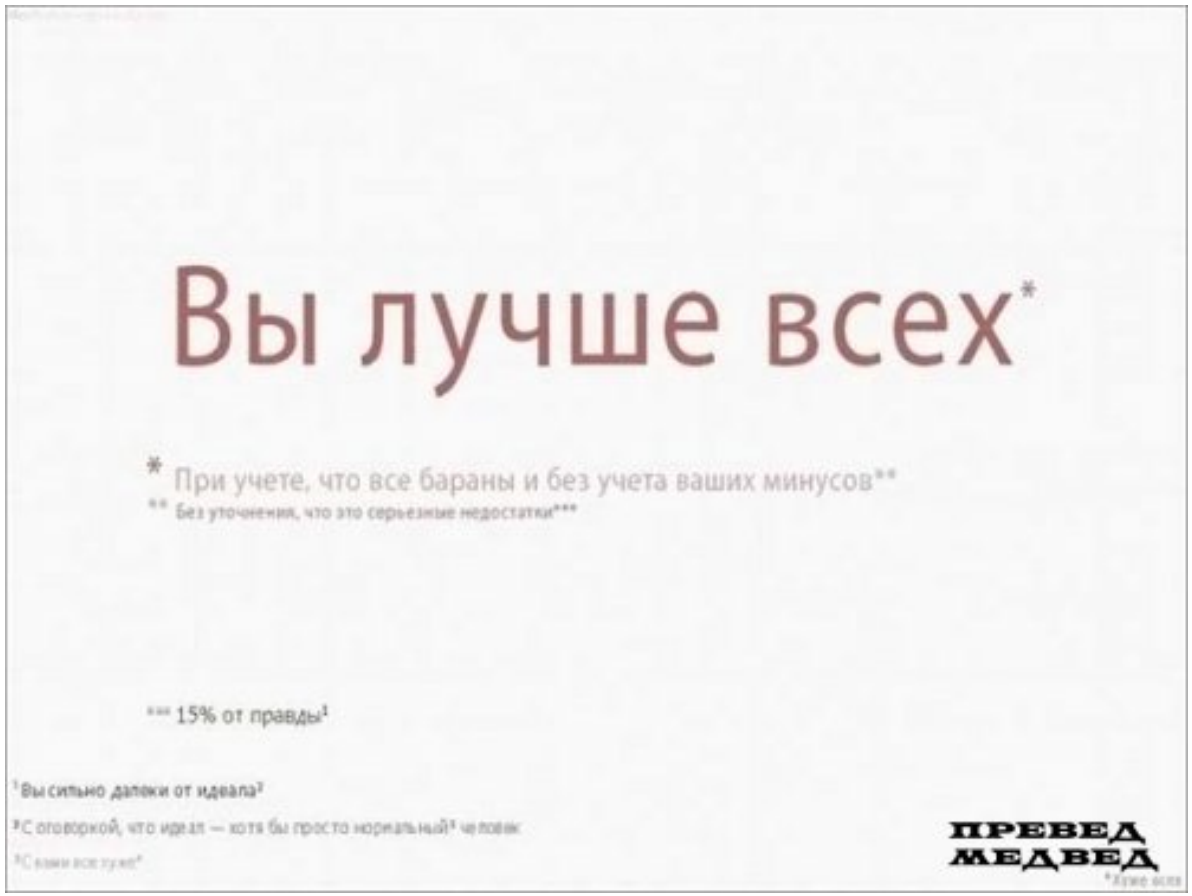
Управляющие структуры в ruby

Информатика
10-11 классы

21 февраля 2012 г.

