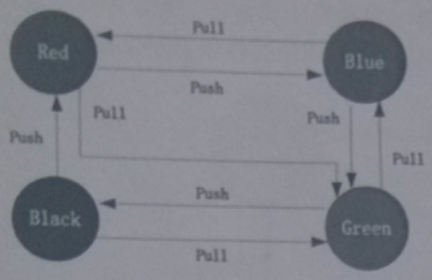
考察知识点：Java语言实现类的方法

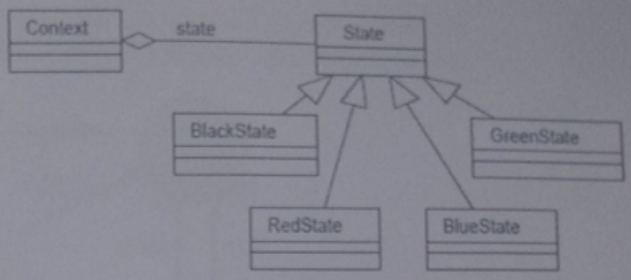
3. 软件设计（20 分）

给定软件可根据鼠标的点击操作改变界面颜色。下图中，左图显示了在不同颜色之间的状态转移关系，即：

- 界面在红色状态下，pull 操作可改变为绿色，push 操作可改变为蓝色；
- 界面在蓝色状态下，pull 操作可改变为红色，push 操作可改变为绿色；
- 界面在绿色状态下，pull 操作可改变为蓝色，push 操作可改变为黑色；
- 界面在黑色状态下，pull 操作可改变为绿色，push 操作可改变为红色.

采用状态机的设计模式，类图如下图中右图所示。其中，RedState, BlueState, BlackState, GreenState 各个类分别代表界面显示为红、蓝、黑、绿四种颜色时相应的状态。

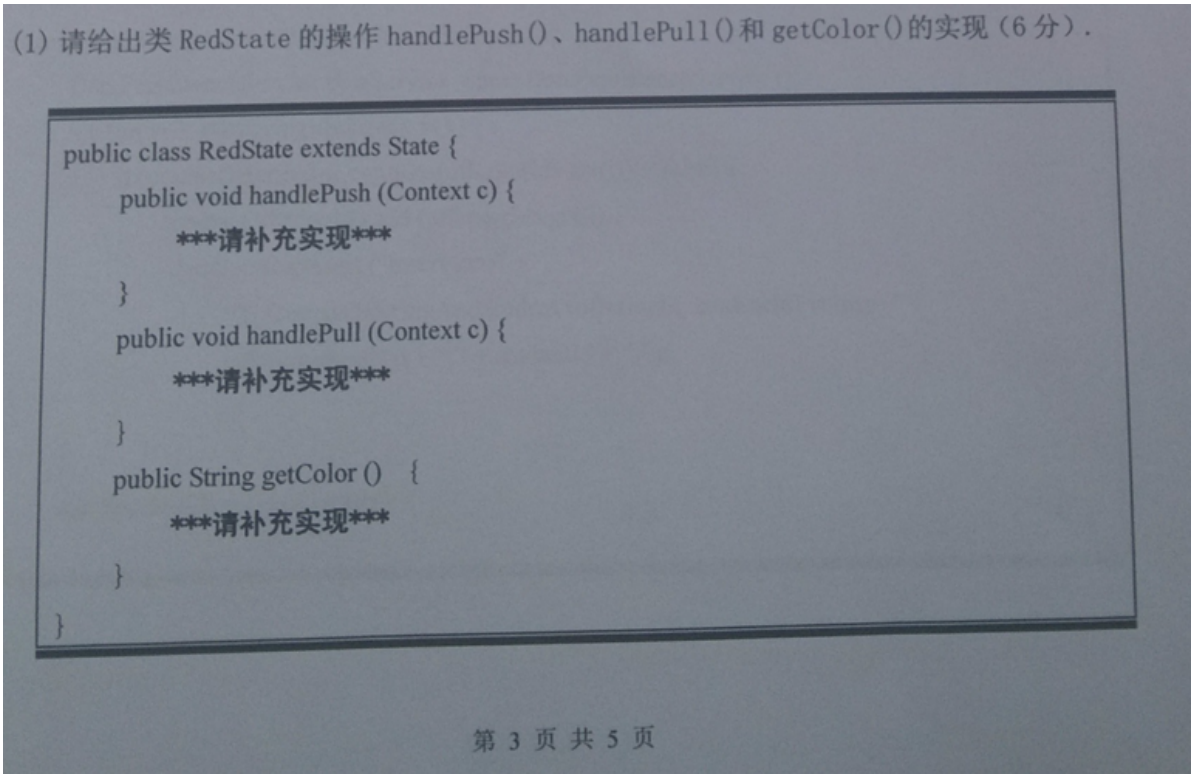




类 Context 和类 State 的设计如下：

```
public class Context {
    private State state = null;
    public Context (State state) {this.state = state;}
    public Context () {this (new RedState());}
    ……
    public void push () {state.handlePush(this);}
    public void pull () {state.handlePull(this);}
    public void setState(State state) {this.state = state;}
}
```

```
public abstract class State {
    public abstract void handlePush (Context c);
    public abstract void handlePull (Context c);
    public abstract String getColor ();
}
```


(1) 请给出类 RedState 的操作 handlePush()、handlePull()和 getColor()的实现（6 分）.

```
public class RedState extends State {
    public void handlePush (Context c) {
        ***请补充实现***
    }
    public void handlePull (Context c) {
        ***请补充实现***
    }
    public String getColor ()  {
        ***请补充实现***
    }
}
```

