37. Битовые операции. Битовые поля.
План ответа:
1. битовые операции: сдвиг влево, сдвиг вправо, битовое «НЕ», битовое «И», битовое
«исключающее ИЛИ», битовое «ИЛИ» и соответствующие им операции составного присваивания;
2. использование битовых операций для обработки отдельных битов и последовательностей битов;
3. различие между битовыми и логическими операциями;
4. битовые поля: описание, использование, ограничения использования.

1. битовые операции: сдвиг влево, сдвиг вправо, битовое «НЕ», битовое «И», битовое «исключающее ИЛИ», битовое «ИЛИ» и соответствующие им операции составного присваивания;

Битовые операции — логические операции, проводимые над каждым битом фигурирующих операндов.
Операнды при этом имеют целый тип и одинаковый размер.
Язык Си поддерживает все битовые операции:

Операция	Название	Нотация	Класс	Приоритет	Ассоциат.
~(унар.)	Побитовое «НЕ»	~X	Префиксные	15	Справа налево
<<	Сдвиг влево	X << Y	Инфиксные	11	Слева направо
>>	Сдвиг вправо	X >> Y	Инфиксные	11	Слева направо
&	Побитовое «И»	X & Y	Инфиксные	8	Слева направо
^	искл. «ИЛИ»	X ^ Y	Инфиксные	7	Слева направо
|	Побитовое «ИЛИ»	X | Y	Инфиксные	6	Слева направо

Инфиксные 2 слева направо
<<= Присваивание со сдвигом влево X <<= Y
>>= Присваивание со сдвигом вправо X >>= Y
&= Присваивание с побитовым «И» X &= Y
^= Присваивание с побитовым исключающим «ИЛИ» X ^= Y
|= Присваивание с побитовым «ИЛИ» X |= Y

Битовые операции применимы только к целочисленным переменным.
Битовые операции обычно выполняют над беззнаковыми целыми, чтобы не было путаницы со знаком.

2. использование битовых операций для обработки отдельных битов и последовательностей битов;

2.1. & -- поразрядная конъюнкция.
На каждом бите проходит конъюнкция/логическое И.

Проверка битов операнда на установление:
Формируем mask, в которой на проверяемые биты установлена единица, а на остальные биты — ноль.
В результате op&mask останутся единицы только в значениях проверяемых с помощью mask битов там, где они установлены в op.

Установка битов операнда в ноль (сброс флагов)
Формируем mask, в которой нужные биты установлены в ноль, а остальные — в один.
В результате op&mask появятся нули только в устанавливаемых с помощью mask битов op, остальные биты останутся прежними.

2.2. | -- поразрядная дизъюнкция.
На каждом бите происходит дизъюнкция/логическое ИЛИ

Установка битов операнда в единицу (установка флагов):
Формируем mask, в которой нужные биты установлены в единицу, а остальные – в ноль.
В результате op|mask появятся единицы только в устанавливаемых с помощью mask битов op, остальные биты останутся прежними.

Проверка битов операнда на сброс проводится аналогично пункту 2.1.2

2.3. ^ -- поразрядная симметрическая разность.
На каждом бите операндов происходит симметрическая разность/сумма по модулю 2/логическое ИСКЛЮЧАЮЩЕЕ ИЛИ/АНТИЭКВИВАЛЕНТНОСТЬ.

Смена значений битов операнда
Формируем mask, в которой нужные биты установлены в единицу, а остальные – в ноль
В результате op^mask изменятся значения только в устанавливаемых с помощью mask битов op.

2.4. ~ -- поразрядная инверсия/логическое НЕ.
Каждый бит операнда изменяет своё значение. При этом результат – обратный код числа op.

2.5. Логические сдвиги числа на некоторое число бит.
Сдвиг влево.
Синтаксис: op<<n_bit.
Иначе говоря,
Все биты сдвигаются на n_bit позиций влево, освободившиеся биты заполнятся нулями, а выдвинутые биты уничтожаются.

Сдвиг вправо.
Синтаксис: op>>n_bit.
Все биты сдвигаются на n_bit позиций вправо, освободившиеся биты заполнятся нулями, а выдвинутые биты уничтожаются.

3. различие между битовыми и логическими операциями;
При проведении логических операций все ненулевые числа (как бы) приводятся к числу (-1), представимому всеми единицами в двоичной записи. Поэтому 00100100&10010010 == 00000000 != (некая единица) == 00100100&&10010010.

4. битовые поля: описание, использование, ограничения использования.
4.1. Стандартный вид объявления:

```
struct имя_структуры
{
   тип имя1: длина;
   тип имяN: длина;
};
```

Битовые поля должны объявляться как целые, unsigned или signed.

4.2. Представление в памяти:
обычным целым числом, размера не менее суммы размеров полей битового поля (контролируется программистом)

4.3. Необходимые действия над битовыми полями:
Извлечение поля из структуры включает следующую последовательность действий:
Конъюнкция с маской битового поля (на битах поля единицы, в остальных местах нули);
Побитовый сдвиг вправо.
Сборка одного числа из битовых полей:
Обнуление числа
Установка битов поля
Сдвиг влево
Замена битового поля
Обнуление нужных битов с помощью маски
Заполнение нужных битов (со сдвигом) поразрядной дизъюнкцией

4.4. Замечание.

Битовые поля применяются тогда, когда нужно компактно записать мн ожество разнообразной перечислимой информации, экономя при этом место в памяти, н о не гонясь за сверхвысокой производительностью алгоритма вопреки стремлению напи сать более читабельный код.

Если требуется более высокая производительность, применяются обыч ные числа, при этом отдельно прописываются всевозможные маски и значения перечисл имых параметров.