1.(P.70 4)一个字长为 16 的机器中，整型数据大小等同于指针，那么无符号整型数据、有符号整型数据以及有符号短整型数据的表示范围分别是多少？

字长是指 CPU 一次能处理的二进制位数，通常等同于：

(1)通用寄存器的位数（如 EAX 在 16 位 CPU 中是 16 位）。

(2)数据总线的宽度（如 16 位 CPU 通常有 16 位数据总线）。

(3)**指针的大小**（如 16 位机器指针是 2 字节）。

机器位数（通常称为“机器字长”或“CPU 位数”）是指：

(1)CPU 寄存器的宽度（如 16 位、32 位、64 位）。

(2)数据总线的宽度（CPU 一次能处理的数据位数）。

(3)地址总线的宽度（决定可寻址的内存空间，如 16 位 → 64KB）。

机器位数=字长=CPU 寄存器宽度（通常）;机器位数决定了 CPU 最自然的整数运算大小。

因此，

机器位数	字长	典型 int 大小	指针大小	最大内存寻址
16 位	16 位	16 位（2 字节）	16 位	64KB

答：

（1）无符号整型数据（unsigned int）：

16 位全用于表示数值，没有符号位。

范围：0 到 2 的 16 次方-1

即：**0 到 65535**

（2）有符号整型数据（int）：

使用补码表示，1 位符号位，15 位数值位。

范围：-2 的 15 次方到 2 的 15 次方-1

即：**-32768 到 32767**

（3）short int（短整型）原本是为了节省内存而设计的，但在 16 位机器中，如果 int 已经是 16 位，short 不能再更小（否则可能影响性能），所以它通常也是 16 位。

有符号短整型数据（short int）：

通常为 16 位。

同样使用补码表示，范围与 int 相同。

即：**-32768 到 32767**

2.（P70 9）对于下列 C 语言代码（节选），给出其在 32 位机器上的输出结果，并解释产生这些结果的原因。

```c
#include <stdio.h>
int main(){
    int x=-1;
    unsigned int uy=4294967295UL;
    int a=(x^uy);
    int b=(x&&uy);
    printf("unsigned x is:%u \n",x);
    printf("signed uy is:%d \n",uy);
    printf("a=%d\n",a);
    printf("b=%d\n",b);
}
```

```
(base) xiaoye@长乐:/mnt/d/hnu/大二课程/计算机系统/作业$ file 2.2
2.2: ELF 32-bit LSB pie executable, Intel 80386, version 1 (SYSV), dynamically linked, interpreter /lib/ld-linux.so.2, BuildID[sha1]=bc45d88498afcf44e74db68333733cf40365fafc, for GNU/Linux 3.2.0, not stripped
(base) xiaoye@长乐:/mnt/d/hnu/大二课程/计算机系统/作业$ ./2.2
unsigned x is:4294967295
signed uy is:-1
a=0
b=1
```

原因：

（1）

在计算机中，负数采用补码（Two’s Complement） 表示，计算方式如下：

-1 的补码计算：

原码（绝对值的二进制）：00000000 00000000 00000000 00000001（即 1）

取反（按位取反）：11111111 11111111 11111111 11111110

加 1（得到补码）：11111111 11111111 11111111 11111111

x 的二进制表示：1111 1111 1111 1111 1111 1111 1111 1111

uy 的二进制表示：1111 1111 1111 1111 1111 1111 1111 1111

（2）x 按无符号数输出,最高位是正权。2 的 31 方+2 的 30 次方+•••+2 的 0 次方（实际上等于 2 的 32 次方-1）。输出 unsigned x is:4294967295。

（3）uy 按有符号数输出。最高位是负权。（-1）*2 的 31 方+2 的 30 次方+•••+2 的 0 次方。输出 signed uy is:-1；

（4）位运算：x 和 uy 异或。在二进制层面上，按位异或，逐位比较 x 和 y，相同为 0，不同为 1。由上可知，两者相同，输出 a=0；

（4）逻辑运算：x 与 uy。如果 x 和 uy 都不为 0，则返回 1（真）。只要有一个是 0，就返回 0（假）。&&不会改变操作数的类型，它只关心 x 和 uy 是否为零。由上可知，x、uy 都不为 0，所以输出 b=1。

3．（P71 11）假如某浮点数类型有1位符号位、8位尾数和1位阶码，则该类型能表示的最大浮点数的十进制数值是多少？

（1）由IEEE浮点表示法可知，一个浮点数的二进制位模式被划分为三个字段：

（1）一个单独的符号位s，直接编码符号位S；

（2）k位的阶码字段exp=e（k-1）e（k-2）・・・e(0)编码阶码E；

（3）n位的小数字段frac=f(n-1)・・・f(0)编码尾数M，但是编码出来的值也依赖于阶码字段的值是否为0。

在位表示确定的情况下，依据不同的exp值，被编码的浮点数可以分成三种不同的情况：

（1）exp的位模式既非全0也非全1，所表示的数为规格化的值。E=e-Bias。【Bias=2的k-1次方-1，k是阶码位数】，M=1+f；

（2）exp的位模式为全0，所表示的数为非规格化的值。E=1-Bias；M=f;

（3）exp的位模式为全1，所表示的数为特殊值。此时，小数域全0表示无穷，s=0对应正无穷，s=1对应负无穷；如果小数域非0，此数值称为NaN，意为“不是一个数”。

本题中，阶码为1，故只有exp全0和全1两种情况。经计算，Bias=0。具体结果如下：

S	exp	frac
1位	1位	8位.

