**算法分析与设计**

1. 给出穷举法的定义、描述其基本步骤，并给出其抽象算法

定义：穷举法就是通过把需要解决问题的所有可能情况逐一试验来找出符合条件的解的方法。

步骤： 首选(first)：产生问题P的第一个候选解；

验证(valid)：检查候选解c是否为问题P的解；

输出(output)：如果c为P的解，则将其输出；

再选(next)：从当前候选解c顺次产生下一个候选解。

算法： 算法名称：抽象穷举法

输入：问题实例P

输出：问题的解

01 c←first(P)

02 **while** c≠A

03 **if** valid(P,c) then

04 output(P,c)

05 c←next(P)

06 end **while**

1. 描述测试素数的试除算法，并编程实现

01 /\*

02 \* 测试素数的试除算法

03 \* 2019年9月4日

04 \*/

05 #include <iostream>

06 **using namespace** std;

07

08 //素数判断函数，如果是素数返回True

09 **bool** PrimeNumJudger(**int** num){

10 **for**(**int** i = 2;i\*i<=num;i++){

11 **if** (num % i == 0){

12 **return false**;

13 }

14 }

15 **return true**;

16 }

17

18 **int** main() {

19 **int** num = 5;

20 **if**(PrimeNumJudger(num)){

21 cout<<"yes"<<endl;

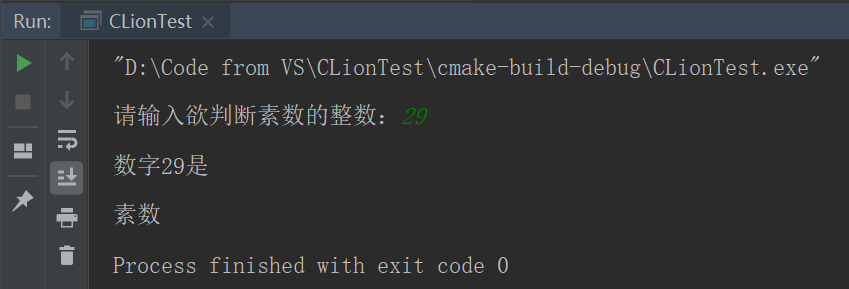
22 }**else**{

23 cout<<"no"<<endl;

24 }

25 **return** 0;

26 }



1. 挑战：如何判断一个很大的数，如1024位，是否为素数

01 /\*

02 \* 大整数测试素数算法

03 \* 2019年9月5日

04 \*/

05

06 #include<iostream>

07 #include<string>

08 #include<cstdio>

09 #include <cstring>

10 **using namespace** std**;**

11

12 **int** bigIntPrime**(char\*** n**,** **int** m**){**

13 **int** ans **=** 0**;**

14 **int** len **=** strlen**(**n**);**

15 **for** **(int** i **=** 0**;** i **<** len**;** i**++)**

16 // ans = (int)((((long long)ans \* 10) % m + (n[i] - '0') % m) % m);

