

Бульонков Михаил Алексеевич  
ИСИ СО РАН

**Базовые**

~~Современные~~ методы и  
понятия программирования

(В круге втором)

# Программа

- Лекции - экзамен
- Семинарские занятия
- Практика на ЭВМ – зачёт (необходимое условие для экзамена)

# Рекомендации

- Разговаривать только с лектором
  - исправлять замеченные ошибки лектора
  - задавать вопросы по ходу лекции
- Входить и выходить в любое время, не отвлекая аудиторию
  - Пропуски лекций не фиксируются
- Распечатать конспекты
  - старосты групп могут получить, написав письмо по адресу [mike@iis.nsk.su](mailto:mike@iis.nsk.su)

# Литература

1. Болски М.И. Язык программирования Си. М.: «Радио и связь». 1988.
2. Керниган Б., Ритчи Д., Фбьюэр А. Язык программирования Си. Задачи по языку Си. М.: «Финансы и статистика», 1985
3. Грогоно П. Программирование на языке Паскаль. М. «Мир», 1982
4. Н.Вирт. Алгоритмы + структуры данных = программы. М.: «Мир», 1985.
5. Пярнпуу А.А. Программирование на Алголе и Фортране. М.: «Наука», 1978
6. Пейган Ф. Практическое руководство по Алголу 68. М.: «Мир», 1979.
7. Языки программирования Ада, Си, Паскаль. М.: «Радио и связь», 1989
8. Геллер Д.П., Фридман Д.П. Структурное программирование на АПЛ. М.: «Машиностроение», 1982
9. Сафонов В.О. Автокод Эльбрус. ЛГУ, 1982.
10. Входной язык для системы АЛЬФА-6 (руководство к пользованию). ВЦ СО АН СССР. Новосибирск, 1976.
11. Ахо А., Хопкрофт Дж., Ульман Дж. Структуры данных и алгоритмы. 384 стр., с ил.; 2000, 4 кв.; Вильямс
12. Ахо А., Сети Р., Ульман Дж. Компиляторы: принципы, технологии и инструменты, – М.: «Вильямс». 2001.

# Что такое программирование?

Заставить кого-то сделать что-то, что нам хочется

- Устав ВС
- Кулинарная книга
- ПДД
- Гипноз, реклама, НЛП
- **Программирование ЭВМ**

# Виды программирования (1)

**Прикладное программирование** –  
создание программ для конечных  
пользователей

- Надёжность, устойчивость, «защита от дурака»
- Интуитивность, удобство  
пользовательского интерфейса
- Эффективность
- Гуманитарные аспекты

# Виды программирования (2)

**Системное программирование** – создание программ для создания программ

- Операционные системы
- Система управления базами данных
- Системы программирования
- Системы автоматизации проектирования (САПР)
- Математические пакеты
- ...

# Виды программирования (3)

**Технология программирования –**  
средства организации процесса  
программирования

- Спецификация
- Проектирование
- Отладка, тестирование
- Документирование
- Сопровождение, версияность

