实验报告 1

树莓派和 Python 编程语言初步

范皓年 1900012739　信息科学技术学院

2020 年 10 月 16 日

1　实验目的

1. 认识树莓派开发板及其供电、连接等
2. 利用树莓派掌握 Linux 系统基本操作
3. 了解 Python 语言，利用树莓派终端和 Python Thonny IDE 进行树莓派上的 python 开发。

2　实验原理

2.1　树莓派微型电脑主板

树莓派是一种麻雀虽小五脏俱全的微型电脑，利用 SD 卡作为内存硬盘，有一定数量的 USB 接口与一个以太网接口，可连键鼠、有网络功能，同时可以连接显示器进行显示。本实验中树莓派作为开发的主机，没有涉及较多复杂的网络连接与硬件接口的相关问题。故不再赘述。

2.2　Linux 系统基本操作

树莓派中运行的，是基于 Debian 的 Linux 发行版即 Raspbian。基本使用方法与 Linux 系统相同，只是针对硬件做了特定的优化。

本实验主要涉及 Linux 系统的**文件操作**、**编辑器使用**与 **python 运行**。

1. 本实验需要的文件操作：

- 更改当前目录：cd dir
- 创建目录：mkdir dir
- 删除目录：rmdir dir
- 复制/移动文件：cp/mv file1 file2

2. vim 使用

- 编辑文件 vim filename
- 保存退出:wq

3. python 运行

- python filename

2.3 Python 语言初步

Python 与 C 语言最直观的不同就是利用强制的缩进而非大括号来表示代码逻辑。

本实验中所需的 Python 基本数据类型包括整型、浮点型、字符串（没有字符型）、布尔值。在做数据类型转换的时候需要使用到 int 函数和 float 函数。

输入函数为 input 函数，参数为输入提示语。返回值为输入值。输出函数为 print 函数，本实验中使用字符串输出，参数可为一个或多个字符串，多个时用逗号隔开，输出时则将多个字符串以空格隔开、顺序输出。

```
name = input('please enter your name: ')
print('hello,', name)
```

条件判断语句与 C 语言的不同在于不需要加括号，但需要使用冒号；另外 else if 需要用 elif 代替。

```
age = 3
if age >= 18:
    print('adult')
elif age >= 6 :
    print('teenager')
else:
    print('kid')
```

循环语句为 for in 循环和 while 循环，典型的使用方法如下：

```
for i in [0,1, 2, 3, 4, 5, 6, 7, 8, 9]:
    print(i)
for i in range(0,10,2): # 从0 循环至9, 步进值为2
    print(i)
n = 1
while n <= 100:
    print(n)
    n = n + 1
```

其中 continue 和 break 等控制语句的用法与 C 语言中相同。

3 实验内容

3.1 示例代码验证

1. 将已经连接好外设的树莓派插电，启动之后打开终端。
2. 进入\home\pi 之后利用 mkdir 命令建立自己的文件夹，
3. 利用 vim 编辑文件 example.py，利用命令行命令 python3 example.py 运行，或者 Thonny IDE 直接编辑并运行。

代码如下：

```
    age = input('Enter your age:\n')
    age = int(age)
    if age >= 18:
        print('You are an adult')
    elif age >= 6:
        print('You are a teenager')
    else:
        print("You are a kid")
```

3.2 冒泡排序

利用 random 模块生成随机列表。共操作 len 次（实际有效的只有 len-1 次，因为第 len 次时最多仅可能有一个元素没有就位）。操作每次将最小的单元挪到列表最后，称为冒泡，依次进行 len-1 次，则所有元素就位。代码如下：

```
    import random as r
    data = []
    for i in range(0,10):
        data.append(r.randint(0, 99))

    l = len(data)

    for i in range(l):
        print(data[i],end = ' ')
    print()
    for i in range(l):
        for j in range(l-i-1):
            if data[j+1] > data[j]:
                data[j], data[j+1] = data[j+1], data[j]

    for i in range(l):
        print(data[i],end = ' ')
```

3.3 简易计算器 (旧版)

