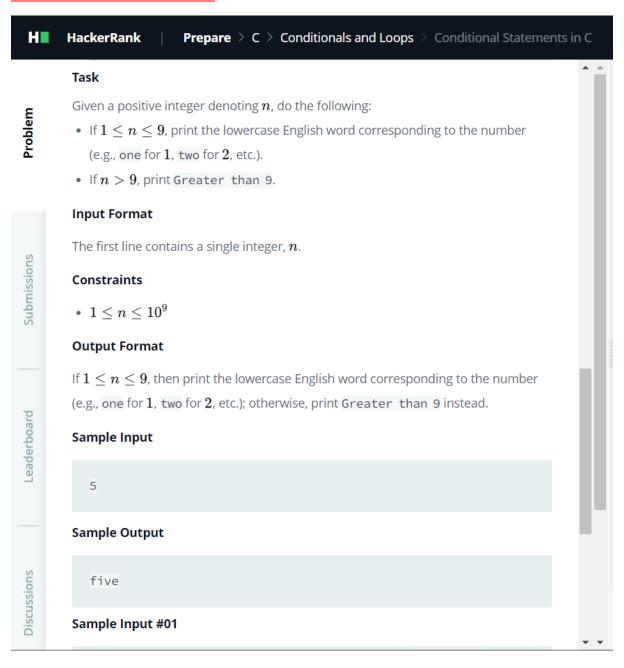
CONDITIONAL STATEMENTS IN C:



```
int main()
11 🗸 {
12
        int number;
13
        scanf("%d",&number);
14 ∨
        if(number==1){
15
            printf("one");
16
        }
17
       else if(number==2){
18
        printf("two");
19
        }
20 🗸
       else if(number==3){
21
        printf("three");
       }
       else if(number==4){
23 🗸
          printf("four");
24
25
       else if(number==5){
26 ∨
           printf("five");
27
28
       else if(number==6){
29 🗸
           printf("six");
31
32 ∨
       else if(number==7){
33
            printf("seven");
        else if(number==8){
35 V
36
            printf("eight");
37
        else if(number==9){
38 ∨
            printf("nine");
39
40
        }
        else{
41 V
            printf("Greater than 9");
42
43
44
       return 0;
45
46
```

Ø	Sample Test case 0	Input (stdin)	Download
\otimes	Sample Test case 1	1 5	
\otimes	Sample Test case 2	Your Output (stdout) 1 five	
		Expected Output	Download
		1 five	

For Loop in C 🛊

Task

For each integer n in the interval [a,b] (given as input) :

- If $1 \le n \le 9$, then print the English representation of it in lowercase. That is "one" for 1, "two" for 2, and so on.
- ullet Else if n>9 and it is an even number, then print "even".
- Else if n>9 and it is an odd number, then print "odd".

Input Format

The first line contains an integer, a.

The seond line contains an integer, b.

Constraints

$$1 \le a \le b \le 10^6$$

Output Format

Print the appropriate English representation, even, or odd, based on the conditions described in the 'task' section.

Note:
$$[a,b]=\{x\in\mathbb{Z}\mid\ a\leq x\leq b\}=\{a,\ a+1,\ldots,b\}$$

Sample Input

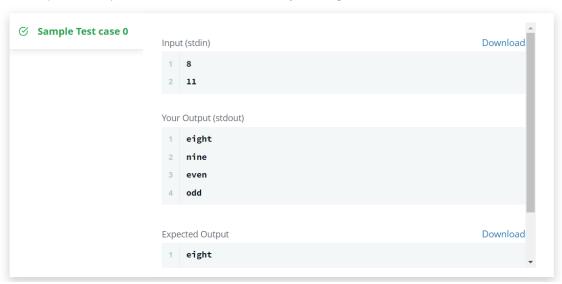
8 11

Sample Output

eight nine

even odd

```
#include <stdio.h>
    int main()
 3 ∨{
4
         int a, b,i;
         scanf("%d\n%d", &a, &b);
5
6 ~
         for(i=a;i<=b;i++){</pre>
 7 ∨
             if (i<10) {
 8 🗸
                 switch(i) {
                    case 1: printf("one\n"); break;
9
10
                    case 2: printf("two\n"); break;
                    case 3: printf("three\n"); break;
                    case 4: printf("four\n"); break;
                    case 5: printf("five\n"); break;
                    case 6: printf("six\n"); break;
14
                    case 7: printf("seven\n"); break;
                     case 8: printf("eight\n"); break;
16
                     case 9: printf("nine\n"); break;
18
19
20 🗸
            else if(i>9 && i%2==0)
                   printf("even\n");
             else if(i>9 && i%2!=0)
22 V
23
                    printf("odd\n");
24
         return 0;
26
```



Print a pattern of numbers from ${\bf 1}$ to ${\bf n}$ as shown below. Each of the numbers is separated by a single space.

