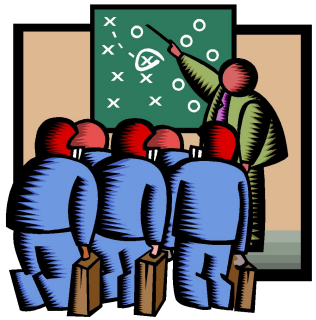
Урок 1 - Вводный

Знакомство

- Ераскин Алексей (Миронюк Антон)
- Разработчик на Java
- Вопросы - aeraskin@prog-school.ru
- Закрытый форум ШП*

(*регистрация на форуме – support@prog-school.ru)



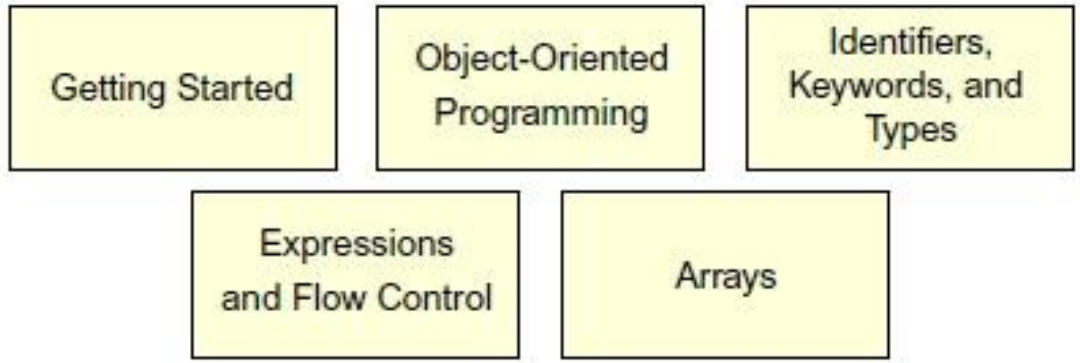
Обзор курса*

- Синтаксис Java и реализация алгоритмов на Java
- ООП в терминах Java
- Коллекции, шаблоны (generics), ввод/вывод данных
- Разработка графического интерфейса (GUI)
- Многопоточность
- Работа с сетью (TCP/IP)
- Создание, отладка и запуск приложений на Java
- Разработка итогового приложения на Java (чат)

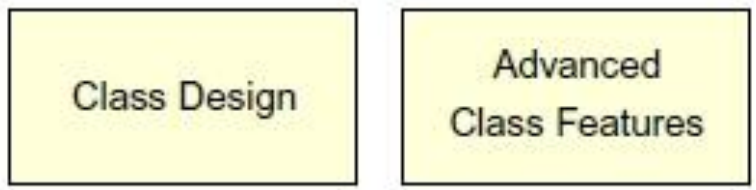
(*программа курса – http://proglive.ru/courses/java1)

Карта курса

1. Основные конструкции языка Java



2. Объектно-ориентированное программирование и особенности его реализации на Java

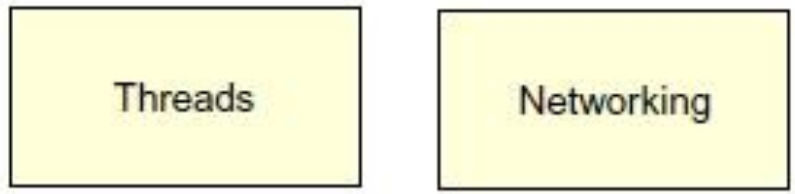


Карта курса

3. Коллекции, шаблоны (generics), ввод/вывод данных, обработка ошибок

4. Разработка программ с графическим пользовательским интерфейсом (GUI)

5. Многопоточность и работа с сетью



Требования к участникам

Начальные знания:

Курс не для полных новичков в IT. Предполагается* понимание схемы работы компьютера, знакомство с двоичным представлением данных в памяти, а также опыт работы с основными алгоритмическими конструкциями (цикл, ветвление, переход) на любом языке.

