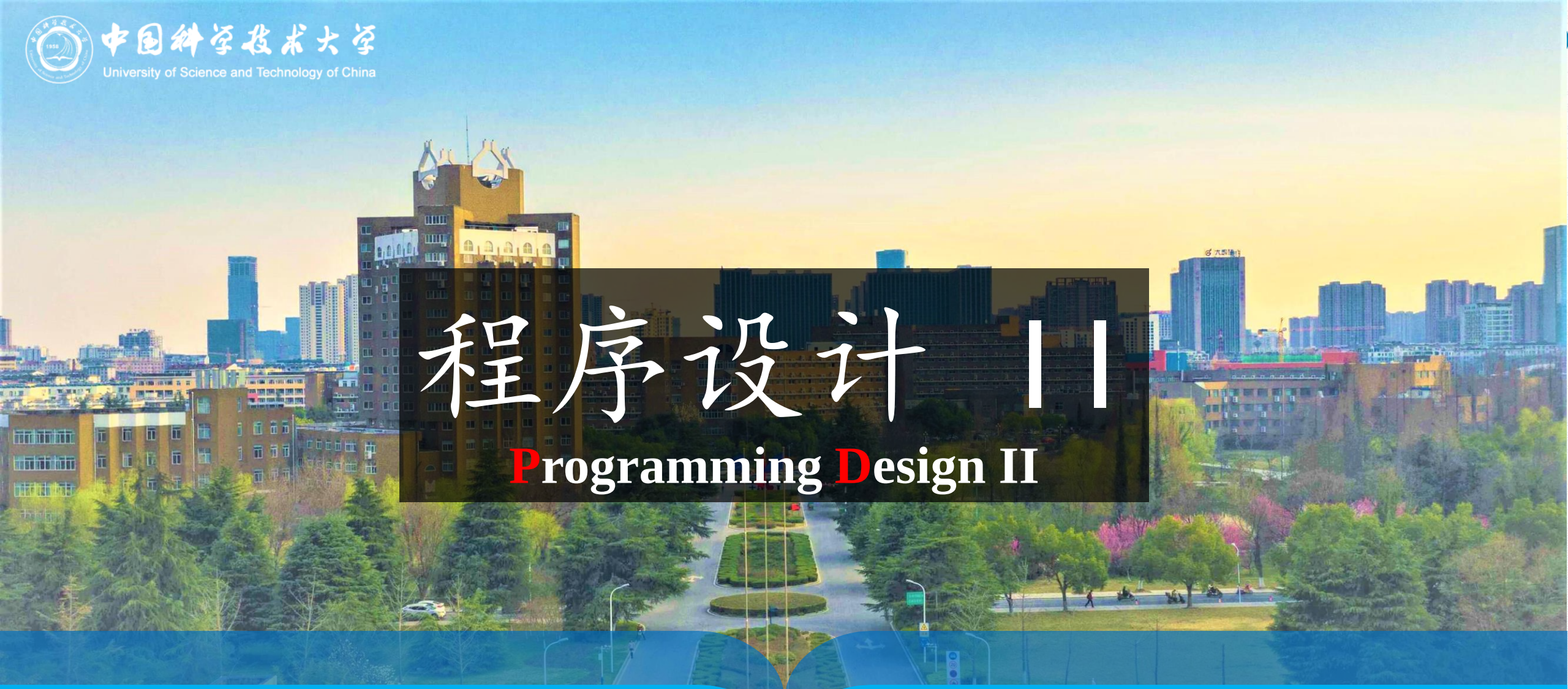

中国科学技术大学
University of Science and Technology of China
程序设计 II
Programming Design II