Input Format

The input will contain a single integer n.

Constraints

 $1 \le n \le 1000$

Sample Input 0

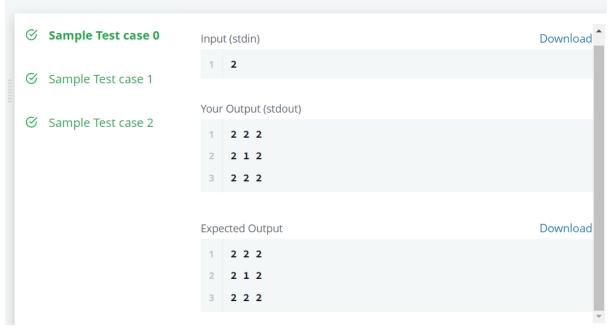
2

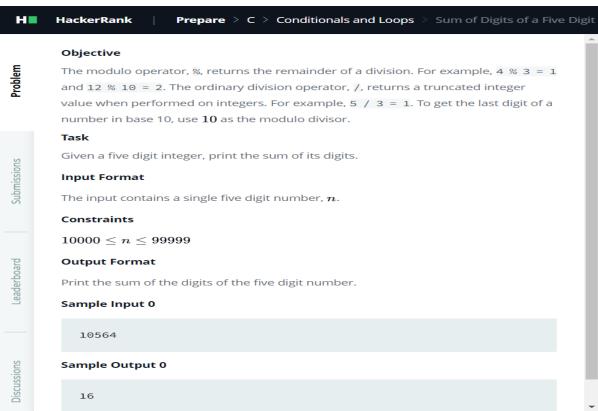
Sample Output 0

```
2 2 2
2 1 2
2 2 2
```

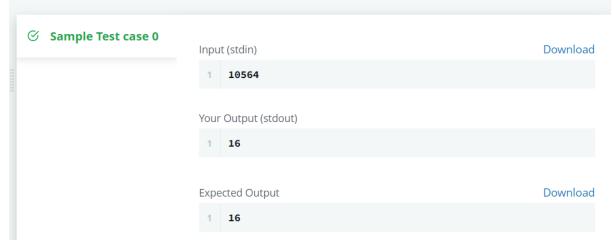
```
#include <stdio.h>
     #include <string.h>
     #include <math.h>
 3
 4
     #include <stdlib.h>
 5
6
     int main()
 7 ~ {
8
9
          int n,i,j;
10
          scanf("%d", &n);
          for(i=1;i<=2*n-1;i++){
12 🗸
                 for(j=1;j<=2*n-1;j++){
   int min_val = i < j ? i : j;
   min_val = min_val < (2 * n - i) ? min_val : (2 * n - i);</pre>
13 🗸
14
                    min_val = min_val < (2 * n - j) ? min_val : (2 * n - j);
16
                 printf("%d ", n - min_val + 1);
}
19
                 printf("\n");
            }
          return 0;
23
     }
```







```
#include <stdio.h>
 2 #include <string.h>
 3
    #include <math.h>
    #include <stdlib.h>
 4
 5 \vee int main() {
        int n,i,sum=0;
 6
        scanf("%d", &n);
 7
       if (n < 10000 || n > 99999){
8 🗸
           printf("Invalid number");
9
10
11 ∨ while(n>0){
12
           sum+=n%10;
13
           n/=10;
14 🗸
        printf("%d",sum);
15
16
       return 0;
17
```



Problem

In this challenge, you will use logical bitwise operators. All data is stored in its binary representation. The logical operators, and C language, use $\mathbf 1$ to represent true and $\mathbf 0$ to represent false. The logical operators compare bits in two numbers and return true or false, 0 or 1, for each bit compared.

- Bitwise AND operator & The output of bitwise AND is 1 if the corresponding bits of two operands is 1. If either bit of an operand is 0, the result of corresponding bit is evaluated to 0. It is denoted by &.
- Bitwise OR operator | The output of bitwise OR is 1 if at least one corresponding bit of two operands is 1. It is denoted by |.
- Bitwise XOR (exclusive OR) operator ^ The result of bitwise XOR operator is 1 if the corresponding bits of two operands are opposite. It is denoted by \oplus .

