**第1关：从自然数中取3个数进行组合之循环算法**

**本关任务：用循环算法找出 5 个自然数中取 3 个数的组合。**

**测试输入：5 3 （n=5，r=3；，表示从1,2,3,4,5自然数中选择 3 个数)**

**预期输出：**

**1 2 3**

**1 2 4**

**1 2 5**

**1 3 4**

**1 3 5**

**1 4 5**

**2 3 4**

**2 3 5**

**2 4 5**

**3 4 5**

**#include <stdio.h>**

**void combloop1(int n, int r)**

**{**

**/\*\*\*\*\*\*\*\*\*\* Begin \*\*\*\*\*\*\*\*\*\*/**

**int i,j,k,t;**

**t=0;**

**for(j=1;j<=n;j++)**

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

**for(k=1;k<=n;k++)**

**if((j!=i)&&(j<i)&&(i<k))**

**{**

**t=t+1;**

**printf("%d %d %d",j,i,k);**

**printf("\n");**

**}**

**/\*\*\*\*\*\*\*\*\*\* End \*\*\*\*\*\*\*\*\*\*/**

**}**

**void main()**

**{**

**int n,r;**

**scanf("%d%d",&n,&r);**

**combloop1(n,r);**

**}**

**第2关：从自然数中取3个数进行组合之递归算法**

**本关任务：用递归算法找出 5 个自然数中取 3 个数的组合。**

**测试输入：5 3 （n=5，r=3；，表示从1,2,3,4,5自然数中选择 3 个数)**

**预期输出：**

**5 4 3**

**5 4 2**

**5 4 1**

**5 3 2**

**5 3 1**

**5 2 1**

**4 3 2**

**4 3 1**

**4 2 1**

**3 2 1**

**#include <stdio.h>**

**int a[100];**

**void combrecur(int n, int r)**

**{**

**/\*\*\*\*\*\*\*\*\*\* Begin \*\*\*\*\*\*\*\*\*\*/**

**int i,j;**

**for(i=n;i>=r;i--){**

**a[r]=i;**

**if(r>1){**

**combrecur(i-1,r-1);**

**}**

**else{**

**for(j=a[0];j>0;j--){**

**printf("%d",a[j]);**

**printf(" ");**

**}**

**printf("\n");**

**}**

**}**

**/\*\*\*\*\*\*\*\*\*\* End \*\*\*\*\*\*\*\*\*\*/**

**}**

**void main()**

**{**

**/\*\*\*\*\*\*\*\*\*\* Begin \*\*\*\*\*\*\*\*\*\*/**

**int n,r;**

**scanf("%d %d",&n,&r);**

**if(n>r){**

**a[0]=r;**

**combrecur(n,r);**

**}**

**/\*\*\*\*\*\*\*\*\*\* End \*\*\*\*\*\*\*\*\*\*/**

**}**

**第3关：[求n的阶乘n!](https://www.educoder.net/tasks/eomjf2n86cbz?coursesId=36vpfaz4)**

**本关任务：用循环和递归算法求 n（小于 10 的正整数） 的阶乘 n!。**

**测试输入：10**

**预期输出：**

**递归算法求得10! = 3628800**

**循环算法求得10! = 3628800**

**#include <stdio.h>**

**float fac\_recursion(int n)**

**{**

**/\*\*\*\*\*\*\*\*\*\*  Begin  \*\*\*\*\*\*\*\*\*\*/**

**if(n==1)**

**return 1;**

**else**

**return n\*fac\_recursion(n-1);**

**/\*\*\*\*\*\*\*\*\*\*  End  \*\*\*\*\*\*\*\*\*\*/**

**}**

**int S=1;**

**float fac\_loop(int n)**

**{**

**/\*\*\*\*\*\*\*\*\*\*  Begin  \*\*\*\*\*\*\*\*\*\*/**

**for(int i=n;i>=1;i--){**

**S\*=i;**

**}**

**return S;**

**/\*\*\*\*\*\*\*\*\*\*  End  \*\*\*\*\*\*\*\*\*\*/**

**}**

**void main()**

**{**

**int n;**

**float y;**

**scanf("%d",&n);**

**y=fac\_recursion(n);**

**printf("递归算法求得%d! = %.0f \n",n,y);**

**y=fac\_loop(n);**

**printf("循环算法求得%d! = %.0f \n",n,y);**

**}**

**第4关：求斐波那契额数列的前10项**

**本关任务：用循环和递归算法求斐波那契额数列的前 10 项。**

**测试输入：10**

**预期输出：**

**递归算法求的前10项为: 1 1 2 3 5 8 13 21 34 55**

**循环算法求的前10项为: 1 1 2 3 5 8 13 21 34 55**

**#include <stdio.h>**

**int fibo\_recur(int n)**

**{**

**/\*\*\*\*\*\*\*\*\*\* Begin \*\*\*\*\*\*\*\*\*\*/**

**if(n==1 || n==2){**

**return 1;**

**}**

**else{**

**return fibo\_recur(n-1)+fibo\_recur(n-2);**

**}**

**/\*\*\*\*\*\*\*\*\*\* End \*\*\*\*\*\*\*\*\*\*/**

**}**

**int fibo\_loop(int n)**

**{**

**/\*\*\*\*\*\*\*\*\*\* Begin \*\*\*\*\*\*\*\*\*\*/**

**int a=1,b=1,c;**

**if(n<=2){**

**return a;**

**}else{**

**for(int i=3;i<=n;i++){**

**c=a+b;**

**a=b;**

**b=c;**

**}**

**return c;**

**}**

**/\*\*\*\*\*\*\*\*\*\* End \*\*\*\*\*\*\*\*\*\*/**

**}**

**void main()**

**{**

**int n,y,i;**

**scanf("%d",&n);**

**printf("递归算法求的前10项为:");**

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

**{**

**y=fibo\_recur(i);**

**printf("%3d",y);**

**}**

**printf("\n循环算法求的前10项为:");**

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

**{**

**y=fibo\_loop(i);**

**printf("%3d",y);**

**}**

**}**

**假设在文本"ababcabccabccacbab"中查找"abccac"，写出采用KMP算法的串匹配过程。**

**1. 构建模式串的最长前缀和最长后缀数组：**

**模式串："abccac"**

**最长前缀和最长后缀数组：[0, 0, 0, 0, 1, 0]**

**2. 开始匹配：**

**文本串："ababcabccabccacbab"**

**模式串："abccac"**

**开始匹配位置：0**

**3. 匹配过程：**

**- 在位置0匹配成功，移动模式串和文本串的指针**

**文本串："ababcabccabccacbab"**

**模式串："abccac"**

**- 在位置1匹配成功，移动模式串和文本串的指针**

**文本串："ababcabccabccacbab"**

**模式串："abccac"**

**- 在位置2匹配失败，根据最长前缀和最长后缀数组移动模式串的指针，移动距离为2**

**文本串："ababcabccabccacbab"**

**模式串："abccac"**

**移动后：**

**文本串："ababcabccabccacbab"**

**模式串：    "abccac"**

**- 在位置4匹配失败，根据最长前缀和最长后缀数组移动模式串的指针，移动距离为4-1=3**

**文本串："ababcabccabccacbab"**

**模式串：    "abccac"**

**移动后：**

**文本串："ababcabccabccacbab"**

**模式串：           "abccac"**

**- 在位置5匹配成功，移动模式串和文本串的指针**

**文本串："ababcabccabccacbab"**

**模式串：                   "abccac"**

**- 匹配成功，返回匹配位置为5**

**2. (简答题)已知数组A[n]的元素为整型，设计算法将其调整为左右两部分，左边所有元素为奇数，右边所有元素为偶数，要求时间复杂度为O(n),空间复杂度为O(1)。**

**双指针。左指针指向数组的开头，右指针指向数组的末尾。然后，不断移动指针，并交换元素的位置，直到左指针和右指针相遇。**

**具体步骤如下：**

**1. 初始化左指针left为0，右指针right为n-1。**

**2. 重复以下步骤，直到左指针和右指针相遇：**

**- 从左指针开始，向右移动left，直到找到一个偶数。**

**- 从右指针开始，向左移动right，直到找到一个奇数。**

**- 如果left < right，交换A[left]和A[right]的值。**

**3. 返回调整后的数组A。**