# Chapter 2 Solution

1. （整数的表示）在 x86-64 机器上，运行下列代码，结果为：

```
int main()
    unsigned int A = 0x11112222;
    unsigned int B = 0x33336666;
    void *x = (void *)&A;
    void *y = 2 + (void *)&B;
    unsigned short P = *(unsigned short *)x;
    unsigned short Q = *(unsigned short *)y;
    printf("0x%x", P + Q);
    return 0;
}
```

A. 0x4444
**B. 0x5555**
C. 0x7777
D. 0x8888

【解】小端机器，从 A 取得 0x2222，从 B 取得 0x3333，因此结果是 0x5555

2. （整数的模运算性质）对于计算机而言，除法往往比乘法慢得多，所以编译器有的时候会把除法操作转变成一次乘法操作和一系列的移位操作。为了简化，这里只考虑这一次乘法。有如下代码：

```
unsigned int a; scanf("%u", &a);
unsigned int x = _____u; // I AM THE MAGIC NUMBER!!!
unsigned int b = a / 3;
unsigned int c = a * x;
```

这一段代码非常的神奇，只要 a 是 3 的倍数，那么 b 和 c 的值就是一样的。那么横线上的 MAGIC NUMBER 应当是：
A. 715827882
B. 1431655765
C. 1431655766
**D. 2863311531**

【解】无符号运算类似于模 2^32，因此计算这 4 个数乘以 3 再对 2^32 取模，只有最后一个算出来是 1

3. （隐式类型转换）下列代码的运行结果为：

```
unsigned int ux = 1;
int y = 2;
printf("%d, ", ux - 2 < y );
printf("%d\n", (!!ux) - 2 < y );
```

A. 0, 0

**B. <u>0, 1</u>**

C. 1, 0

D. 1, 1

【解】x - 2 < y 中，x 是无符号的，因此 x - 2 是无符号的 2^32 - 1，再与有符号的 2 进行比较的时候，2 转换成无符号的 2，因此是 0；(!!x) - 2 < y 中，!运算得到的是有符号数，因此!x 是有符号的 0，从而!!x 是有符号的 1，1 – 2 得到-1， 因此结果是 1

4. （运算优先级）表达式(1 + 4 >> 2) * (1 << 4 / 2)的运算结果是：

A. 16

B. 8

**C. <u>4</u>**

D. 2

【解】(1+4>>2)*(1<<4/2) = ((1+4)>>2)*(1<<(4/2)) = 1 * 4 = 4

5. （运算等价性）有如下的定义：

```
int x = ___;
int y = ___;
unsigned int ux = x;
```

在 x86-64 的机器上，判断如下说法的正确性：

(1) 如果 x > y，那么 ux > y;

**(2) <u>如果 x = 3 且 y = 1，那么表达式(ux > y) > -128 的值与 signed(ux > y) > -128 的值相同;</u>**

**(3) <u>如果 x = 0 且 y = 2，那么表达式((x-y) < 0) + 0u == 0u 的值与 ((x-y) < 0) + 0 == 0 的值相同;</u>**

(4) 如果 x = 0 且 y = 2，那么表达式((ux-y) < 0) + 0u == 0u 的值与 ((x-y) < 0) + 0u == 0u 的值相同;

**(5) <u>表达式 x ^ y ^ (~x) - y 的值与 y ^ x ^ (~y) - x 的值相同;</u>**

(6) 如果 x = 0x80000000，那么表达式 x < 0 和 0x80000000 < 0 的值相同。

【解】

第一问，反例为 x = 1, y = -1

第二问，布尔运算结果都是有符号的

第三问，内层 x-y 得有符号的-2，-2<0 得有符号的 1，1+0u 中 1 转成无符号，因此是 1u，1u==0u，结果为 0；1+0 得 1，1==0，因此结果也为 0

第四问，内层 ux-y 得无符号的 2^32-2，无符号的 2^32-2 和有符号的 0 比，得有符号的 0（布尔运算结果都是有符号的），0+0u==0u 得 1

