

《程序设计艺术与方法》课程实验报告

实验名称	实验一 STL 的熟悉与使用						
姓 名	刘铭源	系院专业	软件工程	班 级	18-4	学 号	2018214937
实验日期			指导教师	徐本柱		成 绩	
一、实验目的和要求 1. 掌握 C++ 中 STL 的容器类的使用; 2. 掌握 C++ 中 STL 的算法类的使用.							
二、实验预习内容 1. 预习 ICPC 讲义, 大致了解 STL 的相关内容。 2. 了解 STL 中一些类 vector list 类的使用方法 3. 了解泛型算法的使用							
三、实验项目摘要 1. 练习 vector 和 list 的使用 2. 练习泛型算法的使用							
四、实验结果与分析（源程序及相关说明） 1.vector 使用 <pre>#include<vector> #include<iostream> #include<ctime> #include<cstdlib> #include<algorithm> using namespace std; int main() { clock_t start,end; start=clock(); vector<int> array1; vector<int>::iterator it1;</pre>							

```
vector<int>::iterator ip;
srand(time(0));
for(int i = 0; i < 10; i++)
{
    array1.push_back(rand()%100);
}

for(it1 = array1.begin(); it1 != array1.end(); it1++)
{
    cout <<*(it1)<<" ";
}

cout <<endl;
array1.insert(array1.begin(),rand());
for(it1 = array1.begin(); it1 != array1.end(); it1++)
{
    cout <<*(it1)<<" ";
}
cout <<endl;

int x = rand()%100;
ip=find(array1.begin(),array1.end(),x);
if(*ip!=x)
{
    cout<<"no,没有找到"<<endl;
    array1.insert(array1.end(),x);
}

else
    cout<<"YES, 找到了"<<*ip<<endl;

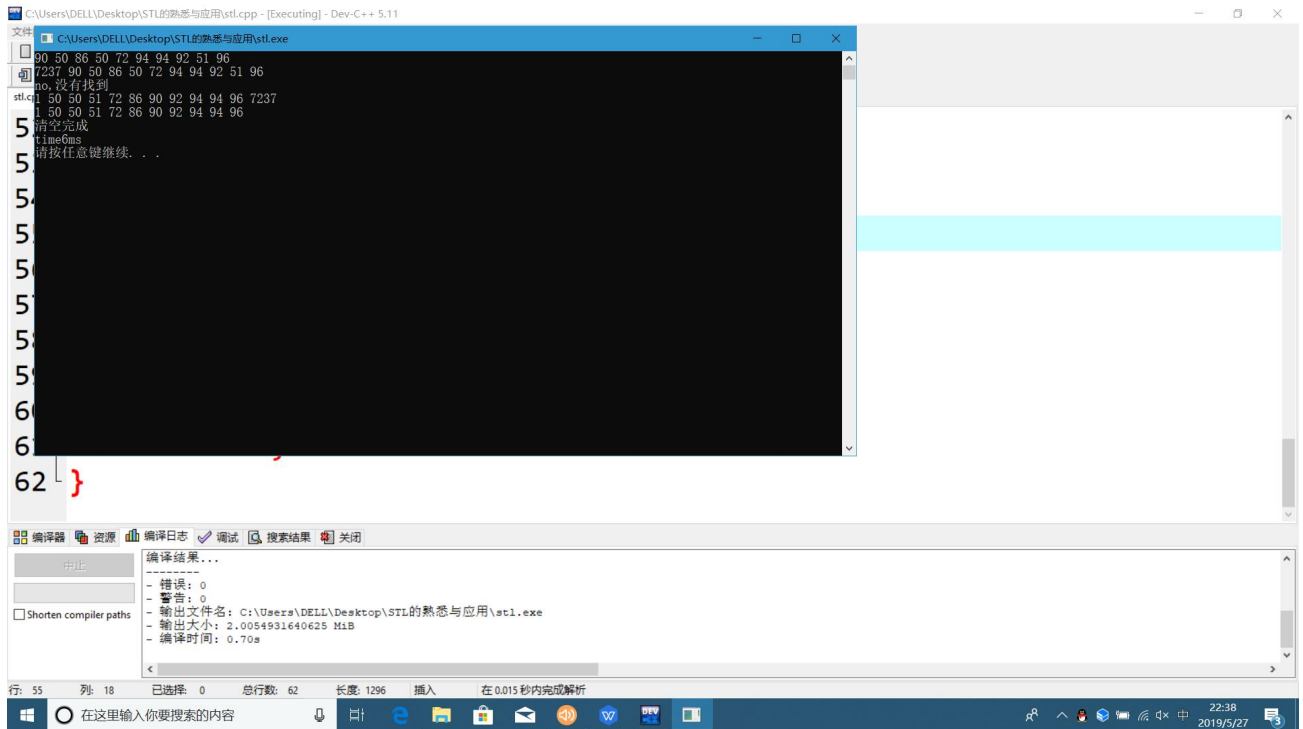
sort(array1.begin(),array1.end());
for(it1 = array1.begin(); it1 != array1.end(); it1++)
{
    cout <<*(it1)<<" ";
}
cout <<endl;

array1.pop_back();
for (it1 = array1.begin(); it1 != array1.end(); it1++)
{
    cout << *it1 << " ";
}
cout <<endl;
```

```

array1.clear();
cout <<"清空完成"<<endl;
end=clock();
cout<<"time"<<end-start<<"ms"<<endl;
system("Pause");
return 0;
}

```



2.泛型编程

```

#include<iostream>
#include<list>
#include<ctime>
#include<cstdlib>
#include<algorithm>
using namespace std;
int main()
{
    clock_t start,end;
    start=clock();
    list<int> l;
    list<int>::iterator it1;
    list<int>::iterator ip;
    srand(time(0));
    for(int i = 0;i < 10; i++)
    {
        l.push_back(rand()%100);
    }
}

