# 实验七 排序

【实验项目】：

1. 理解并掌握直接插入排序、折半插入排序、2-路插入排序、起泡法排序和快速排序的基本概念和方法。
2. 掌握排序算法的基本思想。

## 一、实验目的/

1. 掌握排序算法基本思想的实现。
2. 通过实验掌握直接插入排序、折半插入排序、2-路插入排序的具体实现。
3. 掌握交换排序算法基本思想的实现。
4. 通过实验掌握起泡法排序和快速排序的具体实现。

## 二、实验要求（本次实验要求上交）

**【题目】--验证型实验**

**实现直接插入算法、希尔排序算法、冒泡法排序算法、快速排序算法、直接选择排序算法的排序。其参考算法见（文件夹7-1）**

**【题目2】**

**请在给出的参考算法中，设计随机数来测试排序算法运行时间的程序。写成一个主程序调用各排序函数，查看每次排序的结果，并给出各种排序方法的排序时间比较。初始的时候数组大小可设为10000，然后改成20000或其它值再进行对比，观察输出结果。**

为了产生随机数，可以用到头文件stdlib．h中的两个函数srand()和rand()来设置随机种子以及产生随机数。为了能够计时，用到头文件time．h中的time()和difftime()两个函数，time()用于截取计算机内的时钟，difftime()用于得到两次时钟间隔的时间(秒)。每一种排序方法都单独写成子函数形式，然后用主函数调用它。为了能查看排序前后的效果，可以单独写一个子函数输出数组结果，在排序前后分别调用它，可以看到排序前后的结果。

**参考头文件：**

#include<iostream.h>

#include<stdlib.h>

#include<time.h>

#define N 10000；

#define ElemType int

**随机数的产生：**

double ttl，tt2，tt3，tt4，tt5；

srand(0)；

for(int i=0；i<=N-1；i++)

T[i]=rand()；//产生随机数

print(T,N)； //输出随机数

**time（）的调用：**

tl=time(NULL)

