

# Динамическая диспетчеризация

## Объектно-Ориентированное Программирование

Иван Трепаков

NSU

*Полиморфизм — возможность функции с одним именем иметь разные реализации.*

## Ad hoc полиморфизм

*Ad hoc* (букв. «к этому») — латинская фраза, означающая «для данного случая», «специально для этого».

[Wikipedia](#)

## Ad hoc полиморфизм

*Ad hoc* (букв. «к этому») — латинская фраза, означающая «для данного случая», «специально для этого».

[Wikipedia](#)

- Выбор реализации делается в зависимости от количества и типов формальных параметров функции
  - Перегрузка функций в Java
  - Перегрузка операторов в C++

## Ad hoc полиморфизм

*Ad hoc* (букв. «к этому») — латинская фраза, означающая «для данного случая», «специально для этого».

[Wikipedia](#)

- Выбор реализации делается в зависимости от количества и типов формальных параметров функции
  - Перегрузка функций в Java
  - Перегрузка операторов в C++

```
var x = ...;  
var y = ...;  
var z = x + y; // ???
```

## Ad hoc полиморфизм

*Ad hoc* (букв. «к этому») — латинская фраза, означающая «для данного случая», «специально для этого».

[Wikipedia](#)

- Выбор реализации делается в зависимости от количества и типов формальных параметров функции
  - Перегрузка функций в Java
  - Перегрузка операторов в C++

```
var x = 1;      // int
var y = ...;
var z = x + y;  // ???
```

## Ad hoc полиморфизм

*Ad hoc* (букв. «к этому») — латинская фраза, означающая «для данного случая», «специально для этого».

[Wikipedia](#)

- Выбор реализации делается в зависимости от количества и типов формальных параметров функции
  - Перегрузка функций в Java
  - Перегрузка операторов в C++

```
var x = 1;      // int
var y = 2;      // int
var z = x + y;  // ???
```

## Ad hoc полиморфизм

*Ad hoc* (букв. «к этому») — латинская фраза, означающая «для данного случая», «специально для этого».

[Wikipedia](#)

- Выбор реализации делается в зависимости от количества и типов формальных параметров функции
  - Перегрузка функций в Java
  - Перегрузка операторов в C++

```
var x = 1;      // int
var y = 2;      // int
var z = x + y;  // 3 (int)
```



## Ad hoc полиморфизм

*Ad hoc* (букв. «к этому») — латинская фраза, означающая «для данного случая», «специально для этого».

[Wikipedia](#)

- Выбор реализации делается в зависимости от количества и типов формальных параметров функции
  - Перегрузка функций в Java
  - Перегрузка операторов в C++

```
var x = 1;      // int
var y = 2.0;    // double
var z = x + y;  // ???
```

## Ad hoc полиморфизм

*Ad hoc* (букв. «к этому») — латинская фраза, означающая «для данного случая», «специально для этого».

[Wikipedia](#)

- Выбор реализации делается в зависимости от количества и типов формальных параметров функции
  - Перегрузка функций в Java
  - Перегрузка операторов в C++

```
var x = 1;      // int
var y = 2.0;    // double
var z = x + y;  // 3.0 (double)
```

## Ad hoc полиморфизм

*Ad hoc* (букв. «к этому») — латинская фраза, означающая «для данного случая», «специально для этого».

[Wikipedia](#)

- Выбор реализации делается в зависимости от количества и типов формальных параметров функции
  - Перегрузка функций в Java
  - Перегрузка операторов в C++

## Ad hoc полиморфизм

*Ad hoc* (букв. «к этому») — латинская фраза, означающая «для данного случая», «специально для этого».

[Wikipedia](#)

- Выбор реализации делается в зависимости от количества и типов формальных параметров функции
  - Перегрузка функций в Java
  - Перегрузка операторов в C++

```
String add(int x, int y) {  
    return "ints: " + (x + y);  
}  
String add(String x, int y) {  
    return "mixed: " + (x + y);  
}  
String add(String x, String y) {  
    return "strings: " + (x + y);  
}
```

## Ad hoc полиморфизм

*Ad hoc* (букв. «к этому») — латинская фраза, означающая «для данного случая», «специально для этого».

[Wikipedia](#)

- Выбор реализации делается в зависимости от количества и типов формальных параметров функции
  - Перегрузка функций в Java
  - Перегрузка операторов в C++

```
String add(int x, int y) {  
    return "ints: " + (x + y);  
}  
String add(String x, int y) {  
    return "mixed: " + (x + y);  
}  
String add(String x, String y) {  
    return "strings: " + (x + y);  
}  
add(1, 2)
```

## Ad hoc полиморфизм

*Ad hoc* (букв. «к этому») — латинская фраза, означающая «для данного случая», «специально для этого».