```

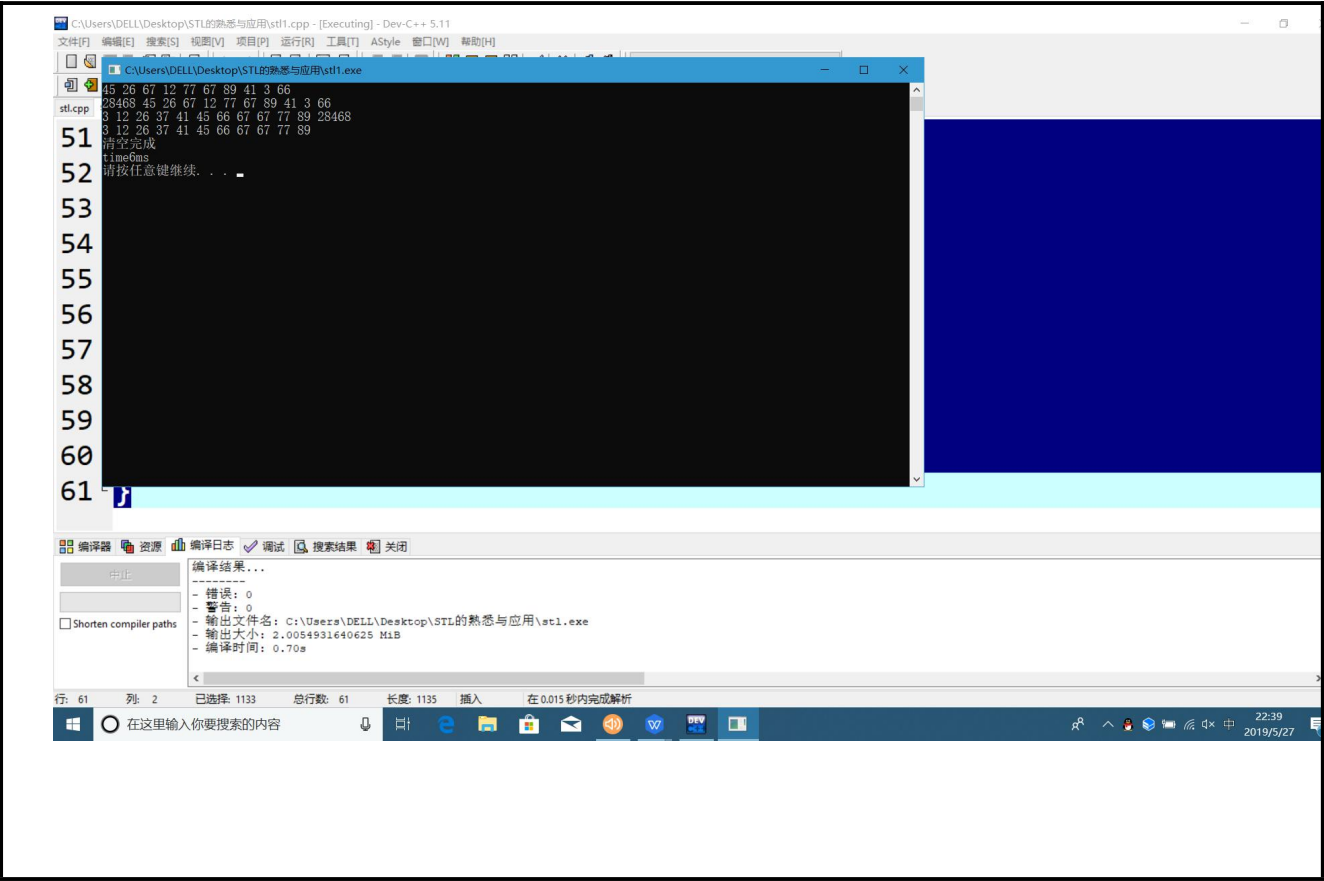
```
for(it1 = l.begin(); it1 != l.end(); it1++)
{
    cout <<*(it1)<<" ";
}

cout <<endl;
l.insert(l.begin(),rand());
for(it1 = l.begin(); it1 != l.end(); it1++)
{
    cout <<*(it1)<<" ";
}
cout <<endl;
int x = rand()%100;
ip = find(l.begin(),l.end(),x);
if(*ip!=x)
{
    l.insert(l.end(),x);
}

else
    cout<<"YES"<<*ip<<endl;
l.sort();
for(it1 = l.begin(); it1 != l.end(); it1++)
{
    cout <<*(it1)<<" ";
}
cout <<endl;
l.pop_back();
for (it1 = l.begin(); it1 != l.end(); it1++)
{
    cout << *it1 << " ";
}
cout <<endl;

l.clear();
cout <<"清空完成"<<endl;
end=clock();
cout<<"time"<<end-start<<"ms"<<endl;
system("Pause");

return 0;
}
```



《程序设计艺术与方法》课程实验报告

实验名称	实验二 搜索算法的实现						
姓 名	刘铭源	系院专业	软件工程	班 级	18-4	学 号	2018214937
实验日期			指导教师	徐本柱		成 绩	

一、实验目的和要求

1. 掌握宽度优先搜索算法。
2. 掌握深度优先搜索算法。

二、实验预习内容

1. 预习 ICPC 讲义中的搜索的内容
2. 了解什么是深度优先搜索和广度优先搜索。

三、实验项目摘要

1. 将书上的走迷宫代码上机运行并检验结果，并注意体会搜索的思想。
2. 八皇后问题：在一个国际象棋棋盘上放八个皇后，使得任何两个皇后之间不相互攻击，求出所有的布棋方法。上机运行并检验结果。
3. 骑士游历问题：在国际棋盘上使一个骑士遍历所有的格子一遍且仅一遍，对于任意给定的顶点，输出一条符合上述要求的路径。
4. 倒水问题：给定 2 个没有刻度容器，对于任意给定的容积，求出如何只用两个瓶装出 L 升的水，如果可以，输出步骤，如果不可以，请输出 No Solution。

