|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **清华大学本科生考试试题专用纸**  考试课程计算机语言与程序设计（卷） 2021年 11月13日  姓名 班级自12学号   |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | 大题 | 题号 | 一 | 二 | 三 | 总分 | | 分数 | **35分** | **45分** | **20分** | | 成绩 | |  |  |  |  |   **试卷提交说明**   1. **在考卷上填写你的姓名、学号；** 2. **所有程序，都只能使用C89/C99标准包含的头文件；并且禁止使用任何全局变量；** 3. **按要求编程，调试运行，将每道题的程序进行复制，将运行结果进行截图，粘贴在考卷对应的题后。答卷中的程序和截图仅做备用，以提交的.c文件为准；** 4. **试卷中示例运行结果的提示语、空行仅做参考，不参与评分，不必在编程中输出与示例完全一致的提示语。只有有实际作用的输入、输出、换行等数据参与评分。** 5. **请严格按照函数要求的形参和返回值定义函数；** 6. **交卷时，提交一个压缩文件包，包括：**    1. **文件包名“学号.zip”，如2021123456.zip**    2. **ZIP文件包内容**       * + - **每道题的c语言源文件，文件名为“学号\_题号.c”,如2021123456\_1.c；**           - **考卷，文件名为“学号\_考卷.docx”**           - **例如：2021123456.zip应包括2021123456\_1.c、2021123456\_2.c、2021123456\_3.c、2021123456\_考卷.docx，共4个文件。** 7. **字符串处理（35分）**   有一种技巧可以对英文句子中的英文单词进行编码，下面是它的工作原理：首先，找到该句子中的最长单词，如“involution”。然后，找到该句子中出现频率最高的单词，如“run”。接着将第二个单词拼接在第一个单词后面，得到新的单词，如“involutionrun”。如果新单词中包含有重复的字母，那么只保留第一个，其余几个丢弃，如“involutr”。现在，将修改过的单词列于字母表的下面，如下所示：   |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | a | b | c | d | e | f | g | h | i | j | k | l | m | n | o | p | q | r | s | t | u | v | w | x | y | z | | **i** | **n** | **v** | **o** | **l** | **u** | **t** | **r** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |   最后，将字母表中没有出现在修改过的单词内的字母，排在修改过的单词之后，即得到编码表：   |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | a | b | c | d | e | f | g | h | i | j | k | l | m | n | o | p | q | r | s | t | u | v | w | x | y | z | | **i** | **n** | **v** | **o** | **l** | **u** | **t** | **r** | **a** | **b** | **c** | **d** | **e** | **f** | **g** | **h** | **j** | **k** | **m** | **p** | **q** | **s** | **w** | **x** | **y** | **z** |   基于这个编码表，我们就可以对英文句子中的英文单词进行编码。请按照要求编写程序，程序结构如下  **注意**：英文句子长度不超过200，且仅包含空格，逗号(,)，句号(.)，与小写英文字母。   1. **函数void Longest(char message[], char lword[])（6分）**   通过参数接收主函数提供的字符串message，找到其中的最长单词，赋值给lword并输出。如果最长的有多个，则选择**最后出现在句子中的那一个**。  如字符串message是 “you can check out any time you like, but you can never leave.”，则得到最长单词lword为“leave”。   1. **函数void Freq(char message[], char fword[])（8分）**   通过参数接收主函数提供的字符串message，找到其中出现频率最高的单词，赋值给fword并输出。如果频率最高的有多个，则选择**最后出现在句子中的那一个**。  如字符串message是 “re so so si do si la, so la si si si si la si la so.”，则得到出现频率最高的单词fword为“si”。  **禁止使用strtok函数**   1. **函数void Concat(char lword[], char fword[], char word[])（8分）**   通过参数接收主函数提供的字符串lword和fword，将fword拼接在lword后面，得到新字符串。如果新字符串中包含有重复的字母，则只保留第一个，其余几个丢弃，最后将该字符串赋值给word并输出。  如字符串lword是 “involution”，字符串fword是 “run”，则最后得到的字符串word为“involutr”。   1. **函数void Encryption(char message[], char word[])（8分）**   通过参数接收主函数提供的字符串message和word，按照题干描述生成编码表，并输出编码表的第二行。最后利用这个编码表对英文句子中的**英文单词**进行编码，并输出编码结果。   1. **主函数main()（5分）**    * 1. 定义字符串message，lword，fword，word。      2. 从键盘读入字符串message。      3. 调用**Longest**函数，找到最长的单词lword并输出。      4. 调用**Freq**函数，找到频率最高的单词fword并输出。      5. 调用**Concat**函数，拼接上述单词并去重，得到word并输出。      6. 调用**Encryption**函数，利用上述单词生成编码表并输出第二行，最后对message进行编码并输出编码结果。   ***[程序示例]***  【Input】  .hey jude dont make it bad,. take a sad song and make it better.  【Output】  better  it  betri  b e t r i a c d f g h j k l m n o p q s u v w x y z  .diy guri rmls kbhi fs ebr,. sbhi b qbr qmlc blr kbhi fs eissip.  【Run】    *请在此处粘贴程序和运行结果截图*   1. **数组处理（45分）**   卷积运算是图像处理中常见的一种运算，通过卷积运算我们可以完成图像的平滑、锐化、边缘提取等功能。卷积运算中涉及到两个量，一个是图像矩阵，另一个是卷积核矩阵。在代码实现中，我们可以通过“卷积核旋转180°”→“旋转后的卷积核与图像矩阵进行互相关运算”来得到卷积运算的结果。两个矩阵互相关运算的定义如下：  设下图中蓝色的5\*5矩阵为A矩阵（元素为格子中心值，即[3 3 2 1 0; 0 0 1 3 1; 3 1 2 2 3; 2 0 0 2 2; 2 0 0 0 1]），蓝色矩阵内阴影中的3\*3矩阵为B矩阵（元素为格子右下角值，即[0 1 2; 2 2 0; 0 1 2]）。则A和B的互相关运算的结果是图中绿色的3\*3矩阵（元素为[12 12 17; 10 17 19; 9 6 14]）。运算过程是，将B矩阵在A矩阵上逐格滑动（左上角开始，从左到右，从上到下），在滑动的每一个位置处，计算A矩阵和B矩阵重叠部分（阴影部分）对应元素的乘积的和，即为互相关运算在该位置处的值（如在第一个位置处，该值为3\*0+3\*1+2\*2 + 0\*2+ 0\*2+1\*0+3\*0+1\*1+2\*2=12）。当从左上角滑动到右下角后，就可以得到最终的互相关运算结果，即绿色矩阵。下面请按要求编写函数逐步实现卷积运算：    **注意**：请使用malloc等函数实现本题所有二维数组（行数列数均小于50）的**动态内存分配**；如果**不使用动态内存分配**（可自拟函数的形参和返回值），**扣15分**。   1. **函数int \*\*InputMatrix(int m, int n)（8分）**   该函数接收两个正整数参数m,n，动态创建一个m行n列的整形二维数组，并要求用户依次输入数组各元素的值，然后返回赋值后的数组。   1. **函数int PrintMatrix(int \*\*matrix, int m, int n)（6分）**   该函数接收一个二维数组 matrix，两个正整数m，n作为形参，其中m，n为matrix的行数和列数。该函数功能为按矩阵形式输出二维数组matrix的各个元素   1. **函数void Rotation180(int \*\*matrix, int m, int n)（12分）**   该函数接收一个二维数组 matrix，两个正整数m，n作为形参，其中m，n为matrix的行数和列数。该函数的功能为将matrix矩阵旋转180°（直接操作传入的matrix数组，无返回值）。   1. **函数int \*\*Corr2(int \*\*matrix1, int \*\*matrix2, int m1, int n1, int m2, int n2)（13分）**   该函数接收两个二维数组 matrix1，matrix2，四个正整数m1，n1，m2，n2作为形参，其中m1，n1为matrix1的行数和列数，m2，n2为matrix2的行数和列数，且满足m1>m2, n1>n2（即matrix1比matrix2的行数和列数都要大）。该函数的功能为计算矩阵matrix1和matrix2的互相关，并将结果以整形二维数组形式返回。   1. **主函数main()（6分）** 2. 从键盘读入两个整数m1,n1作为图像矩阵的行数和列数，并调用函数**InputMatrix**读入图像矩阵元素**，**创建图像矩阵MatrixA。 3. 从键盘读入两个整数m2,n2作为卷积核矩阵的行数和列数，并调用函数**InputMatrix**读入卷积核矩阵元素**，**创建卷积核矩阵MatrixB。 4. 调用**Rotation180**函数，将卷积核矩阵MatrixB旋转180°，并调用**PrintMatrix**函数输出旋转后的卷积核矩阵。 5. 调用**Corr2**函数，计算图像矩阵MatrixA和旋转180°之后的卷积核矩阵的互相关，也即图像矩阵和原始卷积核矩阵的卷积。 6. 调用**PrintMatrix**函数，输出上述卷积的结果。 7. 手动释放动态开辟的内存空间。   ***[程序示例]***  【Input】  3 3  1 2 3  4 5 6  7 8 9  2 2  1 2  3 4  【Output】  输入图像矩阵的行数和列数: 3 3  请依次输入矩阵的元素  1 2 3  4 5 6  7 8 9  输入卷积核矩阵的行数和列数: 2 2  请依次输入矩阵的元素  1 2  3 4  旋转后的卷积核矩阵为  4 3  2 1  卷积结果为  23 33  53 63  【Run】    *请在此处粘贴程序和运行结果截图*   1. **递归编程（20分）**   递归函数是一种特殊的函数类型，通过不断地在函数里面调用函数本身，可以用较少的代码实现一些常见的操作。请**利用递归的思路**完成**十进制以内**的任意进制转换，为了降低难度，**本题只需要考虑正整数和正的纯小数**的转换，具体要求如下：   1. **函数Integer\_tran()（7分）**   该函数的功能为将一个十进制的正整数转换为任意给定的进制并输出到屏幕上。形参和返回值可以自拟，但是形参必须包含待转换的正整数和需要转换成的进制。   1. **函数Decimal\_tran ()（10分）**   该函数的功能为将一个十进制的正的纯小数转换为任意给定的进制并输出到屏幕上，**保留4位小数**，（**四舍五入或者截断保留均可**）。形参和返回值可以自拟，但是形参必须包含待转换的正的纯小数和需要转换成的进制。   1. **主函数main()（3分）** 2. 从键盘读入一个正整数和一个需要转换成的进制（10以内），然后调用**Integer\_tran**函数将该正整数转换为指定进制的值，显示在屏幕上。 3. 从键盘读入一个正的纯小数和一个需要转换成的进制（10以内），然后调用**Decimal\_tran**函数将该正的纯小数转换为指定进制的值，显示在屏幕上。   ***[程序示例]***  【Input】  10 2  0.6 8  【Output】  请输入正整数数值和要转换的进制:10 2  转换完成: 1010  请输入正纯小数数值和要转换的进制:0.6 8  转换完成: 0.4631  【Run】    *请在此处粘贴程序和运行结果截图* |
| **以上为全部题目** |