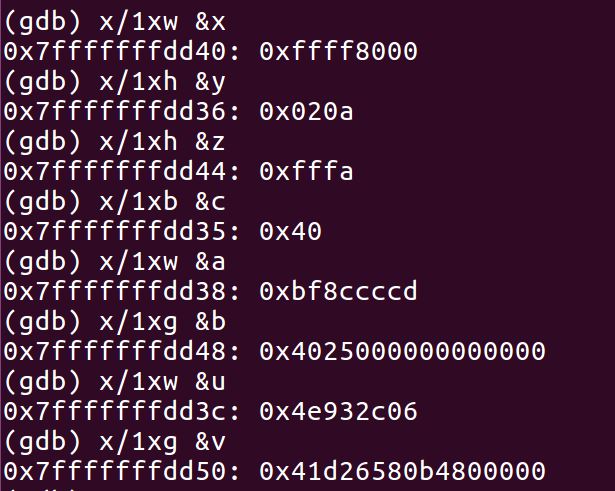
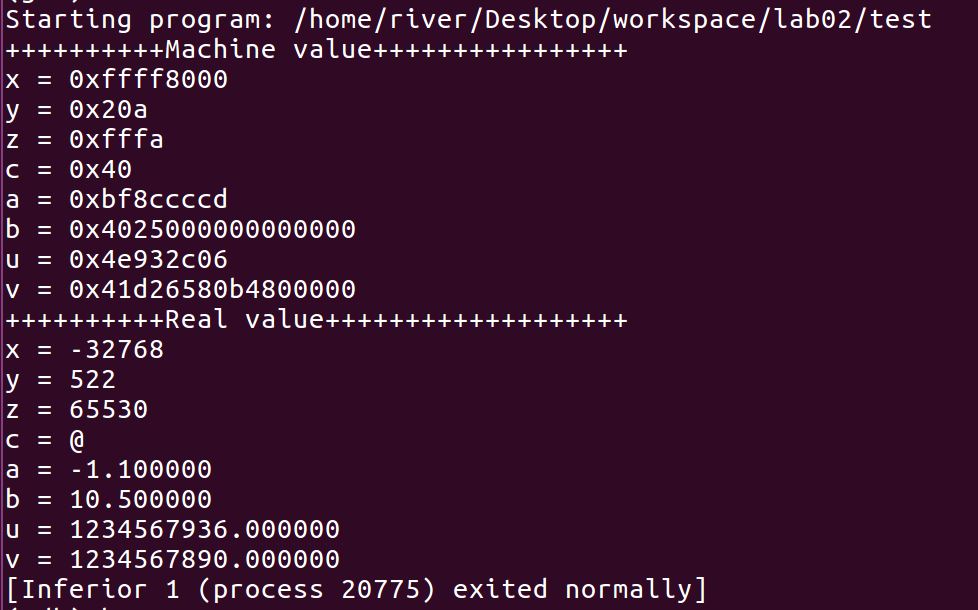
1. 1) int x=-32768, 2) short y=522, 3) unsigned z=65530

4) char c=’@’, 5) float a=-1.1, 6) double b=10.5,

7) float u = 123456.789e4, 8) double v= 123456.789e4

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 变量 | x | y | z | c |
| 机器数 | 0xffff8000 | 0x20a | 0xfffa | 0x40 |
| 变量 | a | b | u | v |
| 机器数 | 0xbf8ccccd | 0x4025000000000000 | 0x4e932c06 | 0x41d26580b4800000 |