17 ans**=(**ans**\***10**+**n**[**i**]-**'0'**)%**m**;**

18 **return** ans**;**

19 **}**

20

21 **bool** primeJudger**(char** **\*** n**){**

22 **for(int** i**=**2**;**i**\***i**<(long long)**n**;**i**++){**

23 **if(**bigIntPrime**(**n**,**i**)==**0**){**

24 **return false;**

25 **}**

26 **}**

27 **return true;**

28 **}**

29 **int** main**(){**

30 **char** n**[**1024**];**

31 cout**<<**"请输入欲判断素数的大整数（<=1024位）："**;**

32 cin**>>**n**;**

33 cout**<<**"数字"**<<**n**;**

34 **if(**primeJudger**(**n**)){**

35 cout**<<**"是素数"**;**

36 **}else{**

37 cout**<<**"不是素数"**;**

38 **}**

39

40

41 **return** 0**;**

42 **}**



1. 在CAAIS中练习洗牌算法和二分搜索算法
2. 编程实现二分搜索算法

01 #include <iostream>

02 #include <cstdlib>

03 #include <ctime>

04 #include<cstdio>

05 #include<algorithm>

06 #define N 100

07 **using namespace** std**;**

08

09 **int\*** createRandArray**(int** **\***n**){**

10 srand**((int)**time**(**0**));** // 产生随机种子 把0换成NULL也行

11 **for** **(int** i **=** 0**;** i **<** N**;** i**++){**

12 n**[**i**] =** rand**()%**100**;**

13 **}**

14 **return** n**;**

15 **}**

16

17 **int\*** sortArray**(int** **\***n**){**

18 sort**(**n**,**n**+**N**);**

19 **return** n**;**

20 **}**

21

22 //查找函数

23 **int** binarySearch**(int** **\***n**,int** key**){**

24 **int** mid**,** low **=** 1**,** high **=** N**;**

25 **while** **(**low **<=** high**)**

26 **{**

27 mid **= (**low **+** high**) /** 2**;**

28 **if** **(**key **==** n**[**mid **-** 1**])** **return** mid**;**//找到结果，返回结果

29 **else if** **(**key **<** n**[**mid **-** 1**])** high **=** mid **-** 1**;**

30 **else** low **=** mid **+** 1**;**

31 **}**

32 **return** 0**;**

33 **}**

34

35 //打印结果函数

36 **void** print**(int** result**)**

37 **{**

38 **if** **(**result**)** printf**(**"\n查找的数字是第%d个\n\n"**,** result**);**

39 **else** printf**(**"查找失败\n\n"**);**

40 **}**

41

42 **int** main**()**

43 **{**

44 **int** n**[**N**];**

45 **int** key**;** // 关键值

46 createRandArray**(**n**);**

47 cout**<<**"随机生成长度为"**<<**N**<<**"的数组"**<<**endl**;**

48 sortArray**(**n**);**

49 cout**<<**"数组排序完毕"**<<**endl**;**

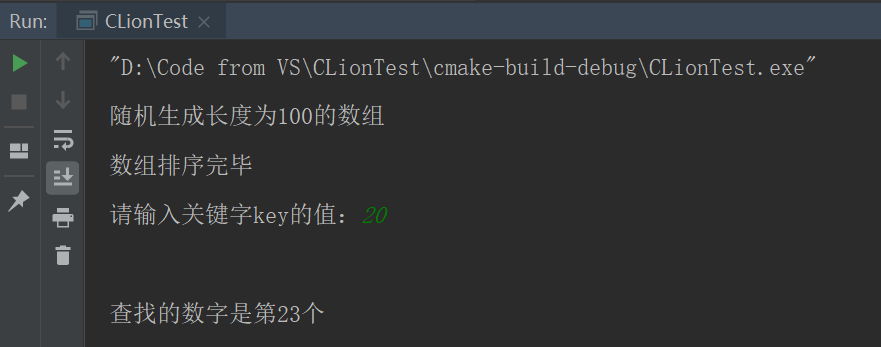
50 cout**<<**"请输入关键字key的值："**;**

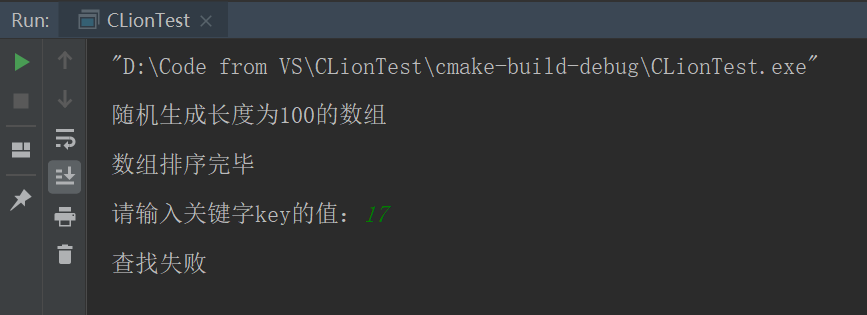
51 cin**>>**key**;**

52 print**(**binarySearch**(**n**,** key**));** // 打印查找结果,如果rearch返回为序号，则打印序号，返回0，则查找失败

53 **return** 0**;**

54 **}**





1. 挑战：编程实现二分搜索算法的递归版