

# 《软件工程导论》课程报告

报 告 题 目: Lesson90:#1 Longest Password

姓 名: 石广钊

学 号: 161180111

指 导 老 师: 方晖

## 目录

1 项目简介 .....	1
1.1 背景 .....	1
1.2 目的 .....	1
2 范围 .....	1
3 需求说明 .....	1
3.1 需求定义 .....	1
3.2 需求规格说明 .....	2
3.3 设计标准 .....	2
3.4 成果交付 .....	3
4 实施方法 .....	3
4.1 方法研究 .....	3
4.2 概念设计 .....	3
4.3 时间表 .....	3
5 设计描述及原型展示 .....	3
5.1 原型测试 .....	6
6 结论 .....	7
7 附录 .....	8
7.1 项目要求原文 .....	8
7.1.1 Lesson 90: Tasks from Indeed Prime 2015 challenge–Longest Password ..	8
7.2 项目代码 .....	9

## 1 项目简介

### 1.1 背景

本项目为南京大学电子学院《软件工程导学》课程作业。项目来源于为 Codility.com 网站 Lesson 90: Tasks from Indeed Prime 2015 challenge 中的任务 1。项目要求写一个程序实现题目中要求的功能，编程语言不限，且本题对时间复杂度没有要求，只需保证正确性即可。

### 1.2 目的

项目要求设计一个函数，对于输入字符串 S，返回其中符合密码要求的最长 word 的长度。其中不同 word 之间以空格分割，word 可作为密码格式需满足三个条件，即：

- (1) 仅包含字母或数字字符 (AZ, az, 09)。
- (2) 需要包含偶数个字母 (0,2,4, ..)。
- (3) 需要包含奇数个数字 (1,3,5, ..)。

实现该功能的编程语言不限，字符串 S 长度 N 长度在 [1..200] 之间。

## 2 范围

本项目中，需要实现一个函数：

```
1 int solution(char *S)
```

编写所得程序可以先通过自己编写的程序进行测试，再通过 codility.com 网站进行测试。

## 3 需求说明

### 3.1 需求定义

- 1、程序能够根据空格位置分割输入的字符串。
- 2、程序能够判断字符串结束的位置。

- 3、 程序能够检测一个 word 是否满足作为密码的三个条件。
- 4、 程序可以返回最长密码的长度，如不存在满足要求的 word，则返回-1。
- 5、 保证返回长度的正确性。

### 3.2 需求规格说明

- 1、 程序中使用切片函数将输入字符串分为多个字符串，或者直接遍字符串逐个处理字符。
- 2、 若遇到空字符 NULL 表示字符串即结束，停止字符串遍历。
- 3、 通过将字符的 ASCII 值与相应范围的数字比较判定字符是否符合要求，统计个字符数量即可判断是否符合要求。
- 4、 对不同测试集进行测试来检测结果的正确性。

### 3.3 设计标准

评分系统:

- 10-9 = 非常重要（必须完成）
- 8-7 = 比较重要（能完成最好）
- 6-4 = 重要而不必要（完成最好但是没有必要）
- 3-0 = 基本不重要

- 1、 准确性（10）-输出结果是否正确。
- 2、 时间复杂度（6）-最坏情况下执行程序所需时间。
- 3、 空间复杂度（6）-程序运行时所占存储空间大小。
- 4、 程序可读性（8）-别人能否读懂此程序。
- 5、 可维护性（7）-修改是否容易。

### 3.4 成果交付

- 1、 课程报告。
- 2、 项目程序代码。
- 3、 自己的以及网站的测试结果。

## 4 实施方法

### 4.1 方法研究

本项目设计不限值设计语言，由于本项目实现功能比较简单，使用不同程序语言编写难度差距不大。

项目成员对 C/C++ 语言和 Python 较为熟悉，因此可用其中一种实现。C/C++ 具有较高的执行效率，而 Python 则拥有丰富的函数可，无需自己实现基础的功能。

考虑到所需实现功能不多，Python 并不会产生较大优势，可选择使用 C 语言实现。

### 4.2 概念设计

使用一个 for 循环遍历字符串 S，并统计字母数量 CharCount 和数字数量 NumberCount，如果在遇到空格前遇到非法字符则计数清零、停止计数直到空格处，否则计数到空格处后根据计数值判断是否为合法密码，是则修改最大长度 MaxLength。