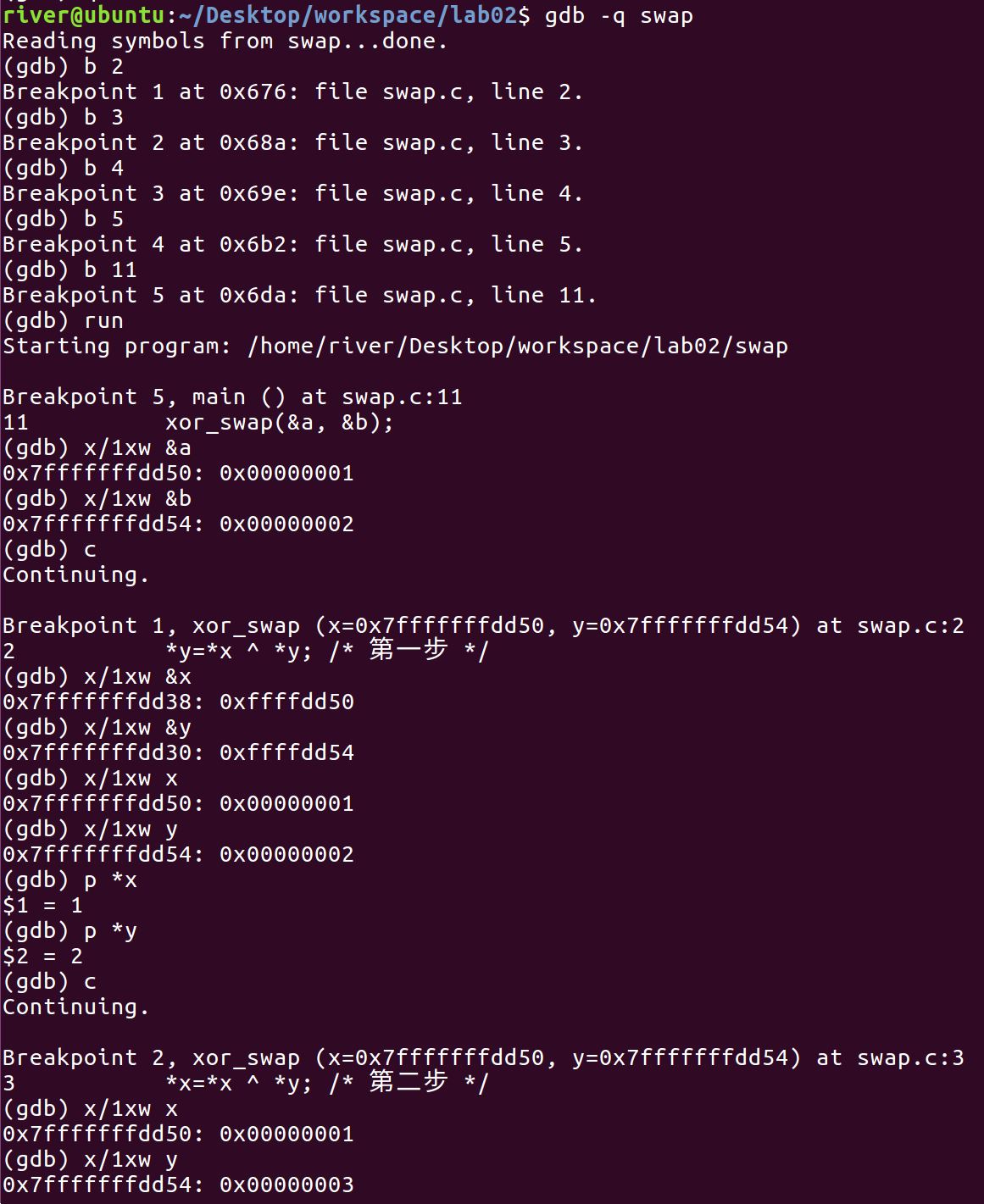


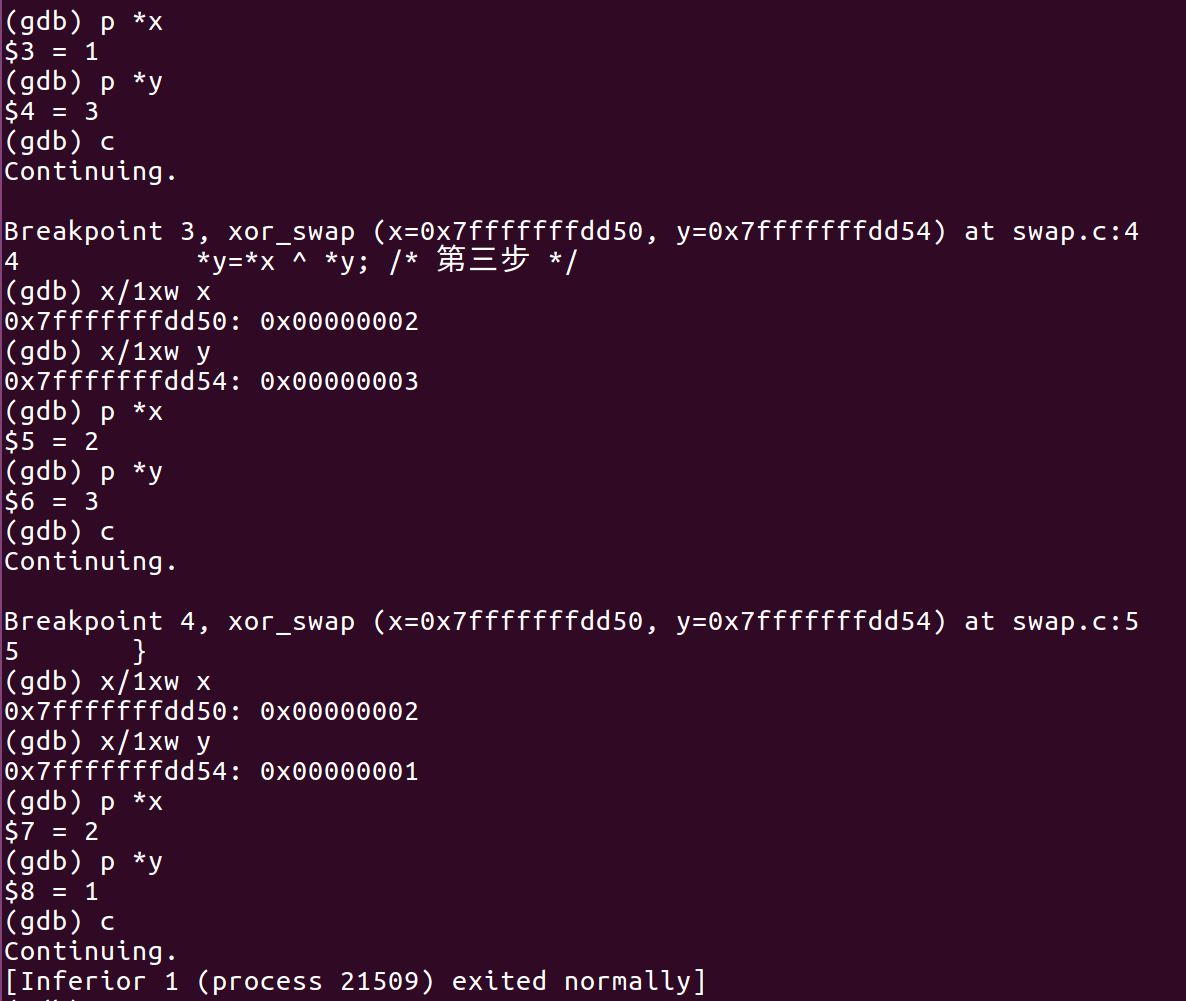
运行代码验证输出与gdb查看的结果一致。

2. 1) 使用gdb命令查看程序变量的取值，填写下面两个表格：

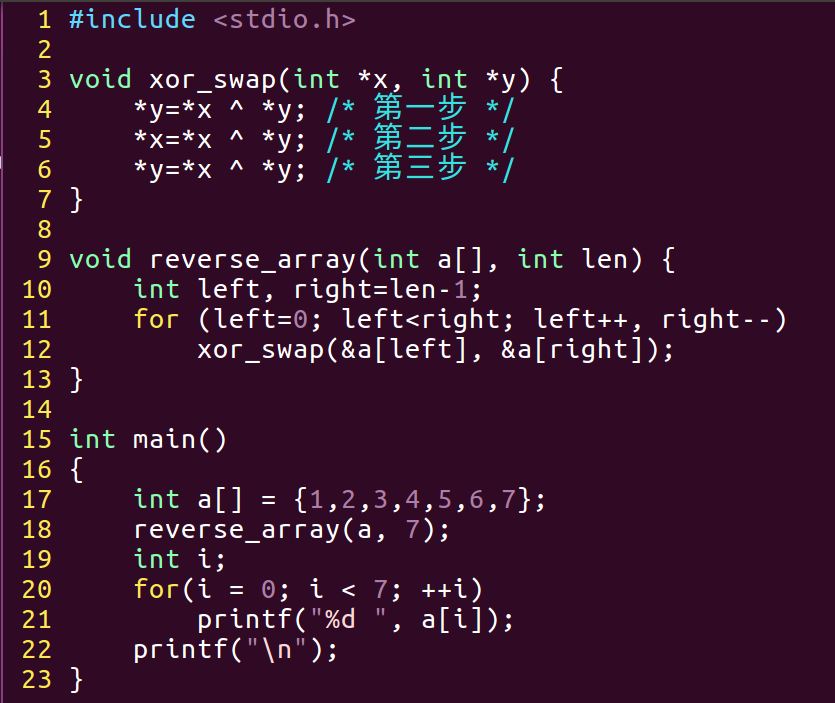
|  |  |  |  |
| --- | --- | --- | --- |
