Переменные и изменяемое состояние в языках программирования

Всеволод Дёмкин
vseloved@gmail.com

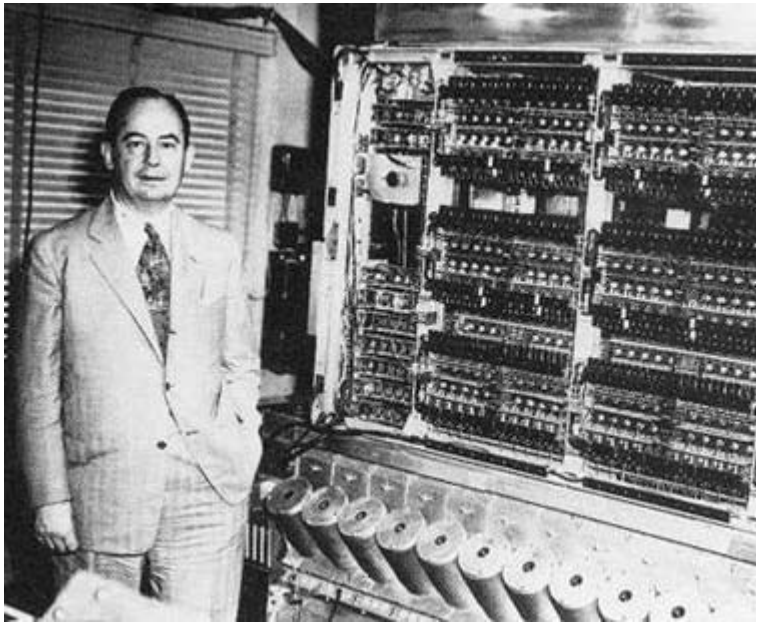
Модель подстановки

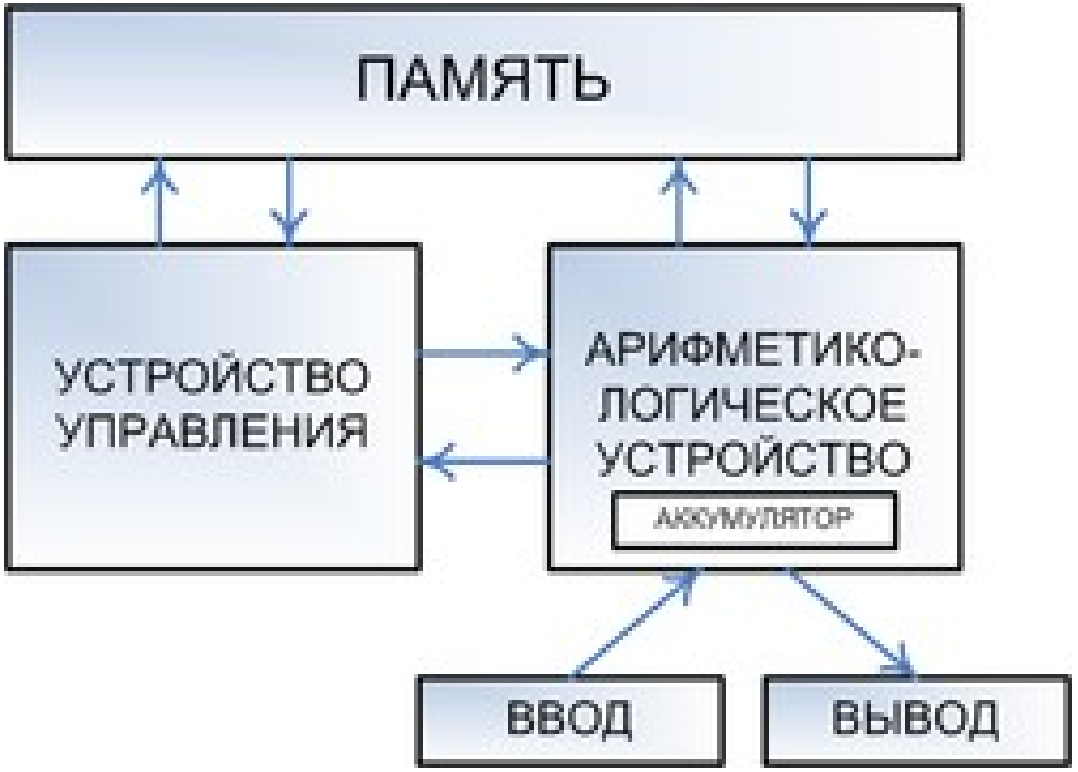
```
function fact (x) =
  { x * fact(x - 1), x > 0
  { 1
```

```
int fact (int x) {
    if (x > 0)
        return x * fact(x-1);
    else
        return 1;
}
```

