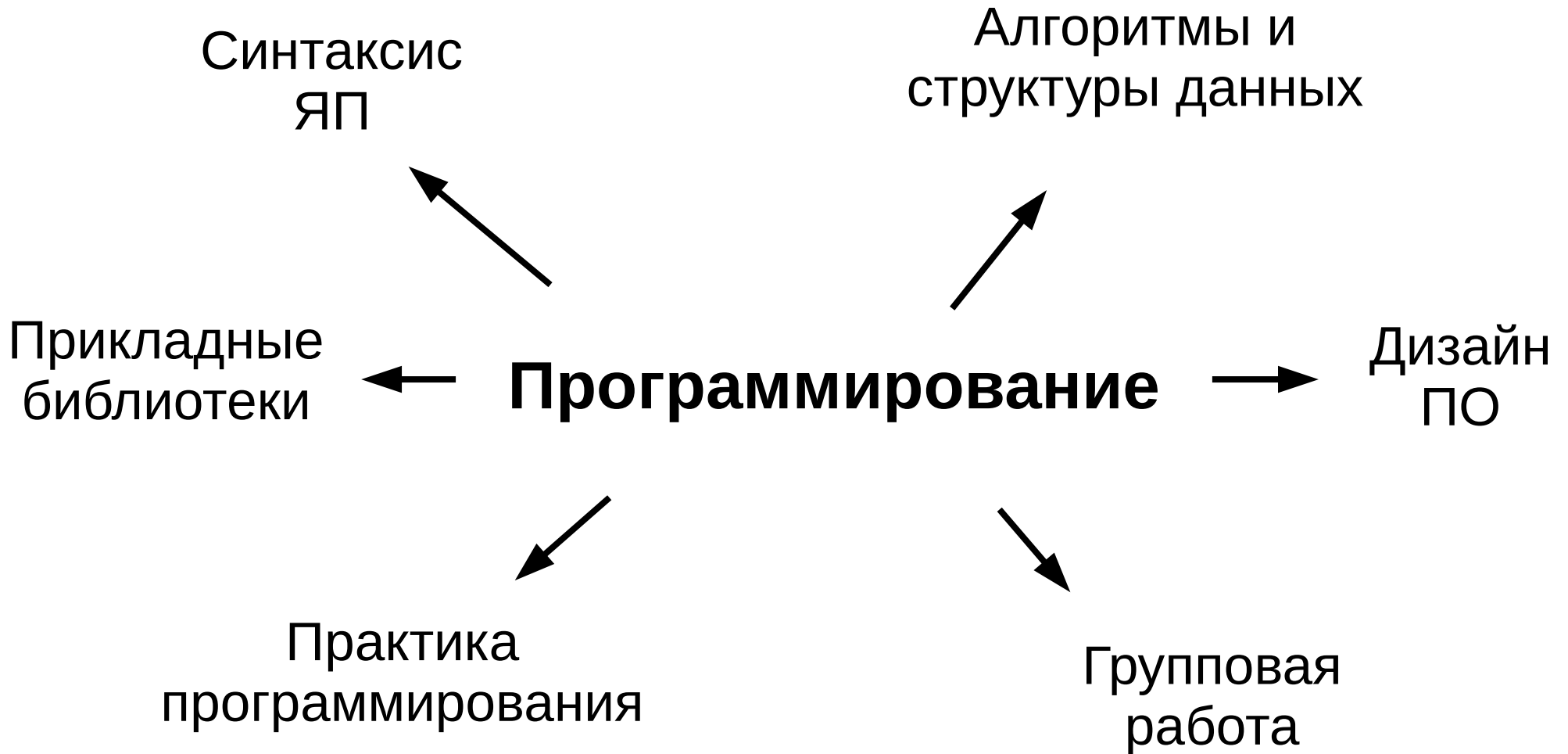


Лекция 1

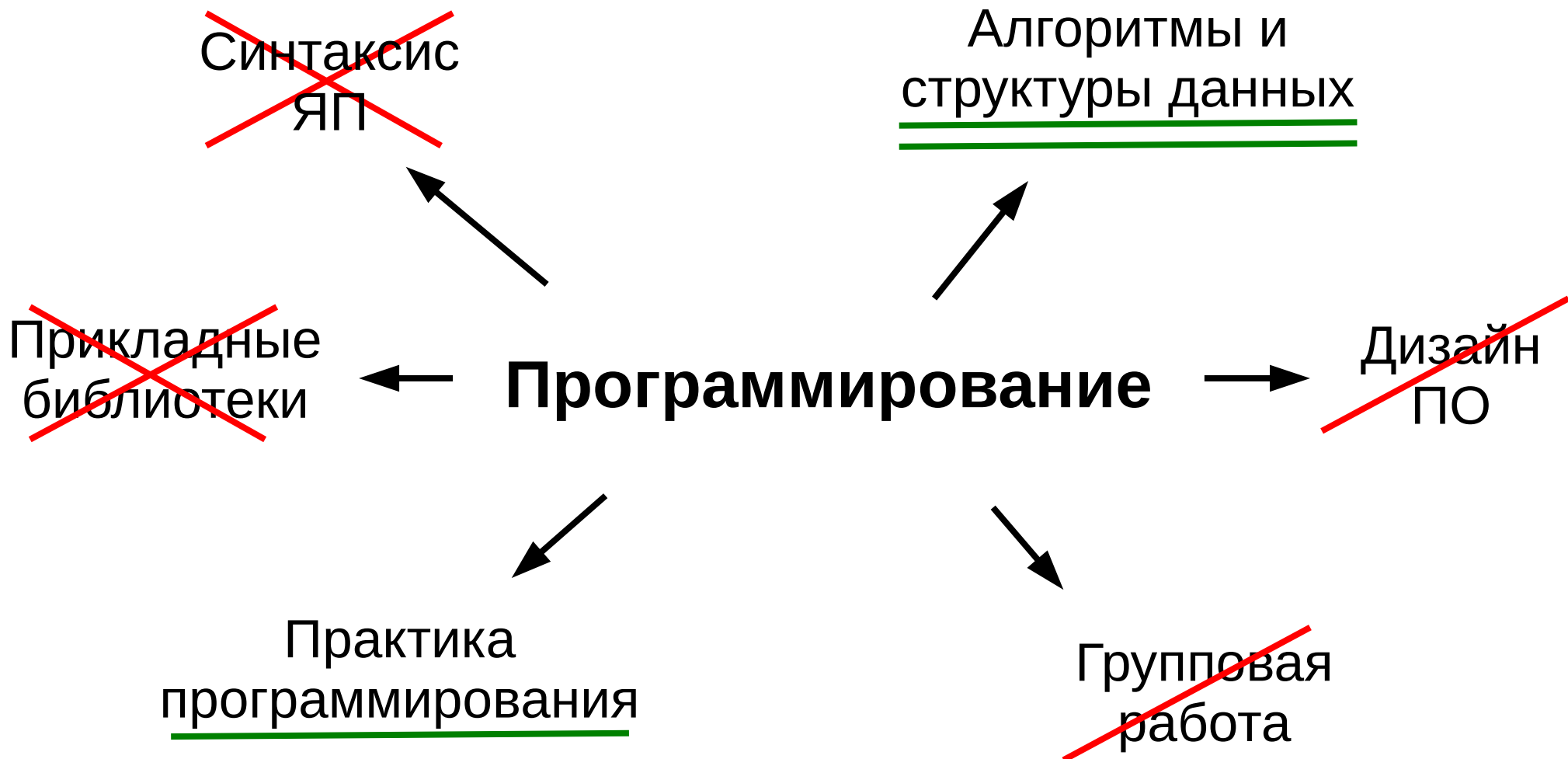
Введение в программирование:
понятие алгоритма,
язык Java,
переменные и типы данных,
ввод/вывод данных,
класс Math

Программирование: общие принципы

Программирование ...