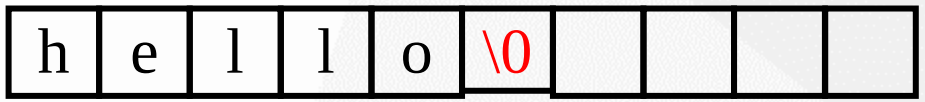
字符串处理

主讲：吴锋

目录
CONTENTS

字符串的基本概念

- 每个字符串是一个特殊的数组，满足两个条件
  - 元素的类型为char
  - 最后一个元素的值为‘\0’, Ascii码就是 0
- 以字符型数组存储
  - 从0号元素开始存储
  - 最大可以存储长度为N-1的字符串，N是数组的大小
- 字符串“hello”在长度为10的字符串数组中的存储



字符串的处理函数

- 将格式化数据写入字符串：sprintf
- 字符串长度查询函数： strlen
- 字符串复制函数：strcpy、strncpy
- 字符串连接函数： strcat
- 字符串比较函数： strcmp、strncmp、stricmp、strnicmp
- 字符串搜索函数： strcspn、strspn、strstr、strtok、strchr
- 字符串大小写转换函数： strlwr、strupr
- 这些函数都要求 #include <string.h>

字符串拷贝和求长度

- 拷贝： char *strcpy(char *dest, const char*src)
- 求长度： int strlen(const char *s)

```
#include <stdio.h>
#include <string.h>
int main() {
    char str1[10]="hello", str2[12];
    strcpy(str2,"hello world");
    printf("length:%d(str1); %d(str2)\n", strlen(str1), strlen(str2));
    strcpy(str1, str2);
    printf("length:%d(str1); %d(str2)\n",
    strlen(str1), strlen(str2));
    printf("%s\n", str1);
    return 0;
}
```

把“hello world”复制到str2

查询str1中字符串的长度

把str2复制到str1

字符串拷贝和求长度

- 输出结果：

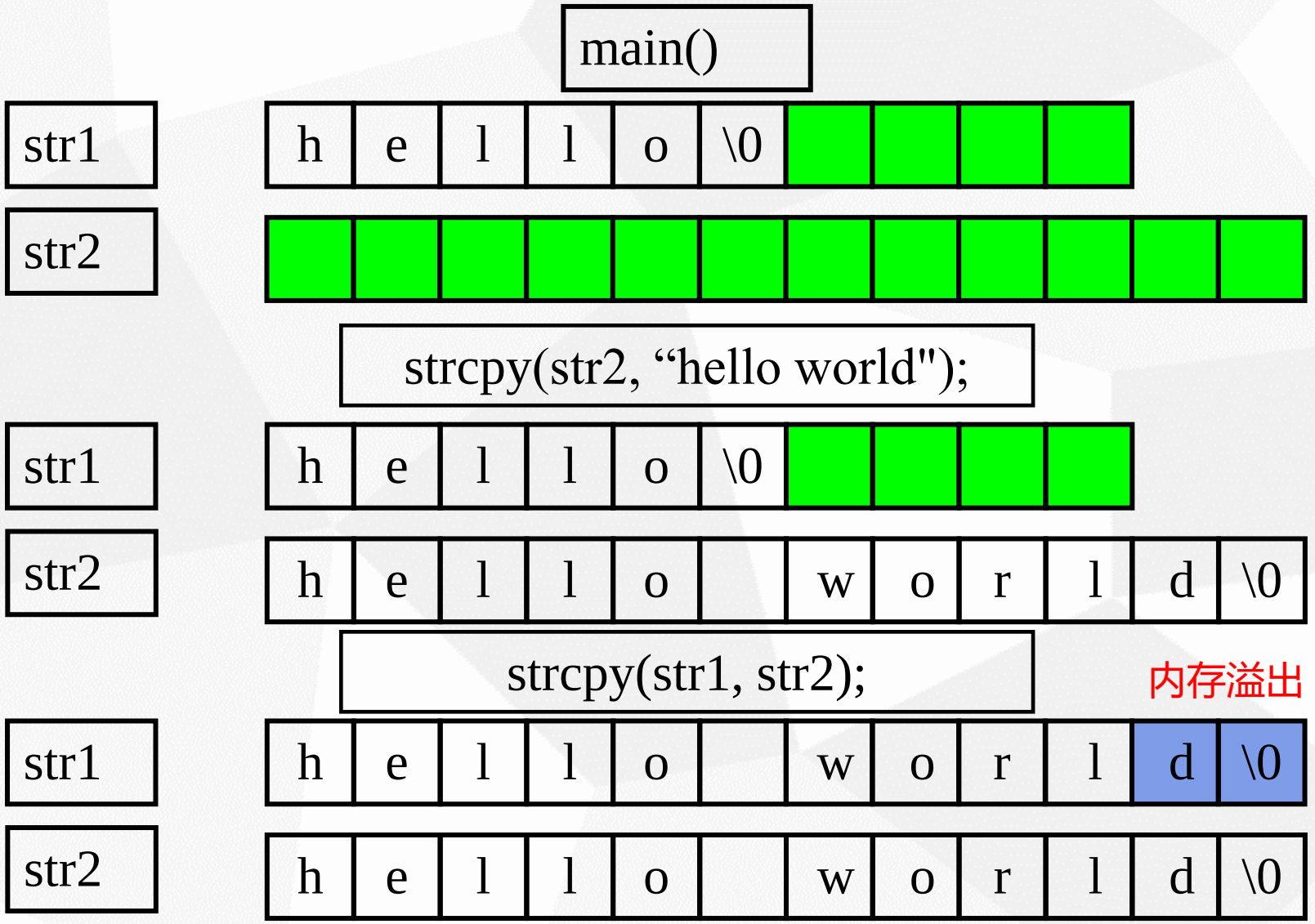
  length:5(str1); 11(str2)

  length:11(str1); 11(str2)

  hello world

- str1存储了11个非’\0’字符？
  - strcpy在复制字符串str2到str1时，不检查str2是否超出了str1的存储容量，而是直接将str2中存储的字符串复制到从str1开始的一段连续区域。
  - 在程序中要特别注意这种情况所引发的程序运行不确定性。

字符串的拷贝


main()
str1
h e l l o \0
str2
strcpy(str2, “hello world");
str1
h e l l o \0
str2
h e l l o w o r l d \0
strcpy(str1, str2);
内存溢出
str1
h e l l o w o r l d \0
str2
h e l l o w o r l d \0

求字符串长度

```
//看s中是否包含 c
int MyStrchr(char * s,  char c) {
    for( int i = 0; i < strlen(s) -1 ; i ++ )
        if( s[i] ==  c)
            return 1;
    return 0;
}
```

- 问题：strlen 是一个O(N)的函数，每次判断 i < strlen(s) – 1 都要执行，太浪费时间了。

字符串的连接

- 连接：**char *strcat(char *dest, const char *src)**

把src内容加到dest后面，同样不会考虑 dest是否够长

```
#include <stdio.h>
#include <string.h>
int main() {
    char str1[100]="hello", str2[10]="^_^";
    strcat(str1," world ");
    printf("%s\n", str1);
    strcat(str1, str2);
    printf("%s\n", str1);
    return 0;
}
```

把”world”添加到str1中原字符串的末尾

把str2中的字符串添加到str1中原字符串的末尾

输出：
hello world
hello world ^_^

字符串的比较

- 比较：**int strcmp(const char *s1, const char *s2)**
  - 区分字符串中字符的大小写
  - 字符串根据字典序
    - 如果s1 < s2，则返回值 < 0
    - 如果s1 == s2，则返回值 == 0
    - 如果s1 > s2，则返回值 > 0
- 比较：**int stricmp(const char *s1, const char *s2)**
  - 不区分字符串中字符的大小写
  - 返回值情况与strcmp相同

