需求

编写一个控制台版本的员工管理系统，员工信息有ID、姓名、年龄、性别，并使用集合跟 面向对象 等知识点实现如下功能，

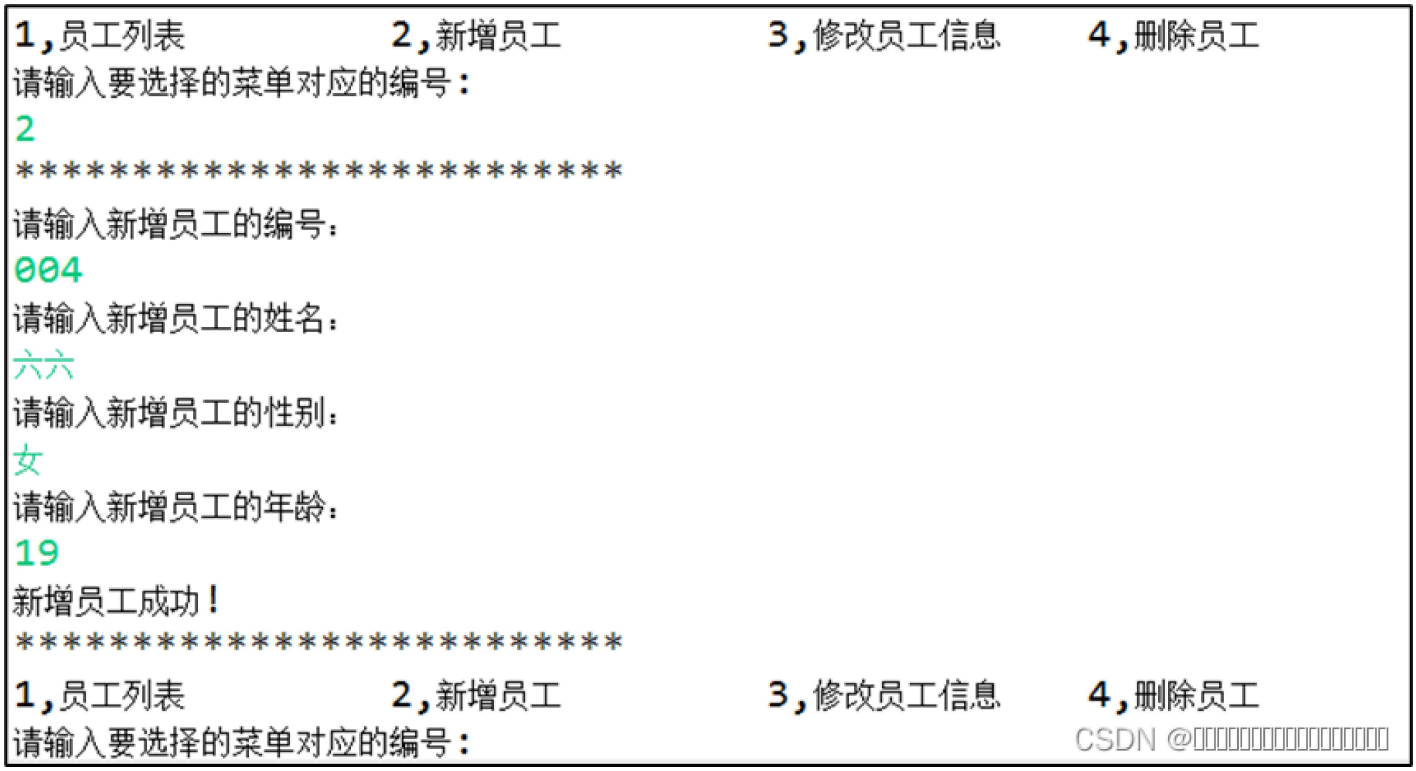
1、 默认 菜单选项 :1员工列表、2新增员工、3修改员工信息、4删除员工，根据1234序号来选中对应的菜单功能，截图如下：

```
1,员工列表        2,新增员工        3,修改员工信息        4,删除员工
请输入要选择的菜单对应的编号：
1
```


