**实验1 算法的时空复杂度分析**

**一、实验目的**

1. 培养学生分析算法复杂度的能力并且能够描述算法复杂性的一些基本理论。

2. 通过教学案例培养学生的逻辑思维能力，能够分析算法复杂性，并且能够计算算法的时空复杂性。

**二、实验要求**

1**．**复习数据结构中时间复杂度的计算。

2．阅读实验程序完成下面的实验内容并上机测试运行。

3．整理并上交**实验报告（内容要求参见实验报告模板）。**

**三、实验内容**

1. 分析顺序查找算法的最好、最坏和平均复杂度

int seqSearch(int a[ ], int n, int k)

{ int i=0;

while (i<n && a[i]!=k)

i++;

if (i<n) return i;

else return -1;

}

2. 有以下算法，其中临时空间为变量i、maxi占用的空间，分析其空间复杂度。

int max(int a[]，int n)

{ int i，maxi=0;

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

if (a[i]>a[maxi])

maxi=i;

return a[maxi];

}

3. 有如下递归算法，分析调用maxelem(a，0，n-1)的空间复杂度。

int maxelem(int a[],int i,int j)

{ int mid=(i+j)/2,max1,max2;

if (i<j)

{ max1=maxelem(a,i,mid);

max2=maxelem(a,mid+1,j);

return (max1>max2)?max1:max2;

}

else return a[i];

}

4. 阅读以下代码，填写缺少的语句, 分析冒泡排序的时间复杂度,并给出运行结果。

#include <iostream.h>

void BubbleSort(int r[ ], int n);

int main()

{

int i, r[]={4,3,9,2,6,1};

cout<<"排序前：";

for(i=0;i<6;i++)

cout<<r[i]<<" ";

cout<<endl;

BubbleSort(r,6);

cout<<"排序后：";

for(i=0;i<6;i++)

cout<<r[i]<<" ";

cout<<endl;

return 0;

}

void BubbleSort(int r[ ], int n)

{

int bound, exchange = n - 1; //第一趟起泡排序的区间是[0,n-1]

（1） //当上一趟排序有记录交换时

{

bound = exchange; exchange = 0;

for (int j = 0; j < bound; j++) //一趟起泡排序区间是[0, bound]

if (r[j] > r[j + 1]) {

（2） ;

//交换记录

（3） //记载每一次记录交换的位置

}

}

}