Histories about the

programmers

Языки программирования

Прошлое, настоящее, будущее

Программирование…

- Программирование — это то, чем занимаются программисты.

  Программистский фольклор

- "Программирование" — как и "любовь" — одно слово, за которым скрывается бесконечное множество занятий.

  Д. Вейнберг

- Любовь — это все. И это все, что мы о ней знаем.

  Э. *Дикинсон*

Предпосылки развития программирования

- 20-х года XIX века - Ч.Бэббидж – ***высказал мысль о предварительной записи порядка действий машины для последующей автоматической реализации вычислений – программе***
- ***Перфокарты, Жозеф Мари-Жаккар***
- Ада Лавлейс - первый в мире программист. ***теоретически разработала некоторые приемы управления последовательностью вычислений, описала одну из важнейших конструкций современного языка программирования - цикл.***

Предпосылки развития программирования

- Джон Моучли, сотрудник Пенсильванского университета - предложил ***системы кодирования машинных команд с помощью специальных символов***.
- Грейс Мюррей Хоппер, - «третий в мире программист 1-го в мире большого цифрового компьютера» - ***подпрограммы, отладка***
- В 1954 г. группа под руководством Г.Хоппер разработала систему, включающую язык программирования и компилятор, которая в дальнейшем получила название MATH-MATIC.

Мэйнстрим – группа самых популярных в данный момент языков программирования, определяющая основное направление и главную тенденцию в развитии языков программирования

Классификация ЯП

- Языки императивного программирования (Fortran, Algol, Pascal, C, Icon)
- Языки функционального программирования (Lisp, ML, Miranda, Haskell)
- Языки логического программирования (Prolog, PARLOG, Mercury)
- Языки объектно-ориентированного программирования (Simula, Smalltalk, Self, C++, Object Pascal))
- Языки web-программирования (Java, Perl, PHP, Python)
- Сценарные языки (HTML, XML)

Императивное программирование

Программа представляет собой последовательность команд, определяющих алгоритм решения задачи. Основная идея - использование памяти для хранения данных. Основная команда- присвоение, с помощью которой определяется и меняется память компьютера. Программа производит преобразование содержимого памяти, изменяя его от исходного состояния к результирующему.

Императивное программирование

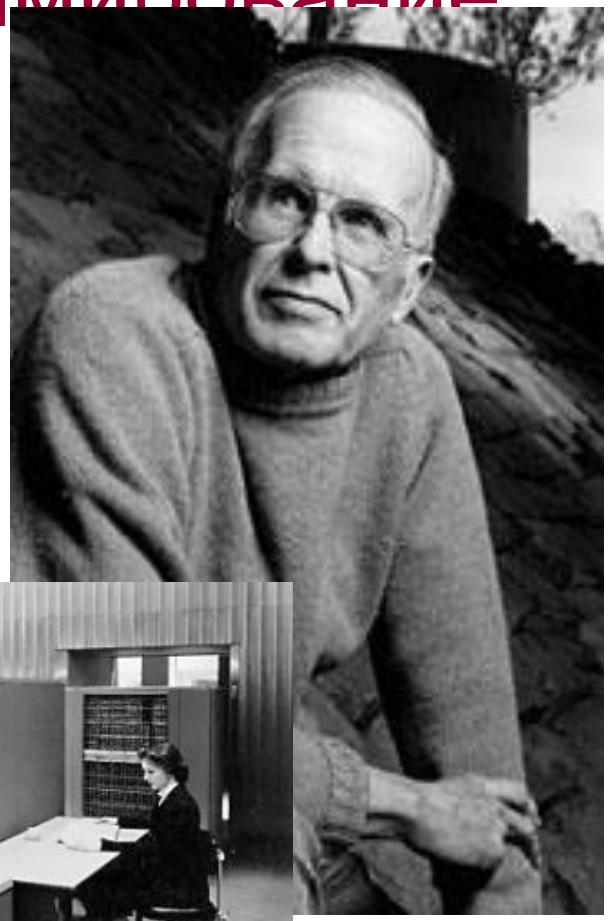
- **Fortran**

Formula Translator

описание – 1954г

реализация – 1957г

разработчик – Джон Бэкус (1925-2007г)



