**深 圳 大 学 实 验 报 告**

**课程名称： C程序设计**

**实验项目名称： 综合实验：循环结构及数组综合应用**

**学院： 电子与信息工程学院**

**专业： 电子信息工程**

**指导教师： 邹文斌**

**报告人： 吴艇 学号：2020281061 班级： 电信4班**

**实验时间： 2021年4月24日**

**实验报告提交时间： 2021年4月26日**

**教务部制**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1. **实验要求**    1. 进一步掌握 C 语言中的选择结构、循环控制的编程方法；    2. 学习 C 语言中的数组定义和引用的编程方法；    3. 理解和掌握冒泡排序法、选择排序法的基本原理，掌握其程序实现方法；    4. 需上交实验报告。 2. **实验环境**    1. Visual Studio 2013 3. **实验内容**   部分国家新冠疫情分析。  定义一个二维char数组用于存放国家的名字  定义两个一维int数组分别存放每个国家的确诊人数和死亡人数  定义一个一维float数组用于存放死亡率，死亡率需通过程序计算，死亡率 = 死亡人数 / 确诊人数。  依次执行如下操作：  （1） 用冒泡排序算法对表1的数据按照确诊人数由多到少的顺序排序，将结果显示在屏幕上并绘制直方图（参照图1），其中屏幕上每一行的显示形式为：  序号 确诊人数 国家 死亡人数 死亡率  （2） 用冒泡排序算法对表1的数据按照死亡率由大到小的顺序排序，将结果显示在屏幕上并绘制直方图（参照图1），其中屏幕上每一行的显示形式为：  序号 死亡率 死亡人数 国家 确诊人数  （3） 用选择排序算法对表1的数据按照死亡人数由多到少的顺序排序，将结果显示在屏幕上并绘制直方图（参照图1），其中屏幕上每一行的显示形式为：  序号 死亡人数 国家 确诊人数 死亡率  **表1：部分国家新冠疫情数据**   |  |  |  |  |  | | --- | --- | --- | --- | --- | | 序号 | 国家 | 确诊人数 | 死亡 | 死亡率 | | 1 | 美国 | 31,990,519 | 571,179 |  | | 2 | 法国 | 5,502,014 | 102,655 |  | | 3 | 土耳其 | 4,550,820 | 37,672 |  | | 4 | 印度 | 16,263,695 | 186,920 |  | | 5 | 英国 | 4,416,588 | 127,638 |  | | 6 | 俄罗斯 | 4,691,290 | 105,718 |  | | 7 | 意大利 | 3,935,703 | 118,699 |  | | 8 | 西班牙 | 3,468,617 | 77,591 |  | | 9 | 巴西 | 14,237,078 | 386,416 |  | | 10 | 德国 | 3,263,415 | 81,326 |  | | 11 | 哥伦比亚 | 2,740,544 | 70,446 |  | | 12 | 阿根廷 | 2,824,652 | 61,176 |  | | 13 | 波兰 | 2,742,122 | 64,707 |  | | 14 | 加拿大 | 1,172,697 | 23,870 |  | | 15 | 伊朗 | 2,358,809 | 68,746 |  | | 16 | 墨西哥 | 2,319,519 | 214,095 |  | | 17 | 中国 | 102,347 | 4,845 |  |   **图1：直方图样例**  国家1：\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  国家2：\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  国家3：\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  国家4：\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  国家5：\*\*\*\*\*\*\*\*\*\*\*\*  国家6：\*\*\*\*\*\*\*   1. **实验过程**   思路：   1. 首先从表格入手，打印出的表格中每一横行都是一个对应关系，但是冒泡排序和选择排序都会对数组内容进行更改，到时候要输出的时候对应关系就乱了，所以我想的是不改变原来数组而且还要把改变后的顺序记录下来。有了思路后我就让各个数据copy了一份，让这部分去比较、修改，而且定义了一个顺序数组，来记录比较后的顺序，到时候直接传入写好的打印函数就可以输出对应的数据了。 2. 然后就是直方图了，直方图主要展示各个数据的大小关系，所以我直接让各组数据的最大值占满，然后其他数据就反映与最大值的百分比。   