day06 【综合练习】

今日内容

- 面向对象综合练习

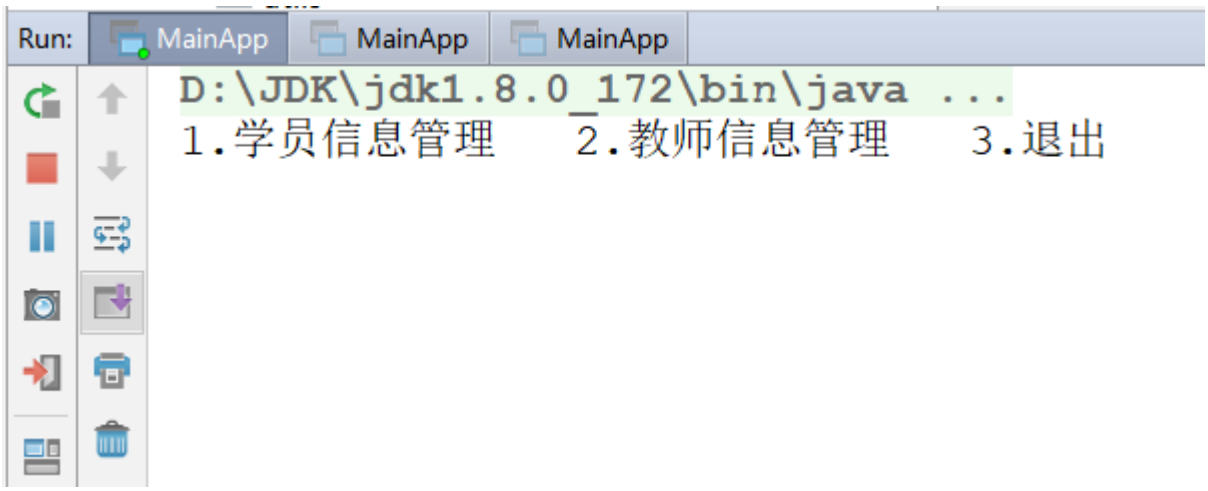
教学目标

- [ ] 能够编写主菜单
- [ ] 能够定义Person类并应用模板模式
- [ ] 能够定义子类Student类并添加特有成员
- [ ] 能够定义子类Teacher类并添加特有成员
- [ ] 能够理解继承在案例中的使用
- [ ] 能够理解模板模式在案例中的使用
- [ ] 能够定义并使用打印Person的静态方法
- [ ] 能够定义并使用打印ArrayList的静态方法
- [ ] 能够理解静态成员和静态方法在案例中的使用

