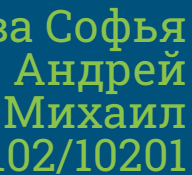




Lua

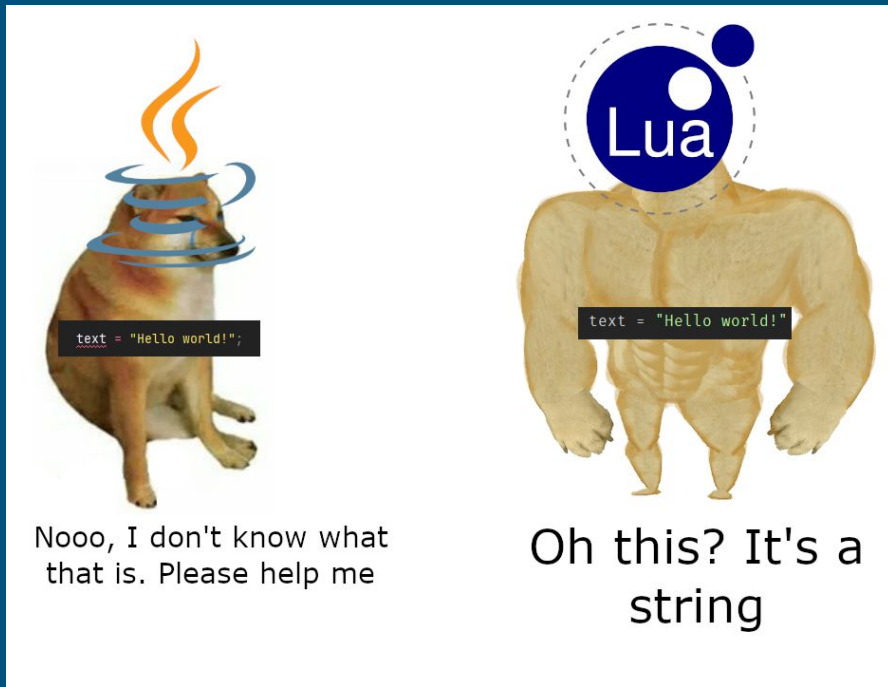
2. Типы и значения



Корпусова Софья
Афанасьев Андрей
Лутченко Михаил
5030102/10201

Введение

Lua — язык с динамической типизацией. У переменных нет predefined типов, любая переменная может содержать любое значение.





Отсутствие значения - `nil`

- Основная задача - отличаться от остальных значений
- Изображает отсутствие подходящего значения
- Глобальные, неинициализированные переменные - `nil`
- Присваиваем `nil` глобальной переменной для удаления

Области видимости

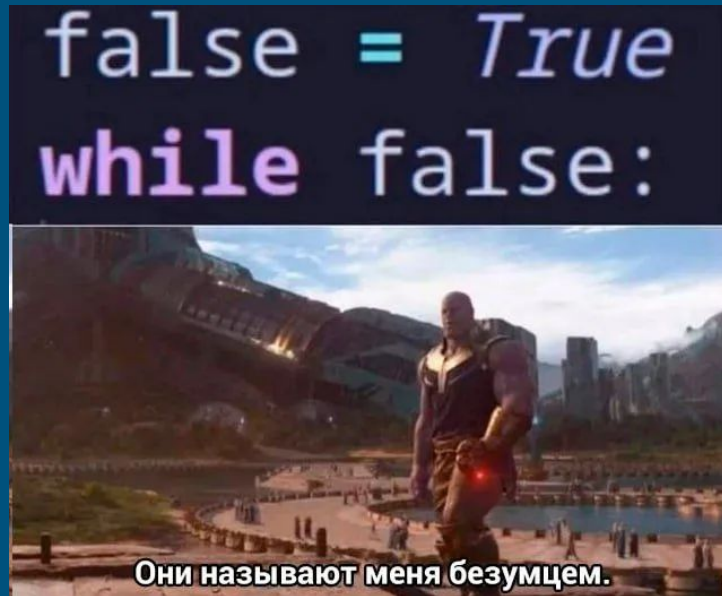
- Область видимости - это строки кода с одинаковым уровнем вложенности

```
-- (1, начало)
if true then
  -- (2, начало)
  for i = 1, 100 do
    -- (3)
  end
  -- (2, конец)
end
-- (1, конец)
```

- Локальные переменные привязаны к области видимости, создаются с помощью `local`

Логические значения - boolean

- `true` или `false`
- `nil` и `false` - ложь, остальное - истина (даже `0` и `""`)



Числа - number

- number = double в C, тип integer в Lua отсутствует
- При использовании вещественного числа для представления целого ошибок округления не происходит
- У дробных чисел могут быть ошибки представления
- Примеры допустимых числовых констант:

4 0.4 4.57e-3 0.3e12 5E+20 0xff (255)

0x1A3 (419) 0x0.2 (0.125) 0x1p-1 (0.5) 0xa.Ъp2 (42.75)