2、 实现员工列表显示功能，执行完功能后回到主页菜单。

```
1,员工列表        2,新增员工        3,修改员工信息        4,删除员工
请输入要选择的菜单对应的编号：
1
****************************
员工编号    员工姓名    员工性别    员工年龄
 001        张三          男          15
 002        李四          男          17
 003        王五          女          20
****************************
1,员工列表        2,新增员工        3,修改员工信息        4,删除员工
请输入要选择的菜单对应的编号：
```


3、 实现新增员工信息功能，如果新增员工编号已存在，则新增失败，给出提示，执行完功能后回到主页菜单。



4、 实现根据员工编号来修改员工信息的功能，执行完功能后回到主页菜单。

思路

1.定义员工类

2.主界面的代码编写

3.查看员工的代码编写

4.新增员工的代码编写

5.修改员工的代码编写

6.删除员工的代码编写

定义员工类

```
/*
    雇员类
 */

public class Emptyee {
    private String index;
    private String name;
    private String gerden;
    private String age;

    public Emptyee() {
    }
```

```
13
14      public Emptyee(String index, String name, String gerden, String age) {
15          this.index = index;
16          this.name = name;
17          this.gerden = gerden;
18          this.age = age;
19      }
20
21      /**
22       * 获取
23       * @return index
24       */
25      public String getIndex() {
26          return index;
27      }
28
29      /**
30       * 设置
31       * @param index
32       */
33      public void setIndex(String index) {
34          this.index = index;
35      }
36
37      /**
38       * 获取
39       * @return name
40       */
41      public String getName() {
42          return name;
43      }
44
45      /**
46       * 设置
47       * @param name
48       */
49      public void setName(String name) {
50          this.name = name;
51      }
52
53      /**
54       * 获取
55       * @return gerden
56       */
57      public String getGerden() {
58          return gerden;
59      }
60
61      /**
62       * 设置
63       * @param gerden
```

```
     */
    public void setGerden(String gerden) {
        this.gerden = gerden;
    }

    /**
     * 获取
     * @return age
     */
    public String getAge() {
        return age;
    }

    /**
     * 设置
     * @param age
     */
    public void setAge(String age) {
        this.age = age;
    }
}
```

主界面编写

```
//思路
/*
        1.用输出语句完成主界面的编写
        2.用Scanner实现键盘录入数据
        3.用switch语句完成操作的选择
        4.用循环完成再次选择
*/

import java.util.Scanner;
import java.util.Arrays;

public static void main(String[] args) {
    //new一个扫描器叫user
    Scanner user = new Scanner(System.in);
    //new一个集合叫array
    ArrayList<Emptyee> array = new ArrayList<>();
    //这里退出程序的方式有很多种,例如return,System.exit(0),flag等,我们这里用OUT标
    OUT:
    //写一个死循环让程序一直处于工作状态,当按下5的时候退出系统
    while (true){
        //打印菜单并输入需要进行的操作
        System.out.println("-----欢迎来到员工管理系统-----");
        System.out.println("1.员工列表      2.新增员工      3.修改员工信息      4.
```

```
        System.out.println("请输入你要选择的菜单对应的编号:");
        String index = user.next();
        //这里进行判断操作的方式有很多种,例如else if()等,我们这里使用switch来进行判
        switch (index){
                //当index为1时我们查看员工列表
            case "1":
                //System.out.println("员工列表");
                //调用查看员工列表的方法
                findAllEmptyee(array);
                break;
                //当index为2时我们新增员工
            case "2":
                //System.out.println("新增员工");
                //调用新增员工的方法
                addEmptyee(array);
                break;
                //当index为3时我们修改员工信息
            case "3":
                //System.out.println("修改员工信息");
                //调用修改员工的方法
                updateEmptyee(array);
                break;
                //当index为4时我们删除员工
            case "4":
                //System.out.println("删除员工");
                //调用删除员工的方法
                deleteEmptyee(array);
                break;
                //当index为5时我们退出系统
            case "5":
                //退出系统前显示提示退出的提示语
                System.out.println("退出系统");
                //System.exit(0);//JVM退出
                break OUT;
                //当index为其他时我们重新输入一下正确的操作
            default:
                System.out.println("请输入正确的操作");
                break;
        }
    }
}
```