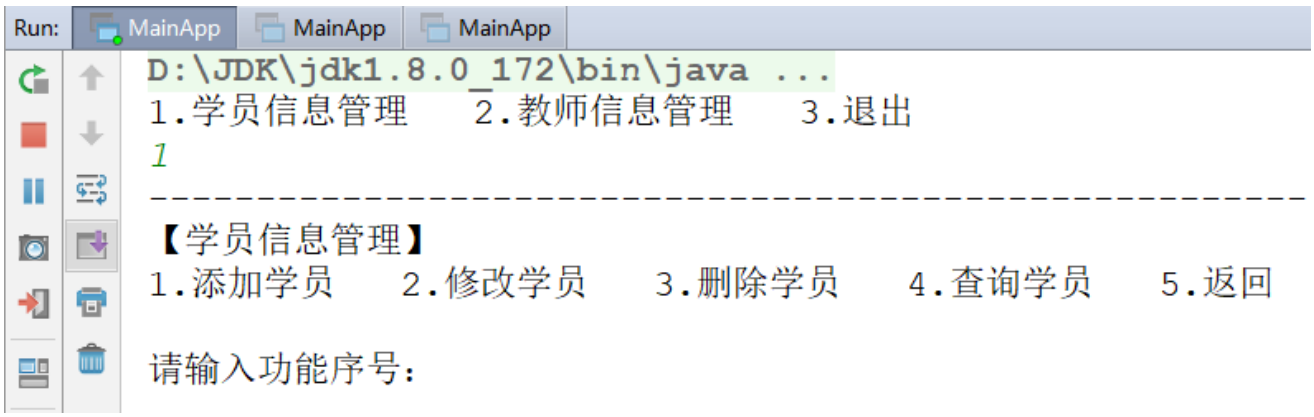
第一章 综合案例-案例演示

1.1 程序启动

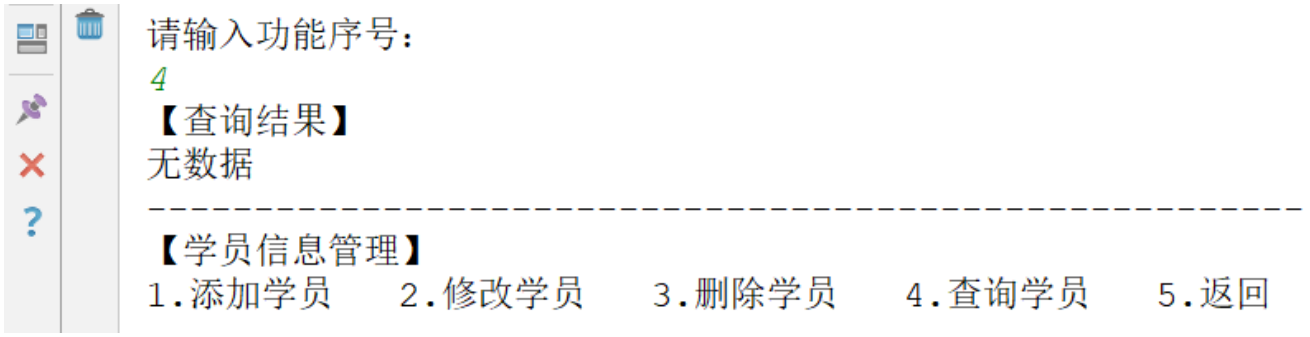
运行com.itheima.main.MainApp类，启动程序：



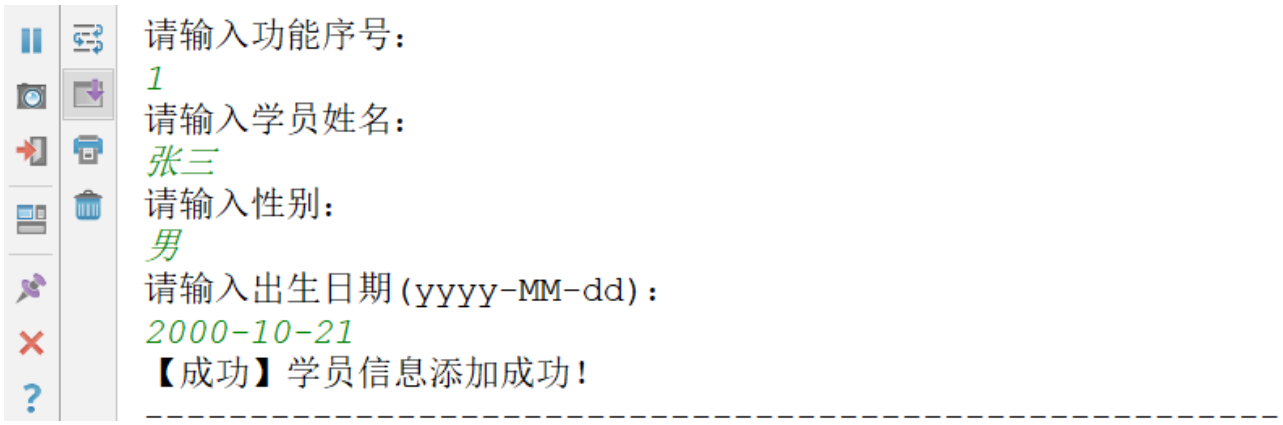
1.2 测试学员信息管理模块



1.3 测试【4.查询学员】

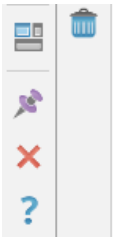


1.4 测试【1.添加学员】



1.5 测试【2.修改学员】

- 输入不存在的编号：



```
请输入功能序号：
2
请输入要修改的学员ID：
10
【错误】学员ID：10 没找到！
----------------------------------------------------------
```

- 输入存在的编号：

```
请输入功能序号：
2
请输入要修改的学员ID：
1
【查询结果】要修改的学员信息：
*************************************************************************************
编号      姓名      性别      生日          年龄      描述
1         张三      男        2000-10-21    18        我是一名：学生 我的工作是：学习Java
*************************************************************************************
请输入新姓名(保留原值输入0)：
王五
请输入新性别(保留原值输入0)：
0
请输入新出生日期(yyyy-MM-dd)(保留原值输入0)：
2000-10-10
【成功】学员信息修改成功！
----------------------------------------------------
```