[Wikipedia](#)

- Выбор реализации делается в зависимости от количества и типов формальных параметров функции
  - Перегрузка функций в Java
  - Перегрузка операторов в C++

```
String add(int x, int y) {  
    return "ints: " + (x + y);  
}  
String add(String x, int y) {  
    return "mixed: " + (x + y);  
}  
String add(String x, String y) {  
    return "strings: " + (x + y);  
}  
add(1, 2) // ints: 3
```

## Ad hoc полиморфизм

*Ad hoc* (букв. «к этому») — латинская фраза, означающая «для данного случая», «специально для этого».

[Wikipedia](#)

- Выбор реализации делается в зависимости от количества и типов формальных параметров функции
  - Перегрузка функций в Java
  - Перегрузка операторов в C++

```
String add(int x, int y) {  
    return "ints: " + (x + y);  
}  
String add(String x, int y) {  
    return "mixed: " + (x + y);  
}  
String add(String x, String y) {  
    return "strings: " + (x + y);  
}  
add(1, 2) // ints: 3  
add("1", 2)
```

## Ad hoc полиморфизм

*Ad hoc* (букв. «к этому») — латинская фраза, означающая «для данного случая», «специально для этого».

[Wikipedia](#)

- Выбор реализации делается в зависимости от количества и типов формальных параметров функции
  - Перегрузка функций в Java
  - Перегрузка операторов в C++

```
String add(int x, int y) {  
    return "ints: " + (x + y);  
}  
String add(String x, int y) {  
    return "mixed: " + (x + y);  
}  
String add(String x, String y) {  
    return "strings: " + (x + y);  
}  
add(1, 2) // ints: 3  
add("1", 2) // mixed: 12
```



## Ad hoc полиморфизм

*Ad hoc* (букв. «к этому») — латинская фраза, означающая «для данного случая», «специально для этого».

[Wikipedia](#)

- Выбор реализации делается в зависимости от количества и типов формальных параметров функции
  - Перегрузка функций в Java
  - Перегрузка операторов в C++

```
String add(int x, int y) {  
    return "ints: " + (x + y);  
}  
String add(String x, int y) {  
    return "mixed: " + (x + y);  
}  
String add(String x, String y) {  
    return "strings: " + (x + y);  
}  
add(1, 2) // ints: 3  
add("1", 2) // mixed: 12  
add("1", "2")
```

## Ad hoc полиморфизм

*Ad hoc* (букв. «к этому») — латинская фраза, означающая «для данного случая», «специально для этого».

[Wikipedia](#)

- Выбор реализации делается в зависимости от количества и типов формальных параметров функции
  - Перегрузка функций в Java
  - Перегрузка операторов в C++

```
String add(int x, int y) {  
    return "ints: " + (x + y);  
}  
String add(String x, int y) {  
    return "mixed: " + (x + y);  
}  
String add(String x, String y) {  
    return "strings: " + (x + y);  
}  
add(1, 2) // ints: 3  
add("1", 2) // mixed: 12  
add("1", "2") // strings: 12
```

## Параметрический полиморфизм

- Реализация функции использует *обобщенный* параметр
  - Generics в Java
  - Templates в C++
  - Type classes в Haskell
- Можно задавать дополнительные ограничения на обобщенный тип
- *Подробнее на следующей лекции*

```
static <T, S> T foo(T x, S y) {  
    return x;  
}
```

```
static <T extends I> boolean bar(I x, I y) {  
    return x.test(y);  
}
```

```
interface I {  
    boolean test(I x);  
}
```

## Полиморфизм подтипов

Статическая

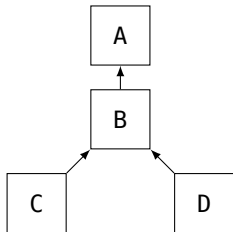
Динамическая

Одиночная

Множественная

# Наследование

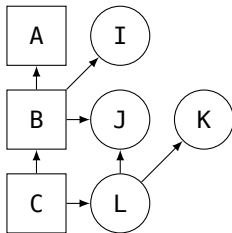
Одиночное



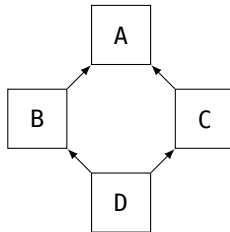
simula



Интерфейсное

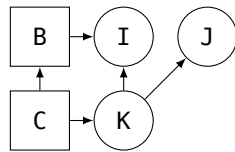
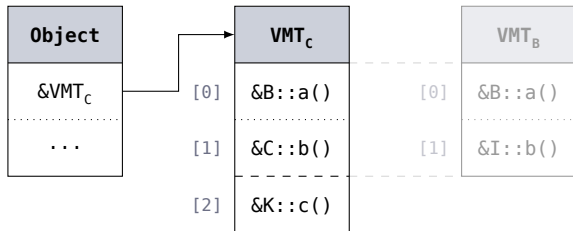