四个数选择计算模式，再输入两个操作数，如果输入不满足自动报错。代码如下：

```
    def calc():

    print(
    '''
    1: +
    2: -
    3: *
    4: /
    0: Exit
    Enter choice:
    '''
        )

    choice = int(input())
    if choice == 0:
        return
    op1 = float(input('Enter first operand: '))
    op2 = float(input('Enter second operand: '))

    ans = 0
    ope = 0
    if choice == 1:
        ans = op1 + op2
        ope = '+'
    elif choice == 2:
        ans = op1 - op2
        ope = '-'
    elif choice == 3:
        ans = op1 * op2
        ope = '*'
    elif choice == 4:
        ans = op1 / op2
        ope = '/'
    print(op1, ope, op2, "=", ans)

def calc2():
    print(
    '''
```

```
    1: +
    2: -
    3: *
    4: /
    0: Exit
    '''
        )
    while True:
        choice = input('Enter choice:')
        if choice[0].isdigit():
            choice = int(choice)
            if 0 <= choice <= 4:
                break
            else:
                print('choice num wrong')
        else:
            print('choice type wrong')
    if choice == 0:
        return
    while True:
        op1 = input('Enter first operand: ')
        if op1.isdigit():
            op2 = input('Enter second operand: ')
        else:
            print('Operand num1 wrong')
            continue
        if op2.isdigit():
            op1 = float(op1)
            op2 = float(op2)
            break
        else:
            print('operand num2 wrong')

    ans = 0
    ope = 0
    if choice == 1:
        ans = op1 + op2
        ope = '+'
    elif choice == 2:
        ans = op1 - op2
        ope = '-'
```

```
    elif choice == 3:
        ans = op1 * op2
        ope = '*'
    elif choice == 4:
        ans = op1 / op2
        ope = '/'
    print(op1, ope, op2, "=", ans)

if __name__ == "__main__":
    # calc()
    calc2()
```

3.4 猜拳游戏

猜拳本来是一个高度随机的游戏，前后局关联性并不大，因而很难找到非常有效的博弈算法。以下即为一个完全随机的版本，不做任何博弈优化，程序除利用随机数给出选择以外只做胜负和平局判断。

程序中使用 isdigit 函数进行了优化，防止除 012 以外的输入进入程序造成程序崩溃，这也就完成了思考题 1。

```
import random
def srp():
    user = 0
    while True:
        user = input("Please input: Scissors(0), \
                    Rock(1), Paper(2), Quit(3)\n")
        if len(user) > 1:
            print('too long')
            continue
        elif user.isdigit():
            user = int(user)
            if user > 3:
                print('num out of range')
                continue
        else:
            print('input not num')
            continue
        if user == 3:
            break;
        computer = random.randint(0, 2)
        if (user == 0 and computer == 2) or (user == 1 and computer == 0) \
            or (user == 2 and computer == 1):
            print('YOU WIN!')
```

```
        elif  computer == user:
            print("IT'S A TIE!")
        else:
            print('YOU LOSE!')
if __name__ == '__main__':
    srp()
```

关于思考题 2，对于计算机的博弈，二者都用随机数表生成结果，不考虑任何博弈策略，几乎不能找到优化的想法。但对于人机对战，通过统计人类的潜意识的选择与博弈，出拳会有一定的偏好，所以通过统计，胜率可以得到一定的提升。经过测试，当人手输入倾向明显的时候（指全部出同一种拳），统计法的胜率几乎可以达到 90% 甚至以上；当相对均匀的输入时，考虑到博弈等综合因素，人仍然会有下意识的选择，通过统计，（观察输出最后一行中各数据占比）差异仍然明显：

```
Please input: Scissors(0), Rock(1), Paper(2), Quit(3)
0
YOU WIN!

151 168 160
```

图 1: 随机算法

```
Please input: Scissors(0), Rock(1), Paper(2), Quit(3)
2
YOU WIN!

143 166 120 429
140 173 116
```

图 2: 统计算法

通过统计算法，机器战胜人类玩家的概率得到了一定的提升。

统计算法的代码如下：

```
import random

def srp2():
    user = 0
    count = [0, 0, 0]
    wtimes = 0
    ltimes = 0
    ttimes = 0
    while True:
        user = input("Please input: Scissors(0), \
        Rock(1), Paper(2), Quit(3)\n")
        if len(user) > 1:
            print('too long')
```

```
14              continue
15          elif user.isdigit():
16              user = int(user)
17              if user > 3:
18                  print('num out of range')
19                  continue
20          else:
21              print('input not num')
22              continue
23          if user == 3:
24              break
25
26          count[user] = count[user] + 1
27          sum = count[0] + count[1] + count[2]
28          computer = random.randint(0, sum)
29          if computer <= count[0]:
30              computer = 1
31          elif computer <= count[0] + count[1]:
32              computer = 2
33          elif computer <= sum:
34              computer = 0
35          if (user == 0 and computer == 2) or (user == 1 and computer == 0) \
36                  or (user == 2 and computer == 1):
37              print('YOU WIN!')
38              wtimes = wtimes + 1
39          elif computer == user:
40              print("IT'S A TIE!")
41              ttimes = ttimes + 1
42          else:
43              print('YOU LOSE!')
44              ltimes = ltimes + 1
45          print("_____")
46          print(count[0], count[1], count[2], sum)
47          print(wtimes, ltimes, ttimes)
48          print("_____")
49
50  if __name__ == '__main__':
51  srp2()
```