Питон
Курс лекций
Лекция первая

© Зайцев Вадим Валерьевич, 2002–2010,
spider.vz@gmail.com

Какое бы название ни имел тот или иной курс, первая лекция обычно не содержит ничего (или почти ничего) из той основной и самой важной части курса, ради которой он и был задуман. Не отступая от этой традиции, мы сегодня расскажем непосвящённым о том, что такое компьютер и зачем он нам нужен и введём несколько основных определений, которые позже неминуемо перейдут в класс очевидных и интуитивно понятных.

На самом деле первая лекция курса нужна лишь для того, чтобы дать почувствовать, что предмет из себя представляет, для чего он нужен и что он может дать лично вам.

Весь материал будет делиться на несколько больших тем, каждая из которых вправе содержать какое-то ненулевое количество малых тем.

1 Введение

1.1 ЭВМ и её обеспечение

Обеспечение компьютера, как известно, делится на две неравные части: аппаратное и программное. Это деление сродни делению человека на душу и тело. Аппаратное обеспечение — это тело, то есть всё, существующее в качестве деталей: корпус, монитор, платы, устройства ввода/вывода информации, различные провода, шлейфы и порты. Программное обеспечение — душа компьютера — куда богаче и разнообразнее: от содержимого микросхем `BIOS` и загрузочных секторов дисков до новой версии Windows, которая может занимать многие гигабайты. Программное обеспечение часто делят на системное и прикладное.

Системное программное обеспечение — это то, которым вы пользуетесь, сами того не замечая, либо к которому прибегаете в самых тяжёлых моментах жизни компьютера. То есть: операционные системы, драйверы и всевозможные утилиты.

Прикладное программное обеспечение является, вообще говоря, предметом роскоши, посему чрезвычайно разнообразно: *файловые менеджеры*, *редакторы* и *просмотрщики* многочисленных форматов файлов, *проигрыватели* музыки и видео, *архиваторы* (числившиеся не так давно в системном), *вычислительные системы*, *сетевые приложения* и *средства подготовки программ*, которые заслуживают того, чтобы остановиться на них подробнее. В эту категорию относят все программы, служащие для производства новых программ.

Спектр средств подготовки программ содержит *редакторы* исходных текстов (обычно обеспечивающих подсветку (выделение некоторых элементов текста, имеющих значение для пользователя: скобок, служебных слов и т.д.) и некоторую поверхностную проверку синтаксиса вводимых конструкций), *трансляторы* (позволяющие собственно запускать программы), *отладчики* (призванные служить благородному делу поиска ошибок в программах, но не всегда помогающие программисту) и в некоторых случаях ещё и *тесты* (профайлеры), позволяющие, например, определить наиболее медленный или наиболее требовательный к ресурсам блок программы.

В последнее время чётко выделились две тенденции, употребляющиеся соответственно в двух выдержавших жестокую конкуренцию семействах операционных систем: **UNIX** и **Windows** (операционные системы некогда переживали бум, теория их строения была сформулирована очень подробно, но, увы, многие разработки так и остались в теоретической области). В юниксах, вообще говоря, всего два редактора: *vi* и *emacs*, и каждый юниксоид, подчас великолепно владея одним из них, с трудом догадывается о том, как выйти из другого. *Emacs*, например, определяет по расширению открываемого файла, какую подсветку ему применять, и в стандартной поставке содержит до сотни подсветок разных языков программирования. Отладчик в юниксе также один, работает на очень низком уровне и редко помогает на практике при использовании языка сколь-нибудь высокого уровня. Таким образом, в поставку языка под юниксовую платформу обычно включается только транслятор (и лишь в редких случаях высокоуровневый отладчик).

Под **Windows** дело обстоит несколько иначе — все вышеперечисленные компоненты спаяны воедино и результат называется *интегрированной средой разработки*. Выглядит это, как вы, вероятно, знаете, как редактор, из которого различными комбинациями клавиш можно вызвать такие действия, как компиляцию исходного текста, выполнение программы, запуск отладчика, и т.д.

1.2 Трансляторы языков программирования

Трансляторы бывают трёх типов: *ассемблеры*, *компиляторы* и *интерпретаторы*. Ассемблеры переводят программу на языке ассемблера в машинные коды. При этом каждой строчке исходного текста ставится в соответствие одна команда процессора (от одного до дюжины байт кода). Компиляторы переводят текст программы на языке высокого уровня в машинные коды. При этом одной строчке исходного текста (которая в языке высокого

уровня может иметь невероятно сложную структуру) может соответствовать много тысяч команд процессора. Интерпретаторы исполняют программу на языке высокого уровня немедленно, строчка за строчкой. Естественно, для этого они должны перевести исходный текст в другое представление, которое не обязано быть машинным кодом. Например, транслятор языка Ява переводит исходный текст в команды так называемой виртуальной машины.

Вообще, справедливо следующее: