**/\*1、一只青蛙一次可以跳上 1 级台阶，也可以跳上2 级。求该青蛙跳上一个n 级的台阶总共有多少种跳法。**

**设计要求：**

**（1）、主函数中输入台阶个数，自定义函数解决计算过程，计算结果在主函数中输出。**

**（2）、分别使用递归方法和常规方法（即循环结构）完成以上设计。**

**输入输出样例1：**

**10**

**89**

**输入输出样例2：**

**20**

**10946 \*/**

**//常规方法**

**#include <stdio.h>**

**int sum(int x);**

**int main()**

**{**

**int n; //n为台阶数**

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

**printf("%d", sum(n));**

**return 0;**

**}**

**int sum(int x)**

**{**

**int i, f1 = 1, f2 = 2;**

**if(x == 1 || x == 2)//当台阶数小于3时**

**return x;**

**else**

**{**

**for(i = 3; i < x + 1; i++)**

**{**

**f1 = f1 ^ f2; //f2和f1交换位置**

**f2 = f1 ^ f2;**

**f1 = f1 ^ f2;**

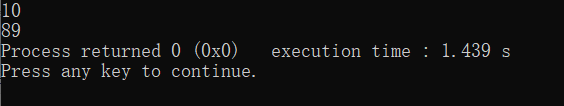
**f2 += f1;**

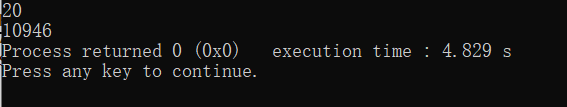
**}**

**return f2;**

**}**

**}**

****

****

**//递归方法**

**#include <stdio.h>**

**int sum(int x);**

**int main()**

**{**

**int n; //n为台阶数**

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

**printf("%d", sum(n));**

**return 0;**

**}**

**int sum(int x)**

**{**

**int f;**

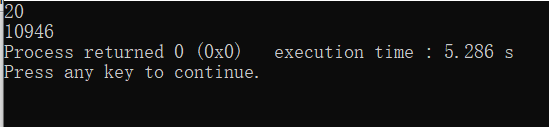
**if(x > 3)**

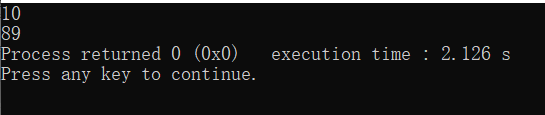
**f = sum(x - 2) + sum(x - 1);**

**else f = x;**

**return f;**

**}**

****

****

**/\*2、 自定义两个函数，分别求两个正整数的最大公约数和最小公倍数，用主函数调用这两个函数并输出结果。**

**要求完成两组测试结果并附上截图。\*/**

**#include <stdio.h>**

**#include <stdlib.h>**

**int gys(int x, int y) //求最大公约数的函数**

**{**

**int t, a;**

**for(a = 1; a < x + 1 && a < y + 1; a++)**

**{**

**if(x % a == 0 && y % a == 0)**

**t = a;**

**}**

**return t;**

**}**

**int gbs(int x, int y) //求最小公倍数的函数**

**{**

**int a;**

**if(x > y)**

**a = x;**

**else a = y;**

**while(a % x != 0 || a % y != 0)**

**++a;**

**return a;**

**}**

**int main()**

**{**

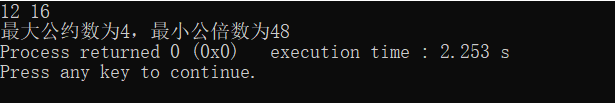
**int m, n;**

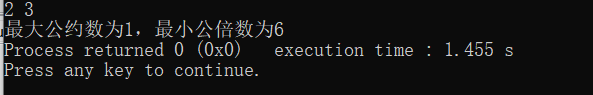
**scanf("%d%d", &m, &n);**

**printf("最大公约数为%d，最小公倍数为%d", gys(m, n), gbs(m, n));**

**return 0;**

**}**

****

****

**/\*3、自定义一个函数，找出一行英文句子中最长的单词出现的开始位置和最长单词的字符个数，英文句子的输入和最终输出结果要求在主函数中实现。**

**输入样例：各单词间保留一个空格**

**I am a student**

**\*/**

**#include <stdio.h>**

**void locate(char a[], int n[]);**

**int main()**

**{**

**char a[100];**

**int n[2] = {0};**

**gets(a);**

**locate(a, n);**

**printf("%d %d", n[0], n[1]);**

**return 0;**

**}**

**void locate(char a[], int n[])**

**{**

**int i, num = 0, loc, longest\_word\_num, location; //location表示最长单词开始位置, num储存单词字符数**

**char c;**

**for(i = 0; (c = a[i]) != '\0'; i++)**

**{**

**++loc;**

**if(c != ' ')**

**{**

**++num;**

**if(num > longest\_word\_num)**

**{**

**longest\_word\_num = num;**

**location = loc;**

**}**

**}**

**else num = 0;**

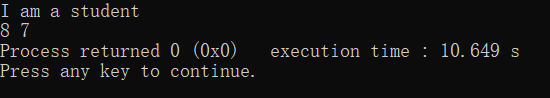
**}**

**location -= longest\_word\_num - 1;**

**n[0] = location;**

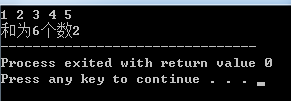
**n[1] = longest\_word\_num;**

**}**

****

**/\* 4、定义一个函数，实现对输入的数组中的偶数进行求和，同时计算偶数的个数，统计结果要求在主函数中输出。**

**输入输出样例：**

** \*/**

**#include <stdio.h>**

**void array\_sum(int a[],int x, int mem[]);**

**int main()**

**{**

**int member[2] = {0},a[100], i, b, n = 0;**

**printf("用输入字母结束循环，每个数字间间隔一个空格\n");**

**for(i = 0; scanf("%d",&b) != 0; i++) //给数组赋值**

**{**

**a[i] = b;**

**n++;**

**}**

**array\_sum(a, n, member);**

**printf("%d %d", member[0], member[1]);**

**return 0;**

**}**

**void array\_sum(int a[],int x, int mem[])**

**{**

**int i, sum = 0, count = 0;**

**for(i = 0; i < x; i++)**

**{**

**if(a[i] % 2 == 0)**

**{**

**sum += a[i];**

**count++;**

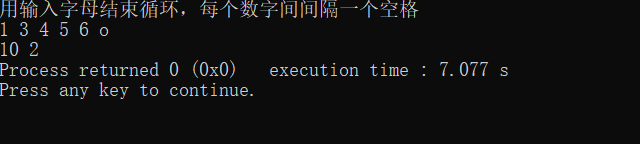
**}**

**}**

**mem[0] = sum;**

**mem[1] = count;**

**}**

****