7.10

7.15

#include<iostream>

using namespace std;

bool find(int, int[], int , int);//查找数组中是否含有某数字

const int arrSize = 20;

int main() {

int arr[arrSize] = {};

int counter = 0;//输入数

int index = 0;//合法元素末索引

while (counter < 20) {

int temp = 0;

cout << "Type in a 10-100 number(inclusive) (" << counter + 1 << " of 20) : ";

cin >> temp;

//输入检查

if (temp < 10 || temp > 100) {

cout << "Illegal Input!" << endl;

continue;

} else {

if (counter == 0) {//首项录入

arr[index] = temp;

index++;

} else {//后续项录入

if (!find(temp, arr, 0, counter - 1)) {

arr[index] = temp;

index++;

}

}

}

counter++;

}

for (int i = 0; i < index; i++)

cout << arr[i] << ' ';

system("pause");

}

bool find(int finding, int arr[], int start, int end) {

if (arr[start] == finding)

return true;

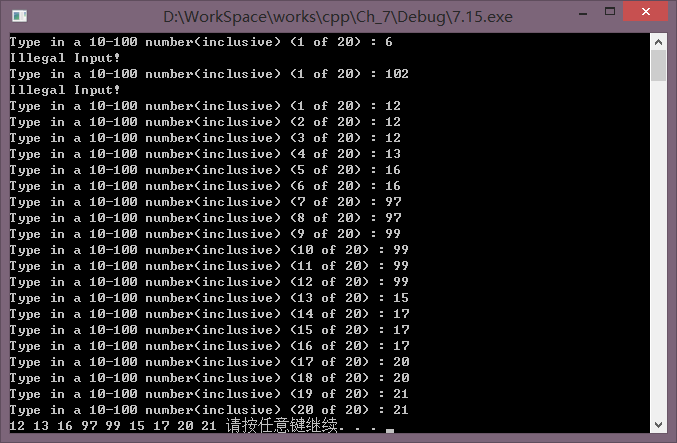
if (start == end)

return false;

else

return find(finding, arr, start + 1, end);

}



7.20

7.32

#include <iostream>

#include <string>

using namespace std;

bool testPalindrome(string, int, int);

int main() {

string str;

while (cin) {

getline(cin, str);

cout << (testPalindrome(str, 0, str.length() - 1) ? "Palindrome!" : "Not a palindrome!") << endl;

}

}

bool testPalindrome(string str, int startIndex, int endIndex) {

//每次检查首尾2个字符是否一致

int temp1 = startIndex, temp2 = endIndex;

//忽略非字母

while (!((str[temp1] >= 'a' && str[temp1] <= 'z') || (str[temp1] >= 'A' && str[temp1] <= 'Z')))

temp1++;

while (!((str[temp2] >= 'a' && str[temp2] <= 'z') || (str[temp2] >= 'A' && str[temp2] <= 'Z')))

temp2--;

if (temp1 > temp2)//若光标移动过数组中间则停止

return true;

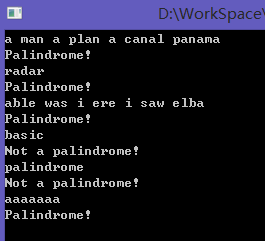
if (str[temp1] != str[temp2])//判断首尾是否相同

return false;

else

return testPalindrome(str, temp1 + 1, temp2 - 1);//继续检查下一组首尾字符

}



7.36

#include <iostream>

#include <string>

using namespace std;

void stringReverse(char[], int);

int main() {

while (cin) {

char str[1024] = {};

cin.getline(str, 1024);

stringReverse(str, 0);

cout << endl;

}

}

void stringReverse(char str[], int start) {

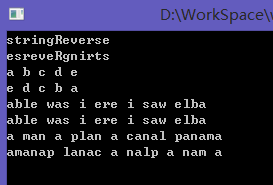
if (str[start] == '\0')

return;

stringReverse(str, start + 1);

cout << str[start];

}



7.37

#include<iostream>

using namespace std;

int\* recursiveMinimum(int[], int, int);

const int arrSize = 10;

int main() {

int arr[arrSize] = { 16,20,8,45,63,12,89,48,76,32 };

for (int i = 0; i < arrSize; i++)

cout << arr[i] << ' ';

cout << endl;

cout << "The minimum is " << \*recursiveMinimum(arr, 0, arrSize - 1) << endl;

system("pause");

}

int\* recursiveMinimum(int arr[], int start, int end) {

if (start == end)

return &arr[start];

if (arr[start] >= arr[end])

return recursiveMinimum(arr, start + 1, end);

else

return recursiveMinimum(arr, start, end - 1);

}