1.6 测试【3.删除学员】

- 输入不存在的编号：


```
请输入功能序号：
3
请输入要删除的学员ID：
10
【错误】学员ID：10 未找到！
----------------------------------------------------------
```

- 输入存在的编号，但取消操作：

```
请输入功能序号：
3
请输入要删除的学员ID：
1
【查询结果】要删除的学员信息：
*************************************************************************************
编号      姓名      性别      生日          年龄      描述
1         王五      男        2000-10-10    18        我是一名：学生 我的工作是：学习Java
*************************************************************************************
【确认】您确定要删除这条信息吗(y/n)？
n
【取消】操作被取消！
----------------------------------------------------
```

- 输入存在的编号，执行删除：

```
请输入功能序号：
3
请输入要删除的学员ID：
1
【查询结果】要删除的学员信息：
--------------------------------------------------------------------------------
编号      姓名      性别      生日          年龄      描述
1         王五      男        2000-10-10    18          我是一名：学生 我的工作是：学习Java
--------------------------------------------------------------------------------
【确认】您确定要删除这条信息吗(y/n)？
y
【成功】数据已被删除！
------------------------------------------------------
```

第二章 综合案例-类设计

2.1 父类Person(抽象)

- 成员属性：
  - id(编号)
  - name(姓名)
  - sex(性别)
  - birthday(生日)
  - age(年龄-由生日计算得出)
- 构造方法：
  - 无参构造
  - 全参构造
- 成员方法：
  - toString()
- 抽象方法：
  - getType()：由各子类实现，返回各自的"类型"字符串。
  - getWork()：由各子类实现，返回各自的"工作"字符串。

2.2 子类Student

- 构造方法
  - 无参构造
  - 全参构造(super调用父类全参构造)
- 重写抽象方法
  - 重写getType()
  - 重写getWork()

2.3 子类Teacher

- 构造方法
  - 无参构造
  - 全参构造(super调用父类全参构造)
- 重写抽象方法
  - 重写getType()
  - 重写getWork()

2.4 工具类Utils类

- 全局变量
  - 学员ID值(添加学员信息时，编号由此ID加1生成)
  - 教师ID值(添加教师信息时，编号由此ID加1生成)
- 全局方法
  - 根据生日计算年龄的方法
  - 打印一个Person对象的方法；
  - 打印一个ArrayList集合的方法；

2.5 启动类

- 定义启动类：MainApp启动程序。

第三章 综合案例-类制作

3.1 父类Person(抽象)

```
public abstract class Person {
    private int id;//编号
    private String name;//姓名
    private String sex;//性别
    private String birthday;//出生日期
    private int age;//年龄--通过出生日期换算
    //构造方法
    public Person() {
    }
    public Person(int id,String name, String sex, String birthday)  {
        this.id = id;
        this.name = name;
        this.sex = sex;
        this.birthday = birthday;
    }
    //getter/setter
```

```
    public int getId() {
        return id;
    }
    public void setId(int id) {
        this.id = id;
    }
    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
    public String getSex() {
        return sex;
    }
    public void setSex(String sex) {
        this.sex = sex;
    }
    public String getBirthday() {
        return birthday;
    }
    public void setBirthday(String birthday) {
        this.birthday = birthday;
    }
    public int getAge()  {
        //通过生日计算年龄
        age = Utils.birthdayToAge(this.getBirthday());
        return age;
    }
    public void setAge(int age) {
        this.age = age;
    }
    //重写toString，同时作为模板
    @Override
    public String toString() {
        return  id + "\t\t" +
                name + "\t\t" +
                sex + "\t\t" +
                birthday + "\t" +
                this.getAge() + "\t\t" +
                " 我是一名：" + getType() + " 我的工作是：" + getWork();
    }
    //模板用到的两个方法，由子类重写
    public abstract String getWork();
    public abstract String getType();
}
```

3.2 子类Student

```
public class Student extends Person {
```

```
    public Student() {
    }
    public Student(int id, String name, String sex, String birthday) {
        super(id, name, sex, birthday);
    }
    @Override
    public String getWork() {
        return "学习Java";
    }
    @Override
    public String getType() {
        return "学生";
    }
}
```

3.3 子类Teacher

```
public class Teacher extends Person {
    public Teacher() {
    }
    public Teacher(int id, String name, String sex, String birthday) {
        super(id, name, sex, birthday);
    }
    @Override
    public String getWork() {
        return "讲课";
    }
    @Override
    public String getType() {
        return "老师";
    }
}
```