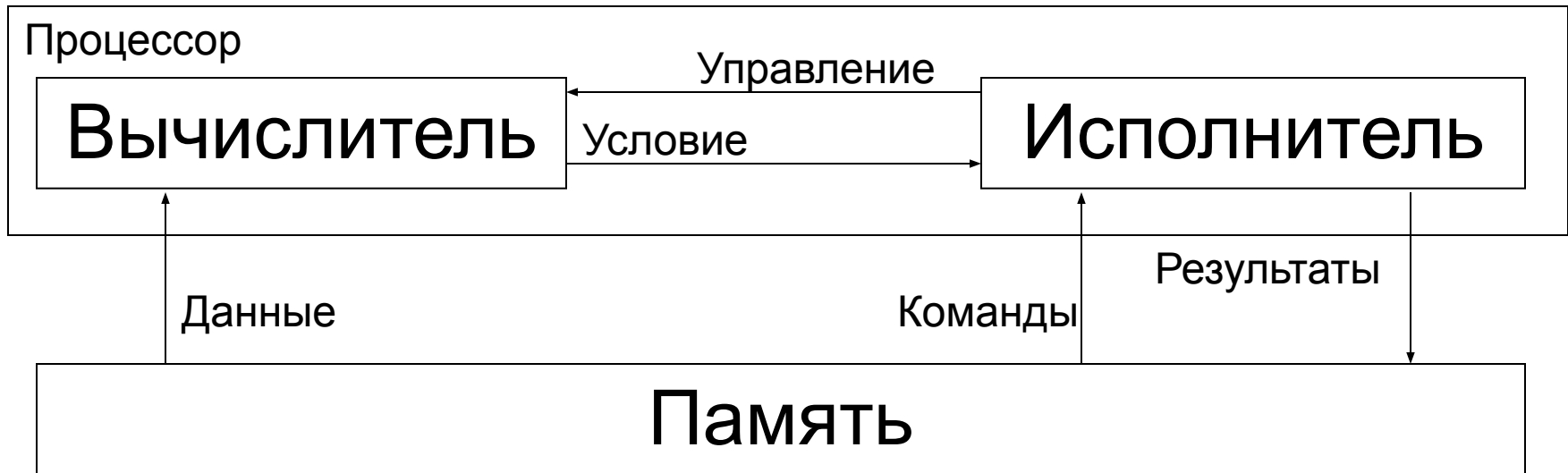


# Виды программирования (4)

**Теоретическое программирование** – программа, как предмет исследования.

- Дискретная математика, кибернетика (структуры данных, алгоритмы)
- Теория вероятности (сложность)
- Алгебра и логика (программа – формула)
- Системный анализ (проектирование)

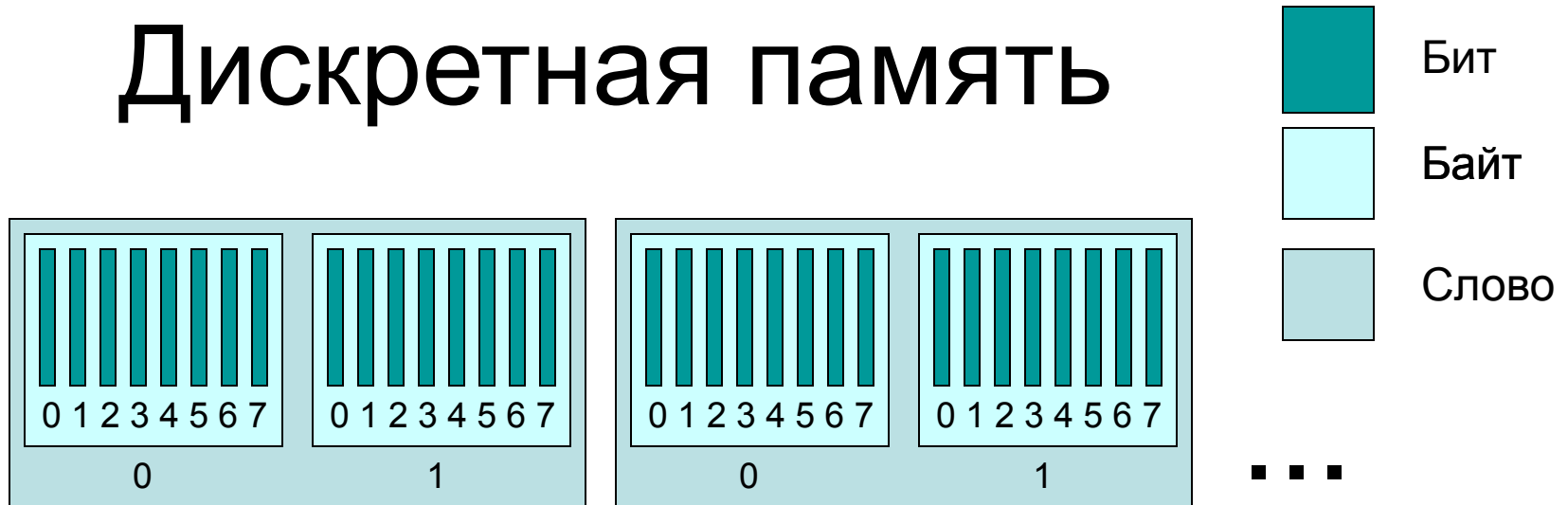
# Логическая модель ЭВМ



Виды команд:

- Арифметические, битовые
- Присваивания, пересылки
- Ввод/вывод
- Управляющие

# Дискретная память



- Бит - элементарная единица хранения информации: 2 значения – 0 и 1 (или 3? Или 10?)
- Байт – минимальная адресуемая группа битов из 8 битов (или 6? или 9? Или 10?)
- Слово – группа из 4-х байтов, с которой может оперировать одна команда (или 2? Или 6?)
- Сегменты, страницы, тэгированная память ....