Вместо введения

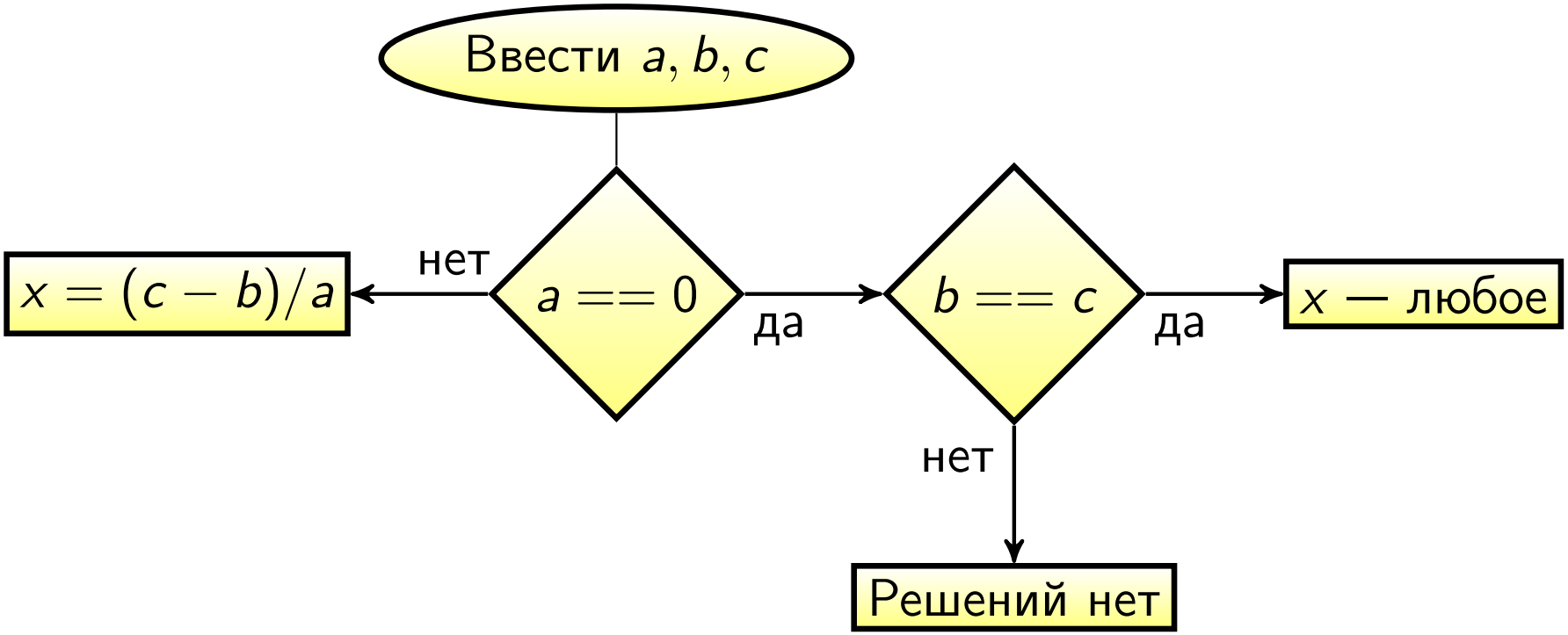


Условия

- Алгоритмы и программы зачастую имеют нелинейную структуру.
- В зависимости от различных параметров системы программы могут работать по-разному.
- Например, при логине на сайте ВКонтакте есть две возможные ситуации:
  1. Вы вводите правильные логин и пароль и попадаете на свою страницу.
  2. Введённая пара “логин–пароль” неверна, и Вас переадресовывает обратно на страницу логина
- Вариантов поведения может быть больше, чем два.
- Такое поведение программ соответствует элементу блок–схемы “Условие” и структуре “Ветвление”.

Блок–схема

Вернёмся к задаче о решении линейного уравнения.



Программа

Listing 1: Решение линейного уравнения

```
a = 5.0
b = 3.0
c = -2.5
if (a == 0)
  if (b == c)
    puts "x - any number"
  else
    puts "there is no solution"
  end
else
  x = (c-b)/a
  puts "x = #{x}"
end
```

Пояснения к программе

- **if** ... **else** ... **end** — оператор условия.
- **if (a == 0)** означает *если значение переменной a равно нулю*.
- В случае, если **a** действительно равно нулю, то выполняется код, расположенный сразу после слова **if**.
- Если же условие ложно (то есть, в нашем случае $a \neq 0$), то выполняется код, расположенный после **else** (*else* переводится как *иначе*). При ложном условии код, расположенный после **if**, просто–напросто игнорируется.
- Условия могут быть вложенными друг в друга. В нашем примере после одного условия сразу же следует другое. Количество “уровней вложенности” не ограничено.
- В конце условия ставится оператор *end*.

