лекция 25

ПРЕПРОЦЕССОР ЯЗЫКА СИ

План лекции

- Препроцессор как часть компилятора
- Этапы работы и внутреннее устройство препроцессора
- Язык препроцессора языка Си
- Алгоритм работы препроцессора языка Си
- Примеры

Общие сведения о языке Си

- Dennis Ritchie
  - Язык для разработки ОС UNIX
  - 1969-1973, Bell Laboratories, США
- Стандарты
  - ANSI (C89)
  - C99 – //, описание переменных не в начале блока, массивы переменной длины
  - C11 – параллелизм, полиморфизм

Лексика языка Си

- Программа на Си -- одна или несколько *единиц компиляции* (файлов)
- Стадии работы компилятора
  - Формирование *лексем* (в том числе работа препроцессора)
  - Синтаксический анализ
  - Семантический анализ
  - Оптимизация
  - Генерация кода

Препроцессор языка Си

- Препроцессор – это интерпретатор специального языка преобразования текстов
- Препроцессор языка Си – это часть компилятора, преобразующая содержимое единицы компиляции в последовательность лексем языка Си
  - Препроцессор GNU C -- библиотека
- Этапы работы препроцессора языка Си
  - Замена *триграфов*
  - Склеивание строк
  - Удаление комментариев
  - Обработка директив

Триграфы языка Си

- Триграфами языка Си называются следующие последовательности символов, начинающиеся с ??

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

Пример

```
??=include <stdio.h>                          /* #          */
int main(void)
??<                                           /* {          */
    char n??(5??);                            /* [ и ]      */
    n??(4??) = '0' - (??-0 ??' 1 ??! 2); /* ~, ^ и|    */
    printf("%c??/n", n??(4??));               /* ??/ = \    */
    return 0;
??>
```

Склеивание строк, удаление комментариев

- Строка единицы компиляции, заканчивающаяся обратной наклонной чертой \, соединяется со следующей строкой

- Символы единицы компиляции, образующие комментарий на языке Си, не включаются в выходную последовательность лексем языка Си

Пример

- //Будет ли исполнена следующая строка??????????????/
  a++;

- ??/ будет проинтерпретирован как '\' в конце строки и продлит комментарий на следующую строку
- a++ будет воспринято как комментарий

Директивы препроцессора Си

- Директивы препроцессора языка Си записываются на специальном языке
  - Грамматика языка препроцессора языка Си отличается от грамматики языка Си
- Строки единицы компиляции на языке препроцессора языка Си начинаются с символа #
- Все остальные строки единицы компиляции являются входными данными для препроцессора языка Си

Внутреннее устройство препроцессора

- Вход -- последовательность байтов в строках единицы компиляции
- Выход -- последовательность лексем для компилятора
- Таблица макросов
  - Состояние препроцессора изменяемое директивами препроцессора
  - Идентификатор макроса и цепочка лексем «правая часть макроса», на которую препроцессор заменяет этот идентификатор в процессе раскрытия
  - Заполняется по ходу работы препроцессора в соответствии с директивами

Константы, макросы, операции

- Константы
  - Целые числа, символы, строки
  - Записываются как соотв. константы в Си
- Макросы
  - Записываются как идентификаторы в Си
  - Предопределенные макросы __FILE__, __LINE__, __FUNCTION__, __TIME__
- Арифметические, побитовые, логические операции
  - Записываются как в языке Си
- Проверка наличия определения макроса
  - defined имя_макроса

Константные выражения

- Константным выражением языка препроцессора языка Си называется выражение, построенное по правилам языка Си из скобок (и) и констант, макросов и операций языка препроцессора языка Си

Ключевые слова препроцессора Си

- define — определить макрос препроцессора
- undef — отменить текущее определение макроса
- include — вставить текст из указанного файла
- if — передать строки до соотв. elif/else/endif на компиляцию, если выражение истинно
- ifdef — то же, что if defined
- ifndef — то же, что if !defined
- else — передать строки до соотв. endif на компиляцию, если соотв. выражение ложно
- elif — то же, что else if
- endif — конец ветки условной компиляции
- line — сообщить компилятору указанные номер строки и имя файла вместо фактических
- error — завершить работу с ошибкой
- pragma — добавить в выходной поток лексем лексемы, зависящие от компилятора

Препроцессор Си

- Пока входной поток байтов не пуст
  - С = строка, полученная заменой триграфов, склейкой строк и удалением комментариев
  - Если С начинается с #, то обработать директиву
  - Иначе для каждой лексемы Л в С
    - Если Л является макросом с определением Х, то положить во входной поток байтов "#undef Л <конец строки> Х #define Л Х <конец строки> "
      - Включает обработку параметров макроса – след. слайд
    - Иначе положить Л в выходной поток лексем

Обработка директив

- #define *макрос* [(*парам* [, *парам …*])] *правая часть*
  - Добавить в таблицу макросов определение
- #undef *макрос*
  - Удалить определение
- #if *константное выражение*
  текст 0
  #else
  текст 1
  #endif
  - Добавить во входной поток байтов текст 0, если выражение истинно, или текст 1, если выражение ложно
- #error сообщение
  - Завершить компиляцию с сообщением

Обработка директив

- #include < *байты* >
- #include " *байты* "
  - Заменить #include на текст из файла с таким именем
  - Две формы отличаются набором директорий, в которых препроцессор ищет файл -- для < > дополнительно просматриваются директории с файлами заголовков, поставляемых вместе с ОС
- #include *последовательность лексем*
  - Последовательность лексем должна раскрываться либо в < *байты* >, либо в " *байты*

Макросы с параметрами

- #define M(a,b,…) nextM
  - Добавить в таблицу макросов замену M(a,b,…) -> nextM
  - Если раскрываем M(A,B,…), то a, b, и т.д. в nextM будет заменено на A, B, и т.д.
  - a##b – объединить a и b в одну лексему
  - #a – строковая константа, значением которой является a
  - #define max(x,y) ((x)<(y)?(x):(y))

Заключение

- Препроцессор как часть компилятора
- Этапы работы и внутреннее устройтсво препроцессора
- Язык препроцессора языка Си
- Алгоритм работы препроцессора языка Си
- Примеры

Препроцессор Си

- #if
  - #if <условие1>
    - <код1>
  - #elif <условие2>
    - <код2>
  - #else
    - <кодN>
  - #endif
  - Вычисляем условия по порядку
  - Когда получаем истину, помещаем во входной поток блок кода до следующей директивы
  - #elif и/или #else могут отсутствовать
  - Можно вложенные #if

Условие:

- defined макрос
- Арифметические, логические, побитовые операции, скобки
- Символы, целые числа
- Макросы, раскрывающиеся в символы или целые числа