(*программа курса по основам – http://proglive.ru/courses/basics)

Пожелания к участникам

- Внимательно изучать все уроки по плану
- Выполнять все дз
- Выполнять все необходимые исправления
- Активно задавать вопросы (форум/почта)
- Изучать дополнительный материал

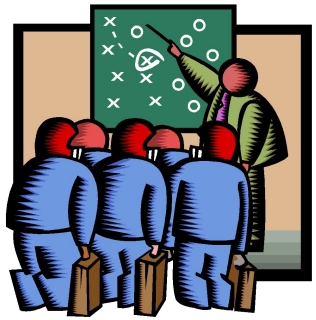
Бонусы и скидки

- Скидки 10%, 15%, 20% (клубные карты)
- Участники, выполнявшие все дз, получат поощрительные подарки от ШП
- По результатам выполнения дз выбирается победитель курса – скидка 50% на любой курс

Сделать к уроку 2

- Изучить видео* урока 1 (закрытый форум ШП)
- Выполнить задание и запросить дополнительное по почте aeraskin@prog-school.ru
- Выполнить дополнительное задание
- Выслать информацию о себе на почту (имя, образование или опыт работы, основные ожидания от курса)
- Указать, желательно ли разбирать в первой половине урока дополнительный материал (логические задачи, вопросы с собеседований, другое)

(*код для просмотра видео – support@prog-school.ru)



Цели урока 1

- Назвать ключевые особенности технологии Java
- Скачать и настроить Java Development Kit (JDK)
- Описать назначение Java Virtual Machine (JVM)
- Написать, скомпилировать и запустить простейшее приложение на Java
- Обсудить различные среды разработки (IDE)
- Познакомиться с IDE Eclipse
- Создать простейшее приложение в IDE Eclipse

Программы

- Программы (software) – набор машинных инструкций
- Для создания программ нужны программные языки
- Программные языки – упрощенная форма человеческих языков

С этой точки зрения программные языки делятся на:

1. Машинный язык
2. Ассемблер
3. Языки высокого уровня

Программы

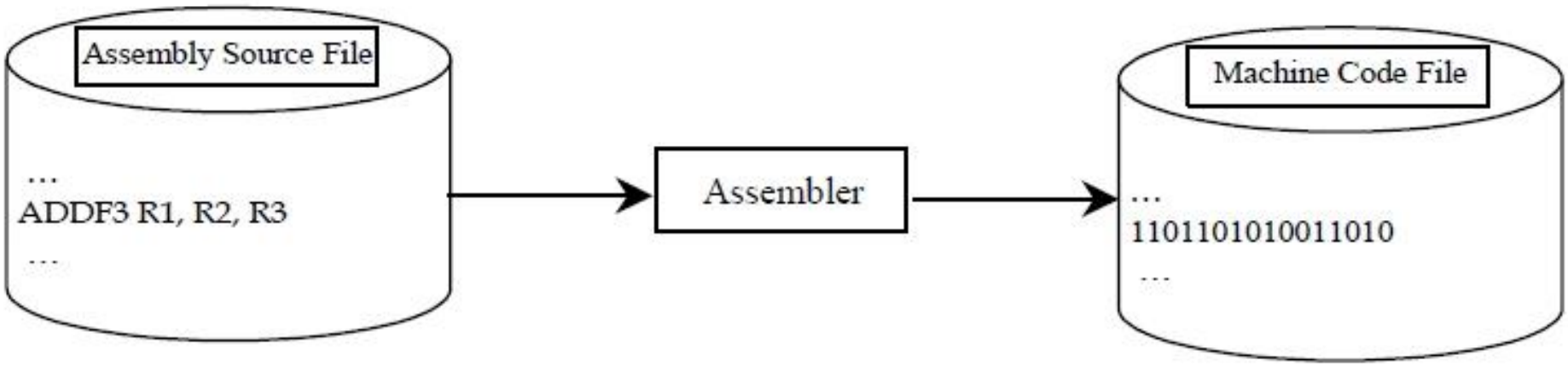
1. Машинный язык:

- Инструкции и данные представляют собой бинарный код вида 10001011101
- Код специфичен для данной машины
- Читать и отлаживать очень и очень сложно

Программы

2. Ассемблер:

- Инструкции представляют собой короткие псевдослова вида ADDF3 R1, R2, R3
- Для запуска программы требуется перевести ассемблерный код в машинный
- Код также специфичен для данной машины
- Читать и отлаживать проще, но все же сложно



Программы

3. Языки высокого уровня (Java, C/C++, C# и т.д.):

- Можно составлять сложные выражения, похожие на предложения на английском языке вида

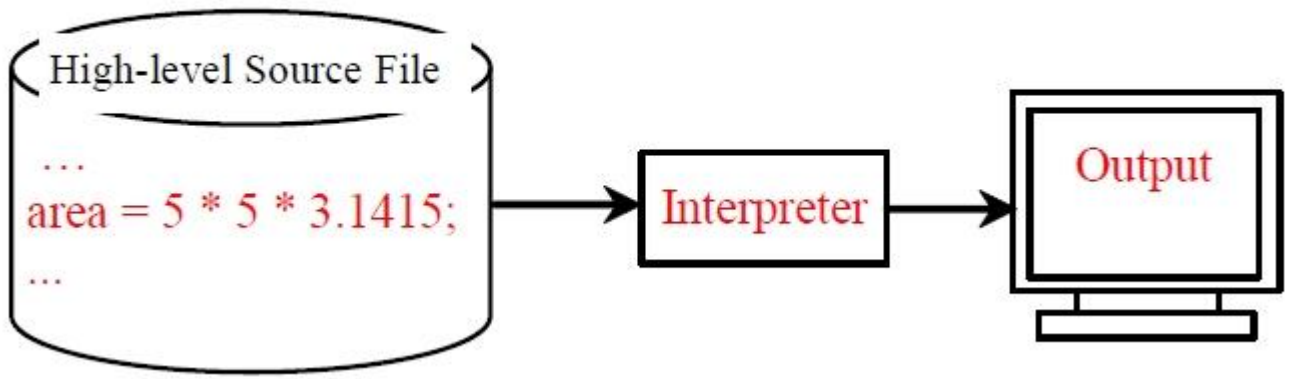
```
int area = 5 * 5 * 3.1415;
```

- Для запуска программы требуется интерпретатор или компилятор
- Код может не быть специфичным для данной машины (в случае с интерпретатором)
- Читать и отлаживать наиболее просто

Интерпретация/компиляция исходного кода (source code)

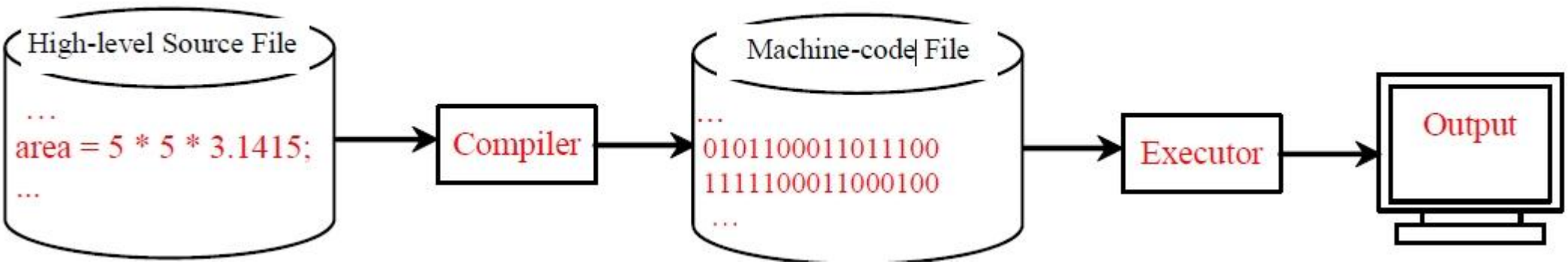
- Программа на языке высокого уровня называется исходным кодом (source code)
- Для запуска исходного кода требуется интерпретатор или компилятор