查看员工的代码编写

```
//思路
/*
        1.用键盘录入先择查看员工
        2.定义一个方法,用于查看员工
                1.判定集合中是否有数据,如果没有显示,提示信息
                2.显示表头信息
                3.将集合中数据取出,按照对应格式显示员工信息,年龄显示补充"岁"
        3.调用方法
*/

//定义一个查看员工列表的方法,这里我们不需要返回所以使用void,因为我们需要对Emptyee进行操
public static void findAllEmptyee(ArrayList<Emptyee> array) {
    //判断集合里面有没有员工信息,如果没有员工信息,则需要先添加信息,如果有则直接显示员工
    if (array.size() == 0){
        System.out.println("没有员工信息,请先添加员工信息再查询");
    }else {
        //显示表头信息
        System.out.println("员工编号\t\t员工姓名\t\t员工性别\t\t员工年龄");

        //将集合中数据取出,按照对应格式显示员工信息,年龄显示补充"岁"
        for (int i = 0 ; i < array.size() ; i++){
            Emptyee s = array.get(i);
            System.out.println(s.getIndex() + "\t" + s.getName() + "\t\t"
        }
    }
}
```

新增员工代码的编写

```
//思路
/*
        1.用键盘录入选择新增员工
        2.定义一个方法,用于新增员工
                1.显示提示信息,提示要输入什么信息
                2.键盘录入员工对象所需要的数据(工号,姓名,性别,年龄)
                3.创建对象,把键盘录入的数据赋值给员工对象的成员变量
                4.将对象添加到集合中(保存)
                5.给出添加成功的提示信息
        3.调用方法
*/

public static void addEmptyee(ArrayList<Emptyee> array){
    //因为这里需要进行输入,所以在这里new一个扫描器叫user,因为这个扫描器的作用范围仅仅
    Scanner user = new Scanner(System.in);
    //这个index定义在这里是因为,把index定义在while循环里边接收数据的地方,那么他的作用
    String index;
    System.out.println("请输入员工编号");
    //这里使用死循环判断工号有没有被占用,如果有则重新输入,没有则继续向下进行增加员工的操
```

```
    while (true){
        index = user.next();
        //这里调用一下判断工号是否重复的方法
        boolean flag = isUsed(array, index);
        //这里进行判断,这个flag在下面会解释为什么要成立才需要重新输入工号
        if (flag) {
            System.out.println("该工号已使用,请重新输入工号");
        }else {
            break;
        }
    }
    System.out.println("请输入员工姓名");
    String name = user.next();
    System.out.println("请输入员工性别");
    String gerden = user.next();
    System.out.println("请输入员工年龄");
    String age = user.next();


    //创建对象,把键盘录入的数据赋值给员工对象的成员变量,这里通过有参构造和无参构造都能赋
    //Emptyee emptyee = new Emptyee(index , name , gerden , age);
    Emptyee emptyee = new Emptyee();
    emptyee.setIndex(index);
    emptyee.setName(name);
    emptyee.setGerden(gerden);
    emptyee.setAge(age);

    //将对象添加到集合中(保存)
    array.add(emptyee);

    //给出添加成功的提示信息
    System.out.println("新增员工成功! ");
}

