(“ABC");
System.out.println(s1 == s2);
```

```
String s1 = “ABC";
String s2 = “ABC";
System.out.println(s1 == s2);
```



Java

«Анти»-особенности языка:

- Стандартная работа с числами;
- Классический синтаксис условий и циклов:
  if, (do) while, for, switch;
- Логические операции и сравнения;
- Работа с консолью;
- Работа со списками.

Java

```java
class HelloWorld {
      public static void main(String[] args) {
      System.out.println("Hello World!");
      }
}
```