T2.

/\*

\* scanf t,n

\*

\* int count (int t, int i)

\* if(i-- > 1)

\* return t+count(t, i)\*10;

\*return t;

\*

\* int sum

\* for(i = 1 ;<=n)

\* 1.count 'tt' //fuc

\* 2.sum += count(n, i)

\*

\*/

T3

/\*

\* scanf n

\*

\* int count = 0;

\*

\* bool IsAbsolutePrime(int num)

\* if(!IsPrime(num)) return false;

\* int sub\_num = swap\_int(num);

\* IsPrime```

\*

\* bool IsPrime(int num)

\* int x; // 要判定是否是素数的数

int flag = 1; // 1 表示该数是素数

for (int i = 2; i <= sqrt(x + 0.5); i++) {

if (x % i == 0) {

flag = 0;

}

}

\*

\* int swap\_int(int num)

\* int bit\_int[4]

\* int times = 0;

\* int return\_value =0;

\* while(num/10)

\* bit\_int[times++] = num%10;

\* num /= 10;

\*

\*

\* while(bit\_int[times]&&times>=0)

\* ret += bit\_int[times--];

\* ret \*= 10;

\*

\*

\* for(int i = 1 <=n )

\* if(Is`Prime(i))

\* count++

\*

\* printf("%d\n",count)

\*/

//

// Created by xiexu on 2022/10/28.

//

/\*

\* difficulty ++(--)

\*

\*

\* int JudgeRadix(``)

\* for(int i= 2 <=40)

\* if(IsRadix(i,``)) return i;

\* return 0;

\*

\*

\* bool IsRadix``

\* int sub\_p = ChangeTenthRadix(p,i)

\* ``

\* ``

\* if(sp\*sq==sr)

\* return true;

\* return false;

\*

\* int ChangeTenthRadix(num, radix)

\*

\* if(num/10)

\* return radix\*(num/10) + num%10

\* else

\* return num

\*

\*

\* scanf p q r

\*

\* printf(, JudgeRadix(p,q,r))

\*

\*

\*/

6 3 1 5 4 9 8 7 2

6 3 1 5 7 9 8 4 2

\*求下一个排列的方法

找出给出的排列的最长连续递降后缀，其长度记为 m(m < n)m(m<n)。

找到最长递降后缀的之前的那个数，记为 p\_kp

k

，在最长递降后缀中找到比大于 p\_kp

k

的数中最小的那个，交换这两个数。

将交换后的序列的后 mm 个数翻转。

\*

\*int sizes = 0;

\* char input[2000] = {0};

\*

\* scanf sizes

\* for(1f0) - scanf(" %c",input[i])

\*

\* int (m)max\_drop\_range = FindMaxDropRange(input, sizes)

\* index\_pro\_char(pk) = sizes-1-m

\* int index\_min\_char = FindMinIndex(input,m, sizes, input[i\_p\_c])

\* SwapChar(input,index\_min\_char,index\_pk)

\* SwapCharArray(input, m, sizes) \*not qsort\*

\* PrintArray(input,sizes)

\*

\*

\* FindMaxDropRange(input, sizes)

\* for(int i = 0; i < sizes - 1; i++) \*?\*

\* if(input[s-1-i] > input[s-2-i])

\* return i+1;

\*

\* FindMinIndex(input,m, sizes,char)

\* for(int i = 0 ;< m-1)

\* if(input[sizes-1-i]>char)

\* return sizes -1 -i;

\*

\* SwapChar(input,index\_min\_char,index\_pk)

\* char tmp = ``

\*

\* SwapCharArray(input, m, sizes)

\* //qsort();

\* char\* left = input +sizes -m,right=input +sizes -1;

\* while(left<right)

\* char tmp``

\* l++

\* r--

\*

\* PrintArray(input,sizes)

\* for(if0 <sizes-1)

\* printf``

\*

\*/

//

// Created by xiexu on 2022/10/28.

//

/\*

\* int n

\* scanf n

\* int square[1000][1000] ={};

\* int x ,y;

\*

\*

\* FillSquare(i,square[1000][1000],int\* pointer\_x,``y,n)

\* Deduce (p\_x,p\_y,n);

\*

\* square[\*p\_x][\*p\_y] = i;

\*

\* Deduce(int\* p\_x,``,int n,int square[1000][1000])

\* \*p\_x++;``;

\*

\* if(p\_x > n-1)

\* p\_x -= n;

\* ``

\*

\* if(s[p\_x][p\_y] !=0)

\* \*p\_x --;

\* \*p\_y -=2;

\*

\*

\*

\*

// \* if(i == 1)

// \* square[0][i/2] = i;

// \* else if()

\*

\*

\* for(if1 <= n\*n)

\* if(i == 1)

\* x = 0;y = n/2;

\* square[x][y] = i;

\* else

\* FillSquare(i,square,&x,&y,n)

\*

\* PrintSquare(square,n)

\*

\*

Modify\_parameter(n, pointer\_col);

\*/