字符串的比较

```c
#include <string.h>
#include <stdio.h>
char string1[] = "The quick brown dog jumps over the lazy fox";
char string2[] = "The QUICK brown dog jumps over the lazy fox";
int main() {
    int result;
    printf("Compare strings:\n\t%s\n\t%s\n\n", string1, string2);
    result = strcmp( string1, string2);
    printf("strcmp: result=%d\n", result);
    result = stricmp( string1, string2);
    printf("stricmp: result=%d\n", result);
    return 0;
}
```

输出：

```
Compare strings:
        The quick brown dog jumps over the lazy fox
        The QUICK brown dog jumps over the lazy fox

strcmp: result=1
stricmp: result=0
```

字符串中查找子串

- 查找子串：char *strstr(char *s1, char *s2);
  - 查找给定字符串在字符串中第一次出现的位置。
  - 如果找到，则返回一个指向s1中第一次出现s2的位置的指针，及子串的起始地址。
  - 如果找不到，则返回 NULL。

字符串中查找子串

```
#include <string.h>
#include <stdio.h>
char str[] = "lazy";
char string[] = "The quick brown dog jumps over the lazy fox";
int main() {
   char *pdest;
   int  result;
   pdest = strstr(string, str);
   result = pdest - string + 1;
   if(pdest != NULL)
      printf("%s found at position %d\n\n", str, result);
   else
      printf("%s not found\n", str);
   return 0;
}
```

在string中搜索str，返回str在string中第一次出现的位置

输出：　lazy found at position 36

字符串中查找字符

- 查找字符：char *strchr(char *s, int c);
  - 查找给定字符在字符串中第一次出现的位置。
  - 如果找到，则返回一个指向s1中第一次出现c的位置的指针，及子串的起始地址。
  - 如果找不到，则返回 NULL。

字符串中查找字符

```
#include <string.h>
#include <stdio.h>
int  ch = 'r';
char string[] = "The quick brown dog jumps over the lazy fox";
int main() {
   char *pdest;
   int result;
   pdest = strchr(string, ch);
   result = pdest - string + 1;
   if( pdest != NULL )
      printf("Result:\tfirst %c found at position %d\n\n", ch, result);
   else
      printf("Result:\t%c not found\n");
   return 0;
 }
```

在string中搜索ch，返回str在string中第一次出现的位置

输出：　Result: first r found at position 12

字符串的部分拷贝

- 部分拷贝：char *strncpy(char *dest, char *src, int maxlen)
  - 将前 maxlen 个字符从src拷贝到dest。
  - 如果src中字符不足 maxlen 个，则连’\0’一起拷贝，’\0’后面的不拷贝。
  - 如果src中字符大于等于maxlen个，则拷贝 maxlen个字符。

字符串的部分拷贝

```c
#include <string.h>
#include <stdio.h>
int main() {
    char s1[20] = "1234567890";
    char s2[] =  "abcd" ;

    strncpy(s1, s2, 5);
    printf("%s\n", s1);

    strcpy(s1, "1234567890");
    strncpy(s1, s2, 4);
    printf("%s\n", s1);
    return 0;
}
```

输出：

abcd

abcd567890

数组函数参数传递

```
#include <stdio.h>
char str1[200] = "Hello,World", str2[100] = "Computer";
void swap(char s1[ ], char * s2) { //交换两个字符串的内容
     char c;
     for(int i = 0; s1[i] || s2[i]; i ++){  //  '\0'的Ascii 码就是 0
          c = s2[i]; s2[i] = s1[i]; s1[i] = c;
     }
     s1[i+1] = s2[i+1] = 0;
}
int main() {
     swap(str1, str2);
     printf("%s\n%s\n", str1, str2);
     return 0;
}
```

输出：
Computer
Hello,World

例题1：Caesar密码

- 问题描述（P110）
  - Julius Caesar 生活在充满危险和阴谋的年代。为了生存，他首次发明了密码，用于军队的消息传递。
  - 假设你是Caesar 军团中的一名军官，需要把Caesar 发送的消息破译出来、并提供给你的将军。消息加密的办法是：对消息原文中的每个字母，分别用该字母之后的第5个字母替换（例如：消息原文中的每个字母A都分别替换成字母F，V替换成A，W替换成B...），其他字符不 变，并且消息原文的所有字母都是大写的。
  - 密码中的字母与原文中的字母对应关系如下：