四、实验结果与分析（源程序及相关说明）

```
1.#include<iostream>
#include<ctime>
using namespace std;
int board[8][8] = {0};
void travel(int a, int b)
{
    int knight_direction[8]={-2,-2,-1,-1,1,1,2,2};
    int knight_num[8]={1,-1,2,-2,2,-2,1,-1};
    int next_d[8]= {0};
    int next_n[8]= {0};
    int exists[8] = {0};
    int i,j,temple_i,temple_j,mine,temple;
    i = a;
    j = b;
    board[i][j] = 1;

    for(int foot= 2; foot <65; foot++)
    {
        for(int c = 0; c < 8; c++)
            exists[c] = 0;
```

```
int p = 0;
for(int k = 0; k < 8; k++)
{
    temple_i = i + knight_direction[k];
    temple_j = j + knight_num[k];
    if(temple_i < 0||temple_j < 0||temple_i > 7||temple_j > 7)
        continue;
    if(board[temple_i][temple_j]==0)
    {
        next_d[p] =temple_i;
        next_n[p] = temple_j;
        p++;
    }
}
if(p == 0)
{
    return ;
}
else if(p==1)
{
    mine = 0;
}
else
{
    for(int q = 0; q < p; q++)
    {
        for(int w = 0; w < 8; w++)
        {
            temple_i = next_d[q] + knight_direction[w];
            temple_j = next_n[q] + knight_num[w];
            if(temple_i < 0||temple_j < 0||temple_i > 7||temple_j > 7)
                continue;
            if(board[temple_i][temple_j]==0)
                exists[q]++;

        }
    }
    temple = exists[0];
    mine = 0;
    for(int d = 1; d< p;d++)
    {
        if(exists[d] < temple)
```

```
        {
            temple = exists[d];
            mine = d;
        }

    }

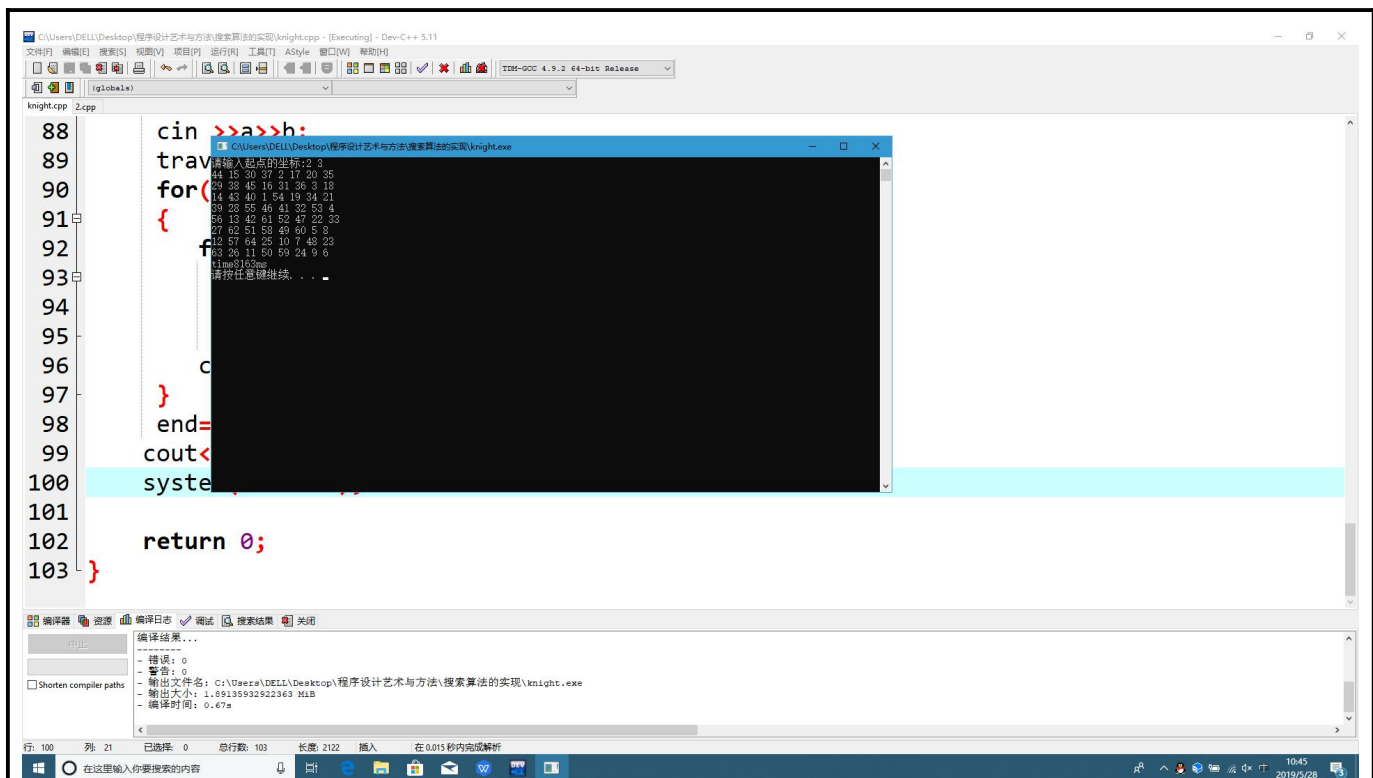
}

    i = next_d[mine];
    j = next_n[mine];
    board[i][j] = foot;
}

return ;
}

int main()
{
    clock_t start,end;
    start=clock();
    int a,b;
    cout<<"请输入起点的坐标:";
    cin >>a>>b;
    travel(a,b);
    for(int i = 0; i < 8; i++)
    {
        for(int j = 0; j < 8; j++)
        {
            cout <<board[i][j]<<" ";
        }
        cout <<endl;
    }
    end=clock();
    cout<<"time"<<end-start<<"ms"<<endl;
    system("Pause");

    return 0;
}
```

