

Hw1

T1

1.
 - o 10011001;
 - o 01001100.
2.
 - o 54;
 - o -19;

T2

1.
 - o -128,127;
2.
 - o $(-2^{N-1}) \sim (2^{N-1} - 1)$

T3

1.
 - o -64

T4

1. 暂时认定int长度为32位，则 $a + b \geq 2147483648$ ($2^{31} = 2147483648$) 时， $a + b$ 数学结果是正数，但程序认定为负数。
2. 由于 a, b 是 unsigned 类型，则即使输入负数，也会被强制转化为正数，且即使 $a + b \geq 4294967296$ ($2^{32} = 4294967296$)，程序结果 $a + b$ 等于数学结果 $\text{mod } 2^{32}$ ，而其结果仍是非负的。
于是，程序仅能输出 "Sum is non-negative\n"。

T5

- 13.0;

T6

- 最小的有限数 1 11111110 111111111111111111111111
- 最小正数 0 00000000 000000000000000000000001

T7 (这题的正常做法是什么)

- (暂时认为题目意思是十进制转化成 浮点型和整型 的二进制表示结果相同)
- 0 (瞪眼出来的)。
- 1318926965, -834214802 (程序跑出来的)。

T8

1.

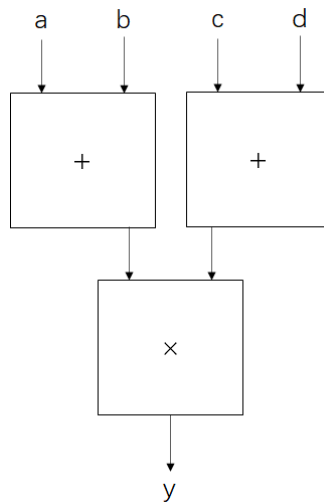
```
1 void swap(int *a, int *b)
2 {
3     *a = *a + *b;
4     *b = *a - *b;
5     *a = *a - *b;
6 }
```

2. ○ 1 | 可能在 "`*a = *a + *b;`" 这一步出现整型溢出问题。

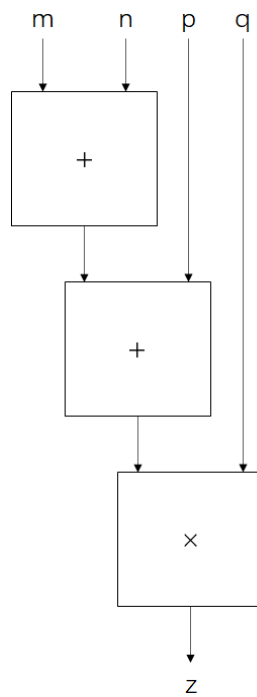
○ 1 // 以下为修正代码。
2 `void swap(int *a, int *b)`
3 `{`
4 `int temp = *a;`
5 `*a = *b;`
6 `*b = temp;`
7 `}`

T9

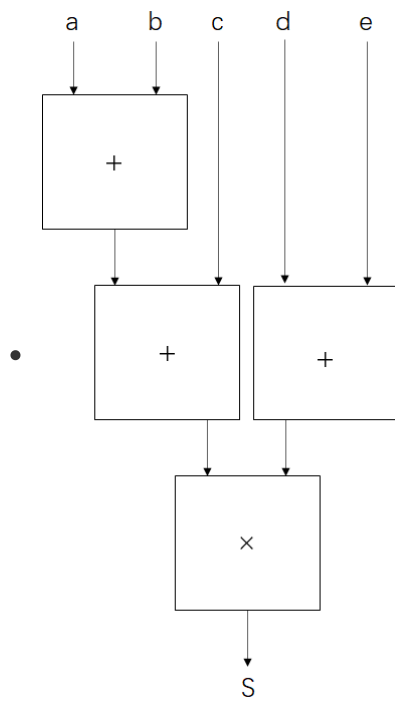
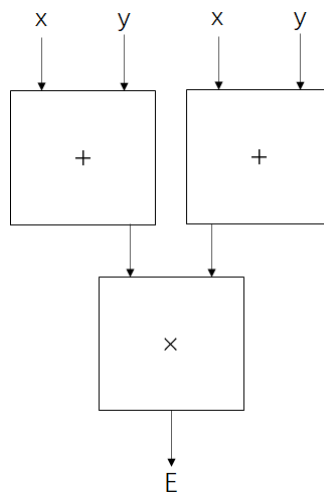
•



•



•



T10

1. 6位 ($2^6 = 64 = 26 \times 2 + 10 + 2$) ;
2. 假设这个字符串有 n 个字符, 则需要 $n \times 6$ 位;
3. 001000 011100 101100 111110 110110 110100 110110 111001 111111.