Множественное



 **Eiffel**

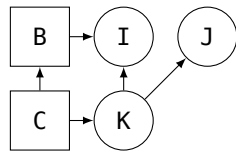
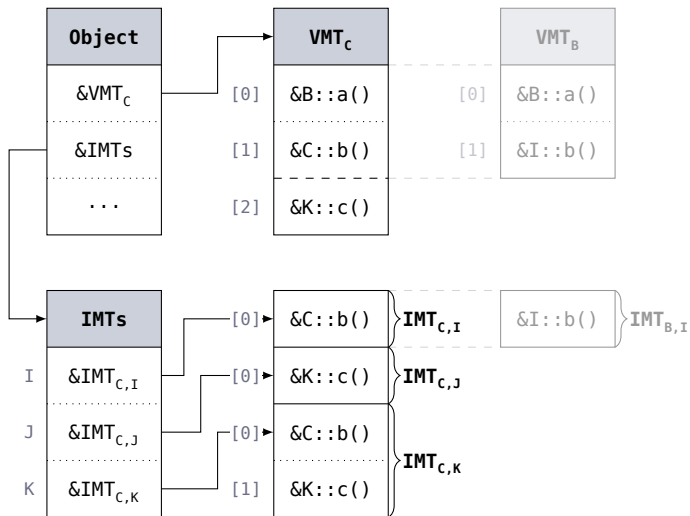
# Диспетчеризация виртуальных методов



Виртуальный вызов  $x.b()$

```
// Формальный тип x: C  
call x.vmt[vnumC,b]
```

# Диспетчеризация виртуальных методов



Виртуальный вызов  $x.b()$

```
// Формальный тип  $x$ : C  
call  $x.vmt[vnum_{C,b}]$ 
```

Интерфейсный вызов  $x.b()$

```
// Формальный тип  $x$ : I  
 $imt_{C,I} := x.imts.find(\&I)$   
call  $imt_{C,I}[vnum_{I,b}]$ 
```



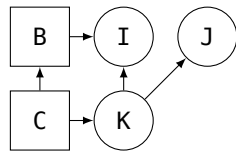
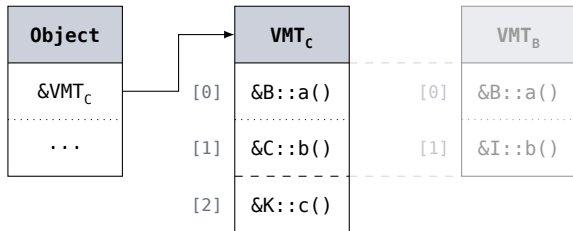
Таблица интерфейсных методов

Полиморфный инлайн кэш



Q&A

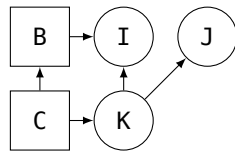
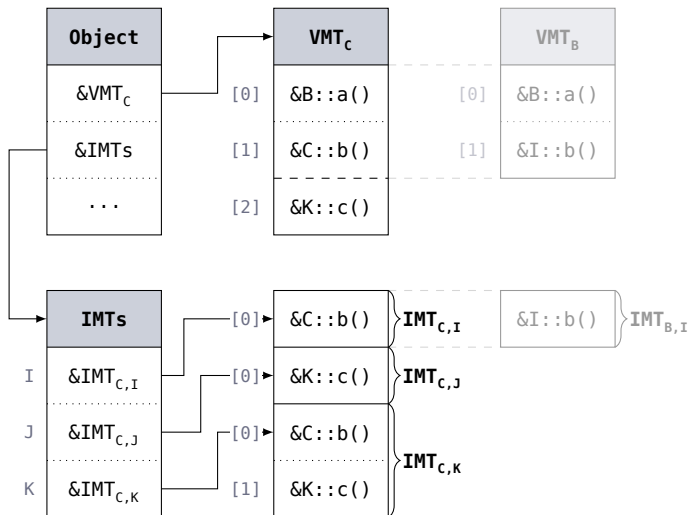
# Таблица виртуальных методов



Виртуальный вызов `x.b()`

```
// Формальный тип x: C  
call x.vmt[vnumC,b]
```

# Таблица виртуальных методов



Виртуальный вызов  $x.b()$

```
// Формальный тип x: C
call x.vmt[vnumC,b]
```

Интерфейсный вызов  $x.b()$

```
// Формальный тип x: I
imtC,I := x.imts.find(&I)
call imtC,I[vnumI,b]
```