Chapter 2

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

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

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. 0, 1
C. 1, 0
D. 1, 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. 4
D. 2

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) 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;

(3) 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;

(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) 表达式 x ^ y ^ (~x) - y 的值与 y ^ x ^ (~y) - x 的值相同;

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

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. _____
B. _____

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			**********	-0.0
**********	01000101			
最小的非规格化负数				
最大的规格化正数				
一				1.0
**********				5.5
正无穷		**********	**********	*****

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)

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：_____

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。

(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。

(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。

(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 进行计算一定不会比用 dx 进行计算更加精确。

(5) 对于任意的双精度浮点数 d，如果 d < 0，那么 d * d > 0。

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

(7) 对于任意的双精度浮点数 d，d == d。

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

(1) s == (int)(float) s

(2) s == (int)(double) s

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

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