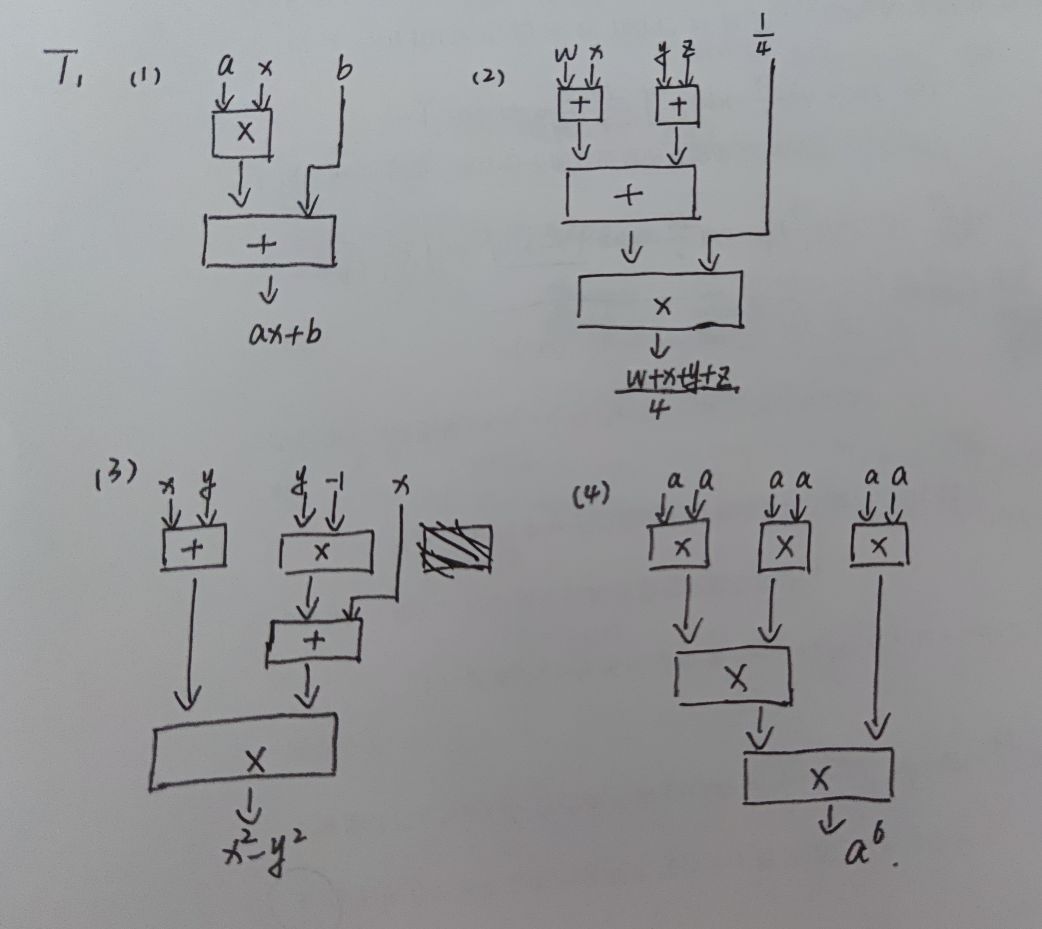
T1



T2

（1）a.10011110(错误，对于正数，原码=反码=补码)

b.00010111(错误，对于负数，补码和原码之间的转化均为非符号位取反+1)

（2）c.-62(相同错误)

d.17（相同错误）

T3

1. 110100
2. 000110
3. 10111
4. 00100

T4

1. 11101011
2. 01111110
3. 0
4. 01110001

T5

IEEE表示：0 10000001 00010011001100110011001(末尾两位应该为010，因为实际上存在进位，最后几位实际为10011，进位为10100)

实现：

#include <iostream>

#include <stdio.h>

//integer:整数 radix:输出的进制 result:结果数组 num:生成r进制所占位数

void transfer\_int(int integer, int radix, int result[], int& num)

{

int i, j, n;

for (i = 0; integer > 0; i++)//除r取余

{

result[i] = integer % radix;

integer = integer / radix;

}

num = i;//r进制位数

for (j = 0; j < i / 2; j++)//输出顺序与计算存入数组的顺序相反

{

n = result[j];

result[j] = result[i - 1 - j];

result[i - 1 - j] = n;

}

}

//小数转r进制(对8进制等均可使用）

//decimal:小数 radix:进制 result:结果数组 num:生成r进制所占位数

void transfer\_dec(double decimal, int radix, int result[], int& num)

{

int i;

for (i = 0; decimal > 0 && i < 50; i++)//乘r取整

{

result[i] = decimal \* radix;

decimal = decimal \* radix;

decimal = decimal - int(decimal);

}

num = i;

}

int main()

{

int integer;//定义整数部分

double decimal;//定义小数部分

int s, E[8] = { 0 }, m[50] = { 0 };//定义浮点数符号位，阶码数，尾数

while (true)