Неполные условия

- Условия могут быть *неполными* (неполное означает отсутствие ключевого слова **else**):

Listing 2: Неполное условие

```
if (a == 0)
  puts "a equal to 0"
  if (b == 0)
    puts "b is equal to 0 too"
  end
end
```

Модификаторы

- Если мы имеем неполное условие и при этом нам нужно выполнить всего одно действие, можно использовать *сокращённую запись условия* (*модификатор*):

Listing 3: Модификатор

```
puts "a is equal to 0" if (a == 0)
```

Отрицательный модификатор

- А если мы хотим сделать какое-либо действие в случае, когда $a \neq 0$?

Listing 4: Простой вариант

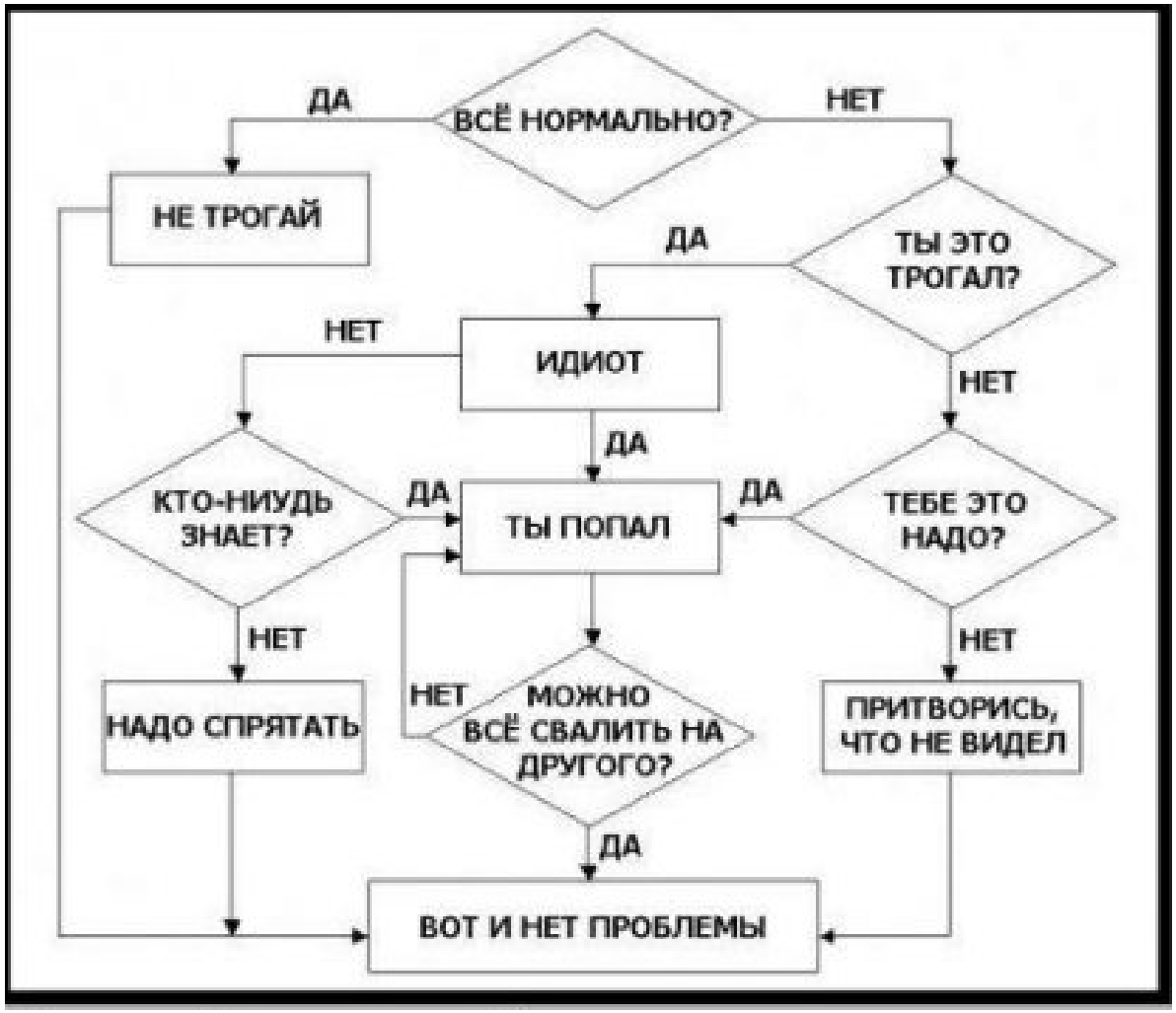
```
puts "a is equal to 0" if (a != 0)
```