| a的存放地址(&a) | b的存放地址(&b) | x的存放地址(&x) | y的存放地址(&y) |
| 0x7fffffffdd50 | 0x7fffffffdd54 | 0x7fffffffdd38 | 0x7fffffffdd30 |

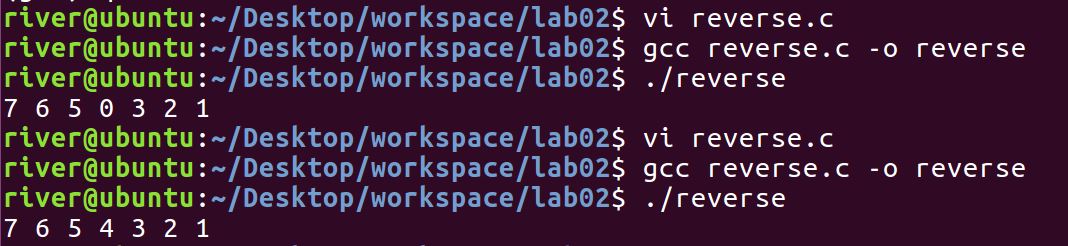
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 执行步数 | x的值(机器) | y的值(机器) | \*x的值(真值) | \*y的值(真值) |
| 第一步前 | 0x7fffffffdd50 | 0x7fffffffdd54 | 1 | 2 |
| 第一步后 | 0x7fffffffdd50 | 0x7fffffffdd54 | 1 | 3 |
| 第二步后 | 0x7fffffffdd50 | 0x7fffffffdd54 | 2 | 3 |
| 第三步后 | 0x7fffffffdd50 | 0x7fffffffdd54 | 2 | 1 |