Строки - `string`

- Последовательность символов
- Можно хранить в виде строк любые бинарные данные
- Строки неизменяемы
- Подвержены автоматическому управлению памятью
- Операция длины `#`

Строковые литералы

- Можно использовать `"` `"` и `\` `'`
- С-образные экранированные последовательности:

<code>\a</code> звонок	<code>\t</code> горизонтальная табуляция
<code>\b</code> возврат на одну позицию	<code>\v</code> вертикальная табуляция
<code>\f</code> перевод страницы	<code>\\</code> обратный слеш
<code>\n</code> перевод строки	<code>\"</code> двойная кавычка
<code>\r</code> возврат каретки	<code>\'</code> одинарная кавычка

- Символ в строке можно задать с помощью числового значения, которое указывается экранированными последовательностями `\ddd` и `\x\hh`, где `ddd` — это не более трех десятичных цифр, а `hh` — две шестнадцатеричные цифры

Длинные строки

- Создаются с помощью `[[]]`
- Экранированные последовательности в этих строках не будут интерпретироваться
- Эта форма игнорирует первый символ строки, если это перевод строки
- Если внутри литерала есть квадратные скобки, то `[=== [] ===]`
- Перенос внутри короткой строки: `\z` - пропускает все последующие символы в строке вплоть до первого пробельного символа

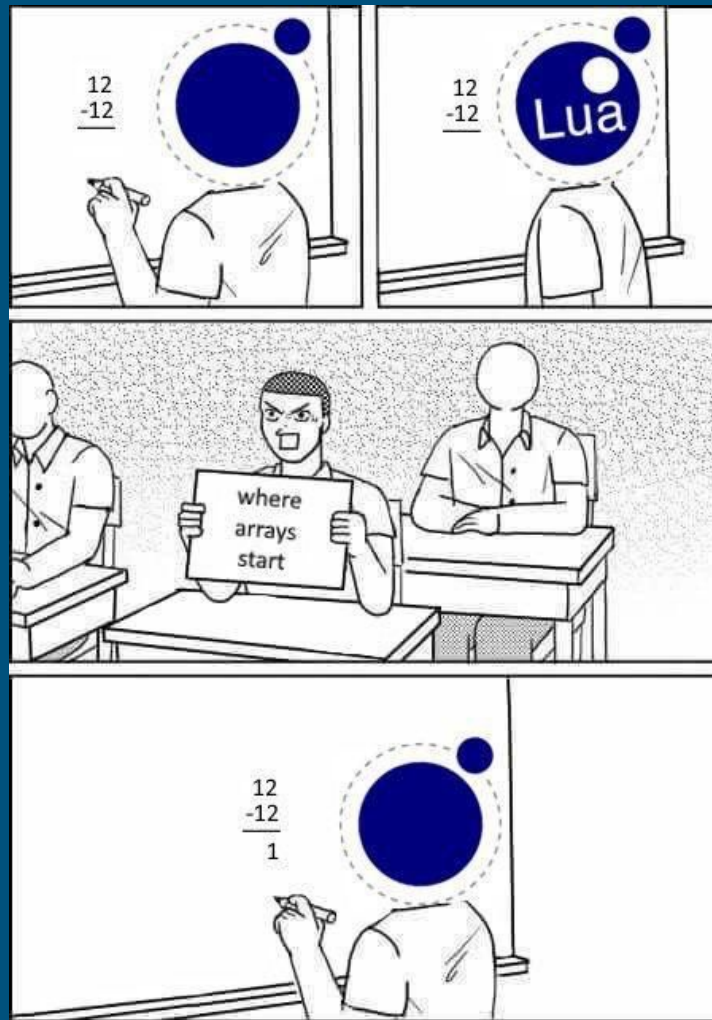
Таблицы - table

- Ассоциативный массив - может быть индексирован не только числами, но и строками или любым другим значением языка
- Единственный механизм структурирования данных в Lua



Таблицы - table

- Переменная типа table хранит ссылку на соответствующий объект
- `a = {}` - создали таблицу и положили ссылку на неё в `a`
- Не существует постоянной связи между переменной и самой таблицей (таблицы анонимны)
- Нет ссылок на таблицу - удаление



Функции - function

- Могут быть записаны в переменные, переданы как параметры в другие функции и возвращены как результат выполнения функций
- Функции анонимны
- Определение функции:

```
function f(x,y)  
    return x*y  
end
```

```
f = function (x,y)  
    return x*y  
end
```

```
function f(x,y) return x*y end
```

Приведение типов

- Неявное: строка преобразуется в число везде, где ожидается число и наоборот
- Явное: функции `tonumber()` и `tostring()`

В следующей презентации:

- Арифметические операции
- Операции сравнения
- Логические операции
- Конкатенация
- Операция длины
- Приоритеты операций

Литература

- [Программирование на языке Lua - Роберту Иерузалимски](#)
- [Разработка скриптов Lua](#)
- [Типы переменных, функции, видимость переменных \(Lua\)](#)
- [Программирование на языках Lua и C](#)
- [Работа с переменными в Lua](#)
- [Работа с таблицами в Lua](#)