# Операционная система

- Внутренние функции
  - Управление ресурсами (память, время, бумага)
  - Реакция на сигналы, аварийные ситуации
  - Статистика
- Внешние функции
  - Создание процессов и их взаимодействие
  - Безопасность, полномочия пользователей
  - Файловая система
  - Интерфейс

# Языки программирования: машинные языки

- Программа
  - Хранится в (кодированном сегменте) памяти
  - Представляется последовательностью 0 и 1
  - Интерпретируется процессором

- Пример:

00010010 10000100

Код команды



Операнд

# Языки программирования: ассемблер

Мнемокод:

- Мнемоники команд вместо их двоичного представления
- Мнемонические названия ячеек памяти: как данных, так и команд

```
.MODEL SMALL
    .DATA
b    DW    5
c    DW    3
a    DW    ?
    .CODE
begin MOV  AX,@DATA
      MOV  DS,AX
      MOV  AX,B
      ADD  AX,C
      MOV  A,AX
      MOV  AH,4CH
      INT  21H
      END  begin
```

# Языки программирования: ассемблер

## Достоинства

- **Понимаемость**
- Простота модификации кода (например, вставка команд)
- Простота и прозрачность преобразования (трансляции) в машинный язык

# Языки программирования: макроассемблер

Макросредства:

- Определение макроса – текстового шаблона с параметрами - расширяемость, повышение уровня абстракции
- Библиотеки макросов - переиспользование кода

Определение:

```
MI MACRO C1,C2,CP,MP
MOV ax,C1 I
MUL C2
MOV CP,dx
MOV MP,ax
ENDM
```

Вызовы в программе:

```
MI DI,A,S1,S2
MI S,2,DI,SI
```



# АЯВУ – алгоритмические языки высокого уровня

- Независимость от машины
  - Универсальные языки
  - Специализированные языки

# АЯВУ – алгоритмические языки высокого уровня

Императивные – последовательность  
изменений состояния памяти

- Algol-60, Fortran, COBOL
- Algol-68, Simula-67, PL/I
- Pascal, C, Ada, Modula-2, C++
- Java, C#

# АЯВУ – алгоритмические языки высокого уровня

- Функциональные – Lisp, Scheme, Miranda, ML, Haskell
  - программа представляется совокупностью определений функций
  - $\lambda$ -исчисление

# АЯВУ – алгоритмические языки высокого уровня

- Математическое определение

$$fact(n) = \begin{cases} n * fact(n - 1), & \text{при } n > 0 \\ 1, & \text{иначе} \end{cases}$$

- Scheme

```
(define (fact n)
  (if (> n 0)
      (* n (fact (- n 1)))
      1)))
```

# АЯВУ – алгоритмические языки высокого уровня

- Логические языки – Prolog
  - программа представляется совокупностью фактов (аксиом) и правил вывода
  - математическая логика

# АЯВУ – алгоритмические языки высокого уровня

- Переписывание термов – Refal
  - программа представляет собой терм со вставленными операциями перобразования
  - унификация
  - нормальные алгоритмы Маркова

# Реализация языков программирования

Семантика языка  $L$  сопоставляет  
программе  $p$  функцию

$$L[p] : D \rightarrow D$$

# Реализация языков программирования

*Интерпретатор* языка  $L$  на языке  $I$

$$I[*int*](p, d) = L[p](d)$$



# Реализация языков программирования

*Интерпретатор* языка  $L$  на языке  $I$

- Итеративно выбирает очередную команду в программе на языке  $L$
- Немедленно выполняет *соответствующую* последовательность действий в языке  $I$ .

# Реализация языков программирования

- *Транслятор (компилятор)* с языка  $L_1$  в язык  $L_2$  – это программа *comp* на языке  $I$ , удовлетворяющая следующему свойству: если  $p$  – программа на языке  $L_1$ , то

$$obj = I[comp](p)$$

программа на языке  $L_2$ , такая что для любых данных  $d$

$$L_2[obj](p) = L_1[p](d)$$

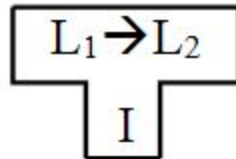
- Компилирует из переведённых команд программу на языке  $L_2$ , но не исполняет её