- Однако для лучшего понимания кода проще, когда все условия — простые. Для этого в ruby есть *ключевое слово unless*, которое можно перевести как *если не*. С ним программа становится проще.

Listing 5: Улучшенный вариант

```
puts "a is equal to 0" unless (a == 0)
```

Пример


ВСЁ НОРМАЛЬНО?
ДА
НЕТ
НЕ ТРОГАЙ
ТЫ ЭТО ТРОГАЛ?
ДА
НЕТ
ИДИОТ
НЕТ
ДА
КТО-НИУДЬ ЗНАЕТ?
ДА
ТЫ ПОПАЛ
ДА
ТЕБЕ ЭТО НАДО?
НЕТ
НЕТ
НЕТ
НАДО СПРЯТАТЬ
НЕТ
МОЖНО ВСЁ СВАЛИТЬ НА ДРУГОГО?
ПРИТВОРИСЬ, ЧТО НЕ ВИДЕЛ
ДА
ВОТ И НЕТ ПРОБЛЕМЫ

Логические операции

- А если мы хотим одновременно проверить несколько условий? Например, если и **a**, и **b** равны нулю. Или же рассмотреть случай, когда хотя бы одна из переменных равна нулю.
- Для этого нужно использовать логические операции: *конъюнкцию* **&&** и *дизъюнкцию* **||**.

Listing 6: Конъюнкция и дизъюнкция

```
if ( (a == 0) && (b == 0) )
  puts "a and b is equal to 0"
end
puts "a or b is equal to 0" if ( (a == 0) || (b == 0) )
```

Сравнения

- Что кроме *проверки на равенство* можно делать в условиях?

Оператор	Описание	Типы переменных
==	равно	любые
!=	не равно	любые
>	больше	integer, float
>=	больше либо равно	integer, float
<	меньше	integer, float
<=	меньше либо равно	integer, float

Таблица: Операторы сравнения

Полное условие

- Рассмотрим реальную задачу решения квадратного уравнения.
- Пусть D — дискриминант уравнения. В ней три варианта:
  1. $D > 0$ — два вещественных корня,
  2. $D = 0$ — один вещественный корень 2 кратности,
  3. $D < 0$ — вещественных корней нет.

Listing 7: Пример полного условия

```
if (D > 0)
  puts "2 real roots"
elsif (D == 0)
  puts "One real root"
else
  puts "No real roots"
end
```

Полное условие

Listing 8: Схема полного условия

```
if (...)
  ...
elsif (...)
  ...
...
elsif (...)
  ...
else
  ...
end
```

- В полном условии добавляется ключевое слово **elsif**, которое переводится как *иначе если*.
- Сначала ruby рассмотрит условие после **if**. Если оно будет ложным, он перейдёт к первому **elsif**. И так далее. Если же все условия окажутся ложными, ruby перейдёт к блоку **else**.
- Кстати, блок **else** не является обязательным!

Квадратное уравнение

- Итак, вернёмся к квадратному уравнению. Напишем программу, высчитывающую все корни (если таковые имеются) квадратного уравнения $ax^2 + bx + c = 0$.
- Немного упростим себе задачу, предположив, что $a \neq 0$.[1]
  1. Вычислим дискриминант уравнения по формуле: $D = b^2 - 4ac$.
  2. Если дискриминант меньше нуля, то решений нет.
  3. Если дискриминант равен нулю, то корень — один. Он равен: $-\frac{b}{2a}$.
  4. Если дискриминант больше нуля, то существует два вещественных корня:

$$x_{1,2} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

[1] Не забудьте сделать самостоятельно алгоритм без такого допущения.