**SouthWest University**

**School of Computer and Information Science**

Program: Computer Science and Technology

Couse Title: C Programming

Chapter Title:

Unit No:

Assignment No.:

Issue Date:

Submission Deadline:

Assessor/Tutor:

Student: 蔡前哲

Student’s Reg. No: 222021321102065

***NOTES TO STUDENTS***

* Check carefully the submission date and the instructions given with the assignment. Late assignments will not be accepted.
* If you are unable to hand in your assignment on time and have valid reasons such as illness, you may apply (in writing) for an extension.
* **If you are caught plagiarizing, you could have your grade reduced to zero, or at worst, you could be excluded from the course.**
* Your submission document should include **screen shots** of VC++ showing your code and the output results of running your code.
* Complete your assignment in Word document and submit it to FTP server
* Your file name should start with your Reg. No, and then include your name, and assignment NO. for example “222012321062259\_李丹\_assignment1.docx”. Remember to link different fields together using the underscore character (\_).

***Formatting Concerns***

#1: You should have a header that explains the program

#2: There should be a space between the prompt and the place where you type in the value.

#3: Your work should be formatted nicely as in the sample output.

**STUDENT’S DECLARATION:**

I confirm that this is all my own work.

Student Signature: 蔡前哲

**Assignment Description**

**Scenario and Tasks：**

**Assignment Description**

**practice on chapter 7 Strings and pointer**

1. Write programs to realize the string functions such as strlen, strcat, strcmp, substr. (array/ pointer/ function)

|  |
| --- |
| Code:  #include<stdio.h>  #include<string.h>  int mystrlen(char \*p)  {  int cnt = 0;  while(\*p != '\0')  {  cnt++;  p++;  }  return cnt;  }  void mystrcat(char \*a, char \*b)  {  int num1 = 0, num2 = 0;  while(\*(a+num1) != '\0')  num1++;  while(\*(b+num2) != '\0')  {  \*(a+num1) = \*(b+num2);  num1++;  num2++;  }  \*(a+num1) = \*(b+num2); //复制\0  }  int mystrcmp(char \*a, char \*b)  {  int flag = 1;  int cmp;  int num1 = 0, num2 = 0;  while(\*(a+num1) != '\0' && \*(b+num2) != '\0')  {  int t = \*(a+num1) - \*(b+num2);  if(t > 0)  {  flag = 0;  cmp = 1; //前面大  break;  }  else if (t < 0)  {  flag = 0;  cmp = -1; //后面大  break;  }  num1++; num2++;  }    if(\*(a+num1) != '\0') //b是a的子串  {  flag = 0;  cmp = 1;  }  if(\*(b+num2) != '\0') //a是b的子串  {  flag = 0;  cmp = -1;  }  if(flag) return 0; //相同  else return cmp;  }  char \*mysubstr(char \*str, int offset, int length)  {  char \*str2;  int num = 0;  for(int i = offset; i < offset+length; i++)  {  \*(str2+num) = \*(str+i);  num++;  }  \*(str2+num) = '\0';  return str2;  }  int main()  {  char a[100], b[100], c[100];  scanf("%s%s", &a, &b);  strcpy(c, a);  int len = mystrlen(a);  printf("%s length: %d\n", a, len);    mystrcat(c, b);  printf("after mystrcat: %s\n", c);    int cmp = mystrcmp(a, b);  printf("after mystrcmp(\"%s\", \"%s\"): %d\n", a, b, cmp);  char \*d = mysubstr(a, 1, 2);  printf("after mysubstr(\"%s\", 1, 2): %s", a, d);  return 0;  } |

|  |
| --- |
| Screen Capture of The Code:  代码图：        运行图：    Two runs, the first two strings entered: "abc" "cdef", the second input: "abc" "abcd". |

1. Input the target string and the pattern string, and write a program to check whether the pattern string is in the target string, and if the pattern string is in the target, return the position in the target string. （Index）app7.14

|  |
| --- |
| Code:  #include<stdio.h>  #include<string.h>  int main()  {  char tstr[100], pstr[100];  printf("Please enter the target string and the pattern string: ");  scanf("%s%s", &tstr, &pstr);    int flag = 0, pos = 0; //是否匹配成功 首字母位置  int tlen = strlen(tstr), plen = strlen(pstr); //两字符串长度  for(int i = 0; i < tlen; i++) //逐位比较字符 外层为target string  {  if(tlen-i < plen) break; //目标串不够长  pos = i;  int num = i; //记录目标子串首字母位置 便于操作下标进行比较  flag = 1;  for(int j = 0; j < plen; j++)  {  if(\*(tstr+num) != pstr[j]) { flag = 0; break; } //字符串不一致  num++;  }  if(flag) break;  }  if(flag)  printf("The pattern string is in the target. The first letter's position is %d.\n", pos);  else  printf("The pattern string is not in the target.");  return 0;  } |

|  |
| --- |
| Screen Capture of The Code:  代码图：    运行图： |

3 、Read a series of words form the file and write a program that finds the “smallest” and the “largest” in a series of words.

