2.57

```
#include <stdio.h>
#include <stdlib.h>
typedef unsigned char *byte_pointer;



void show_bytes(byte_pointer start , size_t len)
{
	size_t i;
	for(i = 0 ; i < len ; i++)
	{
		printf("%.2x",start[i]);
	}
	printf("\n");
}
void show_short(short x)
{
	printf("short	");
	show_bytes((byte_pointer)&x , sizeof(short));
}
void show_long(long x)
{
```

```
        printf("long        " , x , &x);
        show_bytes((byte_pointer)&x , sizeof(long));
}
void show_double(double x)
{
        printf("double        " , x , &x);
        show_bytes((byte_pointer)&x , sizeof(double));
}


int main()
{
        short a = 1;
        long b = 1;
        double c = 1.0;
        show_short(a);
        show_long(b);
        show_double(c);
        return 0;
}
```

2.61

(!(~x)) || (!x) || (!(~(x|111...100))) || (!(x&110...00))

2.73

```
int saturating_add(int x , int y)
{
    int m = INT_MIN;
    int sum = x + y;
    int w = (sizeof(int) << 3) -1;
    int t_min =(( ((m&x) && (m&y) && !(m&sum)) << w )>>w)&INT_MIN;
    int t_max = (((~(m&x)&& ~(m&y)&&(m&sum))<< w)>>w)&INT_MAX;
    return (sum & !(t_max | t_min)) | t_min | t_max;

}
```

2.77

A. K = 17

17*x = x << 4 + x;

B. K = -7

-7*x = x - x<<3;

C. K = 60

60*x = x<<6 - x<<2;

D. K = -112

-112*x = x << 4 - x<<7;

2.81

A. a = ~1^k(k 个 1).

B. b = (1^(k+j))&(~1^j)).

2.85

A. 阶码 2，尾数 M=1.110....0（二进制），小数 f=0.110....0（二进制），V=1.110..0*2^2，位表示：0（符号位）010.....01（指数位）1100....0（小数位）.

B. 阶码 n，尾数 M=1.1.....1，小数 f = 0.11.....1 , V = 1.11....1*2^n，位表示：0（符号位）（指数位无法表示）111111（小数位）.

C. 阶码 (2^(k-1))-2，尾数 M = 1.00000...0，小数 f = 0.0000...0，V=1.0...0*(2^(2^(k-1))-2)，位表示：0（符号位）0111...101（指数位）00....0（小数位）.

2.89

A. 不总是为 1。例子：x = 2147483647 时。

B. 不总是为 1。例子：x = 2147483647 , y = -2147483648 时。

C. 总是为 1。原理：int 所能表示的数的范围及其加法所能得到的数全在 double 类型的规格数范围内，故加法结合律成立。

D. 总是为 1。原理：int 所能表示的数及其乘法所能得到的数的范围全在 double 类型的规格数范围内，故乘法结合律成立。

E. 不总是为 1。例子：x = 0 , z = 2 时。

2.93

```
float_bits float_absval(float_bits f)
{
    int w = (sizeof(float_bits) << 3) -1;
    float_bits Isinfinite = !(~(10...0|f));
    float_bits IsZ = !(10...0&f);
    float_bits IsP = !(~(11....1&f));
    float_bits a =(( (!Isinfinite && IsP)<<w)>>w);
    float_bits b =(( (!Isinfinite && IsZ)<<w)>>w);
    float_bits c =(( Isinfinite<<w)>>w);
    float_bits turn = 011....1&f;
    return (a&f)|(b&turn)|(c&f);
```

```
}
```