Акценты в нашем курсе



Запомнить и осознать

- Вы учитесь программировать, а не какому-то конкретному языку программирования!
 - Вы будете использовать Java, но если вы освоите программирование, то для вас не будет принципиальной разницы на каком языке писать алгоритмы
 -
- В программировании (как впрочем и везде, но здесь особенно) невозможно выучить материал, либо разобраться, понять и научиться, либо все будет плохо
 - Практика первична
 - Это не отменяет теории, т.к. до многих приемов додуматься самостоятельно весьма непросто

Запомнить и осознать

- Не надо пугаться, программирование — достаточно просто (по сравнению с различными разделами математики и физики), но здесь важно до последней детали понимать, что вы делаете
 - Если что-то непонятно — обязательно разобраться (спросить у преподавателя, у своих товарищей, прочить в умных книжках или интернете)

–

Учебные материалы и задачи

- «L:\Лекции\1 курс\Введение в программирование»
- Практические задачи делать как можно быстрее
 - Порядка 15 различных задач
 - Первые практические задачи, возможно, вам покажутся простыми, но последующие будут все сложнее, поэтому не стоит расслабляться
 - Строго следовать инструкциям, которые даны к практическим задачам (если даны)

Запомнить и осознать

- Если преподаватель не принимает задачу и просит вас ее каким-либо образом переделать — он скорее всего прав и ваше решение «кривое» по тем или иным причинам, не стоит закреплять неправильный шаблон
 - Аргумент «Ну она же работает» - не аргумент!
 - Для простых задач состояния «Программа почти работает» не бывает, либо всегда работает правильно, либо решение неверное

Понятие алгоритма

- Одно из определений:

Алгоритм - это точное и понятное предписание исполнителю совершить последовательность действий над заданными объектами, приводящее исполнителя после конечного числа шагов к достижению указанной цели или решению поставленной задачи

Свойства алгоритма

- **дискретность**: состоит из отдельных шагов (команд)
- **результативность**: применение алгоритма обязательно приводит к конечному результату за конечное число шагов
- **массовость**: может применяться многократно при различных исходных данных
- **детерминированность**: выполнение команд в строго определенной последовательности
- **понятность**: должен включать только команды, известные исполнителю (входящие в список команд исполнителя)
- **определенность**: при одинаковых исходных данных всегда выдает один и тот же результат
- **корректность**: дает верное решение при любых допустимых исходных данных

Способы описания алгоритмов

- Словесно-формульное описание (на естественном языке с использованием математических формул)
- Графическое описание в виде блок схем
- Описание на каком-то языке программирования (программа)

Пример алгоритма (программа Софы)

1. Смешать уцтово талвот лыоватол тыволта лобытлваот лобывталот лывоатлот лывоатло тлот:
 - 1/3 ст. подсолнечного масла
 - 1/4 ст. воды
 - 1 ст. сахара
2. Добавить 2-3 размятых банана
3. По желанию добавить 1-2 взбитых яйца
4. Отдельно смешать сухими:
 - 2 ст. муки
 - 1 ч.л. соды
 - 1/2 ч.л. соли
 - 1/2 ч.л. корицы
 - ванилин
5. Все смешать и выпекать, пока не покоричневает

Алгоритмизация

- Алгоритмизация — процесс составления алгоритмов

Программирование (в узком смысле)

- Программирование — запись алгоритмов на языке, понятном компьютеру

Цель изучения данной курса

- Умение составлять алгоритмы (более важная часть)
- Умение записывать алгоритмы в виде программы на языке программирования
- Очень часто эти два умения не разделяют в рамках определения «уметь программировать», но разница есть

Запомнить и осознать

- Начинать что-то кодировать можно только тогда, когда вы знаете / придумали, как решать задачу (или какую-то часть большой задачи), т.е. когда вы составили алгоритм
 - Очень часто задача реально решается на листочке и уже найденное решение записывается в виде программы (это не означает, что код вы должны писать на листочке, но многие преподаватели вполне обоснованно считают, что в этом что-то есть)
- Другими словами:
Сначала думать, потом делать!

Язык Java и средства разработки

Язык Java



Java — сильно типизированный объектно-ориентированный язык программирования, разработанный компанией Sun Microsystems (в последующем приобретённой компанией Oracle). Приложения Java обычно транслируются в специальный байт-код, поэтому они могут работать на любой компьютерной архитектуре, с помощью виртуальной Java-машины. Дата официального выпуска – 23 мая 1995 года.

Компилятор командной строки Java входит в JDK (Java Development Kit), однако интегрированные среды разработки (IDE) для Java распространяются отдельно.

Некоторые особенности Java

- Язык общего назначения, разрабатывался прежде всего для решения прикладных задач (конкурент от Microsoft – C#)
 - Достаточно простой (проще чем C++)
 - «Сборка мусора»
- Строго-типизированный
- Объектно-ориентированный
- Для выполнения программ на устройстве необходимо, если не установлена, установить JRE (Java Runtime Environment), включающий виртуальную машину Java (JVM – Java Virtual Machine)
- Для компиляции программ необходим JDK (Java Development Kit), включающий компилятор командной строки Java
 - Конечные программы компилируются в байт-код Java и исполняются виртуальной машиной Java (JVM)
- Доступна обширная библиотека классов, которая поставляется с JRE
 - JDK включает JRE

Почему Java

- Достаточно простой строго-типизированный язык
- Наиболее востребованный в настоящий момент в индустрии язык
 - для поиска работы знать язык Java и уметь программировать недостаточно, необходимо владеть соответствующим инструментарием и уметь пользоваться широким набором общепризнанных сторонних библиотек
- Очень широкое сообщество разработчиков
 - куча документации, книг, форумов и т.п.
 - множество сторонних библиотек на все случаи жизни
 - и т.д.
- Возможные альтернативы:
 - C / C++ – классический подход
 - Pascal – устаревший подход (Pascal – умирающий язык)
 - Python – современный подход
 - C# – почти то же самое, что Java, но от MicroSoft