Императивное программирование

- **Cobol**

COmmon Business Oriented Language

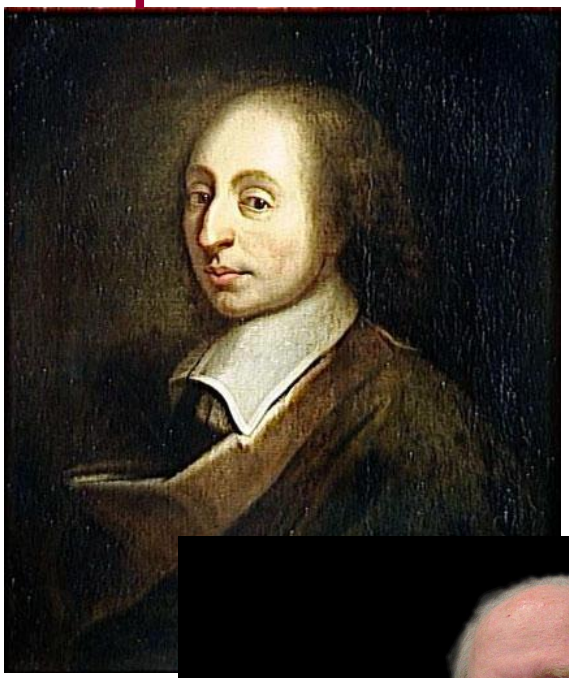
описание и реализация – 1959г

разработчик –

Грейс Мюррей Хоппер (1906-1992г)

Императивное программирование

- **Pascal -Turbo Pascal – Object Pascal-Delphi**
  описание – 1969г
  реализация – 1971г