完整代码：  #include<stdio.h>  char nation[][17] = { "美国","法国","土耳其","印度","英国","俄罗斯","意大利","西班牙","巴西","德国","哥伦比亚","阿根廷","波兰","加拿大","伊朗","墨西哥","中国" };  int confirm[17] = { 31990519,5502014,4550820,16263695,4416588,4691290,3935703,3468617,14237078,3263415,2740544,2824652,2742122,1172697,2358809,2319519,102347 };  int death[17] = { 571179,102655,37672,186920,127638,105718,118699,77591,386416,81326,70446,61176,64707,23870,68746,214095,4845 };  float deathRate[17]; //死亡人数death/确诊人数confirm  //由于不想破环原数组内元素与序号的关系，于是又创建了一个用来排序修改  int copy\_confirm[17] = { 31990519,5502014,4550820,16263695,4416588,4691290,3935703,3468617,14237078,3263415,2740544,2824652,2742122,1172697,2358809,2319519,102347 };  int copy\_death[17] = { 571179,102655,37672,186920,127638,105718,118699,77591,386416,81326,70446,61176,64707,23870,68746,214095,4845 };  float copy\_deathRate[17];  int i; //下标为i的值开始进行冒泡排序  int j; //冒泡排序的次数  int k; //选择排序最小值下标  int n; //计算\*个数  int temp; //储存整型中间值  float ftemp; //储存浮点型中间值  //不想破环原数组，故重新创建序号关系，用于排序  int temparr1[17] = { 0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16 };  int temparr2[17] = { 0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16 };  int temparr3[17] = { 0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16 };  //直方图绘制  //按确诊人数  void confirm\_histogram(int p) {  //默认直方图里依照最大值为100%，即最大值对应的国家“\*”号占满  n = (int)(2\*(0.5+ (50 \* confirm[p]) / 31990519));  printf("%-10s", nation[p]);  while(n--){  printf("\*");  }  printf("\n");  }  //按死亡率  void deathRate\_histogram(int p) {  //默认直方图里依照最大值为100%，即最大值对应的国家“\*”号占满  n = (int)(100 \* (deathRate[p] / 9.230146));  printf("%-10s", nation[p]);  while (n--) {  printf("\*");  }  printf("\n");  }  //按死亡人数  void death\_histogram(int p) {  //默认直方图里依照最大值为100%，即最大值对应的国家“\*”号占满  n = (int)(2 \* (0.5 + (50 \* death[p]) / 571179));  printf("%-10s", nation[p]);  while (n--) {  printf("\*");  }  printf("\n");  }  //输出表格  //按确诊人数  void c\_form(int p) {  printf("%3d|%10d|%8s|%10d|%10f\n", i + 1, confirm[p], nation[p], death[p], deathRate[p]);  }  //按死亡率  void dr\_form(int p) {  printf("%3d|%10f|%10d|%8s|%10d\n", i + 1, deathRate[p], death[p], nation[p], confirm[p]);  }  //按死亡人数  void d\_form(int p) {  printf("%3d|%10d|%8s|%10d|%10f\n", i + 1, death[p], nation[p], confirm[p], deathRate[p]);  }  //按照确诊人数排序  void confirm\_sort() {  printf("确诊人数表格：\n");  printf("序号 确诊人数 国家 死亡人数 死亡率%%\n");  //开始冒泡排序  for (i = 1; i < 17; i++) {  for (j = 0; j < 17 - i; j++) {  //这里不想破环原数组，故修改和判断的都是copy的数组  if (copy\_confirm[j] > copy\_confirm[j + 1]) {  //原来的三个数组中的对应关系不变，故只要知道改变后的位置即可  temp = temparr1[j];  temparr1[j] = temparr1[j + 1];  temparr1[j + 1] = temp;  temp = copy\_confirm[j];  copy\_confirm[j] = copy\_confirm[j + 1];  copy\_confirm[j + 1] = temp;  }  }  }  for (i = 0; i < 17; i++) {  //将改变后的位置传入，以打印表格  c\_form(temparr1[16 - i]);  }  printf("确诊人数直方图：\n");  for (i = 0; i < 17; i++) {  //将改变后的位置传入，以打印直方图  confirm\_histogram(temparr1[16 - i]);  }  }  //按照死亡率排序  void deathrate\_sort() {  printf("死亡率表格：\n");  printf("序号 死亡率%% 死亡人数 国家 确诊人数\n");  //开始冒泡排序  for (i = 1; i < 17; i++) {  for (j = 0; j < 17 - i; j++) {  //修改和判断的都是copy的数组  if (copy\_deathRate[j] > copy\_deathRate[j + 1]) {  //原来的三个数组中的对应关系不变，故只要知道改变后的位置即可  temp = temparr2[j];  temparr2[j] = temparr2[j + 1];  temparr2[j + 1] = temp;  ftemp = copy\_deathRate[j];  copy\_deathRate[j] = copy\_deathRate[j + 1];  copy\_deathRate[j + 1] = ftemp;  }  }  }  for (i = 0; i < 17; i++) {  //将改变后的位置传入，以打印表格  dr\_form(temparr2[16 - i]);  }  printf("死亡率直方图：\n");  for (i = 0; i < 17; i++) {  //将改变后的位置传入，以打印直方图  deathRate\_histogram(temparr2[16 - i]);  }  }  //按照死亡人数排序  void death\_sort() {  printf("死亡人数表格：\n");  printf("序号 死亡人数 国家 确诊人数 死亡率%%\n");  //开始选择排序  for (i = 0; i < 16; i++) {  k = i;  //修改和判断的都是copy的数组  for (j = i + 1; j < 17; j++) {  if (copy\_death[k] > copy\_death[j]) {  k = j;  }  }  if (k != i) {  temp = temparr3[i];  temparr3[i] = temparr3[k];  temparr3[k] = temp;  temp = copy\_death[i];  copy\_death[i] = copy\_death[k];  copy\_death[k] = temp;  }  }  for (i = 0; i < 17; i++) {  //将改变后的位置传入，以打印表格  d\_form(temparr3[16 - i]);  }  printf("死亡人数直方图：\n");  for (i = 0; i < 17; i++) {  //将改变后的位置传入，以打印直方图  death\_histogram(temparr3[16 - i]);  }  }  int main() {  //先把各个国家的死亡率%计算出来，存在数组中  for (i = 0; i < 17; i++) {  deathRate[i] = 100\*((float)death[i] / confirm[i]);  copy\_deathRate[i] = 100\*((float)death[i] / confirm[i]);  }  int c;  while (1) {  printf("按确诊人数排列请按1，按死亡率排列请按2，按死亡人数排列请按3，输入0退出\n");  scanf\_s("%d",&c);  if (c == 1) {  confirm\_sort();  }  else if (c == 2) {  deathrate\_sort();  }  else if (c == 3) {  death\_sort();  }  else if (c == 0) {  return 0;  }  }  }   1. **实验结果**   1.按确诊人数冒泡排序得：    2.按死亡率冒泡排序得：    3.按死亡人数选择排序得：     1. **实验心得**   （本次实验遇到的问题，解决过程，有什么收获等）  关于排序的时候不破换原数组而且要记录排序后的顺序，再由这个新的顺序数组去调用打印表格的函数，看题的时候感觉遥不可及，但一步步来还是可以实现的。每次综合实验都让我复习了很多知识，也让知识得以应用，代码只看不敲是学不会的。  当然，像输出%符号、输出文字左右对齐这些课本上没有的知识，也要能在网上熟练地查到。 |

深圳大学学生实验报告用纸

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 成绩评定：   |  |  |  |  | | --- | --- | --- | --- | | 实验过程（60） | 实验结果（25） | 实验心得（15） | 合计（100） | |  |  |  |  |   指导教师签字： 2021年 月 日 |
| 备注： |

注：1、报告内的项目或内容设置，可根据实际情况加以调整和补充。

2、教师批改学生实验报告时间应在学生提交实验报告时间后10日内。