LAB5 实验报告

1. Program Design

这次实验中，我主要采用 trap 操作里的 PUTS,IN 来实现从键盘的读取还有输出其中，键盘输入字符后，我使用了栈来储存，以实现输入字符串的功能。

然后在密码对比的过程中，通过每一位每一位的对比完成

2. Test Evidence

1. 提示输入“W”

```
Welcome to the bank system! Type 'W' to withdraw some fund.

Input a character>
```

->输入的不是 W,会继续重复，直到输入 W

```
Welcome to the bank system! Type 'W' to withdraw some fund.

Input a character> R
Welcome to the bank system! Type 'W' to withdraw some fund.

Input a character>
```

2. 输入 W,开始密码输入

```
Welcome to the bank system! Type 'W' to withdraw some fund.

Input a character> R
Welcome to the bank system! Type 'W' to withdraw some fund.

Input a character> W
Please input your password:

Input a character>
```

3. 如果输入的密码错误，超过三次则回到最开始的提示输入：

```
Fail!
Please input your password:

Input a character> 4

Input a character> Y
Fail!
Please input your password:

Input a character> 5

Input a character> Y
Fail!
Welcome to the bank system! Type 'W' to withdraw some fund.
```

4. 输入密码正确：

```
Input a character> W
Please input your password:

Input a character> P

Input a character> B

Input a character> 2

Input a character> 2

Input a character> 0
```

```
Input a character> 6

Input a character> 1

Input a character> 2

Input a character> 5

Input a character> 9

Input a character> Y
Success!
```

三.Discussion Questions

1. 我没有使用函数，因为我的程序中没有出现大块重复的代码，有重复的操作也只是一行指令就可以完成
2. 我没有使用递归，因为不需要。实际上我使用了栈，但是没有用到递归
3. 给好的提示，我使用了.STRINGZ 伪指令，将其存在给定的内存里。
4. 我的程序设计，是将密码存在栈里，而这个栈是向上生长的，不太可能出现越界情况，所以一般输入几千长度的密码都是可以存的下的
5. 这里我使用的是栈，实际上应该用队列会比较合适。所以我在密码比对的时候，只能投机取巧，采用倒着比对的办法，从后向前比对。