The file would be as follows:

5

England

China

Japan

USA

Germany

|  |
| --- |
| Code:  #include<stdio.h>  #include<string.h>  int main()  {  FILE \*fp;  fp = fopen("t3.txt", "r");  int n;  char str[100][100];  char sml[100], lag[100];    fscanf(fp, "%d", &n);  for(int i = 0; i < n; i++)  fscanf(fp, "%s", str+i);  strcpy(sml, str[0]);  strcpy(lag, str[0]);  for(int i = 0; i < n; i++)  {  if(strcmp(str[i], sml) < 0) strcpy(sml, str[i]); //有更小的  if(strcmp(str[i], lag) > 0) strcpy(lag, str[i]); //有更大的  }  printf("The largest string is %s.\n The smallest string is %s.", lag, sml);  return 0;  } |

|  |
| --- |
| Screen Capture of The Code:  代码图：    运行图： |

4、one of the oldest known encryption techniques is the Caesar cipher, attributed to Julius Caesar. It involves replacing each letter in a message with another letter that is a fixed number of positions later in the alphabet. (If the replacement would go past the letter Z, the cipher “wraps around” to the beginning of the alphabet.) You may assume that the message does not exceed 80 characters. Characters other than letters should be left unchanged.

Read the message from the file1.txt, and the file would be as follows:

Hello the world

Input the shit amount from the keyboard and write down the encrypted message to the file file2.txt.

|  |
| --- |
| Code:  #include<stdio.h>  #include<string.h>  int main()  {  FILE \*fp;  fp = fopen("file1.txt", "r");  char ch[100], ostr[100], estr[100];  int sa; //加密间隔    while( fgets(ch, 81, fp) != NULL) //读入文件全部内容  strcat(ostr, ch);  fclose(fp);    printf("\nPlease enter the shift amount(from 0 to 26):");  scanf("%d", &sa);  for(int i = 0; i < strlen(ostr); i++)  {  char t = ostr[i];  if(t < 'A' || t > 'z' || ('Z' < t && t < 'a') ) //不是字母 直接复制不处理  {  estr[i] = t;  continue;  }  if('a' < t && t < 'z') //小写字母  estr[i] = (t+sa > 'z') ? (t+sa-26) : (t+sa);  if('A' < t && t < 'Z') //大写字母  estr[i] = (t+sa > 'Z') ? (t+sa-26) : (t+sa);    }  fp = fopen("file2.txt","w");  fprintf(fp, "The encrypted message:\n%s", estr);  fclose(fp);  return 0;  } |

|  |
| --- |
| Screen Capture of The Code:  代码图：    运行图： |

5、Write a program that tests whether two words are anagrams (变位词permutations of the same letters);

Enter the first word: smartest

Enter the second word: mattress

The words are anagrams.

Enter the first word: dumbest

Enter the second word: stumble

The words are not anagrams.

|  |
| --- |
| Code:  #include<string.h>  #include<stdio.h>  int main()  {  char str1[100], str2[100];  int len1, len2, flag = 1, cnt[150];  memset(cnt, 0, sizeof(cnt));  printf("Enter the first word: ");  scanf("%s", str1);  printf("Enter the second word: ");  scanf("%s", str2);  len1 = strlen(str1);  len2 = strlen(str2);  if(len1 != len2) flag = 0; //长度不同 不可能是anagram  else if(!strcmp(str1, str2)) flag = 0; //两字符串相同  else  {  for(int i = 0; i < len1; i++)  cnt[str1[i]]++; //对每个字母计数  for(int i = 0; i < len2; i++)  cnt[str2[i]]--;  for(int i = 'A'; i <= 'z'; i++)  if(cnt[i]) //两单词的字母个数不同 不是anagram  {  flag = 0;  break;  }  }  if(flag)  printf("The words are anagrams.");  else  printf("The words are not anagrams.");    return 0;  } |

|  |
| --- |
| Screen Capture of The Code:  代码图：  运行图： |

6、Write a Program that reverses the words in a sentence.

Enter a sentence: you can cage a swallow can’t you?

