Основные сведения о языке Си

Лекция 2

План лекции

- Метаинформация о языке Си
- Идентификаторы и сущности в языке Си
  - Пространства имен, области видимости, связывание, время жизни, продолжительность хранения
- Лексемы языка Си

Метаинформация

- Dennis Ritchie (1941-2011)
  - Язык для разработки ОС UNIX
  - 1969-1973, Bell Laboratories, США
- Стандарты
  - ANSI (C89)
  - C99
  - C11

Что обозначают идентификаторы?

- Пространство имен – это множество идентификаторов, обозначающих сущности одной из категорий:
  - Переменные, функции, типы и enum-константы
  - Метки для goto
  - Тэги структур, объединений и перечислений после struct, union, enum
  - Элементы структур и объединений после операторов . и ->
- Пространства имен могут пересекаться
  - Сущность, обозначаемая идентификатором определяется по контексту
    - Например, идентификатор после -> обозначает элемент struct/union

Область видимости

- *Область видимости идентификатора* – часть текста программы, где он обозначает одну из сущностей:
  - переменную
  - функцию
  - тэг или элемент struct/union/enum
  - тип
  - метку для goto

Виды областей видимости

- «Функция»
  - Только для меток goto
  - Видны из любой точки в теле функции
- «Файл»
  - Вне всех { } и всех прототипов функции
- «Блок» { }
  - От места объявления до конца блока
- «Прототип функции»
  - Внутри объявления функции; например, область видимости x в «void f(int x);»

Вложенные области видимости

- Если идентификатор обозначает сущности C1 и C2 из одного пространства имен, их области видимости O1 и O2 могут пересекаться
- В этом случае $O1 \subset O2$ или $O2 \subset O1$; частичное перекрытие запрещено правилами языка Си
- Если $O1 \subset O2$, то сущность C1 *скрывает* сущность C2 внутри O1
- Если $O2 \subset O1$, то C2 скрывает C1 внутри O2

Связывание идентификаторов

- Связывание идентификатора – это отождествление различных объявлений идентификатора с одной и той же функцией или объектом (= значением в памяти)
  - Иногда в разных областях видимости
    - Необязательно во всех областях видимости
  - Никогда в разных пространствах имён

Виды связывания идентификаторов

Идентификатор обозначает один и тот же объект или функцию

- во всех единицах компиляции --> внешнее связывание
- в одной единице компиляции --> внутреннее связывание
- в своей области видимости --> связывание отсутствует

Правила связывания 1/2

- Разные идентификаторы обозначают разные функции и объекты
- Идентификатор, видимый во всей единице компиляции и объявленный static, имеет внутреннее связывание
- Если идентификатор объявлен extern в O1, static или extern в O2, и O1 $\subset$ O2, то в O1 он имеет такое же связывание как в O2

Правила связывания 2/2

- Объявление функции без static = объявление extern
- Объявление объекта, видимого во всей единице компиляции, без static = объявление extern
- Не имеют связывания идентификаторы объявленные
  - В прототипе функции
  - Без extern внутри блока
- Объявление одного идентификатора с разными связываниями в одной области видимости приводит к undefined behavior

Время жизни объектов

- Время жизни объекта – часть времени исполнения программы, в течение которого для хранения объекта выделены ячейки памяти
- На протяжении времени жизни объект существует в памяти, имеет постоянный адрес и сохраняет присвоенное значение
- Использование объекта после окончания его времени жизни приводит к undefined behavior
- Значение указателя на объект становится неопределенным, когда заканчивается время жизни объекта

Статическое хранение

- Когда: область видимости «файл», либо связывание static или extern
- Время жизни: все время исполнения программы
- Инициализация:
  - однократно до исполнения программы
  - если начальное значение не задано при описании, то память заполняется нулями

Автоматическое хранение

- Когда: область видимости «блок» и связывание не static и не extern

- Время жизни: от места описания или входа в блок с описанием до окончания исполнения из блока
  - Вход во вложенный блок или вызов функции не заканчивает, а приостанавливает исполнение блока
  - На каждом уровне рекурсии создаётся своя копия объекта

- Инициализация:
  - Каждый раз, когда исполнение проходит через место описания объекта
  - Если описание содержит начальное значение, то это значение
  - Иначе – значение каждый раз становится неопределенным

Потоковое хранение

- C11, thread storage duration
- Статическое хранение в памяти потока
  - thread_local int x; // каждый поток имеет статическую копию x

Лексемы языка Си

- Символы-разделители
- Идентификаторы
- Ключевые слова
- Константы, строковые литералы
- Символы операций и скобки

Символы-разделители языка Си

- Пробелы
- Символы табуляции
- Переводы строк
- Комментарии
  - C89: от /* до */
  - C99: C89 и от // до конца строки
  - Эквивалентно одному пробелу

Идентификаторы языка Си

- Последовательность букв и цифр, начинающаяся с буквы
  - Знак подчеркивания _ является буквой
- Идентификатор функции (переменной), которую можно вызвать (использовать) из другой единицы компиляции, называется *внешним* идентификатором
  - Значимыми являются не менее 6 первых символов
  - Верхний и нижний регистр могут не различаться
- Остальные идентификаторы называются *внутренними*
  - Значимыми являются не менее 31 символа
  - Верхний и нижний регистр различаются