разработчик – Никлаус Вирт

Императивное программирование

- **Turbo Pascal**

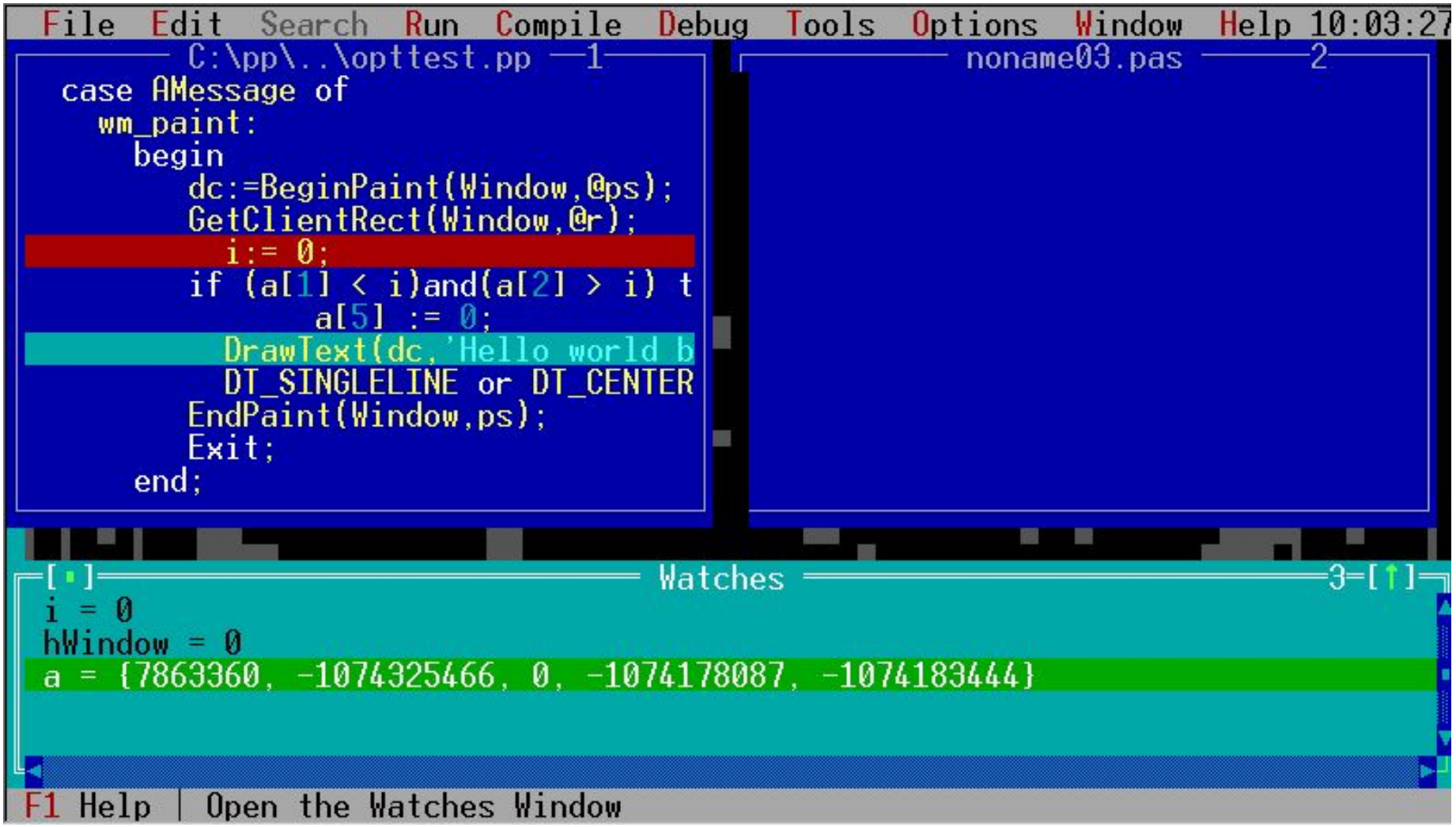
реализация – 1983г

разработчик –

Андерс Хейлсберг

```
Program …
    begin
    ….
    end.
```

Императивное программирование



- Фрагмент программы на языке Free Pascal

Функциональное программирование

- **Программа описывается как математическая функция, зависящая от других математических функций.**

- **Программы являются выражениями, а исполнение программ заключается в вычислении этих выражений**

- **В функциональном программировании не используется память, как место для хранения данных, а, следовательно, не используются промежуточные переменные, операторы присваивания и циклы. Ключевым понятием в функциональных языках является выражение**

Функциональное программирование

- **Lisp**

«Так много дурацких скобок» (Lot of silly parenthesis - "куча глупых скобок" - старинная шуточная расшифровка названия языка Lisp)

реализация – 1958г

разработчик – Джон Маккарти

Функциональное программирование

Задание: Возвратить обращенный список X, если элементы списка X больше соответствующих элементов списка Y, и сам список X в противном случае.

Решение:

```
; Вспомогательные функции:
; 1. пример: (ap '(A B C) '(D E)) => (A B C D E)
(defun ap (x y)
        (cond ((eq x nil)
                (if (eq y nil) nil y))
                (t (if (eq y nil) x (cons(car x) (ap (cdr x)y))))))
; 2. пример: (rev '(A B C)) => (C B A)
(defun rev (x)
        (cond ((eq x nil) nil)
                (t (ap (rev (cdr x)) (cons (car x) nil)))))
; основная функция
(defun f2 (x y)
        (cond
        (
      (and
      (and (> (car x) (car y)) (> (car (cdr x)) (car (cdr y))))
      (> (car (cdr (cdr x))) (car (cdr (cdr y))))
)
(rev x)
        )
(t y)
)
)
```

Функциональное программирование

- **Haskell**

реализация – 1990г

механизм ООП – 1998г

разработчики –

Хаскел Карри,

Саймон Пейтон Джонс .

Логическое программирование

- **Программа – совокупность правил или логических высказываний с причиной и следствием**

Логическое программирование

- **Prolog**

реализация – 1971г
механизм параллельного программирования – 1984г
разработчики –

Алан Колмероер,

Роберт Ковальский

Логическое программирование

```
%trace изменениеБД
/*============================================================
 Пример (напечатать фамилии людей, родившихся до 1975 года,
                и удалить запись о них из цепи) :       */
 domains
   db_selector =  db;бд1;бд2
   пациент = п(фамилия,пол,год_рождения)
   фамилия = string
   пол = мужчина; женщина
   год_рождения = integer
 Predicates
   занесение_пациента_в_БД(фамилия,пол,год_рождения)
   печать_следующего_пациента(db_selector,ref)
   созданиеБД
   распечатка(db_selector)
   изменениеБД(string)
   завершение(string)
```