Reversal of sentence: you can’t swallow a cage can you?

|  |
| --- |
| Code:  #include<string.h>  #include<stdio.h>  int main()  {  char str[100][100], ch; //句子 标点  int num = 0; //单词计数  while(scanf("%s", str+num) != EOF)  num++;    int len = strlen(str[num-1]);  ch = str[num-1][len-1]; //处理最后的符号  str[num-1][len-1] = '\0';  for(int i = num-1; i > 0; i--)  printf("%s ", str[i]);  printf("%s%c", str[0], ch); //输出最后一个单词和符号  return 0;  } |

|  |
| --- |
| Screen Capture of The Cde:  代码图：  运行图： |

7、Read a string from the keyboard /file and write a program to generate the following outputs: Construct a function Sort the string from the minimum character to maximum character. （pointer&function）

|  |
| --- |
| Code:  #include<stdio.h>  #include<string.h>  void bubble(char str[])  {  int len = strlen(str);  for(int i = 0; i < len-1; i++)  {  for(int j = 0; j < len-1-i; j++)  if(str[j] > str[j+1])  {  char t = str[j+1];  str[j+1] = str[j];  str[j] = t;  }  }  printf("%s", str);  }  int main()  {  char str[100];  scanf("%s", str);  bubble(str);  return 0;  } |

|  |
| --- |
| Screen Capture of The Code:  代码图：    运行图： |

8、Read the strings form the file, and then sort the strings (array of pointers and 2-d arrays) by BUBBLE SORT and EXCHANGE MAXIMUM SORT. (function)App7.18 The file would be as follows:

5

England

China

Japan

USA

Germany

|  |
| --- |
| Code:  #include<stdio.h>  #include<string.h>  int main()  {  int n;  char str[100][100];  FILE \*fp;  fp = fopen("t8.txt","r");  fscanf(fp, "%d", &n);  for(int i = 1; i <= n; i++)  fscanf(fp, "%s", str[i]);    //冒泡排序  for(int i = 1; i < n; i++)  for(int j = 1; j <= n-i; j++)  {  if(strcmp(str[j], str[j+1]) > 0) //交换大的字符串到后面  {  char t[100];  memcpy(t, str[j], sizeof(str[j]));  memcpy(str[j], str[j+1], sizeof(str[j+1]));  memcpy(str[j+1], t, sizeof(t));  }  }  for(int i = 1; i <= n; i++)  printf("%s\n", str[i]);  return 0;  } |

|  |
| --- |
| Screen Capture of The Code:  代码图：    运行图： |

**\*9. Many programs use a lot of strings. For different strings, we hope to have a way to judge similar programs. We define a set of operations to make two different strings the same. The specific operation method is:**

**1. Modify a character (such as replacing "a" with "b");**

**2. Add a character (such as changing "abdd" to "aebdd");**

**3. Delete a character (such as changing "travelling" to "traveling");**

**For example, for the two strings "abcdefg" and "abcdef", we think that we can achieve this by increasing/decreasing a "g". The above two options are only needed once. The number of times required for this operation is defined as the distance between two strings, and the similarity is equal to the reciprocal of "distance +1". That is, the distance between "abcdefg" and "abcdef" is 1, and the similarity is 1/2 = 0.5.**

**Given any two strings, can you write an algorithm to calculate their similarity?**

|  |
| --- |
| Code:  #include<stdio.h>  #include<string.h>  int max(int a, int b)  {  return a>b ? a : b;  }  int LCS(char a[], char b[], int x, int y)  {  if(x == 0 || y == 0) return 0; //边界    if(a[x] == b[y])  return LCS(a, b, x-1, y-1) + 1;  else  return max( LCS(a, b, x-1, y), LCS(a, b, x, y-1) );  }  int main()  {  char a[100] = " ", b[100] = " ", t1[100], t2[100];  int alen, blen, lcs = 0;  double ans = 0; //相似度  printf("Please enter two strings:\n");  scanf("%s%s", t1, t2);  alen = strlen(t1);  blen = strlen(t2);  strcat(a, t1); strcat(b, t2); //首位留空 便于处理边界  lcs = LCS(a, b, alen, blen);  //printf("%d\n", lcs);  ans = (double)1/(lcs+1);  printf("Their similarity is %lf", ans);  return 0;  } |

|  |
| --- |
| Screen Capture of The Code:  代码图：    运行图： |

**\*10** **Matrix-vector Multiplication(P323)**

|  |
| --- |
| Code: |

|  |
| --- |
| Screen Capture of The Code:  代码图：  运行图： |

Assessor’s Signature

Date