代码：

#include<stdio.h>

#include<stdlib.h>

#include<time.h>

void Insert\_a(int arr[],int len){ //插入排序

int i,j,value;

for(i=0;i<len-1;i++){

value=arr[i+1];

for(j=i+1;j>0 && value<arr[j-1];j--){ //将比value值更大的元素后移

arr[j]=arr[j-1];

}

arr[j]=value;

}

}

void QuickSort(int arr1[], int low, int high){ //快速排序

if (low < high)

{ int i = low;

int j = high;

int k = arr1[low];

while (i < j)

{

while(i < j && arr1[j] >= k) // 从右向左找第一个小于k的数 {

j--;

}

if(i < j)

{

arr1[i++] = arr1[j];

}

while(i < j && arr1[i] < k) // 从左向右找第一个大于等于k的数

{

i++;

}

if(i < j)

{

arr1[j--] = arr1[i];

}

}

arr1[i] = k; // 递归调用

QuickSort(arr1, low, i - 1); // 排序k左边

QuickSort(arr1, i + 1, high); // 排序k右边

}

}

void display(int arr2[],int len){ //输出函数

for(int i=0;i<=len-1;i++)

printf("%d ",arr2[i]);

printf("\n");

}

void main(){

int i=0,j=100;

int a[100];

int b[100];

clock\_t begin,stop;

clock\_t begin2,stop2;

for(;i<j;i++) //生成随机数组a、b

b[i]=a[i]=rand()%1000;

printf("排序前的数组:\n");

display(a,j);

begin=clock();

Insert\_a(a,j);

stop=clock();

printf("插入排序后的数组:\n");

display(a,j);

printf("插入排序运行时间：%ld\n",(stop-begin/CLOCKS\_PER\_SEC));

begin2=clock();

QuickSort(b,0,j-1);

stop2=clock();

printf("快速排序后的数组:\n");

display(b,j);

printf("快速排序运行时间：%ld\n",(stop2-begin2/CLOCKS\_PER\_SEC));

}

对于n>1的n元素数组，在n很小的时候，存在插入排序比快速排序更快的情况。