作业一：

Calendar类：

获取任意一年的二月有多少天

思路：

1. 键盘录入任意的年份
2. 设置日历对象的年、月、日
   * 年：来自于键盘录入
   * 月：设置为3月，月份是从0开始的，所以设置的值是2
   * 日：设置为1日
3. 3月1日往前推一天，就是2月的最后一天
4. 获取这一天输出即可

作业二：

Vector<Course>：

课程管理每学期都要开设多门课程，每门课程的信息包括课程名称（String）、是否是必修课(boolean)、授课教师（String）、学时(int)、学分(float)；编程实现对课程信息进行创建、添加、删除、修改和查询操作；可以按课程名、是否是必修课、授课教师来进行查询；修改或删除之前要先查询；

作业三：

正则表达式：

输入我的手机号是18837112195，曾经用过18888888888，还用过18812345678，(号码随意，形式一样就行)

* 首先利用获取功能提取出现在用的手机号；
* 其次匹配功能检验该手机号是否符合：第一位为1，第二位为（3,4,5,7,8,9）中的一个，后面9位为任意0-9的数字；
* 其次利用字符串替换功能，将中间四位用\*号加密，比如18837112195——>188\*\*\*\*2195

作业四：

枚举类：

编写菜单系统模拟点菜：编写三个枚举类（前菜类（SALAD,SOUP, SPRING\_ROLLS;）、主菜类（LASAGNE,BURRITO,PAD\_THAI,LENTILS;）、甜点类（BLACK\_FOREST\_CAKE,FRUIT;））实现一个Menu接口，通过括号赋值来定义菜价比如SALAD（35.0），主函数实现点菜流程，每个菜类各点一道菜，最终计算总消费。（一个消费对象即可）

作业五：

Calendar类，Date类，SimpleDateFormat类

修改上周身份证号的作业中将提取的出生年月日转化为年龄输出。

作业六：

有一个给出Student类，在Test类里面利用反射机制实现以下操作：

1. 创建Student类的Class对象
2. 获取完整Student类名并输出
3. 调用无参构造方法创建Student类的实例化对象stu1
4. 调用有参构造方法创建Student类的实例化对象stu2
5. 获取类全部属性信息，并修改
6. 获取全部方法信息
7. 调用类中无参方法display()
8. 调用类中有参方法say()
9. 通过反射来操作属性,设置名字为Bob

public class Student {

private String name;

private int age;

public Student() {

}

public Student(String name,int age) {

this.name = name;

this.age = age;

}

public int getAge() {

return age;

}

public void setAge(int age) {

this.age = age;

}

public void display() {

System.out.println("学生信息： name=" + name + " age=" + age);

}

public void say(String str){

System.out.println("学生说：" + str);

}

}

作业七：利用反射实现的代理模式（真实业务类，代理类，接口）

现给出接口定义、真实业务实现类及测试类定义，编写代理类（MyInvocationHandler）

interface Subject {

public void say(String name,int age);

}

class RealSubject implements Subject{

@Override

public void say(String name, int age) {

System.out.println(name + "今年已经" + age + "岁了。");

}

}

public class Test {

public static void main(String[] args) {

MyInvocationHandler demo = new MyInvocationHandler();

Subject sub = (Subject) demo.bind(new RealSubject());

sub.say("Rollen",23);

}

}

作业8：利用反射完成工厂设计模式

有一个公共接口（Fruit）里面包含一个抽象方法eat(),Apple,Orange类实现了Fruit接口；

有个一个公共接口（Vegetables）里面包含一个抽象方法make(),Carrot,Tomato类实现了Vegetables接口；

设计一个工厂类，可以对通过接口实例化对的方式创建Apple,Orange，Carrot,Tomato类的实例化对象，并调用该对象的方法。

interface Fruit {

public abstract void eat();

}