JDK, версии

- Последняя (на 06.09.2020) версия – JDK 14
 - OpenJDK:
<https://jdk.java.net/14/>
 - Oracle JDK:
<https://www.oracle.com/java/technologies/javase-downloads.html>
- Все примеры будут доступны для JDK 8
 - <https://www.oracle.com/java/technologies/javase-downloads.html>
- Последняя LTS (Long Time Support) версия – JDK 11
 - <https://www.oracle.com/java/technologies/javase-downloads.html>
 - <https://jdk.java.net/archive/>

Средства разработки



- **NetBeans**
 - От Oracle (недавно передано свободному сообществу под эгидой Apache)
 - Вероятно, лучший выбор для начинающих разработчиков (все наиболее просто и понятно, руссифицированный интерфейс у версии 8.2)



- **IntelliJ IDEA**
 - Многие считают лучшей IDE для профессионалов
 - Платная, но есть ограниченная Community Edition и можно получить студенческую лицензию
 - Наиболее прожорливая



- **Eclipse**
 - Самая первая профессиональная IDE (первые версии разрабатывались в IBM, давно передано свободному сообществу)
 - Несколько «марсианская», но если освоиться – вполне удобная

Средства разработки

- В настоящий момент любая из этих сред (NetBeans, IntelliJ IDEA, Eclipse) вам подойдет
- Поэтому выбирайте любую (лучше попробовать все и действительно выбрать)
- В лекциях я буду использовать
IntelliJ IDEA (Community Edition):
<https://www.jetbrains.com/ru-ru/idea/>
 - Примеры проектов будут в IntelliJ IDEA, однако никто не мешает скопировать код в проект в другой среде
-
- Идеальный вариант, чтобы вы могли работать в любой среде (мы же пытаемся из вас сделать профессионалов :)



Средства разработки Online

- Для серьезных проектов не годятся, но для ваших первых проектов / поэкспериментировать – вполне подойдут
 - На первых занятиях, если в компьютерных классах не окажется IDE
- Ищутся в Google, например, по запросу «java web ide», например:
 - <https://www.jdoodle.com/online-java-compiler>
 - http://www.compileonline.com/compile_java8_online.php

https://www.jdoodle.com/online-java-compiler

The screenshot shows the JDoodle online Java compiler interface. The browser's address bar displays the URL `https://www.jdoodle.com/`. The page title is "Java". The main area contains a code editor with the following Java code:

```
1 import java.util.Scanner;
2
3
4 public class MyClass {
5     public static int add(int a, int b) {
6         return a + b;
7     }
8
9     public static void main(String args[]) {
10         Scanner scanner = new Scanner(System.in);
11         System.out.print("Input x: ");
12         int x = scanner.nextInt();
13         System.out.print("Input y: ");
14         int y = scanner.nextInt();
15         int z = add(x, y);
16
17         System.out.println("Sum of x + y: " + z);
18     }
19 }
20
```

Below the code editor, there are several controls:

- CommandLine Arguments ...**: A text input field.
- Interactive mode**: A toggle switch set to **ON**.
- Focus View**: A toggle switch set to **OFF**.
- Version**: A dropdown menu showing **JDK 10.0.1**.
- External Libraries ...**: A button labeled **Add External Library (from Maven Repo)**.

At the bottom, there are buttons for **Execute**, **Save**, **My Projects**, **Recent**, **Collaborate**, **FAQ**, and **More Options**.

Result...

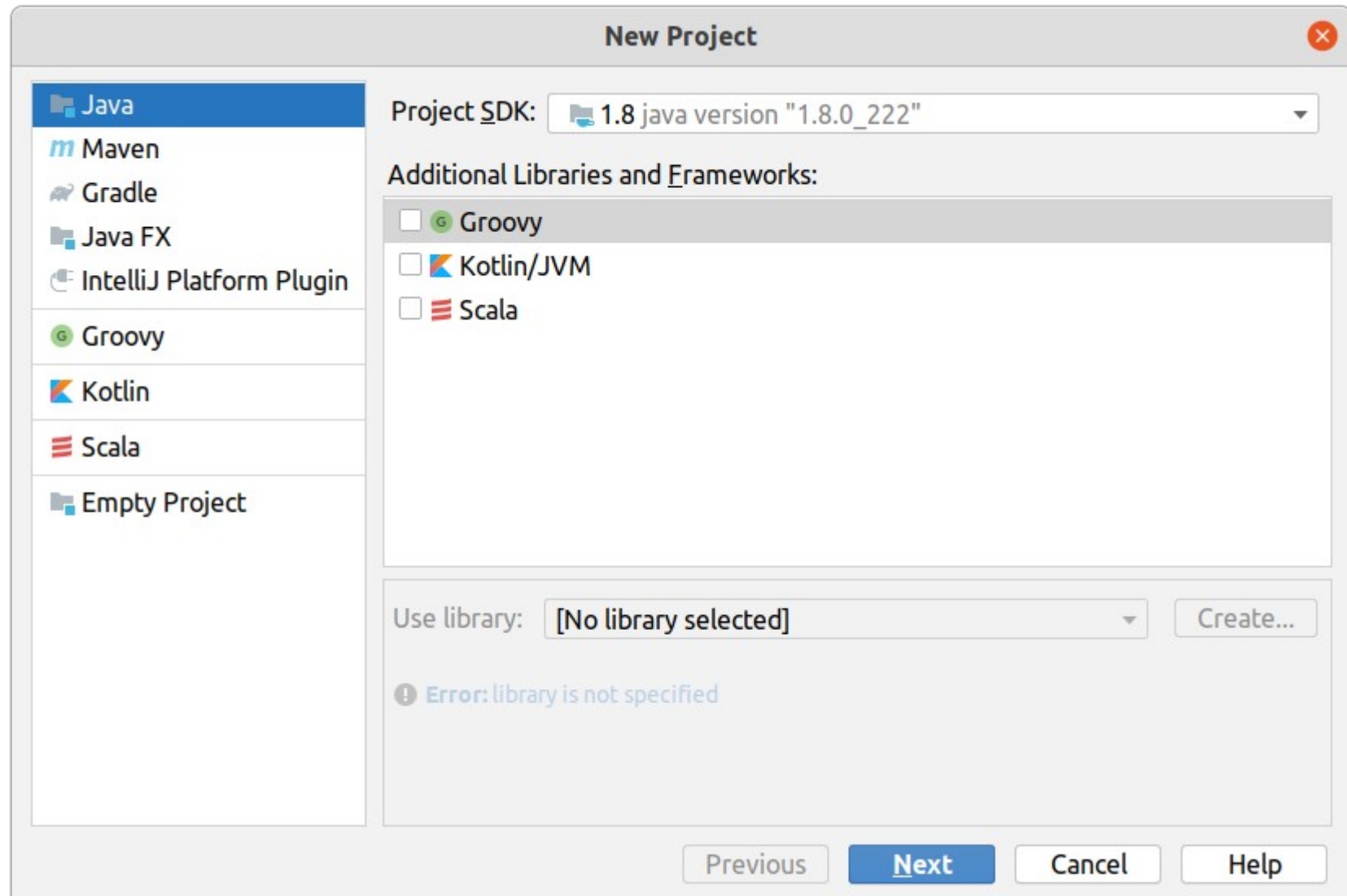
compiled and executed in 3.897 sec(s)

```
Input x: 6
Input y: 4
Sum of x + y = 10
```