|  |
| --- |
| 请将源程序附录于此： |
| #include <stdlib.h>  #include <stdio.h>  #include <time.h>  #include <string.h>  #define N 12000  typedef int KeyType;  typedef char InfoType[10];  typedef struct //记录类型  {  KeyType key; //关键字项  InfoType data; //其他数据项,类型为InfoType  } RecType;  typedef struct //记录类型  {  long key; //关键字项  char data[50];  } Time;  //直接插入排序  void InsertSort(RecType R[],int n)  {  int i,j;  RecType temp;  for (i=1;i<n;i++)  {  temp=R[i];  j=i-1; //从右向左在有序区R[0..i-1]中找R[i]的插入位置  while (j>=0 && temp.key<R[j].key)  {  R[j+1]=R[j]; //将关键字大于R[i].key的记录后移  j--;  }  R[j+1]=temp; //在j+1处插入R[i]  }  printf("\n");  }  //希尔排序算法  void ShellSort(RecType R[],int n)  {  int i,j,d;  RecType temp;  d=n/2; //d取初值n/2  while (d>0)  {  for (i=d;i<n;i++) //将R[d..n-1]分别插入各组当前有序区中  {  j=i-d;  while (j>=0 && R[j].key>R[j+d].key)  {  temp=R[j]; //R[j]与R[j+d]交换  R[j]=R[j+d];  R[j+d]=temp;  j=j-d;  }  }  d=d/2; //递减增量d  }  printf("\n");  }  //冒泡排序算法  void BubbleSort(RecType R[],int n)  {  int i,j;  RecType temp;  for (i=0;i<n-1;i++)  {  for (j=n-1;j>i;j--) //比较,找出本趟最小关键字的记录  if (R[j].key<R[j-1].key)  {  temp=R[j]; //R[j]与R[j-1]进行交换,将最小关键字记录前移  R[j]=R[j-1];  R[j-1]=temp;  }  }  printf("\n");  }  //快速排序  void QuickSort(RecType R[],int s,int t)  {  int i=s,j=t,k;  RecType temp;  if (s<t) //区间内至少存在一个元素的情况  {  temp=R[s]; //用区间的第1个记录作为基准  while (i!=j) //从区间两端交替向中间扫描,直至i=j为止  {  while (j>i && R[j].key>temp.key)  j--; //从右向左扫描,找第1个关键字小于temp.key的R[j]  R[i]=R[j];  while (i<j && R[i].key<temp.key)  i++; //从左向右扫描,找第1个关键字大于temp.key的记录R[i]  R[j]=R[i];  }  R[i]=temp;  QuickSort(R,s,i-1); //对左区间递归排序  QuickSort(R,i+1,t); //对右区间递归排序  }  }  //直接选择排序算法  void SelectSort(RecType R[],int n)  {  int i,j,k;  RecType temp;  for (i=0;i<n-1;i++) //做第i趟排序  {  k=i;  for (j=i+1;j<n;j++) //在当前无序区R[i..n-1]中选key最小的R[k]  if (R[j].key<R[k].key)  k=j; //k记下目前找到的最小关键字所在的位置  if (k!=i) //交换R[i]和R[k]  {  temp=R[i];R[i]=R[k];R[k]=temp;  }  }  printf("\n");  }  void DispSort(RecType R[])  {  int n = 10,k;  for (k=0;k<n;k++)  printf("%2d ",R[k].key);  printf("\n\n");  }  void BijiaoTime(Time D[],int n){  int i,j,k;  Time temp;  for (i=0;i<n-1;i++)  {  k=i;  for (j=i+1;j<n;j++)  if (D[j].key < D[k].key)  k=j;  if (k!=i)  {  temp=D[i];D[i]=D[k];D[k]=temp;  }  }  for(i = 0;i < n;i++){  printf("排序时间第%d快的是:%s\n",i+1,D[i].data);  }  }  int main()  {  char a[5];  int i,n=N,j;  long t1,t2;  RecType R[N],T[N],A[N],B[N],P[N];  Time D[5];  srand(0);  printf("此测试都只显示数组前十个元素\n\n");  for (i=0;i<n;i++){  R[i].key = rand();  T[i].key = R[i].key;  A[i].key = R[i].key;  B[i].key = R[i].key;  P[i].key = R[i].key;  }  for(j = 0;j < 5;j++){  printf("输入你要选择的排序方式,直接选择排序输入S,快速排序输入Q,冒泡排序输入B,希尔排序输入L,直接插入排序输入I:");  gets(&a[j]);  if(a[j] == 'S'){  printf("\n初始关键字:");  DispSort(R);  printf("直接选择排序\n");  t1 = time(NULL);  SelectSort(R,n);  t2 = time(NULL);  D[j].key = difftime(t2,t1);  strcpy(D[j].data,"直接选择排序");  printf("最后结果:");  DispSort(R);  printf("排序的时间为: %lf seconds\n\n",difftime(t2,t1));  }  else if(a[j] == 'Q'){  printf("\n初始关键字:");  DispSort(T);  printf("快速排序\n\n");  t1 = time(NULL);  QuickSort(T,0,n-1);  t2 = time(NULL);  D[j].key = difftime(t2,t1);  strcpy(D[j].data,"快速排序");  printf("最后结果:");  DispSort(T);  printf("排序的时间为: %lf\n\n",difftime(t2,t1));  }  else if(a[j] == 'B'){  printf("\n初始关键字:");  DispSort(A);  printf("冒泡排序\n");  t1 = time(NULL);  BubbleSort(A,n);  t2 = time(NULL);  D[j].key = difftime(t2,t1);  strcpy(D[j].data,"冒泡排序");  printf("最后结果:");  DispSort(A);  printf("排序的时间为: %lf\n\n",difftime(t2,t1));  }  else if(a[j] == 'L'){  printf("\n初始关键字:");  DispSort(B);  printf("希尔排序\n");  t1 = time(NULL);  ShellSort(B,n);  t2 = time(NULL);  D[j].key = difftime(t2,t1);  strcpy(D[j].data,"希尔排序");  printf("最后结果:");  DispSort(B);  printf("排序的时间为: %lf\n\n",difftime(t2,t1));  }  else if(a[j] == 'I'){  printf("\n初始关键字:");  DispSort(P);  printf("直接插入排序\n");  t1 = time(NULL);  InsertSort(P,n);  t2 = time(NULL);  D[j].key = difftime(t2,t1);  strcpy(D[j].data,"直接插入排序");  printf("最后结果:");  DispSort(P);  printf("排序的时间为: %lf\n\n",difftime(t2,t1));  }  else{  printf("输入不合规则");  }  }  BijiaoTime(D,5);  printf("\n");  return 0;  } |
| 请将运行结果图附录于此 |
| C:\Users\Administrator\AppData\Roaming\Tencent\Users\1061892125\QQ\WinTemp\RichOle\RG$MDXP3O2J88}YI1EWKY~F.png  C:\Users\Administrator\AppData\Roaming\Tencent\Users\1061892125\QQ\WinTemp\RichOle\WGWY6G%TKUM]HVPP_YC_0ZM.png  C:\Users\Administrator\AppData\Roaming\Tencent\Users\1061892125\QQ\WinTemp\RichOle\QIM}{AKXPMKS8B]25ISTZY6.png  C:\Users\Administrator\AppData\Roaming\Tencent\Users\1061892125\QQ\WinTemp\RichOle\LX5S{@()WO@1BYZX5K{ZOQK.png  这里快速排序我也不知道怎么回事，当数组的数过多时就一直不能显示出排序后的结果所以就一直没有得出快速排序后的值，下面的排序时间的比较所以就只是另外四个的顺序，快速排序没有包括在里面  C:\Users\Administrator\AppData\Roaming\Tencent\Users\1061892125\QQ\WinTemp\RichOle\LK071[XGS2WK4XY%@VN349R.png |

## 实验上传方法：

将源程序文件和本word文档（添加了相应的源程序和截图）上传。多个文件请将其放入一个文件夹压缩后上传。请于指定时间前上交，过时不补！