第五问，注意优先级关系，先算-再算^

第六问，前者 1 后者 0

6. （运算等价性）补充下列代码，完成 A+B Problem：

```
int x, y;
scanf("%d%d", &x, &y);
int a = x ^ y;
int b = (x & y) << 1;
int ans = ((a ^ b) << __*A*__) + ((a & b) << __*B*__);
printf("%d", ans);
```

A. 0

B. 1

【解】x ^ y 其实是能得到不带进位的结果，x & y 得到进位的情况

7. （浮点数表示）对于 1 符号+3 阶码+4 小数的浮点数(-1)^S*M*2^E，完成下表

| 描述 | 二进制表示 | M（分数） | E | 值 |
|---|---|---|---|---|
| 负零 | **10000000** | **0** | ********** | -0.0 |
| **************** | 01000101 | **21/16** | **1** | **21/8** |
| 最小的非规格化负数 | **10001111** | **15/16** | **-2** | **-15/64** |
| 最大的规格化正数 | **01101111** | **31/16** | **3** | **31/2** |
| 一 | **00110000** | **1** | **0** | 1.0 |
| **************** | **01010110** | **11/8** | **2** | 5.5 |
| 正无穷 | **01110000** | *********** | ********** | ***** |

【解】注意非规格化表示的时候阶码的问题

8. （浮点数表示）float 的格式为 1 符号+8 阶码+23 小数，则下列程序的输出结果是：

```
for (int x = 0; ; x++) {
    float f = x;
    if (x != (int)f) {
        printf("%d", x);
        break;
    }
}
```

A. 死循环

B. 4194305 (2^22 + 1)

C. 8388609 (2^23 + 1)

**D. 16777217 (2^24 + 1)**

【解】表示 16777217 需要 25 位，除了前导 1 以外还要 24 位，float 无法表示

9. （浮点数表示）float 的格式为 1 符号+8 阶码+23 小数，则下列程序的输出结果是：

```
int x = 33554466; // 2^25 + 34
int y = x + 8;
for ( ; x < y; x++) {
    float f = x;
    printf("%d ", x - (int)f);
}
```

输出：**2, -1, 0, 1, -2, -1, 0, 1,**

【解】注意 Round to even

10. （浮点数运算的等价性）判断下列说法的正确性

(1) 对于任意的单精度浮点数 a 和 b，如果 a > b，那么 a + 1 > b。
【正确】

(2) 对于任意的单精度浮点数 a 和 b，如果 a > b，那么 a + b > b + b。
【错误】取 a = INF、b = FLT_MAX 即可。

(3) 对于任意的单精度浮点数 a 和 b，如果 a > b，那么 a + 1 > b + 1。
【错误】错误。取 a = 16777220、b = 16777218 即可。这里要特别注意取 a = INF、b = FLT_MAX 不能构成反例，因为 b + 1 因计算精度无法到达 INF

(4) 设 dx 是一个双精度浮点数，且 dx 不是 INF 或 NAN，并且定义 float fx = dx，那么对于同一个表达式，相比于数学上的真实结果，用 fx 进行计算一定不比用 dx 进行计算更加精确。
【错误】dx 是最小的正的非规格化双精度，那么 dx - dx/2 - dx/2 != 0 但是 fx - fx/2 - fx/2 == 0

(5) 对于任意的双精度浮点数 d，如果 d < 0，那么 d * d > 0。
【错误】d 取最大的非规格化负数

(6) 对于任意的双精度浮点数 d，如果 d < 0，那么 d * 2 < 0。
【正确】

(7) 对于任意的双精度浮点数 d，d == d。
【错误】NAN != NAN

11. （浮点数的类型转换）设 int x; float f; double d; 并且 f 和 d 都不是 NaN，判断下列说法的正确性：

(1) x == (int)(float) x

**(2) x == (int)(double) x**

**(3) f == (float)(double) f**

(4) d == (double)(float) d

【解】 1.4.精度不足