Первая программа
в среде IntelliJ IDEA
(версия 20.2.1, Community Edition)

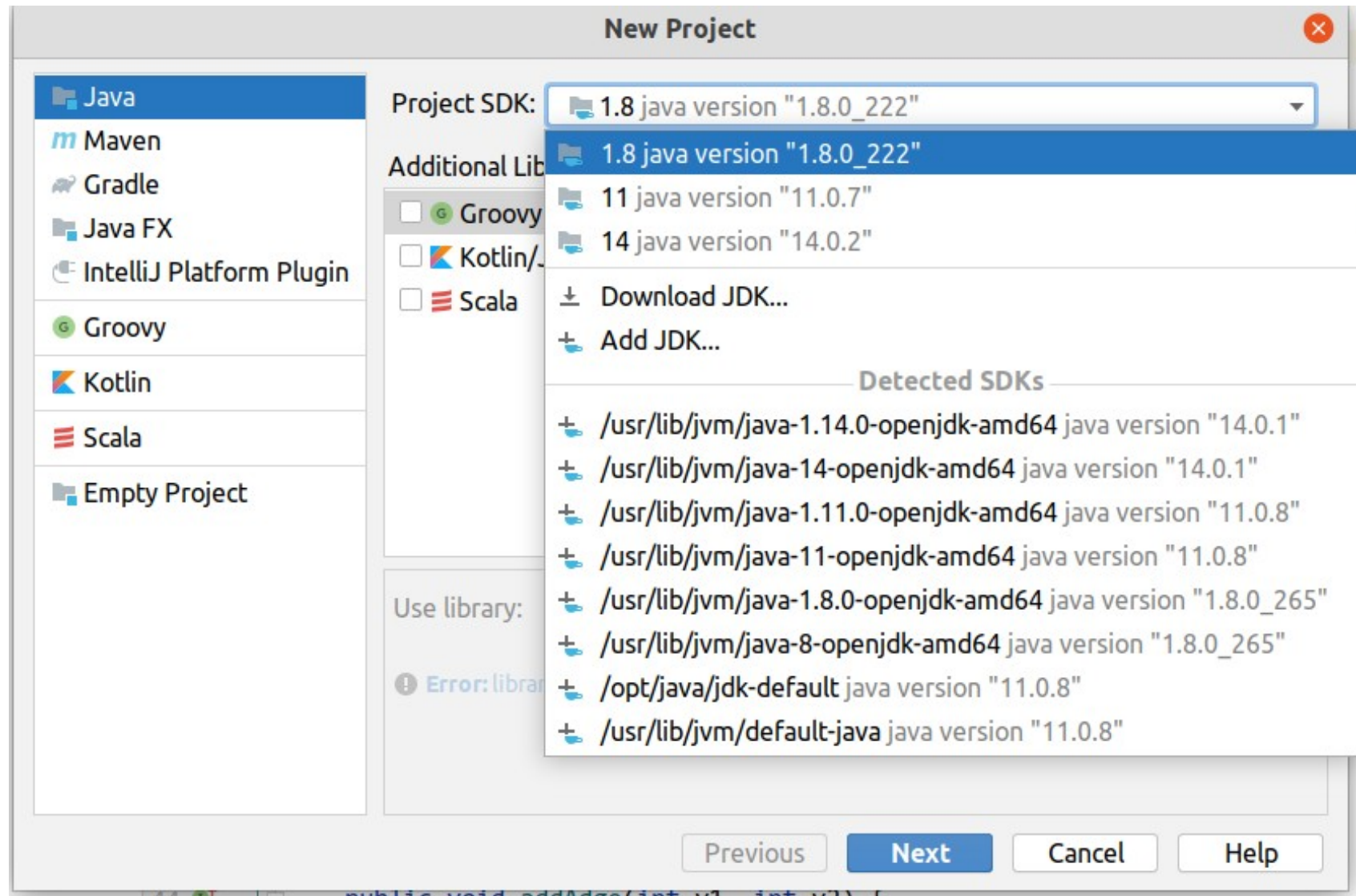
Создание проекта в IntelliJ IDEA

- File → New → Project



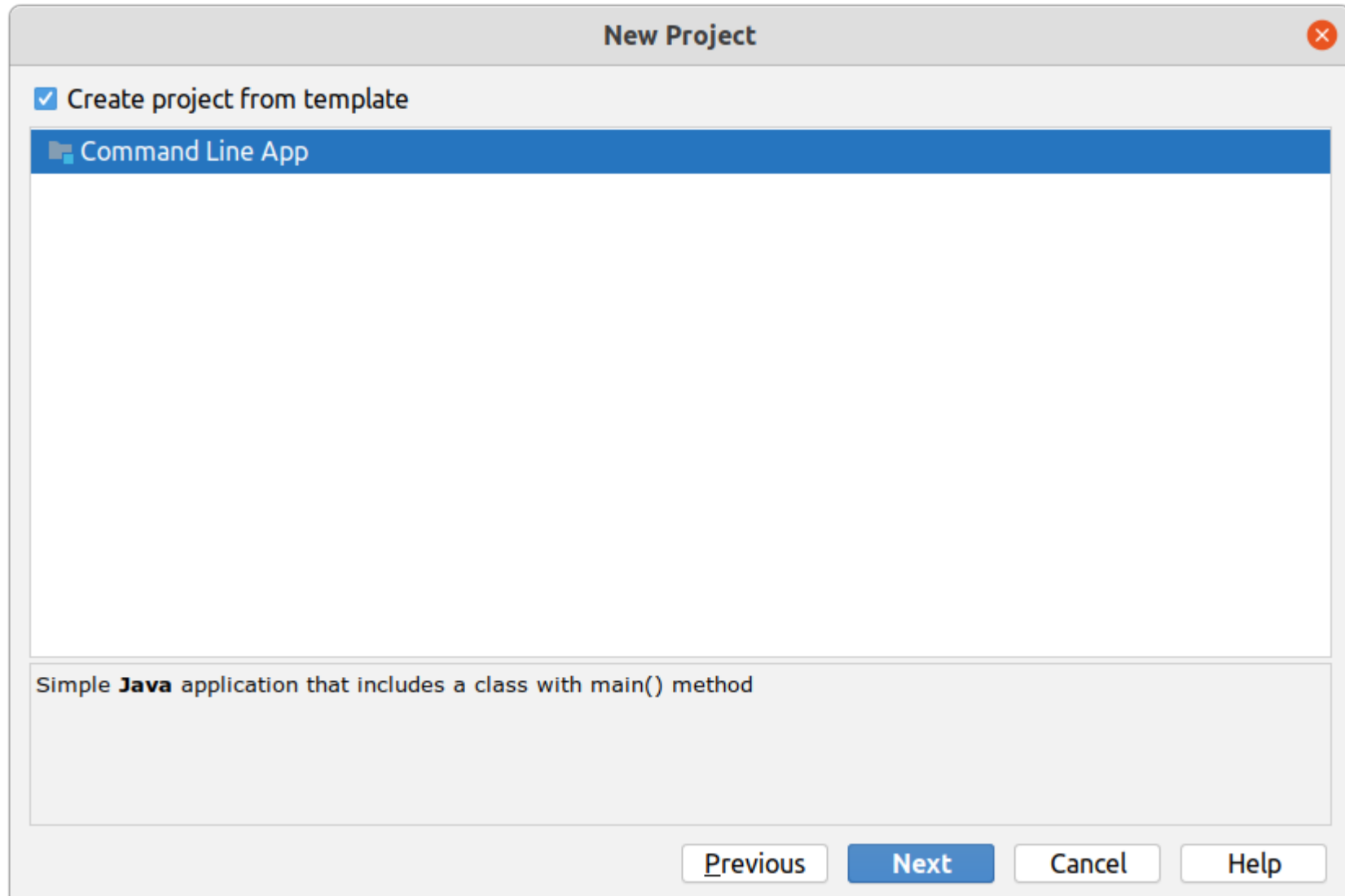
Создание проекта в IntelliJ IDEA

- Выбор / добавление JDK (при необходимости)



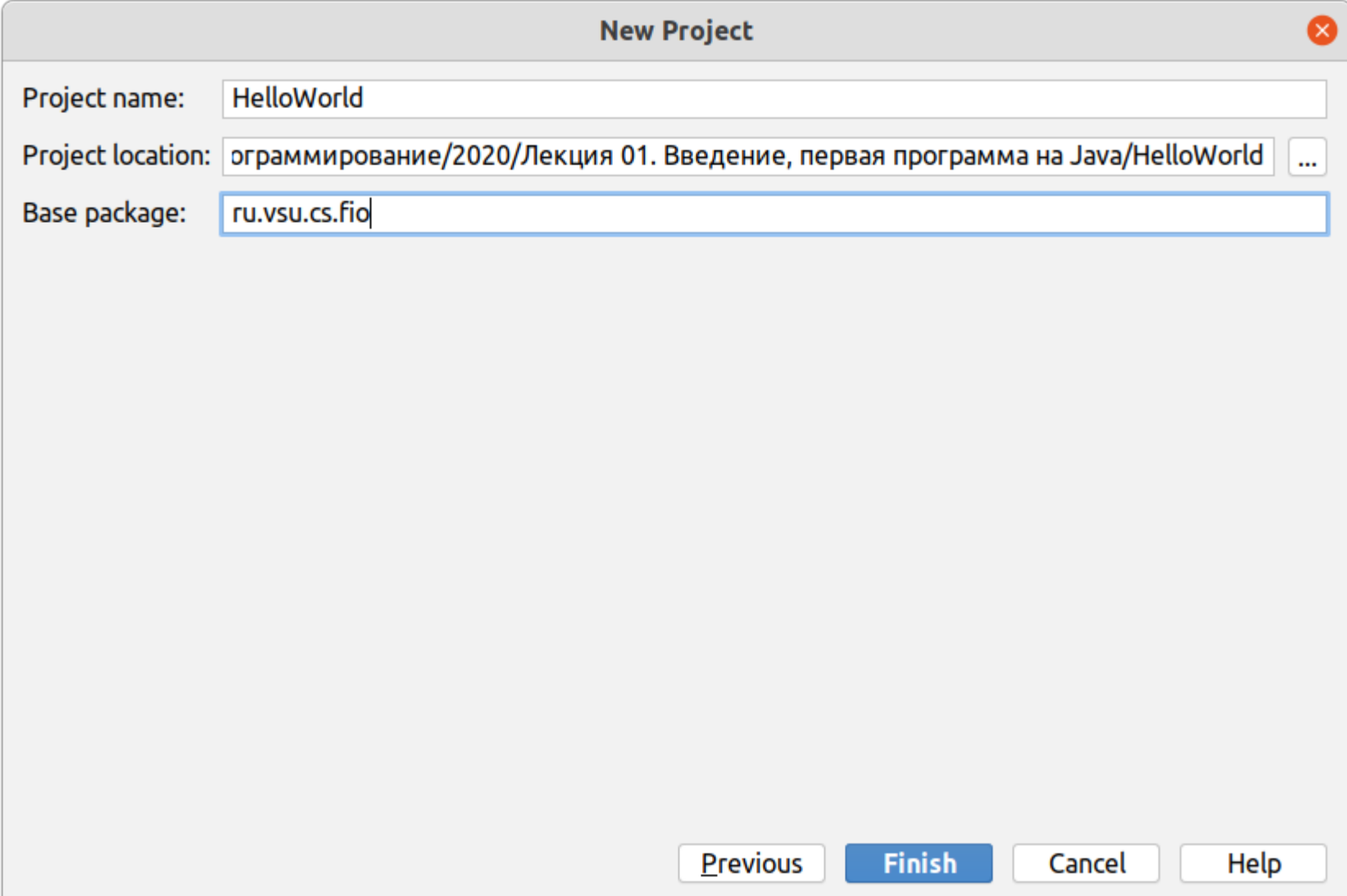
Создание проекта в IntelliJ IDEA

- Выбор шаблона проекта



Создание проекта в IntelliJ IDEA

- Параметры проекта



The image shows the 'New Project' dialog box in IntelliJ IDEA. The dialog has a title bar with 'New Project' and a close button. It contains three input fields: 'Project name' with the value 'HelloWorld', 'Project location' with the value 'ограммирование/2020/Лекция 01. Введение, первая программа на Java/HelloWorld' and a browse button (...), and 'Base package' with the value 'ru.vsu.cs.fio'. At the bottom, there are four buttons: 'Previous', 'Finish' (highlighted in blue), 'Cancel', and 'Help'.