2. 八皇后问题

```
#include<iostream>
#include<conio.h>
#include<citme>
#include<math.h>
#define QUEENS 8
using namespace std;
int iCount = 0;
static int Site[QUEENS];
int m;
void Queen(int n);
void Output();
int Isok(int n);
void Queen(int n)
{
    int i;
    if(n==QUEENS)
    {
        Output();
    }
    for(i=1;i<=QUEENS;i++)
    {
        Site[n]=i;
        if(Isok(n))
```

```
{
    Queen(n+1);

}

}

}

int Isok(int n)
{
    for(int i=0;i<n;i++)
    {
        if(Site[i]==Site[n])return 0;
        if(abs(Site[i]-Site[n])==(n-i))return 0;

    }
    return 1;
}

void Output()
{

    cout<<"第"<<++iCount<<"种解法"<<"    ";
    for(int i=0;i<QUEENS;i++)
    {
        cout<<Site[i]<<" ";
    }
    cout<<endl;
}

int main()
{
    clock_t start,end;
    start=clock();
    cout<<"    "<<"八皇后解法"<<"    "<<endl;
    Queen (0);
    getch();
    end=clock();
    cout<<"time"<<end-start<<"ms"<<endl;
    system("Pause");

    return 0;
}
```

```
C:\Users\DELL\Desktop\4.exe
八皇后解法
第1种解法 1 5 8 6 3 7 2 4
第2种解法 1 6 8 3 7 4 2 5
第3种解法 1 7 4 6 8 2 5 3
第4种解法 1 7 5 8 2 4 6 3
第5种解法 2 4 6 8 3 1 7 5
第6种解法 2 5 7 1 3 8 6 4
第7种解法 2 5 7 4 1 8 6 3
第8种解法 2 6 1 7 4 8 9 5
第9种解法 2 6 8 3 1 4 7 5
第10种解法 2 7 3 6 8 5 1 4
第11种解法 2 7 5 8 1 4 6 3
第12种解法 2 8 6 1 3 5 7 4
第13种解法 3 1 7 5 8 2 4 6
第14种解法 3 5 2 8 1 7 4 6
第15种解法 3 5 2 8 6 4 7 1
第16种解法 3 5 7 1 4 2 8 6
第17种解法 3 5 8 4 1 7 2 6
第18种解法 3 6 2 5 8 1 7 4
第19种解法 3 6 2 7 1 4 8 5
第20种解法 3 6 2 7 5 1 8 4
第21种解法 3 6 4 1 8 5 7 2
第22种解法 3 6 4 2 8 5 7 1
第23种解法 3 6 8 1 4 7 5 2
第24种解法 3 6 8 1 5 7 2 4
第25种解法 3 6 8 2 4 1 7 5
第26种解法 3 7 2 8 5 1 4 6
第27种解法 3 7 2 8 6 4 1 5
第28种解法 3 8 4 7 1 6 2 5
第29种解法 4 1 5 8 2 7 3 6
第30种解法 4 1 5 8 6 3 7 2
第31种解法 4 2 5 8 6 1 3 7
第32种解法 4 2 7 3 6 8 1 5
第33种解法 4 2 7 3 6 8 5 1
第34种解法 4 2 7 5 1 8 6 3
第35种解法 4 2 8 5 7 1 3 6
第36种解法 4 2 8 6 1 3 5 7
第37种解法 4 5 1 5 2 8 3 7
第38种解法 4 6 8 2 7 1 3 5
第39种解法 4 6 8 3 1 7 5 2
第40种解法 4 7 1 8 5 2 6 3
第41种解法 4 7 3 8 2 5 1 6
第42种解法 4 7 5 2 6 1 3 8
第43种解法 4 7 5 3 1 6 8 2
第44种解法 4 8 1 3 6 2 7 5
第45种解法 4 8 1 5 7 2 6 3
第46种解法 4 8 5 3 1 7 2 6
第47种解法 5 1 4 6 8 2 7 3
第48种解法 5 1 8 4 2 7 3 6
第49种解法 5 1 8 6 3 7 2 4
```

```
C:\Users\DELL\Desktop\4.exe
第46种解法 4 8 5 3 1 7 2 6
第47种解法 5 1 4 6 8 2 7 3
第48种解法 5 1 8 4 2 7 3 6
第49种解法 5 1 8 6 3 7 2 4
第50种解法 5 2 4 6 8 3 1 7
第51种解法 5 2 4 7 3 8 6 1
第52种解法 5 2 6 1 7 4 8 3
第53种解法 5 2 8 3 1 4 7 3 6
第54种解法 5 3 1 4 6 8 2 4 7
第55种解法 5 3 1 7 2 8 6 4
第56种解法 5 3 8 4 7 1 6 2
第57种解法 5 7 1 3 8 6 4 2
第58种解法 5 7 1 4 2 8 6 3
第59种解法 5 7 2 4 8 1 3 6
第60种解法 5 7 2 6 3 1 4 8
第61种解法 5 7 2 6 3 1 8 4
第62种解法 5 7 4 1 3 8 6 2
第63种解法 5 8 4 1 3 6 2 7
第64种解法 5 8 4 1 7 2 6 3
第65种解法 6 1 5 2 8 3 7 4
第66种解法 6 2 7 1 3 5 8 4
第67种解法 6 2 7 1 4 8 5 3
第68种解法 6 3 1 7 5 8 2 4
第69种解法 6 3 1 8 4 2 7 5
第70种解法 6 3 1 8 5 2 4 7
第71种解法 6 3 5 7 1 4 2 8
第72种解法 6 3 5 8 1 4 2 7
第73种解法 6 3 7 2 4 8 1 5
第74种解法 6 3 7 2 5 1 4
第75种解法 6 3 7 4 1 8 2 5
第76种解法 6 4 1 5 8 2 7 3
第77种解法 6 4 2 8 5 7 1 3
第78种解法 6 4 7 1 3 5 2 8
第79种解法 6 4 7 1 8 2 5 3
第80种解法 6 8 2 4 1 7 5 3
第81种解法 7 1 3 8 6 4 2 5
第82种解法 7 2 4 1 8 5 3 6
第83种解法 7 2 6 3 1 4 8 5
第84种解法 7 3 1 6 8 5 2 4
第85种解法 7 3 8 2 5 1 6 4
第86种解法 7 4 2 5 8 1 3 6
第87种解法 7 4 2 8 6 1 3 5
第88种解法 7 5 3 1 6 8 2 4
第89种解法 8 2 4 1 7 5 3 6
第90种解法 8 2 5 3 1 7 4 6
第91种解法 8 3 1 6 2 5 7 4
第92种解法 8 4 1 3 6 2 7 5

Process exited after 2.436 seconds with return value 0
```