Интерпретатор читает выражения в исходном коде одно за другим, переводит их в машинный код (или виртуальный машинный код) и сразу выполняет:



Интерпретация/компиляция исходного кода (source code)

Компилятор переводит исходный код в машинную форму и сохраняет ее в отдельном месте (файле), которая может выполняться независимо от этапа компиляции:



Введение в платформу Java и краткая история

- 1991 “Green Project”
- Oak language
- Sun Microsystems
- Java language
- “Write once, run anywhere”
- Oracle (с 27/01/2010)

История версий Java

- JDK 1.0 «Oak» 1996 (first release Oak language)
- JDK 1.1 «Java» 1997 (inner classes, JDBC, RMI, reflection)
- J2SE 1.2 «Java 2»1998 (swing, collections, JIT-compiler)
- J2SE 1.3 «Java 2» 2000 (HotSpot JVM, JNDI API, Java Sound API, Debugging Architecture)
- J2SE 1.4 «Java 2» 2002 (regular expression, logging API, XML/XSLT, Java Web Start)
- J2SE 5.0 «Java 2» 2004 (generics, enumerations, variable arguments, for-each loops) a.k.a. Java 5
- J2SE 6.0 «Java 2» 2006 (JDBC 4.0, Java Compiler API, GUI improvements) a.k.a. Java 6
- J2SE 7.0 «Java 2» 2011 (Da Vinci Machine, NIO, Project Coin) a.k.a Java 7

Java Editions

- Java Stadard Edition (J2SE) – используется для разработки самостоятельных приложений или апплетов, так называемая Core Java

- Java Enterprise Edition (J2EE) – используется для создания приложений на серверной стороне (в терминах приложений с клиент-серверной архитектурой), содержит пакеты для работы с Java Servlets, Java Server Pages (JSP), JDBC и т.д.

- Java Micro Edition (J2ME) – используется для разработки самостоятельных приложений на мобильных устройствах

В рамках этого курса изучается Java Standard Edition.

Особенности Java

- Поддержка ООП
- Встроенный сборщик мусора
- Обнаружение ошибок на этапе компиляции
- Встроенная обработка ошибок (exceptions handling)
- Переносимость кода (write once, run everywhere)
- Поддержка многозадачности на уровне языка
- Динамическая загрузка классов (по необходимости)
- Поддержка работы с высокоуровневыми сетевыми протоколами
- …

Java Virtual Machine (JVM)

- Осуществляет поддержку конкретной аппаратной платформы
- Работает с аппаратно-независимым байт-кодом, полученным на этапе компиляции исходного кода в байт-код
- Байт-код может быть запущен на любом компьютере (win/mac/unix), на котором установлена JVM
- Программная реализация JVM содержится в составе Java Runtime Environment (JRE)
- JRE можно установить отдельно – а можно, в составе Java Development Kit (JDK)

(http://www.oracle.com/technetwork/java/javase/downloads/index.html)

Работа с памятью в Java

- Выделять память физически не требуется
- Высвобождение памяти происходит автоматически с помощью встроенного сборщика мусора
- Сборщик мусора (garbage collector) автоматически проверяет область памяти, где живут объекты Java – Java Heap (куча) – и уничтожает их, если они стали не нужны программе
- Алгоритм работы сборщика мусора зависит от конкретной платформы – а значит, конкретной JVM

Популярные Java IDE

- NetBeans (www.**netbeans**.org)
- Eclipse (www.**eclipse**.org)
- IntelliJ IDEA (www.jetbrains.com/**idea**)

Сделать к уроку 2

- Изучить видео* урока 1 (закрытый форум ШП)
- Выполнить задание и запросить дополнительное по почте aeraskin@prog-school.ru
- Выполнить дополнительное задание
- Выслать информацию о себе на почту (имя, образование или опыт работы, основные ожидания от курса)
- Указать, желательно ли разбирать в первой половине урока дополнительный материал (логические задачи, вопросы с собеседований, другое)

(*код для просмотра видео – support@prog-school.ru)

Конец урока 1

Закрытый форум ШП

aeraskin@prog-school.ru