

//第一题

```
#include <stdio.h>
```

// 定义函数：返回两个整数中较大的那个

```
int max(int a, int b) {
```

```
    return a > b ? a : b; // 三元运算符，a大就返回a，否则返回b
}
```

```
int main() {
```

```
    int num1, num2;
```

```
    printf("请输入两个整数: ");
```

```
    scanf("%d %d", &num1, &num2);
```

```
    int result = max(num1, num2); // 调用max函数
```

```
    printf("最大值是: %d\n", result);
```

```
    return 0;
```

```
}
```

//第二题

```
#include <stdio.h>
```

// 定义函数：返回两个正整数的最小公倍数

```
int lcm(int a, int b) {
```

```
    int max_num = a > b ? a : b; // 最小公倍数至少是较大的数
```

```
    int i = max_num;
```

```
    while (1) { // 循环找符合条件的数
```

```
        if (i % a == 0 && i % b == 0) { // 能同时被a、b整除
```

```
            return i;
```

```
        }
```

```
        i++; // 没找到就继续下一个数
```

```
    }
```

```
}
```

```
int main() {
```

```
    int num1, num2;
```

```
    printf("请输入两个正整数: ");
```

```
    scanf("%d %d", &num1, &num2);
```

```
    int result = lcm(num1, num2); // 调用lcm函数
```

```
    printf("最小公倍数是: %d\n", result);
```

```
    return 0;
```

```
}
```

//第三题

```
#include <stdio.h>
```

// 递归函数：计算n的k次方

```
int power(int n, int k) {  
    if (k == 1) { // 终止条件：k=1时，直接返回n  
        return n;  
    } else { // 递归调用： $n^k = n * n^{(k-1)}$   
        return n * power(n, k - 1);  
    }  
}
```

```
int main() {  
    int n, k;  
    printf("请输入正整数n和k(计算n的k次方) : ");  
    scanf("%d %d", &n, &k);  
  
    int result = power(n, k); // 调用递归函数  
    printf("%d的%d次方是: %d\n", n, k, result);  
    return 0;  
}
```