密码字母：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

原文字母：V W X Y Z A B C D E F G H I J K L M N O P Q R S T U

例题1：Caesar密码

- 输入
  - 最多不超过100个数据集组成。每个数据集由3部分组成。
  - 起始行：START
  - 密码消息：由1到200个字符组成一行，表示Caesar发出的一条消息。
  - 结束行：END
  - 在最后一个数据集之后，是另一行：ENDOFINPUT
- 输出
  - 每个数据集对应一行，是Caesar 的原始消息。

◎ 例题1：Caesar密码

- 样例输入

START
NS BFW, JAJSYX TK NRUTWYFSHJ FWJ YMJ WJXZQY TK YWNANFQ HFZXJX
END
START
N BTZQI WFYMJW GJ KNWXY NS F QNYYQJ NGJWNFS ANQQFLJ YMFS XJHTSI NS WTRJ
END
START
IFSLJW PSTBX KZQQ BJQQ YMFY HFJXFW NX RTWJ IFSLJWTZX YMFS MJ
END
ENDOFINPUT

- 样例输出

IN WAR, EVENTS OF IMPORTANCE ARE THE RESULT OF TRIVIAL CAUSES
I WOULD RATHER BE FIRST IN A LITTLE IBERIAN VILLAGE THAN SECOND IN ROME
DANGER KNOWS FULL WELL THAT CAESAR IS MORE DANGEROUS THAN HE

例题1：Caesar密码

```
#include <stdio.h>
#include <string.h>
int main() {
      char szLine[300];
      while(getline(szLine, 210, stdio)) {  //可用此方式判断数据是否读完
            if(strcmp(szLine,"ENDOFINPUT") == 0)  break;
            getline(szLine, 210, stdio); //读取密文
            for(int i = 0; szLine[i]; i++)
                  if(szLine[i] >= 'A' && szLine[i] <= 'Z') {
                        szLine[i] -= 5;
                        if(szLine[i] < 'A') szLine[i] = 'Z' - ('A' - szLine[i])  + 1;
                  }
            printf("%s\n", szLine);
            getline(szLine, 210, stdio);      //读取 END
      }
      return 0;
}
```

例题2：单词排序

- 问题描述
  - 输入若干行单词（不含空格），请按字典序排序输出。大小写有区别。单词一共不超过100行，每个单词不超过20字符

样例输入：

What
man
Tell
About
back

样例输出：

About
Tell
What
back
man

例题2：单词排序

- 思路：用二维字符数组保存多个单词
  - 如：char Word[100][30];

  则表达式 Word[i] 的类型就是 char *

  Word[i] 就是数组中的一行，就是一个字符串

  Word[i][0] 就是Word[i] 这个字符串的头一个字符

  - 二维字符数组也能初始化，例如：

  char week[7][10]={"Saturday", "Sunday", "Monday", "Tuesday", "Wednesday", "Thursday", "Friday"};

例题2：单词排序

```
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
char Word[100][30];
int MyCompare(const void * e1, const void * e2) {
    return strcmp((char * ) e1, (char * ) e2);
}
int main()  {
    int n = 0; //单词个数
    while(scanf("%s", Word[n]) != EOF && Word[n][0]) n++;
    qsort(Word, n, sizeof(Word[0]), MyCompare);
    for(int i = 0; i < n; i ++)
        printf("%s\n", Word[i]);
    return 0;
}
```

为了对付有可能最后一行读入的是空行

例题3：All in All

- 问题描述
  - 给定两个字符串s和t，请判断s是否是t的子序列。即从t中删除一些字符，将剩余的字符连接起来，即可获得s。
- 输入
  - 包括若干个测试数据。每个测试数据由两个ASCII码的数字和字母串s和t组成， s和t的长度不超过100000。
- 输出
  - 对每个测试数据，如果s是t的子序列则输出“Yes”，否则输出“No”。