Логическое программирование

```
Clauses
   занесение_пациента_в_БД(Фамилия,Пол,ГодРождения) :-
   % Занесение объекта "пациент" в цепь "список пациентов"
         chain_insertz(db,"список пациентов",пациент,
                        п(Фамилия,Пол,ГодРождения),_).

   печать_следующего_пациента(СелекторБД,Ссылка):-
      chain_prev(СелекторБД,Ссылка,ССылкаПред),!,
      ref_term(СелекторБД,пациент,ССылкаПред,п(Фамилия,Пол,Год)),
      write(Фамилия,",",Пол,",",Год,"г."),
      nl,
      печать_следующего_пациента(СелекторБД,ССылкаПред).
   печать_следующего_пациента(_,_):-!.
   созданиеБД:-
         занесение_пациента_в_БД("Иванов",мужчина,1971),
         занесение_пациента_в_БД("Петрова",женщина,1965),
         занесение_пациента_в_БД("Сидоров",мужчина,1961),
         занесение_пациента_в_БД("Иваньков",мужчина,1941),
         занесение_пациента_в_БД("Петрунова",женщина,1975),
         занесение_пациента_в_БД("Сидорков",мужчина,1967),
         write("\nСписок пациентов в исходной БД:"),nl,
         распечатка(db),
         db_close(db).
```

Логическое программирование

```
распечатка(СелекторБД):-
        chain_last(СелекторБД,"список пациентов",ПослСсылка),
        ref_term(СелекторБД,пациент,ПослСсылка,
                 п(ФамилияПац,ПолПац,ГодРождПац)),
        write(ФамилияПац,",",ПолПац,",",ГодРождПац,"г."),nl,
        печать_следующего_пациента(СелекторБД,ПослСсылка),
        write("Конец списка пациентов.\n").

  изменениеБД(FName):-
        db_openinvalid(бд1,FName,in_file), % бд1 только на ЧТЕНИЕ
        db_copy(бд1,"mytemp",in_memory),       % копируем ее в память
        db_open(бд2,"mytemp",in_memory),       % и открываем как бд2
          chain_terms(бд2,"список пациентов",пациент,
                      п(Фамилия,_,ГодРождения),Ссылка),
          ГодРождения < 1975,
          write("Удаляется ",Фамилия),nl,
          term_delete(бд2,"список пациентов",Ссылка),
        fail.
  изменениеБД(_):-!.
```

Логическое программирование

```
завершение(FnameC):-
        db_garbagecollect(бд2),
        db_copy(бд2,FNameC,in_file),
        write("\nСписок пациентов в измененной БД:"),nl,
        распечатка(бд2),
        db_close(бд2),
        db_close(бд1),
        write("\nработа завершена.\n").

  Goal
    clearwindow,
    write("Исходная БД (имя файла?) ------->"),
    readln(FName),
    db_create(db,FName,in_file),
    созданиеБД,
    write("\n\nИзмененная БД (имя файла?) ------->"),
    readln(FNameC),
    изменениеБД(FName),
    завершение(FNameC).
%=============================================================
```

Объектно-ориентированное программирование

- **Выделение объектов и связей между ними, построение иерархии объектов с наследованием свойств и методов предка потомкам.**
- **Вычислительная модель – вызов метода или посылка объекту сообщения**

Объектно-ориентированное программирование

- **C++**

Реализация в 1983 г.

Разработчик –

Бьерн Строуструп

на базе языка C
(1972г, Денис Ритчи).