Ключевые слова языка Си

- ANSI:
  - auto break case char const continue default do
  - double else enum extern float for goto if
  - int long register return short signed sizeof static
  - struct switch typedef union unsigned void volatile while
- C99:
  - _Bool _Complex inline restrict
- C11:
  - _Alignas alignof _Atomic _Generic _Noreturn _Thread_local

Константы языка Си

- Целые
- Символьные
- С плавающей точкой
- Константы перечислимых типов
- Строковые литералы

Целые константы

- Константа записывается в 8-, 10- или 16-ричной системе счисления и может иметь суффиксы u (или U) и/или l (или L)
- 8-ричная запись состоит из цифр и начинается с цифры 0
- 10-тичная запись состоит из цифр и начинается не с цифры 0
- 16-ричная запись состоит из префикса 0x и послед. цифр 0-9 и букв a-f
  - 16-ричные цифры со значения от 10 до 15 обозначаются буквами от a-f
  - Регистр не учитывается
- Константа получает тип с наименьшим диапазоном, содержащим значение константы
  - 10-тичная без суффикса – первый из int, long int, unsigned long int
  - 8- и 16-ричная без суффикса – первый из int, unsigned int, long int, unsigned long int
  - С суффиксом u или U -- первый из unsigned int, unsigned long int
  - С суффиксом l или L -- первый из long int, unsigned long int
  - С суффиксом ul или UL имеет тип unsigned long int

Символьные константы

- Необязательный префикс L и один или нескольких символов в кавычках ' (например 'x' или L'x')
  - В кавычки ' нельзя брать одну кавычку ' или конец строки
  - Значением константы с одним символом внутри является код этого символа в кодировке, принятой на данной машине
  - Значение константы с несколькими символами может зависеть от реализации
- Константа без префикса имеет тип char
- Константа с префиксом L имеет тип wchar_t (описан в stddef.h)

Escape-последовательность	Запись	Escape-последовательность	Запись
новая строка	\n	обратная наклонная черта	\\
горизонтальная табуляция	\t	Знак вопроса	\?
вертикальная табуляция	\v	одиночная кавычка (single quote) \'	\'
возврат на шаг	\b	двойная кавычка (double quote) \"	\"
возврат каретки	\r	восьмеричный код ooo	\ooo
перевод страницы	\f	шестнадцатеричный код hh	\xhh
сигнал "звонок"	\a		

Константы с плавающей точкой

- Целая часть, десятичная точка, дробная часть, e или E, и порядок (возможно, со знаком), и, возможно, суффикс f, F, l или L
- Целая, дробная часть и порядок -- последовательности цифр
- Целая часть или дробная часть (но не обе вместе) могут отсутствовать
- Десятичная точка или E с порядком (но не обе вместе) могут отсутствовать
- Тип определяется суффиксом
  - F или f -- тип float
  - L или l -- тип long double
  - Без суффикса – тип double

Константы перечислимых типов

- Идентификаторы, объявленные как элементы перечисления enum
- Значения определяются внутри enum, имеют тип int

Строковые литералы

- Необязательный префикс L и последовательность символов, в двойных кавычках (например, "..." или L"…")
  - В двойные кавычки нельзя брать одну двойную кавычку или конец строки
  - В строках можно использовать те же escape-последовательности, что и в символьных константах
- Константа без префикса имеет тип массив char
- Константа с префиксом L имеет тип массив wchar_t
- Значение строки хранится в памяти глобальных переменных (static) и инициализируется заданными символами, за которыми идет '\0'
- Поведение программы, пытающейся изменить строковый литерал, не определено
- Написанные рядом строковые литералы объединяются в одну строку
- После любой конкатенации к строке добавляется символ '\0'
- Конкатенация строк с префиксом и без префикса не определена

Операторы и скобки

- Скобки
  - [ ] { } ( )
- Унарные
  - -- ++ ! ~ & * + -
- Бинарные
  - && || << >> -> . , & ^ | * + - / %
  - = == < > <= >= != += -= /= %= <<= >>= &= |= ^=
- Тернарные
  - ?:
- Другое
  - … ;

Перед делением на лексемы

- Удаление комментариев
- Сворачивание три-графов, если разрешено специальной опцией

Три-граф	ASCII	Три-граф	ASCII	Три-граф	ASCII
??=	#	??(	[	??<	{
??/	\	??)	]	??>	}
??'	^	??!	;	??-	~

- Конкатенация (склеивание) строк, оканчивающихся обратной наклонной чертой \
- Работа препроцессора

Заключение

- Общие сведения о языке Си
  - UNIX -- Dennis Ritchie – 1973 -- Bell Laboratories, США
- Идентификаторы и сущности
  - Пространства имен, области видимости, связывание, время жизни, продолжительность хранения
- Лексика языка Си
  - Единица компиляции
  - Стадии работы компилятора
  - Лексемы
    - Символы-разделители
    - Идентификаторы
    - Ключевые слова
    - Константы, строковые литералы
    - Символы операций и скобки