◎ 例题3：All in All

- 样例输入

  sequence subsequence

  person compression

  VERDI vivaVittorioEmanueleReDiItalia

  caseDoesMatter CaseDoesMatter

- 样例输出

  Yes

  No

  Yes

  No

例题3：All in All

- 思路：看s中的每个字符是否在t中出现，而且顺序一致。
  - 设两个指针，分别指向s和t的开头，然后，
  - 如果t的指针指向的字符和s的指针指向的字符相同，则将s的指针向前移动一个字符。 t的指针则不停向前每次移动一个字符，
  - 直到s指针指向末尾的‘\0’说明找到Yes，或 t 的指针指向末尾的‘\0’，说明未找到 No。

例题3：All in All

```
#include <stdio.h>
char s[100010];
char t[100010];
int main() {
    int i,j;
    while (scanf( "%s%s",s,t) > 0 ) {
        for(i = 0,j = 0 ; s[i] && t[j]; j ++)
            if( t[j] == s[i] )i++;
        if (s[i] == 0)  // ‘\0’ 的Ascii 码就是 0
            printf("Yes\n");
        else
            printf("No\n");
    }
    return 0;
}
```

例题4：子串

- 问题描述（P108）
  - 给出一些由英文字符组成的大小写敏感的字符串的集合s，请找到一个最长的字符串x，使得对于s中任意字符串y，x或者是y的子串，或者x中的字符反序之后得到的新字符串是y的子串。
- 输入
  - 输入的第一行是一个整数t (1 <= t <= 10)，t表示测试数据的数目。对于每一组测试数据，第一行是一个整数n (1 <= n <= 100)，表示已经给出n个字符串。接下来n行，每行给出一个长度在1和100之间的字符串。

例题4：子串

- 输出
  - 对于每一组测试数据，输出一行，给出题目中要求的字符串x的长度。

样例输入:

```
2
3
ABCD
BCDFF
BRCD
2
rose
orchid
```

样例输出：

```
2
2
```

◎ 例题4：子串

- 思路：
  - 随便拿出输入数据中的一个字符串。
  - 从长到短找出它的所有子串，直到找到否符合题目要求的。
- 改进：
  - 不要随便拿，要拿输入数据中最短的那个。
  - 从长到短找出它的所有子串，直到找到否符合题目要求的。

例题4：子串

```
#include <stdio.h>
#include <string.h>
int t, n;   char str[100][101];
int searchMaxSubString(char* source);
int main()  {
     int i, minStrLen, subStrLen; char minStr[101];
     scanf("%d", &t);
     while(t--) {
          scanf("%d", &n);
          minStrLen = 100; //记录输入数据中最短字符串的长度
          for (i = 0; i < n; i++) {//输入一组字符串
               scanf("%s", str[i]);
               if ( strlen(str[i]) < minStrLen ) {//找其中最短字符串
                    strcpy(minStr, str[i]); minStrLen = strlen(minStr);
               }
          }
          subStrLen = searchMaxSubString(minStr);//找答案
          printf("%d\n", subStrLen);
     }
     return 0;
}
```

◎ 例题4：子串

```
int searchMaxSubString(char* source) {
    int subStrLen = strlen(source), sourceStrLen = strlen(source);
    int i, j;  bool foundMaxSubStr; char subStr[101], revSubStr[101];
    while (subStrLen > 0 ) {//搜索不同长度的子串，从最长的子串开始搜索
        for (i = 0; i <= sourceStrLen - subStrLen; i++) {
            //搜索长度为subStrLen的全部子串
            strncpy(subStr, source+i, subStrLen);
            strncpy(revSubStr, source+i, subStrLen);
            subStr[subStrLen] = revSubStr[subStrLen] = '\0';
            _strrev(revSubStr);   foundMaxSubStr = true;
            for( j = 0; j < n; j++)  //遍历所有输入的字符串
                if ( strstr(str[j], subStr) == NULL &&
                    strstr(str[j], revSubStr) == NULL ) {
                    foundMaxSubStr = false; break;
            }
            if (foundMaxSubStr)  return subStrLen;
        }
        subStrLen--;
    }
    return 0;
}
```

