Как будем строить язык?

Цели

- Переносимый
- Высокая производительность
- Простой компилятор

Парадигма

- Процедурная (последовательное выполнение команд, объединение их в подпрограммы)

Компиляция

- Однопроходная — то, что используется, обязательно определено выше.

Типизация

- Статическая (тип переменной неизменен)
- Слабая (есть приведение типов)

Язык-надстройка над машинным кодом

- Наличие арифметики указателей
- Каждая операция компилируется в малое количество машинных команд
- Отсутсвуют высокоуровневые возможности

Для какой машины?

- Возможности машины соответствуют средствам языка

Типы памяти

- Ленточная (машина Тьюринга)
- Адресуемая (можно обратиться по произвольному адресу фиксированного размера)
- Ассоциативная (поиск по содержанию)
- Стековая
- Семантическая

Машина с адресуемой памятью

- У любой ячейки есть адрес фиксированного размера
- Есть минимальный адресуемый объём

Машина с ограниченными регистрами

- Каждая операция — не более, чем над N битами
- Набор инструкций ограничен элементарными

Машина с последовательными инструкциями

- Команды выполняются (логически) последовательно.

Инструкции процессоров

- Арифметико-логические операции
- Переход к следующей операции инкрементом её адреса
- Условный прыжок по одному определённому адресу
- Вызов функции с использованием стека и регистра с кодом возврата
- Безусловный переход по определённому адресу

Основные абстракции

- Переменная имеет тип (и соответствующий ему размер)
- Массив — много однотипных переменных
- Адрес ячейки массива =

  адрес начала +

  размер ячейки * номер ячейки
- Нумерация с 0

Объединения

Один адрес, разная интерпретация данных (в том числе разного размера)

Структура

- Структура — разнородные данные
- Адрес элемента структуры =

  адрес начала структуры +

  смещение внутри структуры

Функция

- Входные данные (неопределённое количество)
- Внутренние данные (фиксированное количество)
- Исполняемый код
- Единственная переменная, которая возвращается

Обмен данными между функциями

- Глобальные переменные
- Передача по значению
- Вместо передачи по ссылке - передача указателя

Этапы компиляции и запуска

- Препроцессинг (обработка на уровне текста)
- Компиляция
- Статическое связывание (linking)
- Динамическое связывание и запуск

Область применения

- Ядра операционных систем
- Компиляторы
- Высокопроизводительные приложения

Алфавит языка

- A-Z
- a-z
- 0-9
- ,;.+-^*&=~!/<>(){}[]|%?`":_#

Ключевые слова

- sizeof, typedef, auto, register, extern, static
- char, short, int, void, long, signed, float, double
- struct, enum, union
- do, for, while
- if, else, switch, case, default
- break, continue, goto, restrict

Директивы препроцессора

- #include
- #define
- #ifdef
- #ifndef
- #endif
- #undef

Комментарии

- /*

  комментарий

  */
- //комментарий

Определения

- Тип имя;
-

Операторы и блоки

- ;
- {

  }
- Отступы НЕ влияют на логику программы

Ветвления

if(условие) оператор

else if (условие) оператор

else....

Множественное ветвление из одиночных сравнений

- switch(условие)

  {

  case 1: оператор; break;

  case 2: оператор;

  default:

  }

Цикл с предусловием

- while(условие)

  действие

Цикл с постусловием

- do
- Действие
- while (условие);

for

- for(a=1, b=2; (c>d)&&(e==f); j++)

Указатели

- Специальный тип данных
- Физически — беззнаковое целое число
- & - операция взятия адреса
- * операция разыменования указателя

Указатели на типы

- Тип * - указатель на тип
- Прибавление числа к указателю - прибавление числа, умноженного на размер типа

Массивы

- Массивы — указатели на своё начало

  type a[n];
- a- указатель на начало массива
- Где a[i] ?

  a+ i*sizeof(type)

Структуры

```
struct name
{
int x;
char y;
void *z;
};
```

-
-
-

Указатели на структуры

- struct name var1;
- struct name *var2;
- var1.x=1;
- (*var2).y='c';
- var2->x=3;