Training 9
ビット操作

株式会社イーシーエス　出版事業推進委員会

Lesson1　ビット演算

Point◆◇ビット演算を理解しよう!!

ビット演算はビット単位でのデータの操作やデータの抽出に使われます。
任意のビットを変更できるため大変便利です。しっかりと理解しましょう。
また、論理演算子や比較演算子と記号が似ています。しっかりと区別して覚えましょう。

※ビット演算の入力と出力の関係については巻末資料参照

【問題1】 次の演算結果を 2 進数で答えなさい。(表現範囲は演算に使用するビットの数まで)

① `1001 & 1100`

② `10011 & 10101`

③ `1001 | 1100`

④ `10100 | 10001`

⑤ `1001 ^ 1110`

⑥ `100110 ^ 010011`

⑦ `~1001`

⑧ `~11010`

【問題2】 次の演算結果を 2 進数で答えなさい。

①②⑤⑥は符号なし③④⑦⑧は符号ありとする。(符号ビットは最上位ビット)
ただし、ビットの表現範囲は演算に使用するビットの数までとする。

① `0010 << 2`

② `10001 << 3`

③ `010010 << 2`

④ `10011011 << 4`

⑤ `0100 >> 2`

⑥ `10101 >> 3`

⑦ `001101 >> 2`

⑧ `10101100 >> 4`

【問題3】 次の演算結果を 2 進数で答えなさい。

②は符号なし、⑥は符号ありとする。(符号ビットは最上位ビット)
ただし、ビットの表現範囲は演算に使用するビットの数までとする。

① `1100 & 0100 | 1001`

② `1000 | 1110 >> 2`

③ `1001 ^ 0101 | 1001`

④ `1010 & 1110 ^ 0101`

⑤ `0101 | 1000 & 0110 ^ 1001`

⑥ `~1001 & ( 0011 | 1011 ) >> 1`

【問題4】 次の x の演算結果を 16 進数で答えなさい。

①
```
char x = 0xB4
x &= 0x76
```

②
```
unsigned short x = 0xCA67;
x &= 0x64ED;
```

③
```
unsigned char x = 0x2C;
x |= 0x35;
```

④
```
short x = 0x3A96;
x |= 0x9C19;
```

⑤
```
char x = 0x9C;
x ^= 0x73;
```

⑥
```
unsigned short x = 0x9CA3;
x ^= 0x7635;
```

⑦
```
unsigned char x = 0x93;
x <<= 2;
```

⑧
```
char x = 0xE6;
x >>= 4;
```

【問題5】 以下の条件を満たすビット演算を□に埋めなさい。

変数は①から④は signed char 型の変数、⑤は unsigned char 型の変数とする。
また、ビットは最下位ビットを 0 ビット目として、最上位ビットを 7 ビット目とする。

変数のビット番号	7	6	5	4	3	2	1	0

① 変数 a の 0 ビット目から 3 ビット目の情報をそのままにして、4 ビット目から 7 ビット目の情報を 0 にする。
```
a □= 0x0f;
```

② 変数 b の下位 4 ビットの情報を上位 4 ビットに移動して、下位 4 ビットに 1 をたたせ、変数 c に代入する。
```
c = ( b □ 4 ) □ 0x0f;
```

③ 変数 d の 5 ビット目の情報を 0x00 または 0x01 で取得し、変数 e に代入する。
```
e = ( d □ 5 ) □ 0x01;
```

④ 初期値が 0 の変数 f があり、実行するたびに f の値が 0→1→0→1→0→1･･･となるようにする。
```
f □= 0x01;
```

⑤ 変数 g の下位 4 ビットを変数 i の上位 4 ビットに代入し、変数 h の上位 4 ビットを変数 i の下位 4 ビットを代入する。
```
i = ( g □ 4 ) | ( h □ 4 );
```

Lesson2　演算子の優先順位

Point◆◇演算子の優先順位を理解しよう!!