http://www.willamette.edu/~fruehr/haskell/evolution.html

Архитектура фон Неймана





Архитектура x86

Инструкции процессора
Регистры:
- общего назначения: A, B, C, D
- специальные
Разрядность памяти и способ адресации

http://en.wikipedia.org/wiki/X86_instruction_listings

Факториал на ассемблере
x86 (16bit)

```
fact:
    cmp BX, 0   ; x ?= 0
    jg x_gt_0   ; if (x > 0)
    mov AX, 1   ; AX = 1
    ret         ; return (AX)

x_gt_0:
    dec BX      ; x = x - 1
    call fact   ; AX = fact(x - 1)
    inc BX      ; restore x
    mul BX      ; AX = AX * x
    ret         ; return (AX)
```

Факториал (версия 2)

```
int fact (int x) {
    if (x > 0) {
        int x1 = x - 1;
        int x2 = fact(x1);
        int x3 = x * x2;
        return x3;
    } else {
        return 1;
    }
}
```

Факториал (версия 3)

```
int BASE = 1;

int fact (int x) {
    if (x > 0) {
        int x1 = x-1;
        int x2 = fact(x1);
        int x3 = x * x2;
        return x3;
    } else {
        return BASE;
    }
}
```

Замыкания

```
> (defun inner (x)
    (setf x (+ x 1))
    (print x))

> (defun test1 (x)
    (inner x)
    (print x))

> (test1 1)
2
1 1
```

```
> (defun test2 (x)
    (defun inner ()
      (setf x (+ x 1))
      (print x))
    (inner)
    (print x))

> (test2 1)
2
2 2
```

Замыкания

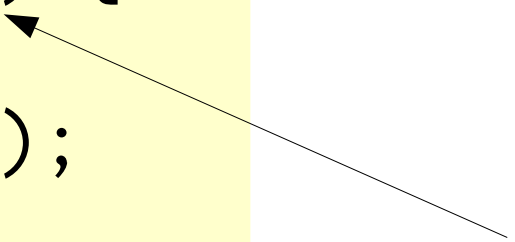
```
> (defun outer ()
    (setf x (+ x 1))
    (print x))
; caught WARNING:
;   undefined variable: X

> (defun test3 (x)
    (outer)
    (print x))

> (test3 1)
Error: The variable X is unbound.
```

Замыкания

```
void test2 (int x) {
    void inner () {
        x += 1;
        printf(x);
    }
    inner();
    printf(x);
}
```



нет x

Подвохи замыканий

```
var funcs = {};
for (var i = 0; i < 3; i++) {
    funcs[i] = function() {
        console.log("My value: " + i);
    };
}

for (var j = 0; j < 3; j++) {
    funcs[j]();
}
```

Ожидали:

```
My value: 0
My value: 1
My value: 2
```

Получили:

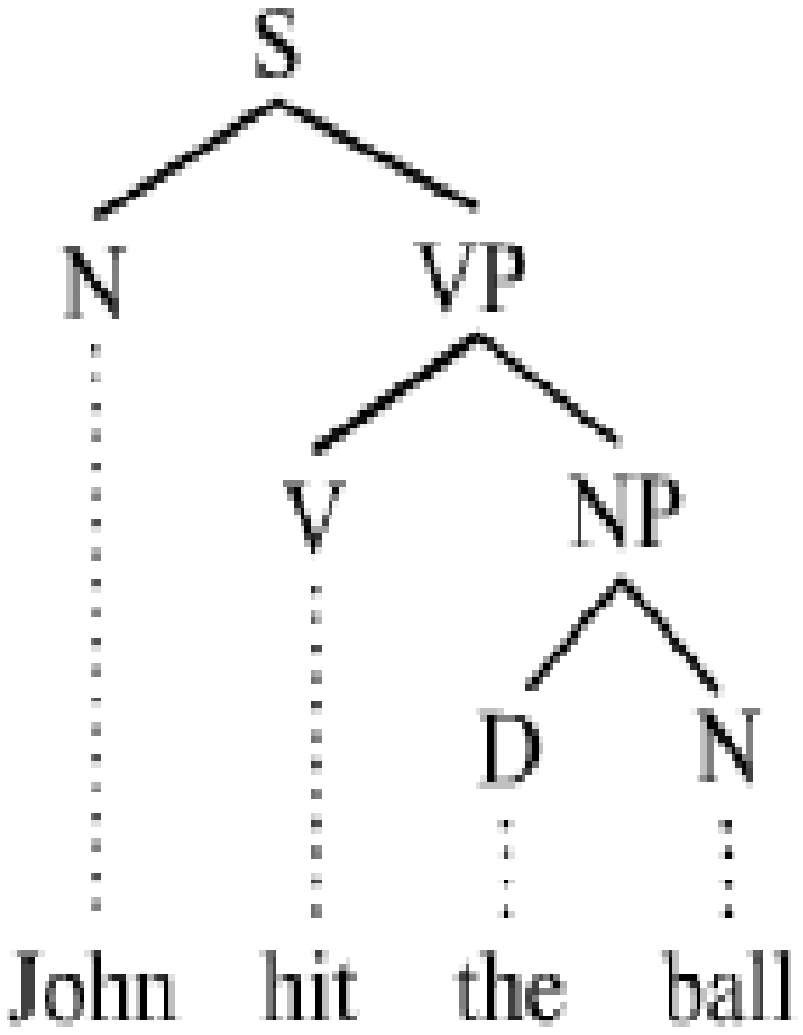
```
My value: 3
My value: 3
My value: 3
```

http://stackoverflow.com/questions/750486/javascript-closure-inside-loops-simple-practical-example

Переменные

Вид переменной	Видимость	Продолжительность жизни
локальная	блочная	ограниченная
лексическая	блочная	бесконечная
глобальная	глобальная	бесконечная
статическая	блочная/файловая	бесконечная

Особые переменные




```
(S (N John)
   (VP (V hit)
       (NP (D the)
           (N ball))))
```

```
(pattern verb-phrase
  (VP (V)
      (NP (D)
          (N))))

(pattern sentence
  (S (or (NP) (N))
     (or (verb-phrase)
         (V))))
```

Проблемы изменяемого состояния

- Проблемы, связанные с использованием глобальных переменных
- Проблемы, связанные с изменением сложных структур
- Проблемы, связанные с конкурентным доступом к данным

Модели управляемого изменяемого состояния

- Монотонное изменяемое состояние
- Невидимое клиенту изменяемое состояние
- Двухфазный цикл жизни
- Инкапсулированное изменяемое состояние
- Координированное изменяемое состояние

Литература

- **SICP 3.1 Assignment and Local State**
  http://mitpress.mit.edu/sicp/full-text/book/book-Z-H-20.html
- **SICP 3.2 The Environment Model of Evaluation**
  http://mitpress.mit.edu/sicp/full-text/book/book-Z-H-21.html

Почитать на досуге

- Изменяемое состояние: опасности и борьба с ними
  http://fprog.ru/2009/issue1/eugene-kirpichov-fighting-mutable-state/
- **Values and Change**
  http://clojure.org/state
- **Equal Rights for Functional Objects**
  http://home.pipeline.com/~hbaker1/ObjectIdentity.html