3.4 工具类Utils类

```
public class Utils {
    public static int stuId ;//学员ID的初始值
    public static int teaId ;//教师ID的初始值
    static {
        stuId = 0;
        teaId = 0;
        //后期可以改为从文件/数据库读取初始值
    }
    public static int birthdayToAge(String birthday)  {
        Date birthDate = null;
        try {//异常处理代码，后面讲
            birthDate = new SimpleDateFormat("yyyy-MM-dd").parse(birthday);
```

```
        } catch (ParseException e) {
            e.printStackTrace();
            return -1;
        }
        //获取当前系统时间
        Calendar cal = Calendar.getInstance();
        //如果出生日期大于当前时间，则返回-1
        if (cal.before(birthDate)) {
            return -1;
        }
        //取出系统当前时间的年、月、日部分
        int yearNow = cal.get(Calendar.YEAR);
        int monthNow = cal.get(Calendar.MONTH);
        int dayOfMonthNow = cal.get(Calendar.DAY_OF_MONTH);

        //将日期设置为出生日期
        cal.setTime(birthDate);
        //取出出生日期的年、月、日部分
        int yearBirth = cal.get(Calendar.YEAR);
        int monthBirth = cal.get(Calendar.MONTH);
        int dayOfMonthBirth = cal.get(Calendar.DAY_OF_MONTH);
        //当前年份与出生年份相减，初步计算年龄
        int age = yearNow - yearBirth;
        //当前月份与出生日期的月份相比，如果月份小于出生月份，则年龄减1，表示不满多少周岁
        if (monthNow <= monthBirth) {
            //如果月份相等，在比较日期，如果当前日，小于出生日，也减1，表示不满多少周岁
            if (monthNow == monthBirth) {
                if (dayOfMonthNow < dayOfMonthBirth) age--;
            }else{
                age--;
            }
        }
        return age;

    }
    //打印ArrayList的方法
    public static void printPersonList(ArrayList personList) {
        System.out.println("---------------------------------------------------------");
        System.out.println("编号\t\t姓名\t\t性别\t\t生日\t\t\t年龄\t\t描述");
        for (int i = 0; i < personList.size(); i++) {
            Object p = personList.get(i);
            System.out.println(personList.get(i));
        }
        System.out.println("---------------------------------------------------------");
    }
    //打印Person的方法
    public static void printPerson(Person person) {
        System.out.println("---------------------------------------------------------");
        System.out.println("编号\t\t姓名\t\t性别\t\t生日\t\t\t年龄\t\t描述");
        System.out.println(person );
        System.out.println("---------------------------------------------------------");
    }
}
```

第四章 综合案例-启动类实现

4.1 主菜单

```
public class MainApp {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        //学生集合
        ArrayList<Student> stuList = new ArrayList<>();
        //教师集合
        ArrayList<Teacher> teaList = new ArrayList<>();
        //主菜单
        while (true) {
            System.out.println("1.学员信息管理    2.教师信息管理    3.退出");
            int op = sc.nextInt();
            switch (op) {
                case 1:
                    studentManage(stuList,sc);
                    break;
                case 2:
                    teacherManage(teaList,sc);
                    break;
                case 3:
                    System.out.println("谢谢使用，拜拜！！");
                    System.exit(0);
                default:
                    System.out.println("你的输入有误，请重新输入！");
                    break;
            }
        }
    }
    //教师信息管理
    private static void teacherManage(ArrayList<Teacher> teaList,Scanner sc) {

    }

    //学员信息管理
    private static void studentManage(ArrayList<Student> stuList,Scanner sc) {

    }
}
```

