小马加编信息学教案(十三)

C++数组(四)

一. 课程内容

1. C++字符数组
2. C++字符数组的读入与输出
3. C++字符数组的处理函数

二. 知识讲解

1. 字符数组

1.1 字符数组的概念

> **数组中的每个元素都是一个字符的数组称为“字符数组”。有时，把一维字符数组又称为“字符串”。**
> 定义字符数组的方法与定义其他类型数组的方法类似。
> 对于字符数组的定义“char s[10] = {'H'，'e'，'l'，'l'，'o'}；”其在计算机内部的存储方式如下：

数组下标	0	1	2	3	4	5	6	7	8	9
元素值	H	e	l	l	o	\0				

1.2 字符数组的赋值

> **C++ 语言规定，可以将字符串直接赋值给字符数组**，例如：

```
char str[30] = {"c.biancheng.net"};
char str[30] = "c.biancheng.net";  //这种形式更加简洁
```

> 数组第 0 个元素为c，第 1 个元素为 . ，第 2 个元素为 b ，后面的元素以此类推。

> **为了方便，你也可以不指定数组长度**，从而写作：

```
char str[] = {"c.biancheng.net"};
char str[] = "c.biancheng.net";  //这种形式更加简洁
```

> **给字符数组赋值时，我们通常使用这种写法，将字符串一次性地赋值（可以指明数组长度，也可以不指明）**，而不是一个字符一个字符地赋值，那样做太麻烦了。

> 但需要注意的是，**字符数组只有在定义时才能将整个字符串一次性地赋值给它，一旦定义完了，就只能一个字符一个字符地赋值了**

1.3 字符串结束标志（重点）

> 字符串是一系列连续的字符的组合，要想在内存中定位一个字符串，除了要知道它的开头，还要知道它的结尾。
> 找到字符串的开头很容易，知道它的名字（字符数组名或者字符串名）就可以；然而，如何找到字符串的结尾呢?
> C++ 语言的解决方案有点奇妙，或者说有点奇葩。

> **在C++语言中，字符串总是以'\0'作为结尾，所以'\0'也被称为字符串结束标志，或者字符串结束符。**

> `'\0'` 是 $ASCII$ 码表中的第 0 个字符，英文称为 `NUL` ，中文称为“空字符”。
> 该字符既不能显示，也没有控制功能，输出该字符不会有任何效果，**它在 C++ 语言中唯一的作用就是作为字符串结束标志**。

> C++ 语言在处理字符串时，会从前往后逐个扫描字符，**一旦遇到 '\0' 就认为到达了字符串的末尾，就结束处理。 '\0' 至关重要，没有 '\0' 就意味着永远也到达不了字符串的结尾。**

> 由 " " 包围的字符串会自动在末尾添加 '\0' 。例如， "abc123" 从表面看起来只包含了 6 个字符，其实不然， C++ 语言会在最后隐式地添加一个 '\0' ，这个过程是在后台默默地进行的，所以我们感受不到。

> 需要注意的是，**逐个字符地给数组赋值并不会自动添加 '\0'**

2. C++字符数组的读入与输出

2.1 字符数组的读入

```
scanf("%s", letter);  //用 scanf 读入整个数组
scanf("%c",  &letter[i]);  //用 scanf 逐个元素读入
cin >> letter;  //用 cin 输入整个数组
cin >> letter[i];  //用 cin 逐个元素输入:
gets(letter);  //用 gets 读入整个数组:
letter[i] = getchar();  //用 getchar 逐个元素读入:
```

2.2 字符数组的输出

```
printf("%s", letter);  //用 printf 输出整个数组
printf("%c", letter[i]);  //用printf逐个元素输出
cout << letter;  //用 cout 输出整个数组
cout << letter[i];  //用 cout 逐个元素输出
puts(letter);  //用 puts 输出整个数组
putchar(letter[i]);  //用 putchar 逐个元素输出
```

2.3 字符串的读入与输出注意

- scanf() 和 gets() 是有区别的:

> **scanf() 读取字符串时以空格为分隔**，遇到空格就认为当前字符串结束了，**所以无法读取含有空格的字符串。**
> **gets() 认为空格也是字符串的一部分，只有遇到回车键时才认为字符串输入结束**，所以，不管输入了多少个空格，只要不按下回车键，对 gets() 来说就是一个完整的字符串。**"话说， gets() 用来读取一整行字符串。**

