```python
class StudentScoreManagement:
    """学生成绩管理系统"""
    def __init__(self):
        # 登录状态
        self.login_status = False
        # 登录账号的权限
        self.permission = None
        # 账号身份验证码
        self.admin_code = "admin1234"
        self.student_code = "1234"
        # 存储学生成绩的字典
        self.Guess = {
            '小明':{'数学':90,'语文':80,'英语':100},
            '小红':{'数学':88,'语文':100,'英语':90}
            }
        # 存储已注册账号的字典
        self.userids = {
            "admin":{
                "admin":{"name":"李老师","pwd":"admin1234"},
                "admin1":{"name":"王老师","pwd":"admin1234"},
                },
            "student":{
                "xx123":{"name":"小明","pwd":"123456"},
                "xx456":{"name":"小红","pwd":"654321"}
                }
        }
        # 储存功能权限的字典
        # 定义不同用户类型及其对应的权限菜单
        self.permission_dict = {
            "admin": [
                "查询成绩",
                "修改成绩",
                "查询排名",
                "添加成绩",
                "查平均分",
                "退出"
            ],
            "student": [
                "查询成绩"
                "查询排名",
                "查平均分",
                "退出"
            ]
        }
        self.menu("欢迎使用学生成绩管理系统！")

    def login(self,*args,**kwargs):
        attempts = 3
        while attempts > 0:
            str1 = "重新" if attempts!= 3 else ""
            userid = input(f"请{str1}输入账号：")
```

```
            password = input("请输入密码：")
            if userid in self.userids["admin"] and password == self.userids["admin"]
[userid]['pwd']:
                self.login_status = True
                self.permission = "admin"
                return self.menu(f"登录成功！欢迎{self.userids['admin'][userid]['name']}
使用系统！")
            elif userid in self.userids['student'] and password ==
self.userids['student'][userid]['pwd']:
                self.login_status = True
                self.permission = "student"
                return self.menu(f"登录成功！欢迎{self.userids['student'][userid]
['name']} 使用系统！")
            else:
                attempts -= 1
                if attempts > 0:
                    print(f"登录失败，您还有 {attempts} 次机会。")
                else:
                    return self.register("login")

    def guess(self,*args,**kwargs):
        str1 = ""
        for name, score in self.Guess.items():
            for k,v in score.items():
                str1 += f"{name}的{k}成绩为：{v}\t"
            str1 += "\n"
        return str1

    def register(self,*args,**kwargs):
        if args:
            print("账号或密码错误！")
        userid = input("请输入账号：")
        if userid in self.userids["admin"] or userid in self.userids["student"]:
            return "该账号已存在！"
        password = input("请输入密码：")
        # 输入身份验证码，判断是否有权限注册老师账号
        check_code = input("请输入身份验证码：")
        if  check_code == self.admin_code and userid not in self.userids["admin"]:
            self.userids["admin"][userid] = {"name":input("请输入姓
名："),"pwd":password}
            self.login_status = True
            self.permission = "admin"
            return f"注册成功，已自动登录！欢迎{self.userids['admin'][userid]['name']}使
用系统！"
        elif check_code == self.student_code and userid not in self.userids["student"]:
            self.userids["student"][userid] = {"name":input("请输入姓
名："),"pwd":password}
            self.login_status = True
            self.permission = "student"
            return f"注册成功，已自动登录！欢迎{self.userids['student'][userid]['name']}
使用系统！"
        else:
            return "身份验证码错误！"
```

```
    def average(self, *args, **kwargs):
        subject_scores = {}
        for name, score in self.Guess.items():
            for subject, score_value in score.items():
                if subject not in subject_scores:
                    subject_scores[subject] = []
                subject_scores[subject].append(score_value)

        average_scores = {}
        for subject, scores in subject_scores.items():
            average_scores[subject] = sum(scores) / len(scores)

        result = ""
        for subject, average_score in average_scores.items():
            result += f"{subject} 平均分：{average_score}，  "
        return result.strip(", ")

    def lst_ranking(self,*args,**kwargs):
        # 学生成绩综合排名
        lst = sorted(self.Guess.items(), key=lambda x: sum(x[1].values()),
reverse=True)
        # 打印排名以及各科成绩
        str1 = ""
        for k,v in lst:
            str1 += f"{k}的排名为：{lst.index((k,v))+1}\n"
            for name, score in v.items():
                str1 += f"{name}成绩为：{score}\t"
            str1 += "\n"
        return str1

    def update_guess(self,*args,**kwargs):
        name = input("请输入学生姓名：")
        guess_object = input("请输入科目：")
        self.Guess[name][guess_object] = int(input("请输入成绩："))
        return f"{name}的{guess_object}成绩已更新为{self.Guess[name][guess_object]}"
    def menu(self,*args,**kwargs):
        # 如果是调用输出返回值
        if args:
            print(args[0])
        # 判断登录状态显示菜单
        if self.login_status:
            i = 1
            for k in self.permission_dict[self.permission]:
                print(f"{i}.{k}", end="\n")
                i += 1
            choice = input("请输入选项：")
            return self.menu_choice(int(choice) - 1)
        else:
            # 未登录的情况，返回注册或者登录菜单的选择
            return self.menu_choice(int(input("1.注册\n2.登录\n3.退出\n请输入选项：")) -
1)
    def menu_choice(self,*args,**kwargs):
        re = ""
```

```
        # 登录状态，未登录 选项执行
        if not self.login_status:
            if args[0] == 0:
                re = self.register()
            elif args[0] == 1:
                re = self.login()
            elif args[0] == 2:
                exit()
            else:
                print(self.menu("输入错误，请重新输入！"))
        else:
            # 老师权限，选择执行
            if self.permission == "admin":
                if args[0] == 0:
                    re =  self.guess()
                elif args[0] == 1:
                    re =  self.update_guess()
                elif args[0] == 2:
                    re =  self.lst_ranking()
                elif args[0] == 3:
                    re =  self.update_guess()
                elif args[0] == 4:
                    re =  self.average()
                elif args[0] == 5:
                    re = ()
                else:
                    re =  self.menu("输入错误，请重新输入！")
            # 学生权限，选择执行
            elif self.permission == "student":
                if args[0] == 0:
                    re =  self.guess()
                elif args[0] == 1:
                    re =  self.lst_ranking()
                elif args[0] == 2:
                    exit()
                else:
                    re =  self.menu("输入错误，请重新输入！")
        self.menu(re)
std = StudentScoreManagement()
```