New Project

Project name: HelloWorld

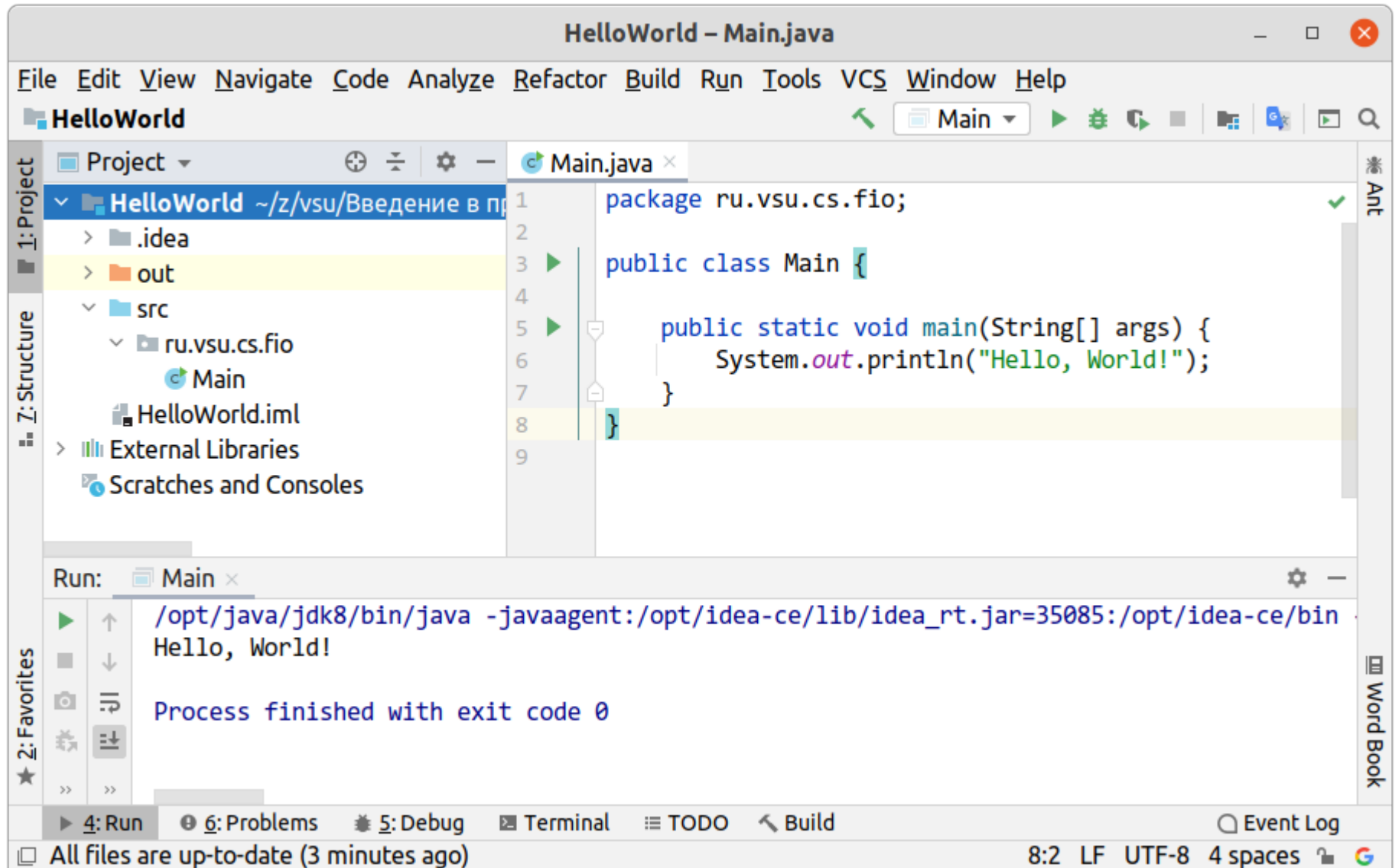
Project location: ограммирование/2020/Лекция 01. Введение, первая программа на Java/HelloWorld ...

Base package: ru.vsu.cs.fio

Previous Finish Cancel Help

Создание проекта в IntelliJ IDEA

- Результат (дописали println и запустили)



Или более простой вариант

- Скопировать с диска L: проект HelloWorld или Lect1Samples, открыть и править
- Каждая задача (проект) – отдельная директория (важно, иначе запутаетесь)

«Hello, World!» на Java

```
package ru.vsu.cs.fio;

import java.util.Scanner;    // в данном проекте
                             // не используется

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


«Hello, World!» на Java

```
package first;
```

```
import java.util.Scanner;    // в данном проекте  
                             // не используется
```

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

Объявление пространства имен проекта (не обязательно)

Импорт необходимых пакетов

Объявление класса (обязательно, пока просто запомнить)

Точка запуска программы (точка входа, запомнить)

Непосредственно код программы (полезные действия)

Запуск программы из IntelliJ IDEA



- Без отладки – Shift + F10
- С отладкой – Shift + F9

Вычисление площади круга

```
package ru.vsu.cs.course1;

import java.util.Locale;
import java.util.Scanner;

public class Program {
    public static void main(String[] args) {
        Locale.setDefault(Locale.ROOT);

        Scanner scanner = new Scanner(System.in);

        System.out.print("Введите радиус круга: R = ");
        double r = scanner.nextDouble();

        double s = Math.PI * r * r;

        System.out.printf(
            "Для круга радиуса R = %1$.3f площадь s = %2$.3f%n",
            r, s
        );
    }
}
```

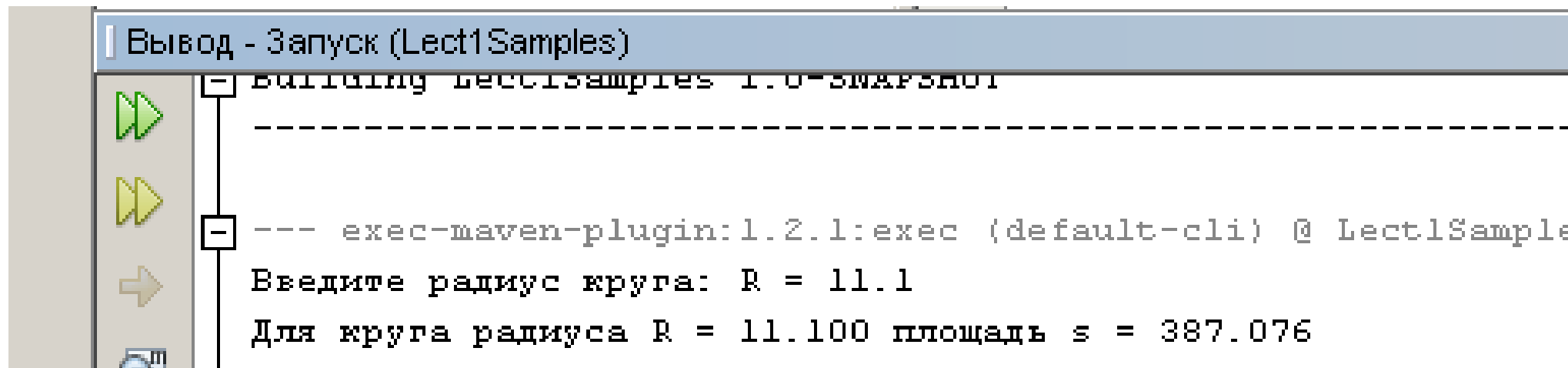
Вычисление площади круга

Программа выполняется по шагам:

```
1) Locale.setDefault(Locale.ROOT);  
2) Scanner scanner = new Scanner(System.in);  
3) System.out.print("Введите радиус круга: R = ");  
4) double r = double.Parse(Console.ReadLine());  
5) double s = Math.PI * r * r;  
6) System.out.printf(  
    "Для круга радиуса R = %1$.3f площадь s = %2$.3f%n",  
    r, s  
);
```

- 1) Первоначальная настройка среды исполнения (не относится к алгоритму)
- 2) Создание сканера (`java.util.Scanner`) для удобства ввода данных
- 3-4) Ввод значение радиуса R (в переменную r)
- 5) Вычисление площади S (в переменную s)
- 6) Вывод результата

Вычисление площади круга (результат выполнения)



