实例 1：用户注册登录

用户管理模块是各种软件中最基本的模块之一，该模块的基本功能是用户注册与登录。虽然每个软件的界面样式有所不同，但注册与登录业务的主要业务逻辑相差无几。这两个业务的流程分别如图 1 和图 2 所示。

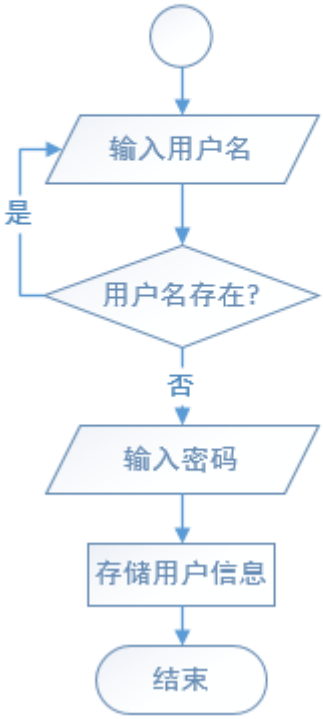

图 1 用户注册业务流程

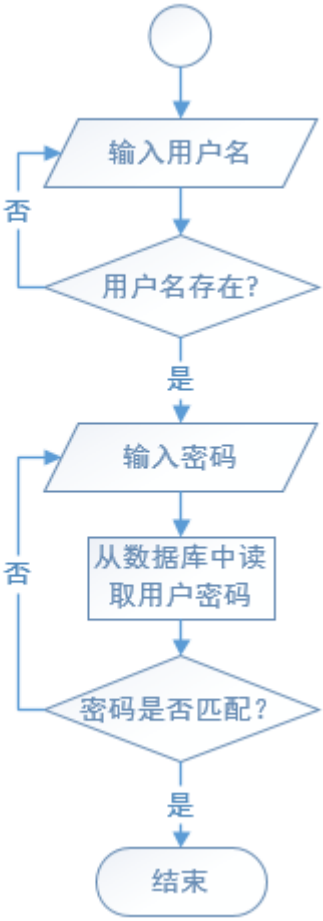

图 2 用户登录业务流程

本实例要求结合数据库，按照以上业务流程实现用户注册登录功能。

实例目标

- 掌握 Python 中数据库的基本使用

实例描述

实例要求使用数据库，因此程序中涉及的数据需存储到数据库中。实例涉及的数据为用户账户信息，这里考虑创建数据库 python，在其中创建包含字段 uname 和 upwd 的表 py_users 存储用户账户信息。

用户注册登录模块主要包含注册和登录两项功能，进入注册登录界面后，用户可选择注册或登录功能：

若选择注册功能，用户需输入用户名与密码，需注意用户名唯一，若输入的用户名与数据库中已有用户名相同，应给出相应提示，并重新接收用户输入；若用户名与密码符合要求，新的用户信息应被存储到数据库中。

结合流程图设计注册和登录的实现流程及函数功能，具体如下。

1.注册函数——registered()

注册函数的流程及功能应为：

(1) 接收用户输入的用户名，判断用户名是否存在，若存在给出提示，继续等待用户输入用户名；

(2) 若用户名不存在，接收用户输入的密码，将用户名和密码一同插入数据库中的用户表，提示“注册成功”。

2.登录函数——login()

