

**实验报告**

**课程名称： 算法设计与分析**

**学 院： 应用技术学院**

**专业班级： 18计科一班**

**学 号： 201833050027**

**姓 名： 叶成宇**

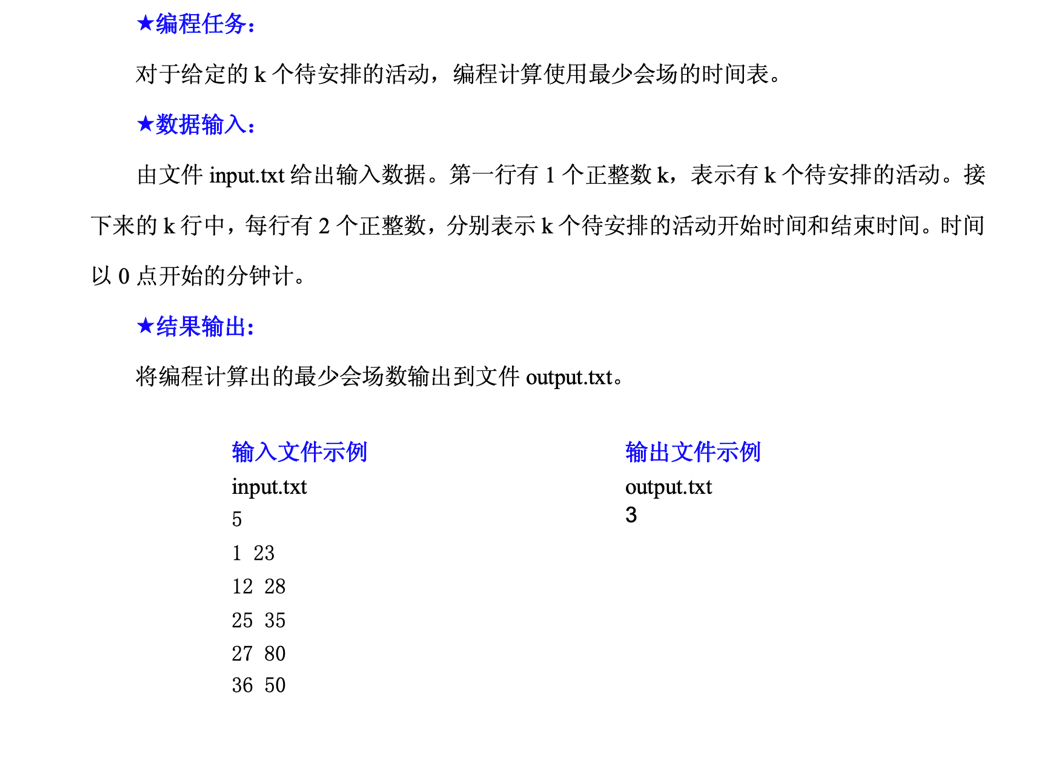
1．实验目的

1、假设要在足够多的会场里安排一批活动，并希望使用尽可能少的会场。设计一个有效的贪心算法进行安排。(这个问题实际上是著名的图着色问题。若将每一个活动作为图的一个 顶点，不相容活动间用边相连。使相邻顶点着有不同颜色的最小着色数，相应于要找的最小 会场数。)

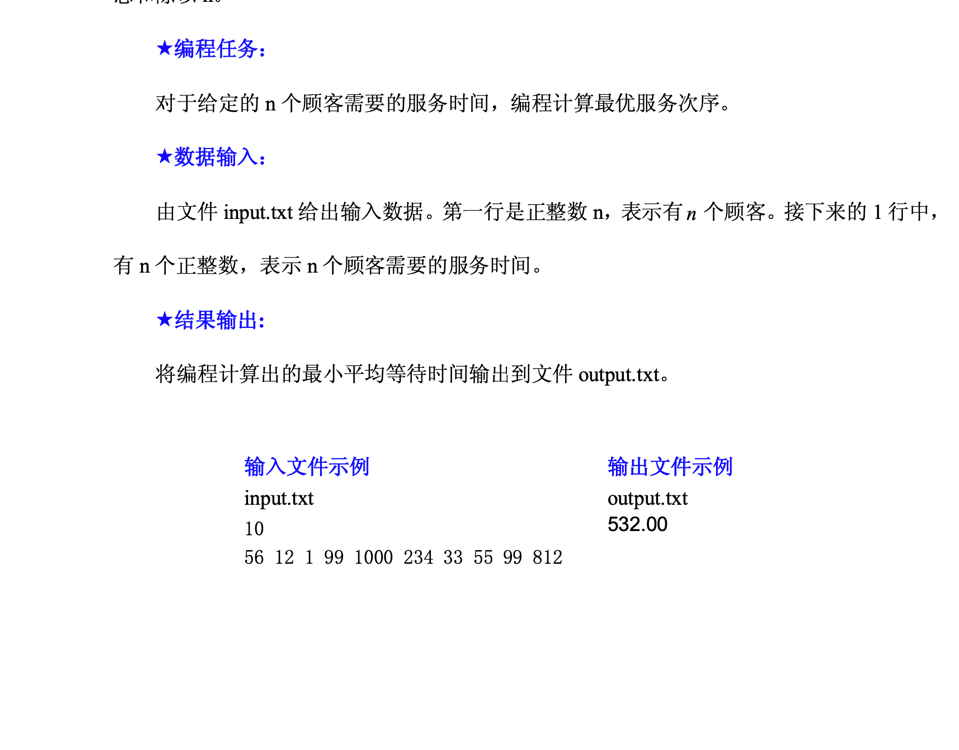
2、设有 n 个顾客同时等待一项服务。顾客 i 需要的服务时间为ti ,1 £ i £ n 。应如何安排 n个顾客的服务次序才能使平均等待时间达到最小?平均等待时间是 n 个顾客等待服务时间的 总和除以 n。