_strrev (char * s)
将字符串 s 颠倒

作业

- 3. 密码(P110)
  - Bob 和 Alice 开始使用一种全新的编码系统。它是一种基于一组私有钥匙的。他们选择了n 个不同的数a1、…、an, 它们都大于0 小于等于n。加密过程如下：待加密的信息放置在这组加密钥匙下，信息中的字符和密钥中的数字一一对应起来。信息中位于i位置的字母将被写到加密信息的第ai 个位置, ai 是位于i 位置的密钥。加密信息如此反复加密，一共加密 k 次。信息长度小于等于n。如果信息比 n 短, 后面的位置用空格填补直到信息长度为n。请你帮助 Alice 和 Bob 写一个程序，读入密钥，然后读入加密次数 k 和要加密的信息，按加密规则将信息加密。假设0 < n <= 200。

作业

- 6. 词典(P111)
  - 你旅游到了国外的一个城市，却不能理解那里的语言。不过幸运的是，你有一本词典可以帮助你。词典中包含不超过100000 个词条，而且在词典中不会有某个外语单词出现超过两次。现在给你一个由外语单词组成的文档，文档不超过100000 行，而且每行只包括一个外语单词。所有单词都只包括小写字母，而且长度不会超过10。请你把这个输入：首先输入一个词典，，每个词条占据一行。每一个词条包括一个英文单词和一个外语单词，两个单词之间用一个空格隔开。。词典之后是一个空行，然后文档翻译成英文，每行输出一个英文单词。如果某个外语单词不在词典中，就把这个单词翻译成“eh”。

作业

- 7. 最短前缀(P111)
  - 一个字符串的前缀是从该字符串的第一个字符起始的一个子串。例如"carbon"的字串是: "c", "ca", "car", "carb", "carbo", 和 "carbon"。注意，这里我们不认为空串是字串, 但是每个非空串是它自身的字串。我们希望能用前缀来缩略的表示单词。例如, "carbohydrate" 通常用"carb"来缩略表示。在下面的例子中，"carbohydrate" 能被缩略成"carboh", 但是 不能被缩略成"carbo" (或其余更短的前缀)，因为已经有一个单词用"carbo"开始

    carbohydrate、cart、carbonic、caribou、carriage、car

  - 一个精确匹配会覆盖一个前缀匹配，例如，前缀"car"精确匹配单词"car". 因此 "car" 是"car"的缩略语是没有二义性的 ,“car”不会被当成"carriage"或者任何在列表中以"car"开始的单词。现在给你一组单词, 要求你找到唯一标识每个单词的最短前缀 。假设输入输入的单词数量不少于2、不多于1000；每个单词的长度至少是1、至多是20。

作业

- 8. 浮点数格式(P112)
  - 输入n(n<=10000)个浮点数，要求把这n 个浮点数重新排列后再输出。每个浮点数中都有小数点、且总长度不超过50 位。

大作业

- 作业提交
  - 2022年5月25日前（发送至助教，建议提前完成，早提交早检查）
  - 文件名：学号-姓名 大作业.rar（报告、代码等）
  - 按照要求撰写实验报告，准备5分钟左右汇报ppt（技术+演示）
- 评分标准
  - 可以独立完成或组队完成（≤5人，要有明确分工并确定贡献百分比）
  - 项目质量：新颖度、代码量、算法难度、结论
  - 运行结果：可执行程序的运行结果
  - 代码风格：源码阅读，算法和风格检查
  - 编译结果：源码和可执行程序的一致性