3. C++ 语言字符数组处理函数

3.1 strlen 函数

> strlen(char a[]) 用于**读取一个字符数组的长度**
> **其原理为从给的地址开始向后累计计数，直到遇到 '\0' 结束符为止**(不包括 '\0')

> ex:

```
char a[1005] = "abcde";
cout << strlen(a) << endl;
```

> 上述代码的输出结果为5

3.2 strcmp 函数

> strcmp 是 string compare 的缩写，意思是**字符串比较**，语法格式为：
> strcmp(str1, str2);
> $str1$ 和 $str2$ 是需要比较的两个字符串。

> 字符本身没有大小之分， strcmp() **以各个字符对应的 $ASCII$ 码值进行比较**。
> strcmp() 从两个字符串的第 0 个字符开始比较，如果它们相等，就继续比较下一个字符，直到遇见不同的字符，或者到字符串的末尾。

> **返回值：**
> 若 $str1$ 和 $str2$ 相同，则返回 0；
> 若 $str1$ 大于 $str2$，则返回大于 0 的值；
> 若 $str1$ 小于 $str2$，则返回小于 0 的值。

```
#include <bits/stdc++.h>
using namespace std;

int main() {
    char a[] = "aBcDeF";
    char b[] = "AbCdEf";
    char c[] = "aacdef";
    char d[] = "aBcDeF";

    printf("a VS b: %d\n", strcmp(a, b));
    printf("a VS c: %d\n", strcmp(a, c));
    printf("a VS d: %d\n", strcmp(a, d));
    return 0;
}
```

> 运行结果

```
a VS b: 32
a VS c: -31
a VS d: 0
```

3.3 strcpy 函数

> strcpy 是 string copy 的缩写，意思是**字符串复制**，**即将字符串从一个地方复制到另外一个地方**，语法格式为：

> `strcpy(str1, str2);`

> `strcpy()` 会把 $str2$ 中的字符串拷贝到 $str1$ 中，字符串结束标志 `'\0'` 也一同拷贝。

> 将 $str2$ 复制到 $str1$ 后，$str1$ 中原来的内容就被覆盖了。

> **另外，`strcat()` 要求 $str1$ 要有足够的长度，否则不能全部装入所拷贝的字符串。**

```
#include <bits/stdc++.h>
using namespace std;

int main() {
    char str1[50] = "xiaomajiabian";
    char str2[50] = "I LOVE C++";
    strcpy(str1, str2);
    printf("str1: %s\n", str1);
    return 0;
}
```

> 运行结果

```
str1: I LOVE C++
```

3. 4 `strcat` 函数

> `strcat` 是 `string catenate` 的缩写，意思是把两个字符串拼接在一起，语法格式为：
> `strcat(str1, str2);`
> $str1$、$str2$ 为需要拼接的字符串。

> `strcat()` 将把 $str2$ 连接到 $str1$ 后面，并删除原来 $str1$ 最后的结束标志 `'\0'` 。
> 这意味着，$str1$ 必须足够长，要能够同时容纳 $str1$ 和 $str2$，否则会越界（超出范围）。
> `strcat()` 的返回值为 $str1$ 的地址。

```
#include <bits/stdc++.h>
using namespace std;

int main() {
    char a[] = "abcd";
    char b[] = "efg";
    strcat(a, b);
    cout << a << endl;
    return 0;
}
```

> 运行结果

```
abcdefg
```

三. 经典例题

1. 数字和

> 输入一个整数 n，求各位上的数字和。

> 输入格式
> 一行一个整数 n。

> 输出格式
> 一行一个整数，表示整数 n 的各位数字之和。

样例输入	样例输出
231344	17

> 数据范围
> n 最多 200 位

2. 进制转换

> 将一个任意n进制数转换为十进制

> 输入格式
> 第一行一个正整数 n
> 第二行一个整数 x

> 输出格式
> 一行一个数字，表示转换得到的十进制数，保证答案在 `int` 范围内

样例输入	样例输出
2 100110	38

> 数据范围
> $1 < n < 10$

3. $honoka$的键盘

> $honoka$ 有一个只有两个键的键盘。
> 一天，她打出了一个只有这两个字符的字符串。
> 当这个字符串里含有 `"VK"` 这个字符串的时候，honoka 就特别喜欢这个字符串。
> 所以，她想改变至多一个字符（或者不做任何改变）来最大化这个字符串内"VK"出现的次数。