Объектно-ориентированное программирование

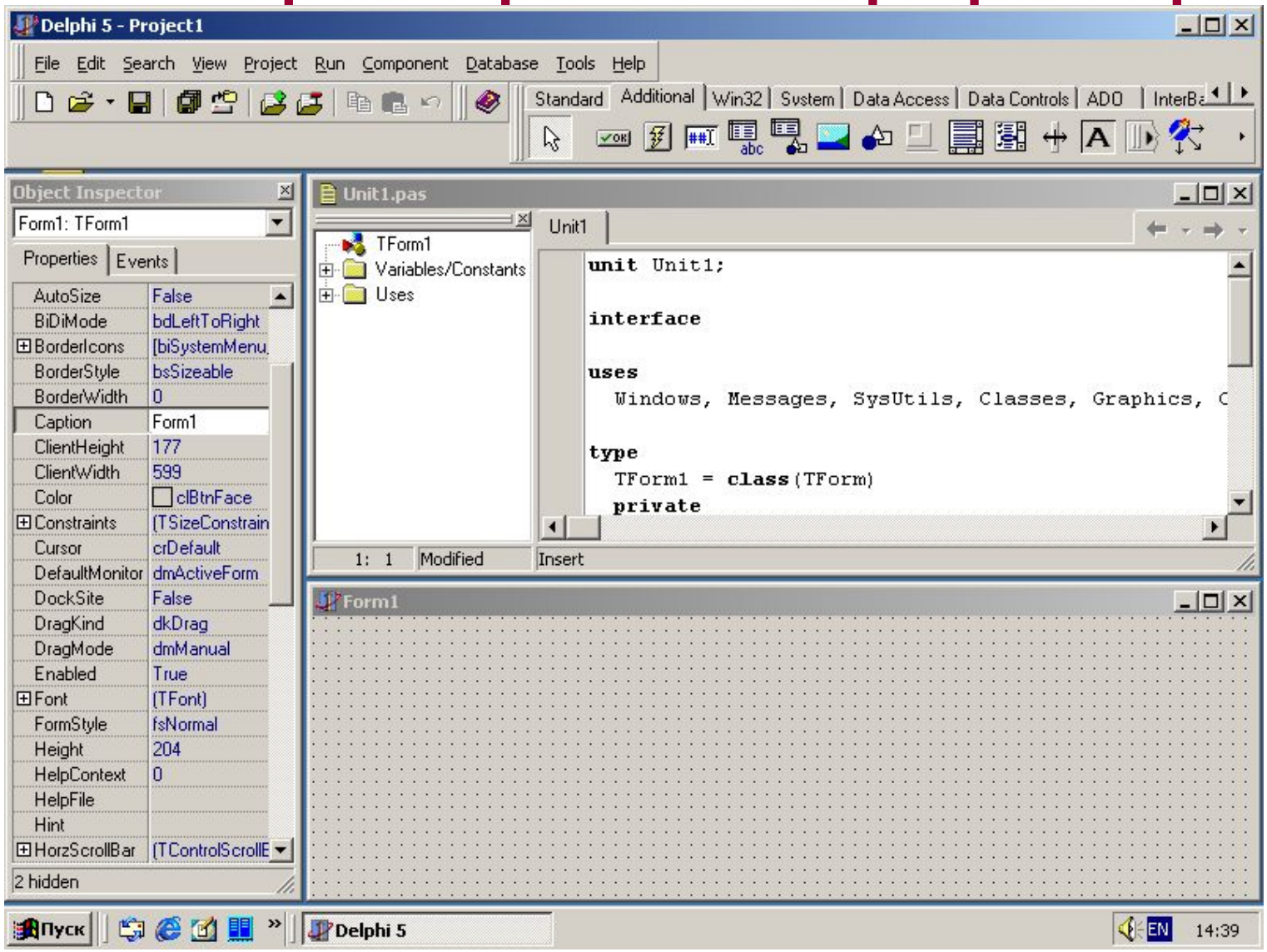
- **Delphi**

реализация – 1995г

разработчик –Чак Язджевски

Язык визуального программирования,
Программа- совокупность модулей, определяющих работу выбранного набора компонент-объектов

Объектно-ориентированное программирование



Рабочее окно языка DELPHI

Языки Web-программирования

- **Языки программирования агентов.**
- **Агенты – сущности, находящиеся в некоторой среде, от которой они получают данные, отражают события, происходящие в среде, интерпретируют их и исполняют команды, воздействующие на среду.**

Языки Web-программирования

- **Python**

реализация – 1991г

разработчик –

Гвидо ван Россум

Программа-совокупность модулей.

Языки Web-программирования

Набор операций над числами - достаточно стандартный как по семантике, так и по обозначениям:

```
>>> print 1 + 1, 3 - 2, 2*2, 7/4, 5%3
2 1 4 1 2
>>> print 2L ** 1000
10715086071862673209484250490600018105614048117055336074437503883703510511249361224931983788156958581275946729175531468251871452856923140435984577574698574803934567774824230985421074605062371141877954182153046474983581941267398767559165543946077062914571196477686542167660429831652624386837205668069376
```

```
>>> print 3 < 4 < 6,      3 >= 5,    4 == 4,    4 != 4  # сравнения
True False True False
```

```
>>> print 1 << 8,   4 >> 2,    ~4    # побитовые сдвиги и инверсия
256 1 -5
```

```
>>> for i, j in (0, 0), (0, 1), (1, 0), (1, 1):
...   print i, j, ":", i & j, i | j, i ^ j  # побитовые операции
...
0 0 : 0 0 0
0 1 : 0 1 1
1 0 : 0 1 1
1 1 : 1 1 0
```

Итого:

Как много языков. Хороших и разных.

Сделай свой выбор!

И не один!