**1.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 变量 | x | y | z | c |
| 机器数 | 0xffff8000 | 0x020a | 0x0000fffa | 0x40 |
| 变量 | a | b | u | v |
| 机器数 | 0xbf8ccccd | 0x4025000000000000 | 0x4e932c06 | 0x41d26580b4800000q |

**2.**

**2.1**

|  |  |  |  |
| --- | --- | --- | --- |
| a 的存放地址(&a) | b 的存放地址(&b) | x 的存放地址(&x) | y 的存放地(&y) |
| 0xbffff0f4 | 0xbffff0f8 | 0xbffff0f4 | 0xbffff0f8 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 执行步数 | x 的值(机器值，  用十六进制) | y 的值(机器值，  用十六进制) | \*x 的值(程序中的  真值，用十进制) | \*y 的值(程序中的真  值，用十进制) |
| 第一步前 | 0x00000001 | 0x00000002 | 1 | 2 |
| 第一步后 | 0x00000001 | 0x00000003 | 1 | 3 |
| 第二步后 | 0x00000002 | 0x00000003 | 2 | 3 |
| 第三步后 | 0x00000002 | 0x00000001 | 2 | 1 |

**2.2**

在最后一次for循环时，left == right = 3. xor\_swap函数参数传递为xor\_swap(&a[3],&a[3]), 此时在这个函数中，x 和 y 地址是相同的。改变 \*y 也会改变 \*x ，在执行按位异或时，会把 \*x 和 \*y都置为0. 修改函数 xor\_swap(int \*x, int \*y) ,在函数汇总判断 x 和 y 是否指向相同地址，若不指向同一地址，函数继续，否则函数返回。

**3.**

|  |  |  |
| --- | --- | --- |
|  | 输 出  True/False | 原因 |
| 语句一 | T | 从int转换为double，因为double有更大的范围，因此不会损失精度。 |
| 语句二 | F | 从int转为float，发生了舍入，值发生了变化。此处被舍入为2147483648.000000，再次转化为int类型时与原来的数不相等 |
| 语句三 | F | float p1 = 3.141592653;超过了float的精度表示范围，会发生舍入，两个数字舍入后相等 |
| 语句四 | T | double result1 = d+(f-f);括号中计算为0，结果和d相等。 |
| 语句五 | F | double result2 = (d+f)-f; f=1e20是一个很大的数，括号中的计算结果超过了double的最大精度，会发生舍入，最后的计算结果不等于d |

**4.**

**4.1**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 机器数  (十六进制) | 真值  (十进制) |  | 机器数  ( 十 六 进制) | 真值  ( 十 进  制) |
| x | 0x66 | 102 | y | 0x39 | 57 |
| ~x | 0x99 | -103 | !x | 0x00 | 0 |
| x&y | 0x20 | 32 | x&&y | 0x01 | 1 |
| x|y | 0x7f | 127 | x||y | 0x01 | 1 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | 机器数  (十六进制) | 真值  (十进制) | OF | SF | CF | AF |
| x1 | 0xb0 | -80 | 0 | 0 | 0 | 0 |
| y1 | 0x8c | -116 | 0 | 0 | 0 | 0 |
| x1+y1 | 0x3c | 60 | 0 | 0 | 0 | 0 |
| x1-y1 | 0x24 | 36 | 0 | 0 | 0 | 1 |
| y1-x1 | 0xdc | -36 | 0 | 1 | 1 | 0 |
| x2 | 0xb0 | 176 | 0 | 1 | 1 | 0 |
| y2 | 0x80 | 140 | 0 | 1 | 1 | 0 |
| x2+y2 | 0x3c | 60 | 0 | 0 | 0 | 0 |
| x2-y2 | 0x24 | 36 | 0 | 0 | 0 | 1 |
| y2-x2 | 0xdc | 220 | 0 | 1 | 1 | 0 |

char类型的表示范围为 -128~127，unsigned char 类型的表示范围为0~255.超过范围会溢出

OF: 带符号整型运算指示溢出状态

SF: 有符号整型的最高有效位。(0指示结果为正，反之则为负)

CF: 无符号整型运算的溢出状态

AF: 算术操作在结果的第3位发生进位或借位则将该标志置1，否则清零