# T-диаграммы

- Интерпретатор

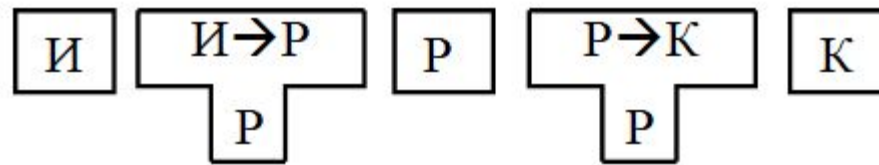


- Транслятор



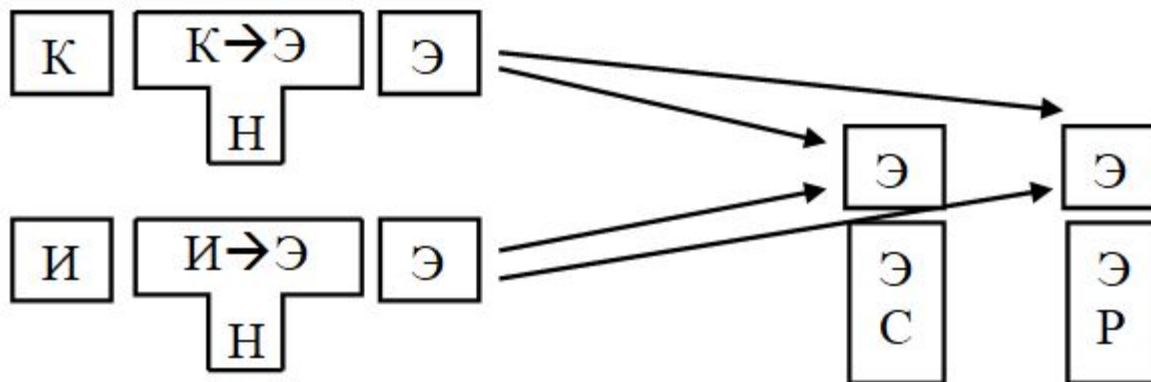
# Многофазная трансляция

- И – итальянский, Р – русский, К – китайский



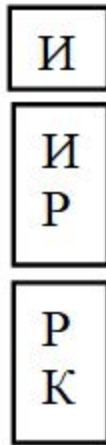
# Общий язык, виртуальные машины

- Э – эсперанто, С – санскрит, Р - русский



# Многоуровневая интерпретация

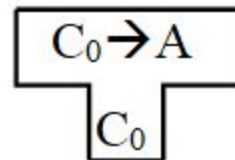
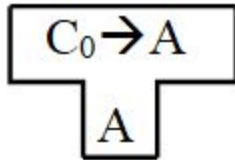
- И – итальянский, Р – русский, К - китайский



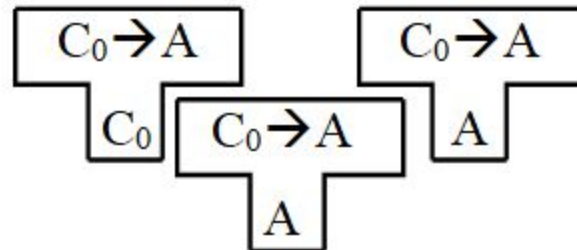
# Метод раскрутки

- $C_0$  = ядро языка (присваивания, простые выражения, безусловный и условный переход, процедуры без параметров)
- $C_1 = C_0$  + сложные выражения
- $C_2 = C_1$  + if, switch
- $L_3 = L_2$  + while, loop
- $L_4 = L_5$  + процедуры с параметрами
- ....
- $C_n$  = полный C
- I = машинный язык

- Инициализация: «с чистого листа»