2．实验内容

1、



2、



3．实验步骤和实验结果

1、

#include<iostream>

#include<cstdio>

#include<algorithm>

using namespace std;

bool b[10001];//用一个bool数组标记这个会场是否已经安排上了

struct meeting

{

int start,end;

}a[10001];

int comp(const meeting &a,const meeting &b)

{

return a.end<b.end;

}

int cmp(const meeting &a,const meeting &b)

{

return a.start<b.start;

}

int main ()

{

int n,ans,counts;

int flag=0;

freopen("/Users/ycy/Desktop/input.txt","r",stdin);

while(scanf("%d",&n)!=EOF)

{

flag++;

memset(a,0,sizeof(a));

ans=0;

counts=n; //假设最开始每场会议都要一个会场

memset(b,false,sizeof(b)); //最开始每场会议都还没安排

for(int i=1;i<=n;i++)

cin>>a[i].start>>a[i].end;

sort(a+1,a+n+1,cmp); //根据会议的结束时间进行排序

for(int i=1;i<=n;i++)

{

int p,t;

if(!b[i]) //如果第i场会议还没安排上，以这场会议作为起点，寻找可以和这场安排一起的有几个

{

ans++;

p=i;

t=i+1;

while(t<=n)

{

if(a[p].end<=a[t].start&&!b[t]) //如果找到一个，这个点安排上

{

counts--; //找到一个能和这场会议安排在一个会场里的，counts--

p=t; //再开始从这个个点往下走

b[t]=true;

}

t++;

}

}

b[i]=true;

}

cout<<ans<<endl;

}

return 0;

}

2、

#include<iostream>

#include<algorithm>

using namespace std;

#define N 10000

int main()

{

int n;

int i;

int a[N];

while(cin>>n)

{

for( i=1;i<=n;i++)

cin>>a[i];

sort(a+1,a+1+n);

int sum=0;

int tmp=0;

for( i=1;i<=n;i++)

{

sum=sum+tmp+a[i];

tmp+=a[i];

}

double ans=0;

ans=(double)sum\*1.0/n;

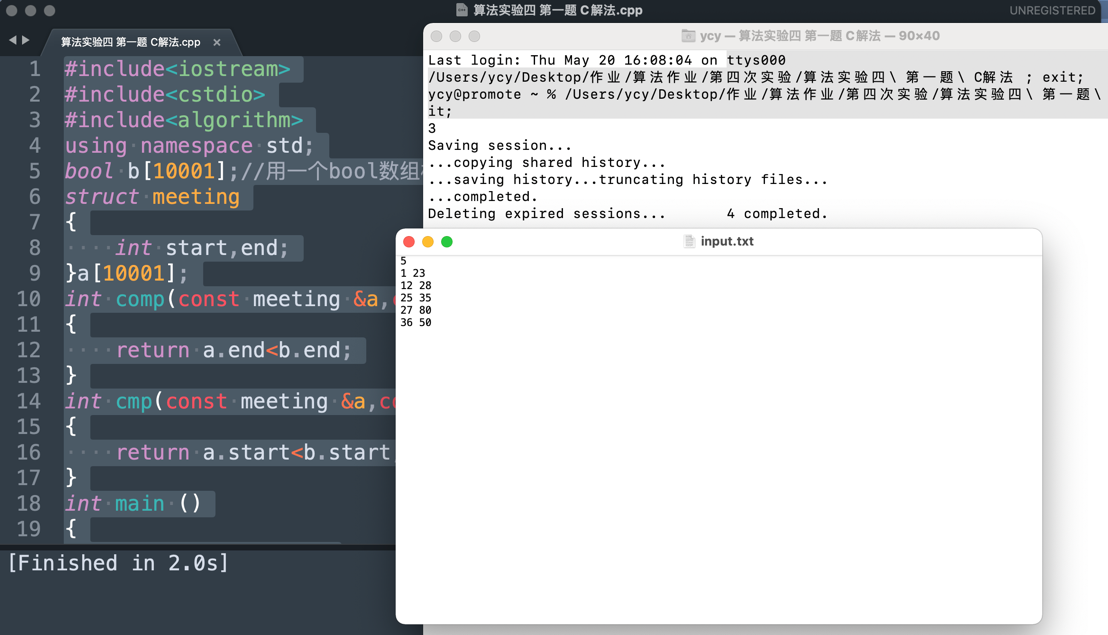
printf("%.2lf\n",ans);

}

return 0;

}

1、



2、

4．分析与讨论

贪心算法是一种对某些求最优解问题的更简单、更迅速的设计技术。贪心算法的特点是一步一步地进行，常以当前情况为基础根据某个优化测度作最优选择，而不考虑各种可能的整体情况，省去了为找最优解要穷尽所有可能而必须耗费的大量时间。贪心算法采用自顶向下，以迭代的方法做出相继的贪心选择，每做一次贪心选择，就将所求问题简化为一个规模更小的子问题，通过每一步贪心选择，可得到问题的一个最优解。虽然每一步上都要保证能获得局部最优解，但由此产生的全局解有时不一定是最优的，所以贪心算法不要回溯