**实验2 芯片测试算法设计**

一、【实验目的】

（1）理解分治策略的设计思想；

（2）熟悉将伪码转换为可运行的程序的方法；

（3）能够根据算法的要求设计具体的实例。

二、【实验内容】

有n片芯片，其中好芯片比坏芯片至少多1片，现需要通过测试从中找出1片好芯片。测试方法是:将2片芯片放到测试台上，2片芯片互相测试并报告测试结果:“好”或者“坏”。假设好芯片的报告是正确的，坏芯片的报告是不可靠的。请设计一个算法，使用最少的测试次数来找出1片好芯片。程序结果返回找到的好芯片的序号。

提示:可参考教材P29页的算法2.3. 测试函数可以采用以下方法。

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

//建立测试函数，参数iA表示主动测试芯片， iB表示被测芯片.返回值为被测芯片的测试值，值为1时表示好芯片，为0时表示坏芯片.可以使用以下函数：

**int X\_test(int iA,int iB)**

**{**

**if(iA==1)**

**return iB;**

**return rand()%2;**

**}**

//应用随机数来表示不确定的值1、0.注意:在主函数中加上随机数种子语句srand(time(NULL));

//算法的输入可以用数组表示，比如:ABc[17]={1,0,0,1,1,1,0,0,1,1,1,1,0,1,0,0,0} 表示17个芯片，其中9片好芯片、8片坏芯片。

三、实验源代码

1.头文件

#include <iostream>

#include <iterator>

#include <stdlib.h>

#include <vector>

class chip{

private:

    bool val;

    int index;

public:

    chip(bool vall,int index){

        this->val=vall;

        this->index=index;

    }

    bool test\_to(chip be\_test){

        if(this->val==true)//好

            return be\_test.val;

        return rand()%2;//不好

    }

    void print(){

        std::cout<<"chips["<<this->index<<"]:"<<this->val<<'\t';

    }

};

void insert\_test(std::vector<chip>& chips){

    int a[]={1,0,1,0,1,0,1,1};

    int i=0;

    for(int& aa:a){

        chip c1(aa,i);

        ++i;

        chips.emplace\_back(c1);

    }

}

void print\_all(std::vector<chip>& chips){

    if(chips.empty()){

    std::cout<<"empty"<<std::endl;

    return;

    }

    for(chip& ch:chips)

        ch.print();

    std::cout<<std::endl;

}

void te\_print(int n){

    std::cout<<"-----"<<n<<"-----"<<std::endl;

}

===============================================================================

#include <iostream>

#include <vector>

#include "e2.hpp"

using namespace std;

void conquer(vector<chip>& chips){

    if(chips.size()==1){

        return;

    }

    vector<chip>::iterator fast;

    vector<chip>::iterator slow;

    fast=chips.begin();

    ++fast;

    slow=chips.begin();

    while (slow!=chips.end() && fast!=chips.end()) {

        vector<chip>::iterator tmp;

        if(fast->test\_to(\*slow) && slow->test\_to(\*fast)){

            auto iter=chips.erase(slow);

            ++slow;

            ++fast;

        }

        else {

            fast=chips.erase(fast);

            slow=chips.erase(slow);

        }

    }

}

int main(){

    vector<chip> chips;

    insert\_test(chips);

    //vector<chip> res;//vv

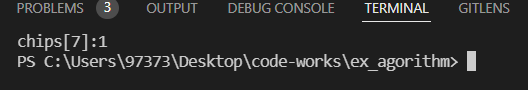
    conquer(chips);

    print\_all(chips);

    return 0;

}

1. 实验结果（运行结果截图）



1. 实验小结

（1）手动执行算法，请分析你的程序是怎么得出结果的，写出步骤，是否找到了好的芯片。（2）你的程序执行结果是否唯一？如果不唯一，有哪些可能的解？（3）此算法对你理解分治策略是否有帮助？什么是分治策略？