```
Вывод - Запуск (Lect1Samples)
Building Lect1Samples-1.0-SNAPSHOT
-----
--- exec-maven-plugin:1.2.1:exec (default-cli) @ Lect1Sample
Введите радиус круга: R = 11.1
Для круга радиуса R = 11.100 площадь s = 387.076
```

Или в консоле:

```
>java -classpath Lect1Samples-1.0-SNAPSHOT.jar ru.vsu.cs.course1.Program
Введите радиус круга: R = 11.1
Для круга радиуса R = 11.100 площадь s = 387.076
```

Основы Java: переменные и типы данных

Переменные

- Переменные — именованные ячейки памяти для хранения значений (можно представить квадратик на листе бумаги, рядом с которым написано его имя)

	...
r =	11.1
s =	387.076
	...

- На самом деле любые данные в памяти хранятся в виде нулей и единиц, однако для понимания происходящего такое представление вполне корректно

Просмотр переменных в режиме отладки

The screenshot shows the NetBeans IDE interface with a Java program being debugged. The main editor displays the source code of `Program.java` with several lines highlighted in red and green. The left sidebar shows the project structure and the 'main' method. The bottom panel displays the 'Variables' window, which lists local variables and their values. Blue arrows point from text labels to specific UI elements: the 'Run' button, a line of code, the 'Variables' window, and the 'main' method in the project tree. A red arrow points to the 'main' method in the breadcrumb navigation.

указатель на строку, которая будет выполнена следующей

ТОЧКИ ОСТАНОВКИ

Выполнить одно действие (один шаг программы) в режиме отладки – F8

СПИСОК ВЫЧИСЛЯЕМЫХ ВЫРАЖЕНИЙ

СПИСОК ЛОКАЛЬНЫХ ПЕРЕМЕННЫХ (а также их значениями)

```
package ru.vsu.cs.course1;

import java.lang.Math;
import java.util.Locale;
import java.util.Scanner;

public class Program {
    public static void main(String[] args) {
        Locale.setDefault(Locale.ROOT);
        Scanner scanner = new Scanner(System.in);

        System.out.print("Введите радиус круга: R = ");
        double r = scanner.nextDouble();

        double s = Math.PI * r * r;

        System.out.printf(
            "Для круга радиуса R = %1$.3f площадь s = %2$.3f\n",
            r, s
        );
    }
}
```

Имя	Тип	Значение
<input checked="" type="checkbox"/> $\text{Math.PI} * r * r$	double	387.07563084879837
<input checked="" type="checkbox"/> $r * 10$	double	111.0
<Введите новый параметр наблюдения>		
<input type="checkbox"/> Статически		
<input type="checkbox"/> args	String[]	#79(length=0)
<input type="checkbox"/> scanner	Scanner	#231
<input type="checkbox"/> r	double	11.1

Отладка (Lect1Samples)

Присваивание значений переменной

- Значения переменных могут меняться в ходе выполнения программы (можно представить, что в клетку на листе бумаги вы можете записать новое значение, предварительно стерев старое – будем считать, что значения вы пишете карандашом)
- Например, следующая инструкция
 $r = 0.75;$
запишет в ячейку r значение 0.75
- Правильно читать « r присвоить 0.75», а не « r равно 0.75»

Объявление переменных

- Перед тем, как переменные использовать, их надо объявить.
- Переменные объявляются следующим образом (указывается тип и имя переменной):

```
int a; // a = 0 (значение по умолчанию)
```

- При объявлении переменной можно сразу же ее инициализировать каким-то значением:

```
double b = 1.5;
```

- Можно объявить сразу несколько переменных:

```
int c, d; // c = 0, d = 0
```

```
String
```

```
    s1 = "String 1",
```

```
    s2;
```

Константы

- Константы можно понимать, как переменные, значения которых нельзя поменять во время выполнения программы (можно представить, как квадратики на листе бумаги, в которых значения записаны ручкой, т.е. нельзя стереть)
- Значения констант задаются при их объявлении:

```
final int N = 100;  
final String HELLO = "Hello, World!";
```
- Иногда константы называют всеми большими буквами (чтобы по названию сразу было ясно, что это константа)
- Константы, как правило, объявляются в начале программы или функции
- Хорошим стилем считается, если в программе конкретные значения (например, конкретные числа: 10, 7.5 и т. п.) встречаются только в объявлениях констант

Идентификаторы в Java

- Идентификаторы — это названия переменных, функций, классов и т. д.
- Язык Java чувствителен к регистру символов программы, в том числе и к идентификаторам (А и а — разные переменные)
- Названия идентификаторам (переменным и т.д.) надо давать осмысленные, тогда программы будет самодокументированной
 - Не бояться «длинных» идентификаторов (linesCount, isPointInRect и т. п.)
 - Не стоит использовать в идентификаторах русские символы (Java допускает)
 - Самостоятельно прочитать соглашения по именованию идентификаторов языка Java

Область видимости переменных

- Область видимости переменной — фрагмент программы, в котором данная переменная доступна для использования
- Для простоты пока можно считать, что область видимости переменной задается фигурными скобками (переменная доступна от места своего объявления до соответствующей закрывающейся фигурной скобки, включая вложенные области видимости)
- Нельзя объявлять переменную с тем же именем, которая уже доступна в данной области видимости (за исключением глобальных для функции переменных)
- **Переменные нужно объявлять в максимально узкой области видимости как можно позже (это позволяет предотвратить некоторые глупые ошибки)**
 - Не стоит объявлять все необходимые переменные заранее, как в Pascal'e