For example, for integers 3 and 5,

```
3 = 00000011 (In Binary)
5 = 00000101 (In Binary)
AND operation
                    OR operation
                                       XOR operation
 00000011
                      00000011
                                         00000011
                                        ^ 00000101
& 00000101
                    00000101
  00000001 = 1
                      00000111 = 7
                                          00000110 = 6
```

You will be given an integer \emph{n} , and a threshold, $\emph{k.}$ Foreachnumberifrom1throughn

 $, find the maximum value of the logical and, or and xor when compared against all integers through \\ \sqcap$ $that are greater than \verb||i.Consider a value only if the comparison returns are sultless than \verb||k\$|. Print the are greater than a value of the comparison returns are sultless than a value of the comparison returns are sultless than a value of the comparison returns are sultless than a value of the comparison returns are sultless than a value of the comparison returns are sultless than a value of the comparison returns are sultless than a value of the comparison returns are sultless than a value of the comparison returns are sultless than a value of the comparison returns are sultless than a value of the comparison returns are sultless than a value of the comparison returns are sultless than a value of the comparison returns a value of the com$ results of the and, or and exclusive or comparisons on separate lines, in that order.

HackerRank **Prepare** > C > Conditionals and Loops > Bitwise Operators

Problem

Submissions

_eaderboard

Discussions

n = 3

k = 3

Example

The results of the comparisons are below:

```
a b
    and or xor
    0 3 3
1 2
     1
         3
           2
1 3
2 3
     2
        3 1
```

For the and comparison, the maximum is 2. For the or comparison, none of the values is less than k, so the maximum is 0. For the xor comparison, the maximum value less than k is 2. The function should print:

2 0 2

Function Description

Complete the calculate_the_maximum function in the editor below.

calculate_the_maximum has the following parameters:

- int n: the highest number to consider
- int k: the result of a comparison must be lower than this number to be considered

Prints

Submissions

Discussions

The only line contains 2 space-separated integers, n and k.

Print the maximum values for the and, or and xor comparisons, each on a separate line.

Constraints

- $2 \le n \le 10^3$
- $2 \le k \le n$

Sample Input 0

5 4

Sample Output 0

3

3

Explanation 0

$$n=5,k=4$$

$$S = \{1, 2, 3, 4, 5\}$$

All possible values of \boldsymbol{a} and \boldsymbol{b} are:

1.
$$a = 1, b = 2; a \& b = 0; a | b = 3; a \oplus b = 3;$$

2.
$$a = 1, b = 3; a \& b = 1; a | b = 3; a \oplus b = 2;$$

3.
$$a = 1, b = 4; a \& b = 0; a | b = 5; a \oplus b = 5;$$

4.
$$a = 1, b = 5$$
; $a \& b = 1$; $a | b = 5$; $a \oplus b = 4$;

5.
$$a = 2$$
, $b = 3$; $a \& b = 2$; $a | b = 3$; $a \oplus b = 1$;

6.
$$a = 2, b = 4; a \& b = 0; a | b = 6; a \oplus b = 6;$$

7.
$$a = 2, b = 5; a \& b = 0; a | b = 7; a \oplus b = 7;$$

8.
$$a = 3, b = 4$$
; $a \& b = 0$; $a | b = 7$; $a \oplus b = 7$;

9.
$$a=3,\,b=5;\,\,a\;\&\;b=1;\,\,a\mid b=7;\,\,a\oplus b=6;$$

10.
$$a = 4, b = 5$$
; $a \& b = 4$; $a | b = 5$; $a \oplus b = 1$;

- The maximum possible value of a&b that is also <(k=4) is 2, so we print 2 on first line.
- The maximum possible value of a|b that is also <(k=4) is 3, so we print 3 on second line.
- The maximum possible value of $a\oplus b$ that is also <(k=4) is 3, so we print 3 on third line.

ions

```
#include <stdio.h>
void calculate_the_maximum(int n, int k) {
   int max_and = 0, max_or = 0, max_xor = 0;
     for (int i = 1; i <= n; i++) {
         for (int j = i + 1; j <= n; j++) {
             int current_and = i & j;
             int current_or = i | j;
             int current_xor = i ^ j;
             if (current_and > max_and && current_and < k) {</pre>
                 max_and = current_and;
             if (current_or > max_or && current_or < k) {</pre>
                 max_or = current_or;
             if (current_xor > max_xor && current_xor < k) {</pre>
                 max_xor = current_xor;
     printf("%d\n%d\n", max_and, max_or, max_xor);
/int main() {
     int n, k;
     scanf("%d %d", &n, &k);
     calculate_the_maximum(n, k);
    return 0;
```

You have passed the sample test cases. Click the submit button to run your code against all the test

⊘ Sample Test case 0

Input (stdin)

1 5 4

Your Output (stdout)

```
1 2 2 2 3 3 3
```

Expected Output

```
1 2
2 3
3 3
```