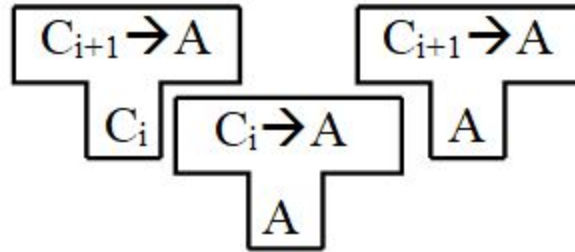
- Первый шаг





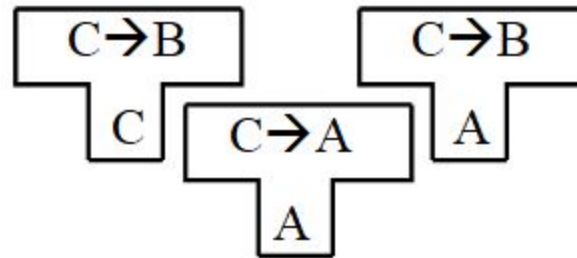
# Метод раскрутки

- Шаг раскрутки



# Кросс-компиляция

- Перенос существующего транслятора на другую машину

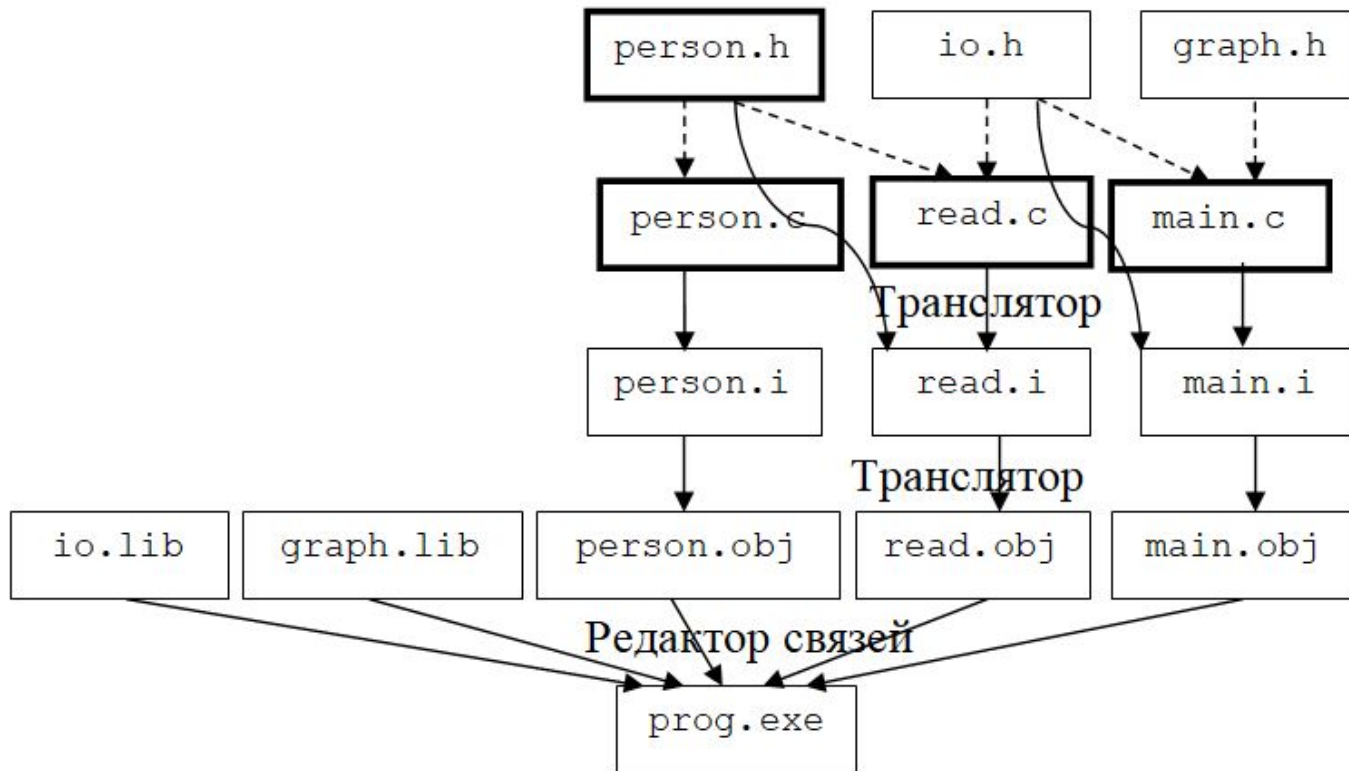


# Система программирования

- **Текстовый редактор** – текст программы (.c, .h)
- **Включаемые файлы** – predetermined macros (.h)
- **Препроцессор** – preprocessing of input texts (.c в .i)
- **Транслятор** – translate program to machine language (.obj)
- **Библиотеки** – «заготовки» programs in machine language (.lib)
- **Редактор связей** – assemble ready program from parts (.exe)
- **Загрузчик** – load program into memory for execution

# Система программирования

- Система построения – build, make



# Система программирования

- **Справочная система** – контекстная помощь
- **Отладка** – пошаговое исполнение, точки останова, просмотр текущих значений переменных...
- **Тестирование** – проверка правильности работы программы на заранее заготовленных примерах
- **Профилирование** – частота исполнения фрагментов программы

# Система программирования

- **Документирование** – комментирование текста программы, создание технической, пользовательской и системной документации
- **Управление хранением исходных текстов** – версионность, совместная работа
- **Средства анализа исходных текстов** – перекрёстные ссылки, проверка выполнения инвариантов, обнаружение потенциальных ошибок исполнения
- **Рефакторинг** – языково-ориентированная модификация текста программы.