作业二：Shell 脚本的测试结果

吴泓鹰
数学与应用数学 (强基计划)　　3210101890

2022 年 6 月 28 日

1　Shell 脚本的源码与测试

在源码的编写上，我以教材[1]上第 40 页的

`Try It Out Case III: Executing Multiple Statements`

为模板，进行了简单的本地化改动得到了本报告所展示的 Shell 脚本。

1.1　源码展示

```
#!/bin/sh
yes_or_no() {
        echo "你的名字是 $* 吗 ?"
        while true #这里的"true"可以用": "代替
        do
        echo -n "请输入 yes 或者 no: "
        read x
        case "$x" in
        y | yes | Y | Yes) return 0;;
        [Nn]* ) return 1;;
        * ) echo "请回答 yes 或者 no! "
        esac
```

[1] N. Matthew and R. Stones, Beginning Linux Programming, 4th Edition, Wiley Publishing, Inc., Indianapolis, 2008.

```
13          done
14  }
15  echo "这个程序的用户有 $*"
16  if yes_or_no "$1"
17  then
18  echo "你好呀 $1, 真是个好名字 ~"
19  else
20  echo "打扰了 ~"
21  fi
22  exit 0
```

1.2 测试结果

(a) 输入:

```
./shell_hw why yhw
Y
```

输出:

```
why@why-virtual-machine:~/mathsoft/homework2$ ./shell_hw why yhw
这个程序的用户有 why yhw
你的名字是 why 吗 ?
请输入 yes 或者 no: Y
你好呀 why, 真是个好名字 ~
```

(b) 输入:

```
./shell_hw why
Noooo!
```

输出:

```
why@why-virtual-machine:~/mathsoft/homework2$ ./shell_hw why
这个程序的用户有 why
你的名字是 why 吗 ?
请输入 yes 或者 no: Noooo!
打扰了 ~
```

(c) 输入:

```
/shell_hw why yhw
?
...
爬
nanodesu
```

输出:

```
why@why-virtual-machine:~/mathsoft/homework2$ ./shell_hw why yhw
这个程序的用户有 why yhw
你的名字是 why 吗 ?
请输入 yes 或者 no: ?
请回答 yes 或者 no!
请输入 yes 或者 no: ...
请回答 yes 或者 no!
请输入 yes 或者 no: 爬
请回答 yes 或者 no!
请输入 yes 或者 no: nanodesu
打扰了~
```

2 Shell 脚本的解析与学习

在这一部分，我将对上述脚本的源码与输出结果进行解析，从而展示我在今天的课堂与课外中所学习到的有关 Shell 脚本的内容。

2.1 源码与输出结果的解析

我们从将对源码上往下进行解析:

- 在源码中的第一行与第四行都使用到了`#`, 但实际上他们的功能并不相同; 第一行中`#!/bin/sh`是一种特殊的注释，`#!`告诉了系统应该执行`/bin/sh`这个 Shell 程序; 而第四行中的`#`仅仅是告诉系统其后的内容属于注释，不需要执行。

- 第二行中的`yes_or_no(){...}`是一个函数，它可以被后面的程序所调用执行`{...}`中的内容。
- 源码中的`echo`可以用于输出其后的变量内容（需要在变量名前加上`$`）或者字符串，在`echo`后加上`-n`可以使输出后不自动换行。
- while 循环语句

```
while 循环条件 do
循环内容
done
```

正如上述代码展示的一样，在满足循环条件的情况下重复的执行循环内容; 在源码中，条件`true`即验证返回值是否为真，其中的`return 0`表明返回值为假，`return 1`表明返回值为真。

- 源码中的`read`可以用于读取用户所输入的一个字符串。
- case 条件语句

```
case 变量 in
条件 ) 执行内容;;
条件 ) 执行内容;;
...
esac
```

正如上述代码展示的一样，从上到下依次对变量验证条件，在变量满足条件的时候执行其后的内容，注意：这里的两个双引号`;;`是必要的。

- if 条件语句

```
if 条件
then
内容1
else
内容2
fi
```

正如上述代码展示的一样，验证条件是否成立，是则输出内容 1, 否则输出内容 2。

- 源码中的`exit`将确保脚本返回一个有意义的退出码，一般而言退出码 0 表示程序成功运行，其他退出码则会有不同的含义。

对测试结果的分析：

- 注意到我们在执行文件`./shell_hw`后加入了字符`why`与`yhw`，这将在脚本中作为参数变量出现，即源码中出现的`$*,$1`，它们分别表示脚本程序的所有参数与第一个参数，因此在测试结果 (a) 中分别出现了`why yhw`于`why`。
- 在测试结果 (b) 与 (c) 中我们发现输入`Noooo!,nanodesu`也能输出，这是因为`[]*`可以将只要包含方括号中内容的输入识别为条件成立。