4.2 学员信息管理二级菜单

```
public class MainApp {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        //学生集合
        ArrayList<Student> stuList = new ArrayList<>();
        //教师集合
        ArrayList<Teacher> teaList = new ArrayList<>();
        //主菜单
        while (true) {
            System.out.println("1.学员信息管理    2.教师信息管理    3.退出");
            int op = sc.nextInt();
            switch (op) {
                case 1:
                    studentManage(stuList,sc);
                    break;
                case 2:
                    teacherManage(teaList,sc);
                    break;
                case 3:
                    System.out.println("谢谢使用，拜拜！！");
                    System.exit(0);
                default:
                    System.out.println("你的输入有误，请重新输入！");
                    break;
            }
        }
    }
    //教师信息管理
    private static void teacherManage(ArrayList<Teacher> teaList,Scanner sc) {

    }

    //学员信息管理
    private static void studentManage(ArrayList<Student> stuList,Scanner sc) {

        //二级菜单
        while (true) {
            System.out.println("-----------------------------------------------------
");
            System.out.println("【学员信息管理】");
            System.out.println("1.添加学员    2.修改学员    3.删除学员    4.查询学员    5.返回");
            System.out.println();
            System.out.println("请输入功能序号：");
            int op = sc.nextInt();
            switch (op) {
                case 1:
                    addStudent(stuList, sc);
                    break;
                case 2:
                    updateStudent(stuList, sc);
                    break;
                case 3:
                    deleteStudent(stuList, sc);
```

```
                    break;
                case 4:
                    selectAll(stuList, sc);
                    break;
                case 5:
                    return;//结束方法
                default:
                    System.out.println("你的输入有误，请重新输入！");
                    break;
            }
        }
    }
    //添加学员
    private static void addStudent(ArrayList<Student> stuList, Scanner sc) {

    }
    //修改学员
    private static void updateStudent(ArrayList<Student> stuList, Scanner sc) {

    }
    //删除学员
    private static void deleteStudent(ArrayList<Student> stuList, Scanner sc) {

    }
    //查询所有学员
    private static void selectAll(ArrayList<Student> stuList, Scanner sc) {

    }
}
```

4.3 查询所有学员

```
//查询所有学员
private static void selectAll(ArrayList<Student> stuList, Scanner sc) {
    System.out.println("【查询结果】");
    if (stuList.size() == 0) {
        System.out.println("无数据");
        return;
    }
    Utils.printPersonList(stuList);//调用工具类打印
}
```

4.4 添加学员

```
//添加学员
private static void addStudent(ArrayList<Student> stuList, Scanner sc) {
    System.out.println("请输入学员姓名：");
    String name = sc.next();
    System.out.println("请输入性别：");
    String sex = sc.next();
    System.out.println("请输入出生日期(yyyy-MM-dd)：");
    String birthday = sc.next();

    stuList.add(new Student(++Utils.stuId,name,sex,birthday));

    System.out.println("【成功】学员信息添加成功！");
}
```

4.5 修改学员

```
//修改学员
private static void updateStudent(ArrayList<Student> stuList, Scanner sc) {
    System.out.println("请输入要修改的学员ID：");
    int stuId = sc.nextInt();
    //查询
    for (int i = 0; i < stuList.size(); i++) {
        Student stu = stuList.get(i);
        if (stu.getId() == stuId) {
            System.out.println("【查询结果】要修改的学员信息：");
            //打印
            Utils.printPerson(stu);
            //执行修改
            System.out.println("请输入新姓名(保留原值输入0)：");
            String newName = sc.next();
            System.out.println("请输入新性别(保留原值输入0)：");
            String newSex = sc.next();
            System.out.println("请输入新出生日期(yyyy-MM-dd)(保留原值输入0)：");
            String newBirthday = sc.next();

            if (!"0".equals(newName)) {
                stu.setName(newName);
            }
            if (!"0".equals(newSex)) {
                stu.setSex(newSex);
            }
            if (!"0".equals(newBirthday)) {
                stu.setBirthday(newBirthday);
            }
            System.out.println("【成功】学员信息修改成功！");
            return;
        }

    }
    System.out.println("【错误】学员ID：" + stuId + " 没找到！");
```

```
}
```

4.6 删除学员

```
//删除学员
private static void deleteStudent(ArrayList<Student> stuList, Scanner sc) {
    System.out.println("请输入要删除的学员ID：");
    int stuId = sc.nextInt();
    //查询
    for (int i = 0; i < stuList.size(); i++) {
        Student stu = stuList.get(i);
        if (stu.getId() == stuId) {
            System.out.println("【查询结果】要删除的学员信息：");
            Utils.printPerson(stu);
            System.out.println("【确认】您确定要删除这条信息吗(y/n)？");
            String str = sc.next();
            if ("y".equals(str)) {
                stuList.remove(i);
                System.out.println("【成功】数据已被删除！");
                return;//结束方法
            }else{
                System.out.println("【取消】操作被取消！");
                return;
            }

        }

    }
    System.out.println("【错误】学员ID：" + stuId + " 未找到！");
}
```

第五章 课堂练习

5.1参考学员管理实现教师管理模块