> 给出原来的字符串，请计算她最多能使这个字符串内出现多少次 "VK" 。
> (只有当 'V' 和 'K'; 正好相邻时，我们认为出现了 "VK" 。)

> 输入格式
> 第一行给出一个数字 n ,代表字符串的长度。第二行给出一个字符串 s。

> 输出格式
> 第一行输出一个整数代表所求答案。

样例输入	样例输出
2 VK	1
2 VV	1
1 V	0
20 VKKKKKKKKKVVVVVVVVVK	3
4 KVKV	1

> 数据范围
> 对于 100% 的数据，$1 <= n <= 100$ 。

四. 提高巩固

1. 标题统计($NOIP2018$普及组第一题)

> 凯凯刚写了一篇美妙的作文，请问这篇作文的标题中有多少个字符?
> 注意：标题中可能包含大、小写英文字母、数字字符、空格和换行符。
> 统计标题字 符数时，空格和换行符不计算在内。

> 输入格式
> 输出只有一行，一个字符串 s

> 输出格式
> 输出只有一行，包含一个整数，即作文标题的字符数（不含空格和换行符）。

样例输入	样例输出

样例输入	样例输出
234	3
Ca 45	4

> 数据范围
> 规定 $|s|$ 表示字符串 s 的长度（即字符串中的字符和空格数）。
> 对于 40% 的数据，$1 \leq |s| \leq 5$，保证输入为数字字符及行末换行符。
> 对于 80% 的数据，$1 \leq |s| \leq 5$，输入只可能包含大、小写英文字母、数字字符及行末换行符。
> 对于 100%的数据，$1 \leq |s| \leq 5$，输入可能包含大、小写英文字母、数字字符、空格和行末换行符。

> **样例说明**
> 【输入输出样例 1 说明】
> 标题中共有 3 个字符，这 3 个字符都是数字字符。
> 【输入输出样例 2 说明】
> 标题中共有 5 个字符，包括 1 个大写英文字母， 1 个小写英文字母和 2 个数字字符， 还有 1 个空格。由于空格不计入结果中，故标题的有效字符数为 4 个。

2. ISBN号码($NOIP2008$普及组第一题)

> 每一本正式出版的图书都有一个 ISBN 号码与之对应
> ISBN 码包括9位数字、1位识别码和3位分隔符，其规定格式如 "x-xxx-xxxxx-x"
> 其中符号“-”是分隔符（键盘上的减号），最后一位是识别码
> 例如 0-670-82162-4 就是一个标准的 ISBN 码。
> ISBN 码的首位数字表示书籍的出 版语言，例如 0 代表英语；
> 第一个分隔符 "-" 之后的三位数字代表出版社，例如 670 代表维京出版社；
> 第二个分隔之后的五位数字代表该书在出版社的编号；
> 最后一位为识别码。
> 识别码的计算方法如下：
> 首位数字乘以1加上次位数字乘以2……以此类推，用所得的结果 mod 11 ，所得的余数即为识别码
> 如果余数为10，则识别码为大写字母 X 。
> 例如 ISBN 号码 0-670-82162-4 中的识别码4是这样得到的：
> 对 067082162 这9个数字，从左至右，分别乘以 1，2，…，9 ，再求和，即 0×1+6×2+……+2×9=158
> 然后取$158 mod 11$的结果4作为识别码。
> 你的任务是编写程序判断输入的 ISBN 号码中识别码是否正确，
> 如果正确，则仅输出 “Right” ；如果错误，则输出你认 为是正确的 ISBN

> 输入格式
> 输入只有一行，是一个字符序列，表示一本书的 ISBN 号码（保证输入符合 ISBN 号码的格式要求）。

> 输出格式
> 输出一行，假如输入的 ISBN 号码的识别码正确，那么输出 “Right” ，否则，按照规定的格式，输出正确的 ISBN 号码（包括分 隔符 "-" ）。

样例输入	样例输出
0-670-82162-4	Right
0-670-82162-0	0-670-82162-4

3. 拼数($NOIP1998$普及组)

> 设有n个正整数，将它们联接成一排，组成一个最大的多位整数。
> 例如：$n=3$时，3个整数$13,312,343$联接成的最大整数为：34331213
> 又如：$n=4$时，4个整数$7,13,4,246$联接成的最大整数为：7424613

> 输入格式
> 第一行，一个正整数n。
> 第二行，n个正整数。

> 输出格式
> 一个正整数，表示最大的整数

样例输入	样例输出
3 13 312 343	34331213

> 数据范围
> $n \leq 20$