{

double number;//输入浮点数

printf("请输入一个浮点数：");

scanf\_s("%lf", &number);

int i=0, j=0;

if (number >= 0)

{

s = 0;

}

else

{

s = 1;

number = -number;

}//按照输入数据的正负分类

integer = (int)number;

decimal = number - integer;

int m\_n1, m\_n2;

transfer\_int(integer, 2, m, m\_n1);

transfer\_dec(decimal, 2, m + m\_n1, m\_n2);

//规格化，计算阶数, 尾数

int pn = 0;//阶数

if (integer > 0)//小数点左移

{

pn = m\_n1 - 1;//阶数

for (i = 0; i < 23; i++)//去掉首位默认的1

{

m[i] = m[i + 1];

}

}

else//小数点右移

{

for (i = 0; m[i] == 0; i++)

{

}

pn = -i - 1;

for (j = 0; j < 23; j++)//去掉左边无效的0和第一个的1

{

m[j] = m[j + 1 + i];

}

}//计算阶码

int p1[8], pn1;

transfer\_int(pn + 127, 2, p1, pn1);//阶数转二进制

if (pn1 < 8)//不足8位左边补0

{

for (j = 0; j < 8 - pn1; j++)

{

E[j] = 0;

}

for (int k = 0; k < pn1; k++)//得出完整p

{

E[j + k] = p1[k];

}

}

else

{

for (i = 0; i < 8; i++)//得出完整p

{

E[i] = p1[i];

}

}

//依次输出符号位，阶码和尾数

printf("%d ", s);

for (i = 0; i < 8; i++)

{

printf("%d", E[i]);

}

printf(" ");

for (i = 0; i < 23; i++)

{

printf("%d", m[i]);

}

printf("\n");

}

return 0;

}T6

1011.28125

T7

（1）

a. 10000101

b. 11111111

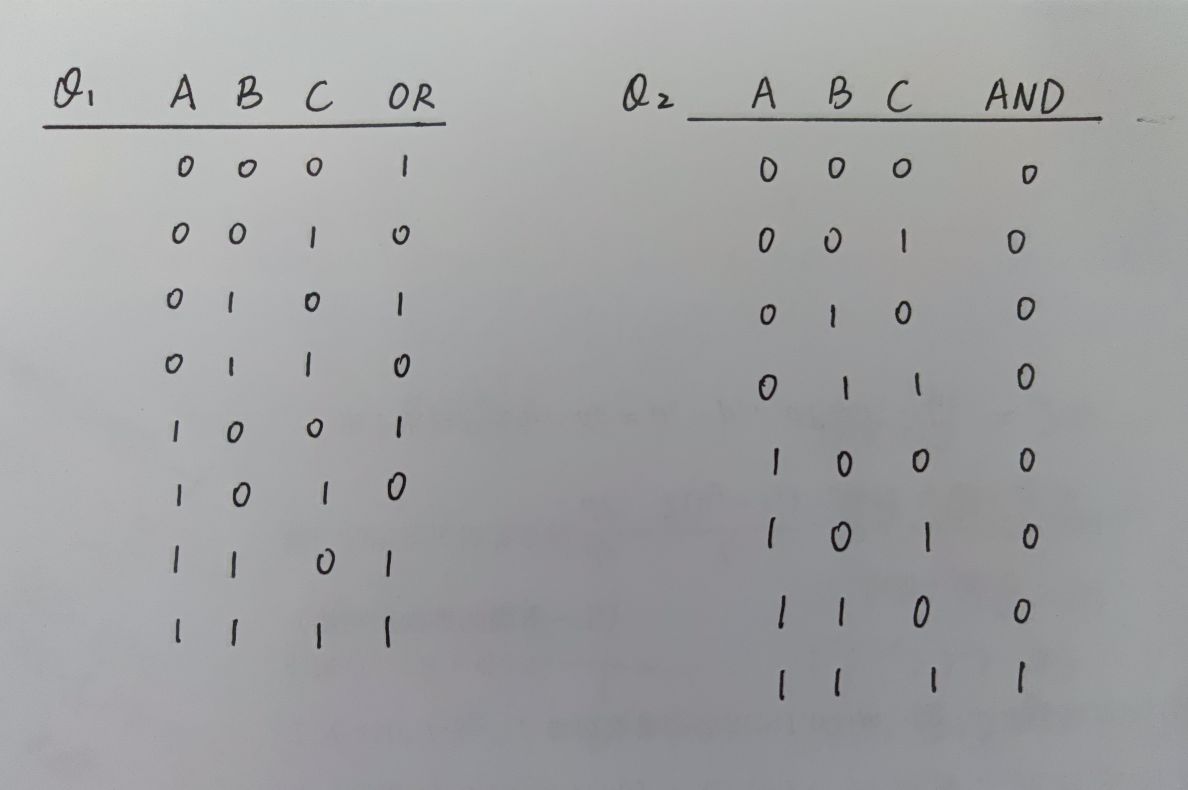
c. 10101111

（2）

d. X9BFD

e. X5007

T8



T9

1. BQoN
2. 用于加密传输8bit字节码

T10

(

T11

“没做出来”