实验名称	实验三计算几何算法的实现						
姓 名	刘铭源	系院专业	软件工程	班 级	18-4	学 号	2018214937
实验日期			指导教师	徐本柱		成 绩	

一、实验目的和要求

(1) 理解动态规划的基本思想、动态规划算法的基本步骤。

(2) 掌握动态规划算法实际步骤。.

二、实验项目摘要

(1) 将讲义第三章第三节中的凸包代码上机运行并检验结果。

(2) 完成讲义第三章的课后习题，上机运行并检验结果。

(3) 思考：
判线段相交时，如果有个线段的端点在另一条线段上，注意可能与另一条线段上的 端点重合，思考这样的情况怎么办。

(4) 房间短路问题： 给顶一个内含阻碍墙的房间，求解出一条从起点到终点的短路径。房间的边界 固定在 $x=0, x=10, y=0$ 和 $y=10$ 。起点和重点固定在(0,5)和(10,5)。房间里还有 0 到 18 个 墙，每个墙有两个门。输入给定的墙的个数，每个墙的 x 位置和两个门的 y 坐标区间， 输出短路的长度。

三、实验项目摘要

1. 将书上的走迷宫代码上机运行并检验结果，并注意体会搜索的思想。

2. 八皇后问题：在一个国际象棋棋盘上放八个皇后，使得任何两个皇后之间不相互攻击，求出所有的布棋方法。上机运行并检验结果。

3. 骑士游历问题：在国际棋盘上使一个骑士遍历所有的格子一遍且仅一遍，对于任意给定的顶点，输出一条符合上述要求的路径。

4. 倒水问题：给定 2 个没有刻度容器，对于任意给定的容积，求出如何只用两个瓶装出 L 升的水，如果可以，输出步骤，如果不可以，请输出 No Solution。

四、实验结果与分析（源程序及相关说明）

1. 计算凸包

```
#include<iostream>
#include<algorithm>
#include<cstdio>
#include<ctime>
#include<cmath>
#define MAXN 1010
using namespace std;
struct point
```

```
{
    double x,y;
    double len;
};
point pt[MAXN];
point s[MAXN];
point A;
int NUM;

double Cross(point a,point b,point c)
{
    return (b.x-a.x)*(c.y-a.y)-(b.y-a.y)*(c.x-a.x);
}

double dis(point a,point b)
{
    return sqrt(a.x-b.x)*(a.x-b.x)+(a.y-b.y)*(a.y-b.y);
}

void Find(int n)
{
    int temp=0;
    point a=pt[0];
    for(int i=0;i<n;i++)
    {
        if(pt[i].y<a.y||pt[i].y==a.y&&pt[i].x<a.x)
        {
            a=pt[i];
            temp=i;
        }
    }
    pt[temp]=pt[0];
    pt[0]=a;
    A=a;
}

bool cmp(point a,point b)
{
    double x=Cross(A,a,b);
    if(x>0) return true;
    else if(x<0) return false;
    a.len=dis(A,a);
    b.len=dis(A,b);
    return a.len<b.len;
```

```

}

int Graham(int n)
{
    int top=2;
    s[0]=pt[0];
    s[1]=pt[1];
    s[2]=pt[2];
    pt[n]=pt[0];
    for(int i=3;i<n;i++)
    {
        while(Cross(s[top-1],s[top],pt[i])<=0&&top>1)
        {
            top--;
        }
        s[++top]=pt[i];
    }
    return top;
}