```

判断工号是否被使用

```
//思路
/*
        1.定义一个方法,对学号是否被使用进行判断
                1.如果与集合中的某一个学生学号相同,返回true
                2.如果都不相同,则返回false
        2.在添加学生录入学号后调用该方法
                1.如果返回true,弹出提示,重新输入学号
                2.如果返回false,正常添加学生对象
*/

//定义一个判断学号重复的方法,这里因为后面有需要返回的值,且是Boolean类型的值所以使用boo
```

```java
public static boolean isUsed(ArrayList<Emptyee> array , String index){
    //这里默认初始值是false
    boolean flag = false;

    //通过for循环进行集合的遍历
    for (int i = 0 ; i < array.size() ; i++) {
        Emptyee emptyee = array.get(i);
        //这里判断集合里边的工号和输入的工号是否重复了,成立则说明有相同的工号,所以把fl
        if (emptyee.getIndex().equals(index)){
            flag = true;
            break;
        }
    }

    return flag;
}
```

修改员工代码的编写

```java
//思路
/*
        1.用键盘录入选择修改员工
        2.定义一个方法,用于修改员工
                1.显示提示信息
                2.键盘录入要修改员工的工号
                3.键盘录入要修改的学生信息
                4.修改对应的学生信息
                5.给出修改成功提示
        3.调用方法
*/

public static void updateEmptyee(ArrayList<Emptyee> array){
    //因为这里需要进行输入,所以在这里new一个扫描器叫user,因为这个扫描器的作用范围仅仅
    Scanner user = new Scanner(System.in);
    //定义个count来当作后边判断修改时工号是否一致
    int count = -1;
    System.out.println("请输入要修改员工的工号");
    String index = user.next();

    //通过遍历集合来查找集合里的工号和需要修改的工号是否存在
    for (int i = 0 ; i < array.size() ; i++){
        Emptyee emptyee = array.get(i);
        //如果存在,把i的值给count保存起来
        if (emptyee.getIndex().equals(index)){
            count = i;
            break;
        }
```

```
    }

    //这里判断count的值有没有被改变,如果被没有被改变,则说明需要修改工号的信息不存在或没
    if (count == -1){
        System.out.println("修改失败,请确认员工号是否正确");
    }else {
        //重新输入员工的信息
        System.out.println("请输入员工新姓名");
        String name = user.next();
        System.out.println("请输入员工新性别");
        String gerden = user.next();
        System.out.println("请输入员工新年龄");
        String age = user.next();

        //修改员工信息
        Emptyee emptyee = new Emptyee();
        emptyee.setIndex(index);
        emptyee.setName(name);
        emptyee.setGerden(gerden);
        emptyee.setAge(age);
        array.set(count , emptyee);

        //给出提示修改成功的语句
        System.out.println("修改成功! ");
    }
}
```

删除员工的代码编写

```
//思路
/*
        1.用键盘录入选择删除员工
        2.定义一个方法,用于删除员工
                1.显示提示信息
                2.键盘录入要删除的员工的工号
                3.遍历集合,将这个工号对应的员工对象从集合删除
                4.给出删除成功提示
        3.调用方法
*/

public static void deleteEmptyee(ArrayList<Emptyee> array){
    //因为这里需要进行输入,所以在这里new一个扫描器叫user,因为这个扫描器的作用范围仅仅
    Scanner user = new Scanner(System.in);
    //定义个count来当作后边判断删除时工号是否一致
    int count = -1;
    System.out.println("请输入要删除员工的工号");
    String index = user.next();
```

```
        //通过遍历集合来查找集合里的工号和需要删除的工号是否存在
        for (int i = 0 ; i < array.size() ; i++){
            Emptyee emptyee = array.get(i);
            //如果存在,把i的值给count保存起来
            if (emptyee.getIndex().equals(index)){
                count = i;
                break;
            }
        }

        //这里判断count的值有没有被改变,如果被没有被改变,则说明需要删除工号的信息不存在或没
        if (count == -1){
            System.out.println("删除失败,请确认员工号是否正确");
        }else {
            array.remove(count);
            System.out.println("删除成功! ");
        }
    }
```