C 言語の全ての演算子は処理される優先順位が決まっています。
複数の演算子を使用した処理には、優先順位を意識してプログラムを作りましょう。

※演算子の優先順位については巻末資料参照

【問題 1】次の x の演算結果を 10 進数で答えなさい。

①
```
int x = 0;
x = 60 - 15 * 3;
```

②
```
int x = 0;
x = 10 * 4 / 2;
```

③
```
int x = 0;
x = 30 & 15 * 3;
```

④
```
int x = 0;
x = 10 | 4 + 15;
```

⑤
```
int x = 15;
x = 20 + 10 * x >> 2;
```

⑥
```
int x = 4;
x = 6 | 8 + 5 * x << 3;
```

⑦
```
double x = 2.0;
double y = 3.0;
x = (double)(int)( pow( x + 2.5 , ++y ) / 4 ) + 0.5;
```

※標準ライブラリ double pow(double x, double y)は引数「x」の「y」乗を計算するライブラリ

⑧
```
int x = 30;
double y = 10.5;
x = 11 * abs( x - (int)pow( y , 2.0 ) ) >> 2 ;
```

※標準ライブラリ int abs (int x);は引数 x の絶対値を int 型で計算するライブラリ

解答

Training9 ビット操作

Lesson1 ビット演算

問題１	①1000	②10001	③1101
	④10101	⑤0111	⑥110101
	⑦0110	⑧00101	

問題２	①1000	②01000	③001000
	④10110000	⑤0001	⑥00010
	⑦000011	⑧11111010	

問題３	①1101	②1011	③1101
	④1111	⑤1101	⑥0100

問題４	①0x34	②0x4065	③0x3D
	④0xBE9F	⑤0xEF	⑥0xEA96
	⑦4C	⑧FE	

問題５	①a &= 0x0f;	②c = (b << 4) |(^) 0x0f;
	③e = (d >> 5) & 0x01;	
	④f ^= 0x01;	
	⑤i = (g << 4) | (h >> 4);	

【解説】
データ型の決まっているデータ先頭ビットは signed(符号あり)の場合、符号ビットとしてとらえます。
signed の符号ビットが 1 の時、先頭ビット以前には 1 が存在します。
unsigned(符号なし)の場合、先頭ビットに 1 が存在していても先頭ビット前は 0 が存在します。

Lesson2 演算子の優先順位

問題１	①15	②20	③12	④27
	⑤42	⑥230	⑦102.5	⑧220

【解説】問題１ ⑦
演算子の優先順位が高い順に演算していけば x の値を求めることができます。(演算順序は下記参照)

1. ++y を演算する
   演算結果 x = (double)(int)(pow(x + 2.5 , 4.0) / 4) + 0.5
2. x + 2.5 を演算する。
   演算結果 x = (double)(int)(pow(4.5 , 4.0) / 4) + 0.5
3. pow(4.5 , 4.0)を演算する。
   演算結果 x = (double)(int)(410.0625 / 4) + 0.5
4. (410.0625 / 4)を演算する。
   演算結果 x = (double)(int)102.515625 + 0.5
5. (int)102.515625 を演算する。
   演算結果 x = (double)102 + 0.5
6. (double)102 を演算する。
   演算結果 x = 102.0 + 0.5
7. 102.0 + 0.5 を演算する。
   演算結果 x = 102.5

【解説】問題１ ⑧
⑦と同様に優先順位が高い順に演算していけば x の値を求めることができます。(演算順序は下記参照)

1. pow(y , 2.0)を演算する
   演算結果 x = 11 * abs(x -(int)110.25) >> 2
2. (int)110.25 を演算する。
   演算結果 x = 11 * abs(x -110) >> 2
3. x - 110 を演算する。
   演算結果 x = 11 * abs(-80) >> 2
4. abs(-80)を演算する。
   演算結果 x = 11 * 80 >> 2
5. 11 * 80 を演算する。
   演算結果 x = 880 >> 2
6. 880 >> 2 を演算する。
   演算結果 x = 220

巻末資料

演算子の優先順位

順位	演算子
1	関数呼び出し ( ) [ ] -> . ++(後置) --(後置)
2	! ~ ++(前置) --(前置) *(間接参照) &(アドレス) sizeof
3	キャスト
4	*(乗算) / %
5	+ -
6	<< >>
7	< > <= >=
8	== !=
9	&(ビット積)
10	^
11	|
12	&&
13	||
14	?:(条件演算子)
15	= += -= *= /= %= &= |= ^= <<= >>=
16	,(順次演算子)