遍历结束后返回 MaxLength，若 MaxLength 等于 0 则返回-1。

### 4.3 时间表

## 5 设计描述及原型展示

最终设计程序代码如下：

```
1 #include <stdbool.h>
2 int solution(char *S) {
3     int MaxLength = 0;
4     int CharCount = 0;
```

```
5  int NumberCount = 0;
6  bool PassWord = true;
7  for(int i = 0; S[i] != 0; i++) {
8      // 0--48, 9--57; A--66, Z--90; a--97, z--122, 空格--32
9      // 未遇到空格且此字符串合法
10     if(PassWord && S[i] != ' ') {
11         if(S[i] <= '9' && S[i] >= '0') {
12             NumberCount++;
13         }
14         else if((S[i] >= 'a' && S[i] <= 'z') \
15             || (S[i] >= 'A' && S[i] <= 'Z')) {
16             CharCount++;
17         }
18         else {
19             PassWord = false;
20             NumberCount = 0;
21             CharCount = 0;
22         }
23     }
24     else {
25         // 合法字符串遇到空格
26         if(PassWord) {
27             if(MaxLength < NumberCount + CharCount \
28                 && NumberCount % 2 == 1 \
29                 && CharCount % 2 == 0)
30                 MaxLength = NumberCount + CharCount;
31             NumberCount = 0;
32             CharCount = 0;
33             PassWord = true;
34         }
35         // 不合法字符串空格
36         else if(S[i] == ' ') {
37             PassWord = true;
38         }
39         // 不合法字符串非空格
40     }
41 }
```

```

42 // 最后一个字符串是否合格
43 if(PassWord) {
44     if(MaxLength < NumberCount + CharCount \
45         && NumberCount % 2 == 1 \
46         && CharCount % 2 == 0)
47         MaxLength = NumberCount + CharCount;
48 }
49 return MaxLength == 0? -1 : MaxLength;
50 }

```

第一行 `#include <stdbool.h>` 用于包含 `stdbool.h` 头文件起原因是 linux 环境下 `stdio.h` 中没有包含该文件，因此直接使用 `bool` 类型会报错。

之后的几行定义函数以及定义和初始化变量。

for 循环 `for(int i = 0; S[i] != 0; i++)` 遍历字符串 `S`，当遇到空字符是遍历结束。

```

1 // 未遇到空格且此字符串合法
2 if(PassWord && S[i] != ' ') {
3     if(S[i] <= '9' && S[i] >= '0') {
4         NumberCount++;
5     }
6     else if((S[i] >= 'a' && S[i] <= 'z') \
7         || (S[i] >= 'A' && S[i] <= 'Z')) {
8         CharCount++;
9     }
10    else {
11        PassWord = false;
12        NumberCount = 0;
13        CharCount = 0;
14    }
15 }

```

当遇到的不是空格且当前字符串状态为合法时，如果为数字则 `NumberCount` 加 1，若为字母则 `CharCount` 加 1，否则表示遇到了非法字符，改变状态为非法并清零计数器。

```

1 else {
2     // 合法字符串遇到空格
3     if(PassWord) {
4         if(MaxLength < NumberCount + CharCount \

```

```

5         && NumberCount % 2 == 1 \
6         && CharCount % 2 == 0)
7         MaxLength = NumberCount + CharCount;
8         NumberCount = 0;
9         CharCount = 0;
10        PassWord = true;
11    }
12    // 不合法字符串空格
13    else if(S[i] == ' ') {
14        PassWord = true;
15    }
16    // 不合法字符串非空格
17    }

```

如不满足之前的条件，则存在三种情况：合法字符串遇到了空格、不合法字符串且未遇到空格或不合法字符串遇到空格。对第一种情况判定密码是否合法并与最大值比较，若合法且大于最大值，则赋值给最大值，然后还原计数状态；第二种情况不做处理；第三种情况需还原计数状态。

之后的几行测试最后一个 word，然后返回结果。

## 5.1 原型测试

编写测试主函数如下：

```

1 int main(void) {
2     char str[] = "test aa1 a0A pass007 ?xy1";
3     printf("%s\n", str);
4     printf("%d\n", solution(str));
5     return 0;
6 }

```

改变字符串的为以下的不同值测试不同情况下的结果。

```

1     str1 = "" // 测试空字符串
2     str2 = "as2 235 f ewf" // 包含多个连续空格的字符串
3     str3 = "hi 34% dqew11 *gtDGF Df34F1A d" // 普通字符串
4     str4 = "ss 33 f34#^ EW" // 无合法密码的字符串

