```
"""
    模拟银行管理系统
    功能菜单
        1. 开户
        2. 存款
        3. 取款
        4. 转账
        5. 查询
        6. 退出
"""
'''
```
这行代码的作用是导入Python标准库中的getpass模块。getpass模块提供了一个函数，用于在命令行中安全地获取用户输入的密码，而不显示输入的字符。

实现原理
getpass模块的核心是getpass.getpass()函数。这个函数会提示用户输入密码，并且输入的密码不会在屏幕上显示，从而避免了密码泄露的风险。

用途
getpass模块在需要安全地获取用户密码的场景中非常有用，例如在脚本中需要用户输入密码时，使用getpass可以避免密码在命令行中明文显示。
```
'''
import getpass
'''
```
这段Python代码定义了一个名为`MENU`的字典，用于存储银行系统中的菜单选项及其对应的中文描述。每个菜单选项都是一个键值对，其中键是英文标识符，值是对应的中文描述。具体来说：

- `"create_account"`: 开户
- `"save_money"`: 存款
- `"withdraw_money"`: 取款
- `"transfer_money"`: 转账
- `"query_account"`: 查询
- `"exit_system"`: 退出

实现原理
字典（Dictionary）是Python中的一种数据结构，用于存储键值对。在这个例子中，键是菜单选项的英文标识符，值是对应的中文描述。字典的键是唯一的，每个键都映射到一个值。

用途
这段代码的用途是定义一个菜单，用于在银行系统中展示用户可以选择的操作。通过使用字典，可以方便地将英文标识符转换为中文描述，从而提高用户体验。

注意事项
1. **键的唯一性**：字典中的键必须是唯一的，不能重复。如果重复定义相同的键，后面的值会覆盖前面的值。
2. **键值对的访问**：可以通过键来访问对应的值。例如，`MENU["create_account"]`会返回"开户"。
3. **扩展性**：如果需要添加新的菜单选项，只需在字典中添加新的键值对即可。
4. **国际化**：如果需要支持多语言，可以定义多个字典，每个字典对应一种语言，然后根据用户选择的语言来展示对应的菜单。

