**《程序设计基础1》实验报告**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **年级、专业、班级** | | **大一计算机类01班** | | **姓名** | | **孙莹莹** | **学号** | **20204104** |
| **实验题目** | **排序算法的设计与实现** | | | | | | | |
| **实验时间** | **2020年11月8日** | | **实验地点** | | **DS1421** | | | |
| **学年学期** | **2020-2021（1）** | | **实验性质** | | **□验证性 ■设计性 □综合性** | | | |
| 一、实验目的  1． 了解学习冒泡、插入及归并排序算法。  2． 理解并实现简单的分治算法。  3． 熟悉for循环语法。 | | | | | | | | |
| 二、实验项目内容  1. 使用随机数模块生成包含20个整数的随机数列表，然后使用冒泡排序算法对其进行排序并输出。（不可以使用列表自身的排序功能，排序过程应封装成函数）  2. 使用随机数模块生成包含20个整数的随机数列表，然后使用插入排序算法对其进行排序并输出。（不可以使用列表自身的排序功能，排序过程应封装成函数）  3. 使用随机数模块生成包含20个整数的随机数列表，然后使用归并排序算法对其进行排序并输出。（不可以使用列表自身的排序功能，排序过程应封装成函数） | | | | | | | | |
| 三、实验的算法（伪代码或者流程图）和源程序  1.冒泡排序：  import random  def fun(n):      martix = [random.randint(0,100) for c in range(n)]      return martix  print(fun(20))  def bubbleSort(seq):      for i in range(len(seq)-1,0,-1):          for j in range(0,i):              if seq[j] >seq[j+1]:                  seq[j],seq[j+1] = seq[j+1],seq[j]  seq = fun(20)  bubbleSort(seq)  print(seq)  2.插入排序：  import random  def fun(n):      martix = [random.randint(0,100) for c in range(n)]      return martix  print(fun(20))  def insertionSort(arr):        for i in range(1, len(arr)):     key = arr[i]            j = i-1          while j >=0 and key < arr[j] :                  arr[j+1] = arr[j]                  j -= 1          arr[j+1] = key      arr = fun(20)  insertionSort(arr)  print ("排序后的数组:")  for i in range(len(arr)):      print ("%d" %arr[i])  3.归并排序  import random  def fun(n):      martix = [random.randint(0,100) for c in range(n)]      return martix  print(fun(20))  def merge(arr, l, m, r):      n1 = m - l + 1      n2 = r- m        # 创建临时数组      L = [0] \* (n1)      R = [0] \* (n2)        # 拷贝数据到临时数组 arrays L[] 和 R[]      for i in range(0 , n1):          L[i] = arr[l + i]        for j in range(0 , n2):          R[j] = arr[m + 1 + j]        # 归并临时数组到 arr[l..r]      i = 0     # 初始化第一个子数组的索引      j = 0     # 初始化第二个子数组的索引      k = l     # 初始归并子数组的索引        while i < n1 and j < n2 :          if L[i] <= R[j]:              arr[k] = L[i]              i += 1          else:              arr[k] = R[j]              j += 1          k += 1        # 拷贝 L[] 的保留元素      while i < n1:          arr[k] = L[i]          i += 1          k += 1        # 拷贝 R[] 的保留元素      while j < n2:          arr[k] = R[j]          j += 1          k += 1    def mergeSort(arr,l,r):      if l < r:              m = int((l+(r-1))/2)              mergeSort(arr, l, m)          mergeSort(arr, m+1, r)          merge(arr, l, m, r)      arr = [12, 11, 13, 5, 6, 7]  n = len(arr)  print ("给定的数组")  for i in range(n):      print ("%d" %arr[i]),    mergeSort(arr,0,n-1)  print ("\n\n排序后的数组")  for i in range(n):      print ("%d" %arr[i]), | | | | | | | | |
| 四、实验结果及分析和（或）源程序调试过程、实验总结与体会  一、冒泡排序    二、插入排序    **三、归并排序**    在实验中存在的问题：如何将所理解的算法，转化为可编写程序  心得和体会是遇见问题时，可以适当学习他人的代码，在理解算法以及每一句语句的意义后，设法自己模仿，编写自己的程序 | | | | | | | | |