```



经测试结果准确无误。

使用 codility.com 网站进行测试，得到结果如下：

expand all		Example tests	
▶	example example test	✓	OK
expand all		Correctness tests	
▶	simple short and simple tests	✓	OK
▶	one_character one character words	✓	OK
▶	one_word tests that contains one word only	✓	OK
▶	even_letters all words have even number of letters	✓	OK
▶	odd_digits all words have odd number of digits	✓	OK
▶	odd_length it's sufficient to test validity of characters and if length of word is odd	✓	OK
▶	all_alphanumeric all words contain only alphanumeric characters	✓	OK
▶	extra_characters valid passwords joined with some invalid characters	✓	OK
▶	large_random random tests	✓	OK
▶	maximum biggest possible tests with mixed types of words	✓	OK

对不同情况下测试结果均准确，满足设计要求。

## 6 结论

从测试结果可以看到，实验程序设计满足要求。程序能够保证输出结果的准确性；程序时间复杂度为线性时间，即使在输入较复杂的情况下仍然能够较快输出结果；程序遍历字符

串而非使用切片操作复制为多个字符串，保证了较低的空间复杂度；程序可读性和可维护性均较强。

设计结果满足设计要求。

## 7 附录

### 7.1 项目要求原文

#### 7.1.1 Lesson 90: Tasks from Indeed Prime 2015 challenge—Longest Password

You would like to set a password for a bank account. However, there are three restrictions on the format of the password:

it has to contain only alphanumerical characters (a–z, A–Z, 0–9);

there should be an even number of letters;

there should be an odd number of digits.

You are given a string  $S$  consisting of  $N$  characters. String  $S$  can be divided into words by splitting it at, and removing, the spaces. The goal is to choose the longest word that is a valid password. You can assume that if there are  $K$  spaces in string  $S$  then there are exactly  $K + 1$  words.

For example, given "test 5 a0A pass007 ?xy1", there are five words and three of them are valid passwords: "5", "a0A" and "pass007". Thus the longest password is "pass007" and its length is 7. Note that neither "test" nor "?xy1" is a valid password, because "?" is not an alphanumerical character and "test" contains an even number of digits (zero).

Write a function:

```
def solution(S)
```

that, given a non-empty string  $S$  consisting of  $N$  characters, returns the length of the longest word from the string that is a valid password. If there is no such word, your function should return  $-1$ .

For example, given  $S = \text{"test 5 a0A pass007 ?xy1"}$ , your function should return 7, as explained above.

Assume that:

$N$  is an integer within the range  $[1..200]$ ;

string S consists only of printable ASCII characters and spaces.

In your solution, focus on correctness. The performance of your solution will not be the focus of the assessment.

## 7.2 项目代码

```
1  /*****
2      > File Name: longestpassword.c
3      > Author: ShiGuangzhao
4      > Mail: Guangzhao_Shi@163.com
5      > Created Time: 2019年12月02日 星期一 21时34分10秒
6      *****/
7
8  #include <stdbool.h>
9  #include <stdio.h>
10
11 int solution(char *S) {
12     int MaxLength = 0;
13     int CharCount = 0;
14     int NumberCount = 0;
15     bool PassWord = true;
16     for(int i = 0; S[i] != 0; i++) {
17         // 0--48, 9--57; A--66, Z--90; a--97, z--122, 空格--32
18         // 遇到空格或此字符串不合法
19         if(PassWord && S[i] != ' ') {
20             if(S[i] <= '9' && S[i] >= '0') {
21                 NumberCount++;
22             }
23             else if((S[i] >= 'a' && S[i] <= 'z') \
24                 || (S[i] >= 'A' && S[i] <= 'Z')) {
25                 CharCount++;
26             }
27             else {
28                 PassWord = false;
29                 NumberCount = 0;
30                 CharCount = 0;
```

```
31     }
32 }
33 else {
34     // 合法字符串遇到空格
35     if(PassWord) {
36         if(MaxLength < NumberCount + CharCount \
37             && NumberCount % 2 == 1 \
38             && CharCount % 2 == 0)
39             MaxLength = NumberCount + CharCount;
40         NumberCount = 0;
41         CharCount = 0;
42         PassWord = true;
43     }
44     // 不合法字符串空格
45     else if(S[i] == ' ') {
46         PassWord = true;
47     }
48     // 不合法字符串非空格
49 }
50 }
51
52 // 最后一个字符串是否合格
53 if(PassWord) {
54     if(MaxLength < NumberCount + CharCount \
55         && NumberCount % 2 == 1 \
56         && CharCount % 2 == 0)
57         MaxLength = NumberCount + CharCount;
58 }
59 return MaxLength == 0? -1 : MaxLength;
60 }
61
62 int main(void) {
63     char str[] = "test aa1 a0A pass007 ?xy1";
64     printf("%s\n", str);
65     printf("%d\n", solution(str));
66     return 0;
67 }
```