2) 运行reverse.c，并说明输出这种结果的原因，修改代码以得到正确的逆序数组

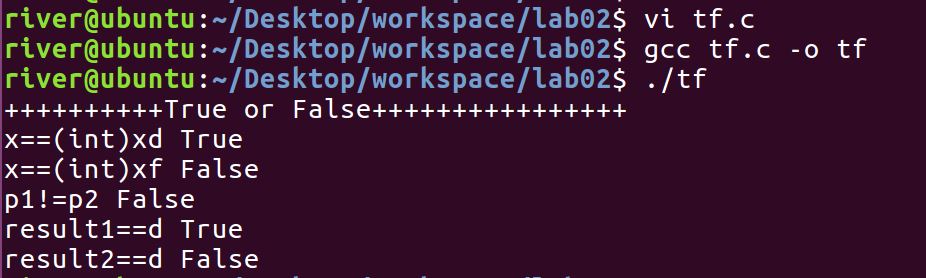


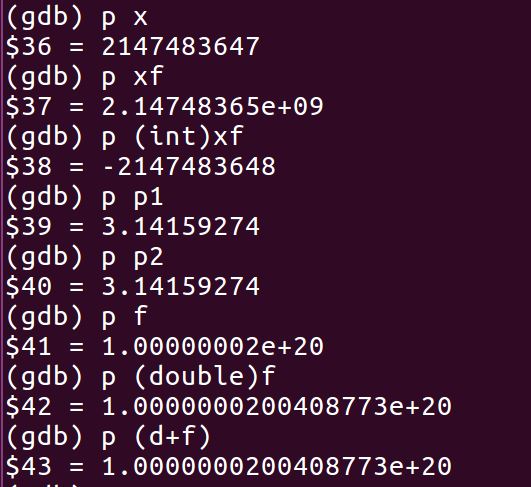


循环条件是left<=right，即left==right==3时执行xor\_swap(&a[3], &a[3])

两个相同的二进制数进行异或操作得到0，则执行三次a[3]=a[3] ^ a[3]后a[3]的值为0

3. 解释语句输出为 False 的原因并填写在表格中





|  |  |  |
| --- | --- | --- |
|  | 输出 | 原因 |
| 语句一 | True |  |
| 语句二 | False | int有32位精度，而float只有24位，将INT\_MAX赋值给float xf时有数据被舍入，得2.1478365e+09大于2147483647，强制转化为int时符号位是1，识别为负数，原int x为正数 |
| 语句三 | False | 3.141592653和3.141592654所表示的精度均超出float所能表示的范围，p1与p2只精确到3.14159274，相等 |
| 语句四 | True |  |
| 语句五 | False | 执行(d+f)-f时f由float强制转化为double，1.00000002e+20变为1.0000000200408773e+20，d相对f数值过小，相加时可忽略不计，double f与(d+f)近似相等，result2为0 |

4. 请在单步运行过程中完成下面的表格：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 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 | 0xffffff99 | 4294967193 | !x | 0xffffff00 | 4294967040 |
| x & y | 0x20 | 32 | x && y | 0x1 | 1 |
| x | y | 0x7f | 127 | x || y | 0x1 | 1 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | 机器数  (十六进制) | 真值  (十进制) | OF | SF | CF | AF |
| x1 | 0x7fffffff | 2147483647 | 0 | 0 | 0 | 0 |
| y1 | 0x1 | 1 | 0 | 0 | 0 | 0 |
| sum\_x1\_y1 | 0x80000000 | 2147483648 | 1 | 1 | 0 | 1 |
| diff\_x1\_y1 | 0x7ffffffe | 2147483646 | 0 | 0 | 0 | 0 |
| diff\_y1\_x1 | 0x80000002 | 2147483650 | 0 | 1 | 1 | 1 |
| x2 | 0x7fffffff | 2147483647 | 0 | 0 | 0 | 0 |
| y2 | 0x1 | 1 | 0 | 0 | 0 | 0 |
| sum\_x2\_y2 | 0x80000000 | 2147483648 | 1 | 1 | 0 | 1 |
| diff\_x2\_y2 | 0x7ffffffe | 2147483646 | 0 | 0 | 0 | 0 |
| diff\_y2\_x2 | 0x80000002 | 2147483650 | 0 | 1 | 1 | 1 |

2) 写出上面表格中每个标识位变化的原因，可直接在上表中注明。

sum\_x1\_y1: 01111111111111111111111111111111

+ 00000000000000000000000000000001

= 10000000000000000000000000000000,

结果溢出了，OF为1，sum符号为1，SF为1，第三位发生进位，AF为1。

diff\_y1\_x1: 00000000000000000000000000000001

- 01111111111111111111111111111111

= 10000000000000000000000000000010

sum符号为1，SF为1，最高有效位发生借位，CF为1，第三位发生借位，AF为1

sum\_x2\_y2: 01111111111111111111111111111111

+ 00000000000000000000000000000001

= 10000000000000000000000000000000,

结果溢出了，OF为1，sum符号为1，SF为1，第三位发生进位，AF为1。

diff\_y2\_x2: 00000000000000000000000000000001

- 01111111111111111111111111111111

= 10000000000000000000000000000010

sum符号为1，SF为1，最高有效位发生借位，CF为1，第三位发生借位，AF为1