这段代码可以作为银行系统中的一个基础模块，用于实现用户界面和功能选择。

```
'''
MENU = {
    "create_account":"开户",
    "save_money":"存款",
    "withdraw_money":"取款",
    "transfer_money":"转账",
    "query_account":"查询",
    "exit_system":"退出"
    }
'''

'''
# 定义一个字符串变量CARDIDS，值为"6666"
CARDIDS = "6666"
# 定义一个字符串变量CARDNUM，值为"11111"
CARDNUM = "11111"
# 定义一个字符串变量CARDIDE，值为"0000"
CARDIDE = "0000"
# 定义一个包含两个字典的列表，每个字典代表一张银行卡的信息
card = [{'id': '6666111110000', 'name': 'li', 'password': '123', 'money': 1100.22},
{'id': '6666111120000', 'name': 'wa', 'password': '123', 'money': 1.0}]
# 定义一个字典，用于表示不同状态码对应的含义
STATUS = {"001":"非法访问","002":"密码错误","003":"卡号不存在","004":"余额不足","005":"撤销操作"}

def menu(*args,**kwargs):
    # 定义菜单函数，返回菜单字符串
    # 定义变量i，初始值为1
    i = 1
    # 定义字符串变量str，初始值为"请选择功能：\n"
    str = f"请选择功能：\n"
    # 遍历MENU字典中的键值对
    for key,value in MENU.items():
        # 将键值对中的值添加到字符串变量str中，格式为"{i}.{value}\n"
        str += f"{i}.{value}\n"
        # i自增1
        i += 1
    return str

def create_account(*args,**kwargs):
    # 定义开户函数，返回卡号
    # 声明全局变量CARDNUM
    global CARDNUM
    # 输入用户姓名
    name = input("请输入您的姓名：")
    # 输入用户密码
    password = getpass.getpass("请输入您的密码：")
    # 输入用户存款金额
    money = float(input("请输入您的存款金额："))
    # 生成用户ID
    ID = CARDIDS + CARDNUM + CARDIDE
    # 将用户信息添加到card列表中
    card.append({"id":ID,"name":name,"password":password,"money":money})
```

```
    # CARDNUM自增1
    CARDNUM = str(int(CARDNUM) + 1)
    # 返回ID
    return ID

def update_money(*args,**kwargs):
    # 定义存款/取款函数，返回金额或错误代码
    # 判断args列表的第一个元素是否为1
    if args[0] == 1:
        # 如果是1，则将k赋值为"存"
        k = "存"
    # 否则，判断args列表的第一个元素是否为0
    elif args[0] == 0:
        # 如果是0，则将k赋值为"取"
        k = "取"
    else:
        return STATUS["001"]    #001错误代码表示非法访问
    # 检查账户
    index = check_account()
    # 如果账户存在
    if index != "":
        # 输入金额
        price = float(input(f"请输入您要{k}款的金额："))
        # 如果args[0]为1，表示存款
        if args[0] == 1:
            # 将金额加到账户余额中
            index["money"] += price
        # 如果args[0]为0，表示取款
        elif args[0] == 0:
            # 如果账户余额大于等于取款金额
            if index["money"] >= price:
                # 将金额从账户余额中减去
                index["money"] -= price
            # 如果账户余额小于取款金额
            else:
                # 返回余额不足的错误代码
                return STATUS["004"]    #004错误代码表示余额不足
        # 如果args[0]不为1或0，表示非法访问
        else:
            # 返回非法访问的错误代码
            return STATUS["001"]    #001错误代码表示非法访问
        # 返回金额
        return price

def transfer_money(*args,**kwargs):
    # 定义转账函数，返回True或错误代码
    # 检查账户
    index = check_account()
    # 如果账户存在
    if index != "":
        # 输入要转账的卡号
        otherid = input("请输入您要转账的卡号：")
        # 检查要转账的账户
```

```
        index2 = check_account(otherid)
        # 判断index2的类型是否为字符串
        if type(index2) != str:
            # 输入转账金额
            price = float(input("请输入您要转账的金额："))
            # 打印核对对方信息和转账金额
            print(f"核对对方信息与转账金额：{index2['name']}的卡号{index2['id']}，您要转账
{price}元")
            # 输入确认转账信息是否正确的选项
            reok = input(f"请确认转账信息是否正确（Y/N）：")
            # 如果确认转账信息正确
            if reok.upper() == "Y":
                # 如果转账金额小于等于余额
                if index["money"] >= price:
                    # 转账金额从index中减去
                    index["money"] -= price
                    # 转账金额加到index2中
                    index2["money"] += price
                    # 返回True
                    return True
                else:
                    # 返回余额不足的错误代码
                    return STATUS["004"]    #004错误代码表示余额不足
            else:
                # 返回撤销操作的错误代码
                return STATUS["005"]    #005错误代码表示撤销操作

def check_account(*args,**kwargs):
    # 定义查卡函数，返回卡信息或错误代码
    # 判断args是否为空
    if not args:
        # 如果为空，则提示用户输入卡号
        cardid = input("请输入您的卡号：")
    # 如果不为空，则将args的第一个元素赋值给cardid
    else:
        cardid = args[0]
    # 初始化found为False
    found = False
    for _ in card:  # 遍历card列表中的每一个元素
        if _["id"] == cardid:  # 如果当前元素的id与cardid相等
            found = True  # 将found设置为True
            if not args:  # 如果args为空
                password = getpass.getpass("请输入您的密码：")  # 提示用户输入密码
                if _["password"] == password:  # 如果密码正确
                    return _  # 返回当前元素
                else:
                    return STATUS["002"]    #002错误代码表示密码错误
            else:
                return _  # 返回当前元素
    if not found:  # 如果遍历完都没有找到匹配的卡号
        return STATUS["003"]  # 003 错误代码表示卡号不存在
#主函数
if __name__ == '__main__':
```

```python
# 无限循环，直到用户选择退出
while True:
    # 打印菜单
    print(menu())
    # 获取用户输入的选择
    choice = input("请输入您的选择：")
    # 如果用户选择开户
    if choice == "1":
        # 调用开户函数，获取返回值
        re = create_account()
        # 打印开户成功信息，并显示卡号
        print(f"开户成功，您的卡号是{re},请妥善保管")
    # 如果用户选择存款
    elif choice == "2":
        # 调用存款函数，获取返回值
        re = update_money(1)
        # 如果返回值不是字符串，则打印存款成功信息，并显示存款金额
        print(f"存款成功，您存入的金额为{re}元") if type(re)!= str else print(re)
    # 如果用户选择取款
    elif choice == "3":
        # 调用取款函数，获取返回值
        re = update_money(0)
        # 如果返回值不是字符串，则打印取款成功信息，并显示取款金额
        print(f"取款成功，您取款的金额为{re}元") if type(re)!= str else print(re)
    # 如果用户选择转账
    elif choice == "4":
        # 打印当前卡信息
        print(card)
        # 调用转账函数，获取返回值
        re = transfer_money()
        # 如果返回值不是字符串，则打印转账成功信息
        print("转账成功") if type(re)!= str else print(re)
    # 如果用户选择查询账户
    elif choice == "5":
        # 调用查询账户函数，并打印返回值
        print(check_account())
    # 如果用户选择退出
    elif choice == "6":
        # 退出循环
        exit()
    # 如果用户输入非法选项
    else:
        # 打印错误信息
        print(STATUS["001"])    #001错误代码表示非法访问
```