```

```

int init(int n)
{
    Find(n);
    sort(pt,pt+n,cmp);
    NUM=Graham(n);
}

int main()
{
    clock_t start,end;
    start=clock();
    int n;
    cin>>n;
    srand(time(0));

    for(int i=0;i<n;i++)
    {
        pt[i].x=rand()%100;
        pt[i].y=rand()%100;
    }

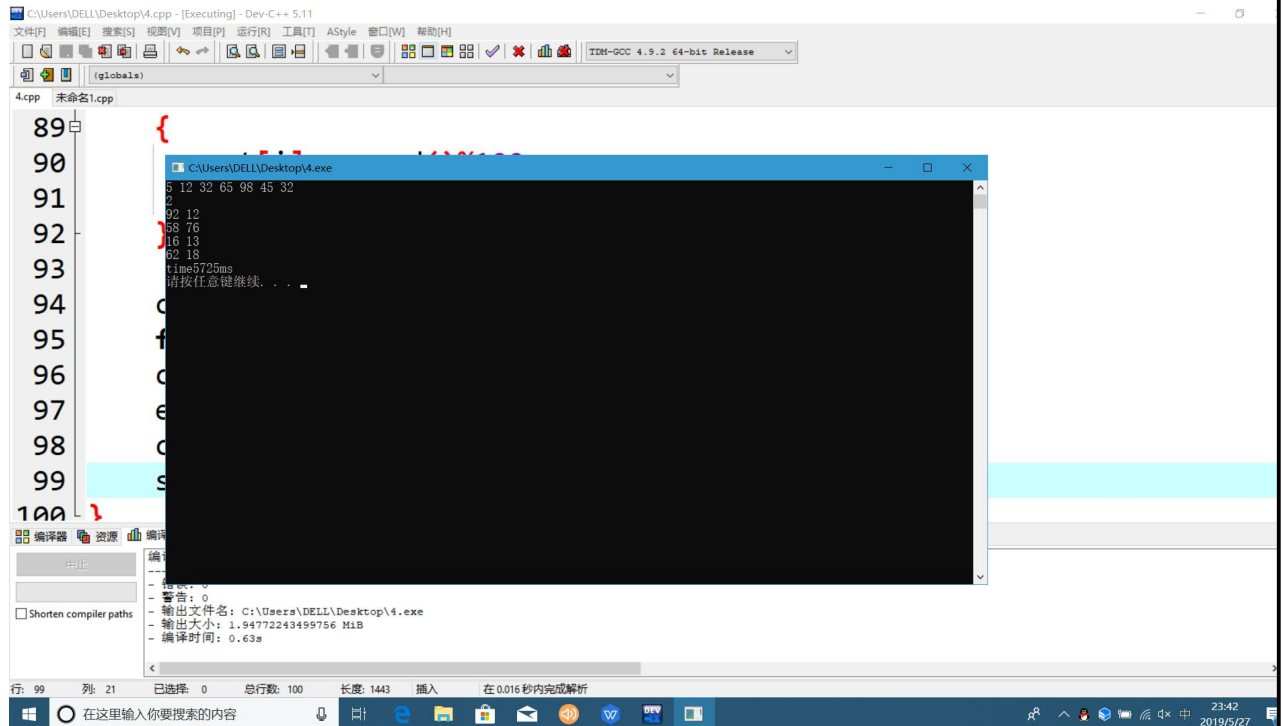
    cout<<init(n)<<endl;
    for(int i=0;s[i].x!=0;i++)
    cout<<s[i].x<<" "<<s[i].y<<endl;
}

```

```

end=clock();
cout<<"time"<<end-start<<"ms"<<endl;
system("Pause");
}

```



```

2.#include <iostream>
#include <utility>
#include <vector>
#include<ctime>
#include <algorithm>
using namespace std;
typedef pair<double, double> Point;
Point startPoint;
double direction(Point a, Point b, Point c)
{
    Point a1,a2;
    a1.first =c.first -a.first ;
    a1.second=c.second-a.second;
    a2.first =b.first -a.first;
    a2.second=b.second-a.second;
    return (a1.first*a2.second-a1.second*a2.first);
}
bool on_segment(Point a, Point b, Point c)
{
    double min_x,max_x,min_y,max_y;
    if(b.first<c.first)

```

```

{
    min_x = b.first;
    max_x = c.first;
}
else
{
    min_x = c.first;
    max_x = b.first;
}

if(b.second<c.second)
{
    min_y = b.second;
    max_y = c.second;
}
else
{
    min_y = c.second;
    max_y = b.second;
}

if (a.first >=min_x && a.first <=max_x && a.second>=min_y && a.second<=max_y)
    return true;
else
    return false;
}

bool sortByPolarAngle(const Point &a1, const Point &a2)
{
    double d=direction( startPoint, a1, a2);
    if (d<0)
        return true;
    else if (d>0)
        return false;
    else if (d==0 && on_segment(startPoint, a1, a2) )
        return true;
    else if (d==0 && on_segment(a2,startPoint,a1) )
        return true;
    else
        return false;
}

void find_convex_hull(vector<Point> & point)
{
    Point p0=point[0];
    int k=0;