◎ 大作业

• 实验报告格式

中国科学技术大学

本科实验报告（模板）

课程名称 _____

实验名称 _____

姓　　名 _____

学　　院 _____

系　　别 _____

专　　业 _____

年　　级 _____

学　　号 _____

任课教师 _____

2018 年　11　月　27　日

一、如何写实验报告

针对“计算机程序设计”课程实验的特点，建议在书写实验报告时应包括如下内容：

1. 实验目的

实验作为教学的一个重要环节，其目的在于更深入地理解和掌握课程教学中的有关基本概念，应用基本技术解决实际问题，从而进一步提高分析问题和解决问题的能力，我们着手做一个实验的时候，必须明确实验的目的，以保证达到课程所指定的基本要求。在写实验报告时，要进一步确认是否达到了预期的目的。

2. 实验内容

在本书中，每一个实验都安排了多个实验题目，根据教学安排、进度、实验条件、可提供的机时、学生的基础等因素，可以选择其中的几个或全部。因此，在实验报告中，实验内容是指本次实验中实际完成的内容。在每一个实验题目中，一般都提出了一些具体要求，其中有些具体要求是为了达到实验目的而提出的。因此，在实验内容中，不仅要写清楚具体的实验题目，还要包括具体要求。

3. 算法与流程图

算法设计是程序设计过程中的一个重要步骤。在本书中，对于某些实验题目给出了方法说明，有的还提供了流程图。如果在做实验的过程中，使用的算法或流程图和书中的不一样，或者书中没有给出算法和流程图，则在实验报告中应给出较详细的算法说明与流程图，并对主要符号与变量作相应的说明。

4. 程序清单

程序设计的产品是程序，它应与算法或流程图相一致。程序要具有易读性，符合结构化原则。

5. 运行结果

程序运行结果一般是输出语句所输出的结果。对于不同的输入，其输出的结果是不同的。因此，在输出结果之前还应注明输入的数据，以便对输出结果进行分析和比较。

6. 调试分析和体会

这是实验报告中最重要的一项，也是最容易忽视的一项。

实验过程中大量的工作是程序调试，在调试过程中会遇到各种各样的问题，每解决一个问题就能积累一点经验，提高编程的能力。因此，对实验的总结，最主要的是程序调试经验的总结。调试分析也包括对结果的分析、尚存在的问题和拟解决的方法等。

体会主要是指通过本次实验是否达到了实验目的，有哪些基本概念得到了澄清等。

大作业：聊天机器人

- 实验目的
  - 用C/C++等编程语言实现简单的聊天机器人
- 实验基本要求
  - 编写代码实现聊天的基本功能，可与程序使用者用自然语言一问一答；
  - 聊天内容可以是通用，也可以是特定范围，比如，假定聊天机器人为科大学生，与人聊天科大的学习、生活情况
- 加分项
  - 提供交互界面、支持中文等
  - 应用机器学习（ML）、自然语言理解（NLP）等

◎ 大作业：聊天机器人

- 聊天机器人，是一种通过自然语言模拟人类进行对话的程序。聊天机器人的研究源于经典的图灵测试。
- 最早的聊天机器人ELIZA诞生于1966年，由麻省理工学院（MIT）开发。ELIZA的实现技术仅为关键词匹配及人工编写的回复规则。
- 受到ELIZA的启发，Richard S. Wallace博士在1995年开发了ALICE系统。ALICE采用的是启发式模板匹配的对话策略，随着ALICE一同发布的AIML（Artificial Intelligence Markup Language）目前被广泛应用在移动端虚拟助手的开发中。
- Neuralconvo 是在 2016 年由 Julien Chaumond 和 Clément Delangue 使用深度学习训练创造出来的现代聊天机器人。它通过读取上千部电影脚本来“学习”，并识别出文本中的模型，然后到它受训练的语句中寻找类似的模型，然后生成一句新的句子返回给你。

大作业：自主选题

- 实验目的
  - 自定
- 实验要求
  - 书写实验报告，图文并茂。
- 注：须经老师认可