```
int a;    // a = 0
...
{
    int a1 = 10;
    ...
    {
        // нельзя:
        // int a1 = 10;
        int a2 = 20;
    }
}
...
{
    // можно:
    int a1 = 20;
    ...
}
```

Типы данных

- Задают область допустимых значений переменных
- Описывают правила хранения различных данных в виде нулей и единиц в памяти и допустимые операции над данными
- Базовые типы языка Java (их больше, но в подавляющем большинстве случаев этих достаточно):
 - Числа
 - Целые (**int**)
 - Вещественные (**double**)
 - Символы (**char**)
 - Строки (**String**)
 - Логические значения (**boolean**)

Строгая типизация

- Строгая типизация означает, что
 - Для любого выражения в программе можно формально вывести его тип
 - При компиляции программы весь код (все операции) проверяется на совместимость или возможность преобразования типов, несовместимость считается ошибкой
- Строгая типизация позволяет:
 - Находить некоторые типы ошибок на этапе компиляции (т.е. разработки программы)
 - Генерировать более оптимальный код
- Строгая типизация делает программы более многословными и не допускает применения некоторых приемов, доступных в языках с динамической типизацией

Целочисленные типы данных

Тип	Размер (байт)	Диапазон
byte	1	от -128 до 127
short	2	от -32768 до 32767
char	2	беззнаковое целое число, представляющее собой символ UTF-16 (буквы и цифры)
int	4	от -2147483648 до 2147483647
long	8	от -9223372036854775808 до 9223372036854775807

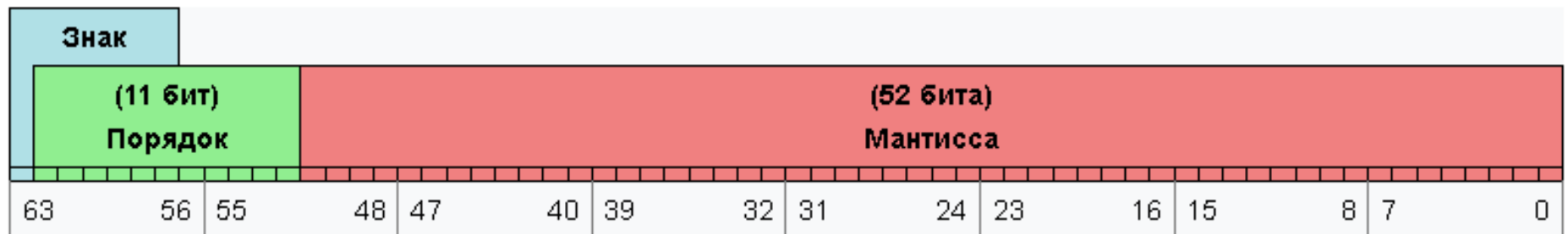
Типы с плавающей точкой (вещественные типы)

Тип	Размер (байт)	Диапазон
float	4	от $-1.4\text{e-}45\text{f}$ до $3.4\text{e+}38\text{f}$
double	8	от $-4.9\text{e-}324$ до $1.7\text{e+}308$

Вещественные типы данных.

Сравнение на равенство.

- Представление в памяти double



```
Console.WriteLine(9876543210.0123456789);           // 9876543210.01235
Console.WriteLine(0 == 1E-324);                       // true
Console.WriteLine(105.3256 - 105.32 == 0.0056);      // false
```

- Вычисления накапливают ошибку в дальних знаках после запятой, поэтому корректно сравнивать вещественные числа с заданной точностью, например, так:

```
Math.abs(a - b) < 1e-10
```

Объяснение, почему $105.3256 - 105.32 \neq 0.0056$

```
// Вещественные числа представлены в памяти в двоичном виде:  
// часть битов определяет несколько значащих знаков, часть - порядок,  
// т.е. положение этих знаков относительно запятой.  
// Это означает, что никаких точных десятичных значений в double не бывает  
// - только приближенные
```

```
double a = 105.3256, b = 105.32, c = 0.0056;  
System.out.println(a - b == c);    // напечатает false  
// объяснение  
System.out.printf("%f - %f = %f\n", a, b, a - b);  
System.out.printf("%.20f:%n %s\n", a, doubleToBinaryStr(a));  
System.out.printf("%.20f:%n %s\n", b, doubleToBinaryStr(b));  
System.out.printf("%.20f:%n %s\n", a - b, doubleToBinaryStr(a - b));  
System.out.printf("%.20f:%n %s\n", c, doubleToBinaryStr(c));
```

```
false  
105.325600 - 105.320000 = 0.005600  
105.325600000000000000000000:  
  0100000001011010010101001101011010100001011000011110010011110111  
105.320000000000000000000000:  
  0100000001011010010101000111101011100001010001111010111000010100  
0.0056000000000000116000:  
  00111111011101101111000000000110100011011011100011000000000000  
0.0056000000000000000000:  
  0011111101110110111100000000011010001101101110001011101011000111
```

Запомнить и осознать

- Вещественные числа в `double` всегда приближительны (в десятичном представлении только 15-16 знаков значащих)
- Вычисления дополнительно накапливают ошибку в младших значащих знаках, поэтому сравнивать `double` числа оператором равенства (`==`) нельзя
- `double` числа, полученные в вычислениях, можно сравнивать только с заданной точностью, например, так:

`Math.abs(a - b) < 1e-10`

double NaN и Infinity

- При некорректных вычислениях в double, например, делении на 0, вопреки распространенному мнению не происходит ошибок, а возникают специальные значения NaN (Not a Number) и Infinity
- Это надо учитывать и дополнительно проверять получившийся результат вычислений в случае возможности некорректных данных

```
double a2 = 100;  
double b2 = 0;  
double b2 = 0; c2 = a2 / b2;  
System.out.println(c2);           // Infinity  
System.out.println(c2 == c2);     // true  
System.out.println(Double.isInfinite(c2)); // true
```

```
double c3 = Math.sqrt(-1);  
System.out.println(c3);           // NaN  
System.out.println(c3 == c3);     // false  
System.out.println(Double.isNaN(c3)); // true
```

Логический и символьный тип

Тип	Число байт	Допустимые значения
<code>boolean</code>	1	<code>true</code> или <code>false</code>

Запомнить и осознать

- Не умничайте и используйте перечисленные выше 5 типов (про остальные базовые типы забудьте, пока без них не сможете обойтись)
 - `int`
 - `double`
 - `String`
 - `char`
 - `boolean`

Типы бинарных операторов

Оператор	Тип левого операнда A	Тип правого операнда B	Тип выражения: A Оператор B
+ - * / %	int	int	int
+ - * /	int	double	double
+ - * /	double	int	double
+	X	String	String
+	String	X	String
Битовые: << >> &	int	int	int
Логические: &&	boolean	boolean	boolean
Логические: < <= > >= == !=	Int, double, char	int, double, char	boolean

Типы унарных операторов

Оператор	Тип правого операнда A	Тип выражения: Оператор A
Битовые: ~	int	int
Логические !	boolean	boolean

Самостоятельно изучить (поэкспериментировать)

- Приоритеты операторов

Вычисление площади круга

```
package ru.vsu.cs.course1;

import java.lang.Math;
import java.util.Locale;
import java.util.Scanner;

public class Program {
    public static void main(String[] args) {
        Locale.setDefault(Locale.ROOT);

        Scanner scanner = new Scanner(System.in);

        System.out.print("Введите радиус круга: R = ");
        double r = scanner.nextDouble();

        double s = Math.PI * r * r;

        System.out.printf(
            "Для круга радиуса R = %1$.3f площадь s = %2$.3f%n",
            r, s
        );
    }
}
```

Код в функции main (или любой другой функции)

- Последовательность инструкций, понятных компилятору Java:
 - Объявление переменных
 - Вычисление значений и присвоение полученного результата переменным
 - Вызов процедур и функций (методов в терминах ООП)
 - Различные инструкции управления ходом вычисления программы
 - Условный оператор
 - Операторы циклов

Совет (настоятельная рекомендация)

- Обязательно смотрите проекты-примеры к лекции (и экспериментируйте с ними):
 - Lect1Samples