```



```

        for (int i=1;i<point.size();i++)
        {
            if (point[i].second<p0.second || point[i].second==p0.second && point[i].first<p0.first)
            {
                p0=point[i];
                k=i;
            }
        }

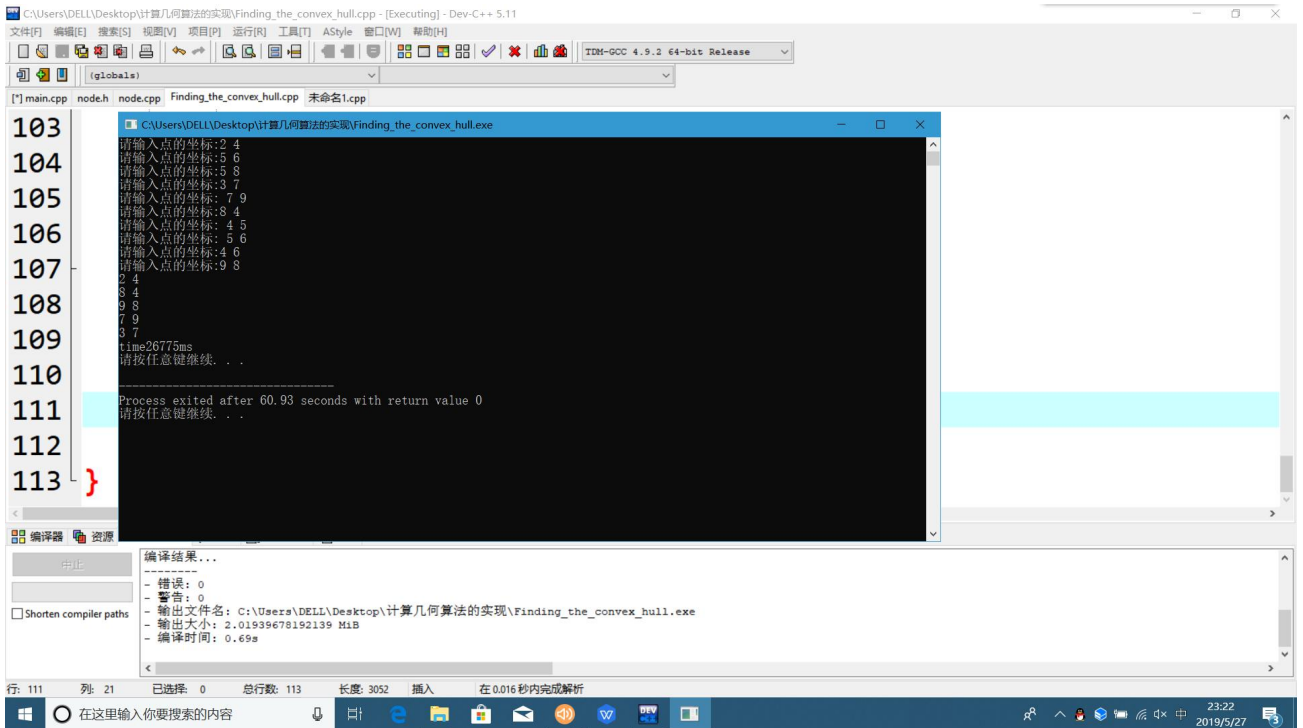
        point.erase(point.begin()+k);
        point.insert(point.begin(),p0);
        vector<Point> convex_hull;
        do {
            convex_hull.push_back(point[0]);
            startPoint=point[0];
            point.erase(point.begin());
            sort(point.begin(),point.end(),sortByPolorAngle);
            if (point[0]==convex_hull[0])
                break;
            point.push_back(convex_hull[convex_hull.size()-1]);
        } while (1);
        for (int i=0;i<convex_hull.size();i++)
        {
            cout<<convex_hull[i].first<<' '<<convex_hull[i].second<<endl;
        }
    }
int main()
{
    clock_t start,end;
    start=clock();
    vector<Point> a;
    for(int i = 0; i < 10;i++)
    {
        int x,y;
        cout <<"请输入点的坐标:";
        cin>>x>>y;
        Point  b;
        b.first = x;
        b.second = y;
        a.push_back(b);
    }
    find_convex_hull(a);
    end=clock();