登录函数的流程及功能应为：

(1) 接收用户输入的用户名，判断用户名是否存在，若不存在提示“用户名不存在”，等待用户重新输入用户名；

(2) 若用户名存在，接收用户输入的密码，根据用户名从数据库中查询相应密码，并与用户输入的密码进行匹配；

(3) 若密码匹配成功提示“登录成功”，匹配失败提示“密码有误”，等待用户重新输入密码，重复过程(2)(3)。

代码实现

```
"""
1.创建表
create table py_users(
id int unsigned auto_increment not null primary key,
uname varchar(20) not null,
upwd char(40) not null,
is_delete bit not null default 0
);
"""
import pymysql
# 用户注册
def mysql_registered(conn,cur):
 # 1.获取有效用户名
 sql_select = 'select * from py_users where uname=%s'
 uname = input('用户名：')
 while True:
```

```
        # 执行 sql 语句
        params = [uname]
        result = cur.execute(sql_select,params)
        if result == 1:
            print('用户名已存在，请重新输入')
            uname = input('用户名：')
        else:
            break
    # 3.获取密码
    upwd = input('密  码：')
    # 4.插入数据库
    sql_insert = 'insert into py_users(uname,upwd) values(%s,%s)'
    params = [uname,upwd]
    result = cur.execute(sql_insert,params)
    conn.commit()
    # 5.插入结果判断
    if result == 1:
        print('注册成功')
    else:
        print('注册失败')
    # 6.关闭连接
    cur.close()
# 用户登录
def mysql_login(conn,cur):
    result = 0
    while not result:
        uname = input('用户名：')
        sql = 'select upwd from py_users where uname=%s'
        params = [uname]
        result = cur.execute(sql,params)
        if result==1:           # 如果找到了用户
            mysql_upwd = cur.fetchone()
            while True:
                upwd = input('密码：')
                if upwd == mysql_upwd[0]:
                    print('登录成功')
                    break
                else:
                    print('密码错误，请重新输入')
```

```
        else:
            print('用户名不存在')
    # 关闭连接
    cur.close()
# 打印菜单
def menu():
    print("----功能选择----")
    print("1.注册")
    print("2.登录")
# 主函数
def main():
    # 1.打印菜单
    menu()
    # 2.连接数据库
    conn = pymysql.connect(host='localhost',port=3306,database='python',
        user='root',password='chan1121',charset='utf8')
    cur=conn.cursor()
    # 3.功能选择
    sel = input("注册（1）or 登录（2）？")
    # 选择功能
    if sel == '1':
        mysql_registered(conn,cur)
    elif sel == '2':
        mysql_login(conn,cur)
if __name__ == '__main__':
    main()
```

以上代码结合 MySQL 数据库设计实现了用户注册登录功能，其中第一段注释为用户数据表的创建命令，此段注释中的内容应在运行程序之前在 MySQL 终端中执行。

注释之后的代码为注册登录功能的实现，这些代码主要分为四个函数：注册函数 mysql_registered()、登录函数 mysql_login()、菜单函数 menu()和主函数 main()。下面分别对这四个函数进行说明。

1.mysql_registered()函数

mysql_registered()函数接收两个参数，其中参数 conn 为数据库连接，参数 cur 为数据库游标对象。此函数首先接收用户输入的用户名，在 while 循环中判断用户名是否有效（是否未被占用）；获取有效用户名后接收用户输入的密码，利用游标对象操作数据库，将用户名和密码插入到数据库表中；最后判断数据库操作是否成功。

2.mysql_login()函数

mysql_login()函数同样接收参数 conn 和 cur，该函数在获取用户输入的用户名后，会根据用户名到数据库中查找相应密码，若查询记录为空，说明用户名不存在，给出提示并继续

接收用户名；若查询到相应密码，使用变量 mysql_upwd 记录密码信息，并在循环中接收用户输入的密码，与数据库中查询到的密码进行匹配，若匹配成功则提示登录成功，否则提示密码错误，并继续密码。

3.menu()函数

menu()函数用于打印菜单。

4.main()函数

考虑到注册和登录功能都需要操作数据库，程序将数据库的打开与关闭提取到了 main()函数中，并在其间调用 menu()打印菜单、接收用户输入的功能选项，根据功能选项执行不同的功能。

代码测试

运行程序，根据提示选择不同功能，测试结果分别如下。

1.注册

（1）一般情况

```
----功能选择----
1.注册
2.登录
注册（1）or 登录（2）？1
用户名：coding
密　码：1907
注册成功
```

（2）用户名已存在

```
----功能选择----
1.注册
2.登录
注册（1）or 登录（2）？1
用户名：chan
用户名已存在，请重新输入
用户名：lili
密　码：1907
注册成功
```

2.登录

```
----功能选择----
1.注册
2.登录
注册（1）or 登录（2）？2
用户名：jim
用户名不存在
```

```
用户名：coding
密码：0321
密码错误，请重新输入
密码：1907
登录成功
```