$Bias = 2^{K-1}-1$（其中 $K=1$）$\Rightarrow Bias = 0$

非规格化的值：

s	exp	frac	E	Value
0	0	0000 0000	1	0
0	0	0000 0001	1	$(2^{-8})\times 2 = \frac{1}{128}$
……				
0	0	1111 1111	1	$(2^{-1}+\cdots+2^{-8})\times 2 =$
……				

特殊值：

0	1	0000 0000	n/a	inf
0	1	0000 0001		NaN
……				

最大浮点数的十进制为：1.9921875

（2）但本题只有 1 位阶码，如果不区分规格化和特殊值的话，将 exp 全 1，看作规格化值，那么可表示的最大浮点数为 0 1 11111111 ，该值的 M=1+f，E=1，换算为十进制的结果为 3.9921875。

最大浮点数的十进制为：3.9921875

4.（P71 17）考虑下列基于 IEEE 浮点格式的 7 位浮点表示。两个格式都没有符号位——它们只能表示非负的数字。

1）格式 A

• 有 k=3 个阶码位。阶码的偏置值是 3；

• 有 n=4 个小数位

2）格式 B

• 有 k=4 个阶码位。阶码的偏置值是 7；

• 有 n=3 个小数位

请以格式 A 表示的数[101 1110]转换为格式 B 表示的数。

答：格式 A 表示的数[101 1110]转换为格式 B 表示的数[1001 111]。

IEEE 表示法见上一题表达，不再赘述，具体分析过程如下。

格式A表示的数[101 1110]

exp不是全0,也不是全1 $\Rightarrow$ 它是规格化的值.

$E = e - Bias \qquad M = 1 + f$

格式A: $Bias = 3 \quad e = 5 \quad f = \frac{1}{2} + \frac{1}{4} + \frac{1}{8} = \frac{7}{8}$

$\downarrow$

十进制: $E = 2 \quad M = \frac{15}{8} \qquad 2^2 \times \frac{15}{8} = \frac{15}{2}$

$\downarrow$

二进制: $111.1 \Leftrightarrow 1.111 \times 2^2$

$\downarrow$

格式B: 由 1.111×2^2 知, $E=2$

格式B有k=4个阶码位,阶码的偏置是7

$e = E + Bias = 7 + 2 = 9 \xrightarrow{\text{二进制}} 1001$

格式B有n=3个小数位. $M = 1 + f$

$frac = 111$

综上,以格式A表示的数[101 1110]转换为格式B表示的数[1001 111]。

4.假设一个基于 IEEE 浮点格式的 9 位浮点表示有 1 个符号位、4 个阶码位（k=4）和 4 个尾数位（n=4），请写出正数中最小的非规格化数、最大的非规格化数、最小的规格化数、最大的规格化数的二进制位表示。

答：正数中最小的非规格化数、最大的非规格化数、最小的规格化数、最大的规格化数的二进制位表示分别为 0 0000 0001、0 0000 1111、0 0001 0000、0 1110 1111。分析如下：

1个符号位、4个阶码位（k=4）、4个尾数位（n=4）

$Bias = 2^{k-1} - 1 = 7$

IEEE浮点数表示方法

- exp非全0或全1　规格化的值　$E = e - Bias$　$M = 1 + f$
- exp全0　非规格化的值　$E = 1 - Bias$　$M = f$
- exp全1　小数域全0　$S=0$　$+\infty$　$S=1$　$-\infty$
- 　小数域非0　NaN

s	exp	frac	E	value	
非规格化数					
0	0000	0000	-6	0	
0	0000	0001	-6	$\frac{1}{16}\times(2^{-6})=\frac{1}{1024}$	正数中最小的非规格化数.
		递增			
0	0000	0010	-6	$\frac{1}{8}\times(2^{-6})$	
……	……				
0	0000	1111	-6	$(\frac{1}{2}+\frac{1}{4}+\frac{1}{8}+\frac{1}{16})\times 2^{-6}=\frac{15}{1024}$	正数中最大的非规格化
规格化数					
0	0001	0000	-6	$2^{-6}=\frac{1}{64}$	正数中最小的规格化数.
		递增			
0	0001	0001	-6	$(\frac{1}{16}+1)\times 2^{-6}$	
……					
0	1110	1110	7	$(1+\frac{1}{2}+\frac{1}{4}+\frac{1}{8})\times 2^{7}$	
0	1110	1111	7	$(1+\frac{1}{2}+\frac{1}{4}+\frac{1}{8}+\frac{1}{16})\times 2^{7}=248$	正数中最大的规格化数.
特殊值.					
0	1111	0000	n/a	inf	
0	1111	0001		NaN	
……					
0	1111	1111		NaN	