```

```

cout<<"time"<<end-start<<"ms"<<endl;
system("Pause");
return 0;
}

```

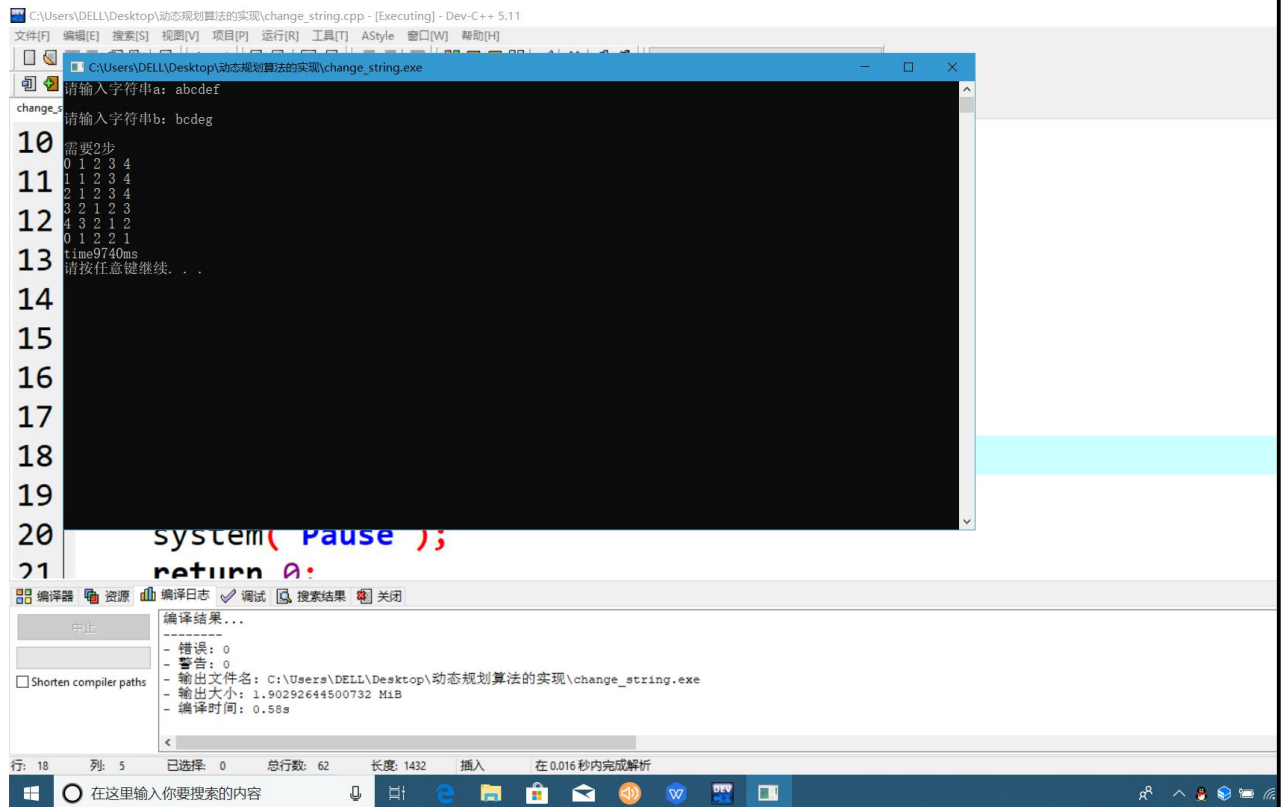


实验名称	实验四动态规划						
姓 名	刘铭源	系统专业	软件工程	班 级	18-4	学 号	2018214937

实验日期		指导教师	徐本柱	成 绩	
一、实验目的和要求 <ol style="list-style-type: none"> 1. 理解动态规划的基本思想、动态规划算法的基本步骤。 2. 掌握动态规划算法实际步骤。 					
二、实验预习内容 <ol style="list-style-type: none"> 1. 预习 ICPC 讲义，大致了解动态规划产生的背景 2. 了解动态规划的概念以及所适用的情况 3. 了解动态规划里的经典例题 					
三、实验项目摘要 <ol style="list-style-type: none"> 1. 求两个字符串最长公共子序列（区别于字串） 求一个字符串转变为另一个字符串的最小步骤数目（步骤仅包括替换、删除和插入）用 					
四、实验结果与分析（源程序及相关说明） <pre> #include<iostream> #include<string> #include<ctime> using namespace std; int mine(int x1,int x2, int x3); void change(string s1, string s2); int main() { clock_t start,end; start=clock(); string s1, s2; cout <<"请输入字符串 a: "; cin>>s1; cout <<endl; cout <<"请输入字符串 b: "; cin>>s2; cout <<endl; change(s1,s2); end=clock(); </pre>					

```
cout<<"time"<<end-start<<"ms"<<endl;
system("Pause");
return 0;
}
int mine(int xx1,int xx2, int xx3)
{
    int tt1, tt2;
    tt1 = min(xx1,xx2);
    tt2 = min(tt1,xx3);
    return tt2;
}
void change(string s1, string s2)
{
    const char* n_s1 = s1.c_str();
    const char* n_s2 = s2.c_str();
    int temple;
    int b[100][100];
    for(int i = 0; i < s1.length(); i++)
        b[0][i] = i;
    for(int j = 0; j < s2.length(); j++)
        b[j][0] = j;
    for(int k = 1; k <= s1.length() ; k++)
        for(int m = 1; m <=s2.length(); m++)
        {
            if( n_s1[k-1] == n_s2[m-1])
                b[k][m] = b[k-1][m-1];
            else
            {
                temple = mine((b[k-1][m-1]),(b[k][m-1]),(b[k-1][m]));
                b[k][m] = temple + 1 ;
            }
        }
}

cout <<"yes,需要"<<(b[(s1.length() )][(s2.length() )])<<"步"<<endl;
for(int l=0; l < s1.length(); l++)
{
    for(int n = 0; n < s2.length(); n++)
    {
        cout << b[l][n]<<" ";
    }
    cout <<endl;
}
}
```



```
C:\Users\DELL\Desktop\动态规划算法的实现\change_string.cpp - [Executing] - Dev-C++ 5.11
文件[F] 编辑[E] 搜索[S] 视图[V] 项目[P] 运行[R] 工具[T] AStyle 窗口[W] 帮助[H]
C:\Users\DELL\Desktop\动态规划算法的实现\change_string.exe
请输入字符串a: abcdef
请输入字符串b: bcdeg
change_s
10 需要2步
11 0 1 2 3 4
12 1 1 2 3 4
13 2 1 2 3 4
14 3 2 1 2 3
15 4 3 2 1 2
16 0 1 2 2 1
17 time9740ms
18 请按任意键继续. . .
19
20 system("Pause");
21 return 0;
```

```
#include<iostream>
#include<string>
#include<ctime>
using namespace std;
void md(string a,string b);

int main()
{
    clock_t start,end;
    start=clock();
    cout<<" 输入两个字符串"<<endl;
    string a,b;
    cin>>a>>b;
    cout<<"结果为: "<<endl;
    md(a,b);
    cout<<endl;
    end=clock();
    cout<<"time"<<end-start<<"ms"<<endl;
    system("Pause");
}
void md(string a,string b)
{
    int a_length=a.length();
```

```

int b_length=b.length();
int D[a_length+1][b_length+1];
D[0][0]=0;
for(int i=1;i<=b_length;i++)
{
    D[0][i]=i;
}
for(int i=1;i<=a_length;i++)
{
    D[i][0]=i;

    for(int i=1;i<=a_length;i++)
    {
        for(int j=1;j<=b_length;j++)
            D[i][j]=min(min(D[i-1][j]+1,D[i][j-1]+1),(a[j-1]==b[i-1]?D[i-1][j-1]:D[i-1][j-1]+1));/?
    }

    cout<<D[a_length][b_length];
}

```

The screenshot shows a C++ IDE with the following components:

- Editor:** Displays the C++ code for calculating the edit distance between two strings. The code uses a 2D array `D` to store the results of subproblems. The strings being compared are `safjksjf` and `safjhejf`.
- Output Window:** Shows the execution results. It prompts the user to input two strings, which are `safjksjf` and `safjhejf`. The result is `17`, and the execution time is `4591ms`. It also prompts the user to press any key to continue.
- Compiler Output Window:** Shows the compilation results. It indicates that there are no errors or warnings, and the output file is `C:\Users\DELL\Desktop\4.exe`. The output size is `1.90240669250488 Mib` and the compilation time is `0.56s`.
- Taskbar:** Shows the Windows taskbar with the Start button, a search bar, and several application icons. The system clock shows the time as `23:48` on `2019/5/27`.