第 3 页 共 5 页

参考解答：
没有参考答案加持，如有问题，欢迎DM！
(1)

```
public class RedState extends State{
    // 界面在红色状态，push操作可改变成蓝色
    public void handlePush(Context c){
        c.setState(BlueState);
    }

    // 界面在红色状态，pull操作可改变成绿色
    public void handlePull(Context c){
        c.setState(GreenState);
    }

    public void String getColor(){
        return this.state;
    }
}
```

考察知识点：UML用例图

(书本P224页对于这种图的描述较为模糊，复习提纲中第43点提及到：`掌握用例图的概念、内容和方法`，故拿两道习题来练练手)

包含关系和扩展关系的区别

匿名用户
推荐于2017-12-16

包含关系下，去掉子用例，父用例将不能使用；扩展关系则不会，扩展关系中子用例是对父用例的增强型描述

本回答被网友采纳

86 评论(3) 分享 举报

5．以下给出了“老年人监护系统”中的用例及其描述，使用 UML 用例图描述该系统，并给出用例之间的联系。

(1)摔倒动作检测：从楼梯传感器和摄像头中获取输入数据，用以检测是否有人摔倒。

(2)摔倒事件报警：如果检测到某位老人摔倒，那么发送一条报警消息到手机上，同时该报警信息会被发送到用例“事件日志”中进行记录。

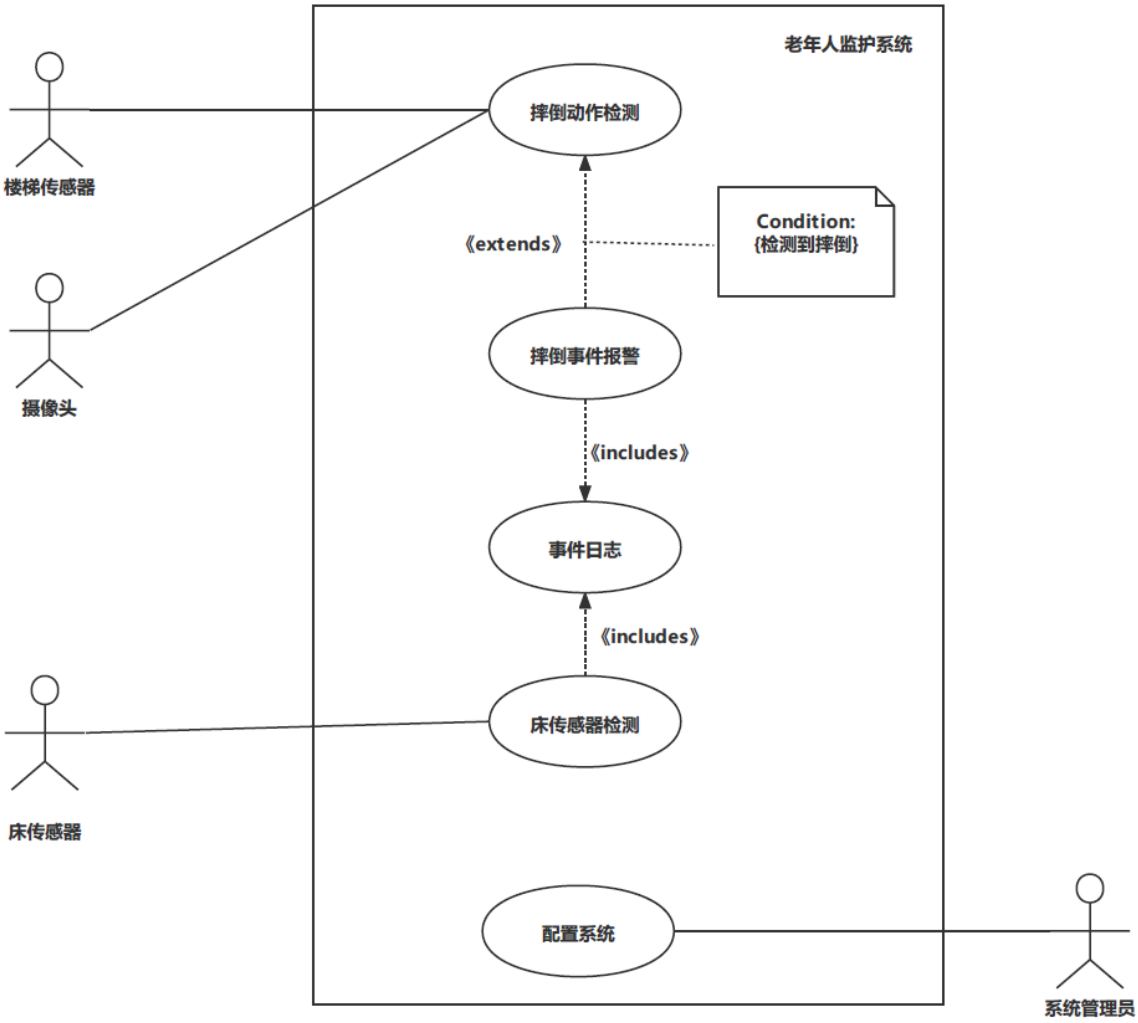
(3)事件日志：将发生的事件记录在数据库中。

(4)床传感器监测：从安装在床位上的床传感器中获取脉搏、呼吸等数据，并发送到用例“事件日志”中处理。

(5)配置系统：系统管理员对系统进行各种配置操作。

摔倒事件报警是对摔倒动作的增强型描述
去除摔倒时间报警和床传感器检测，事件日志将无法使用。

看不清楚图片，请戳我！



考察知识点：UML用例图

6．在图书管理系统中：

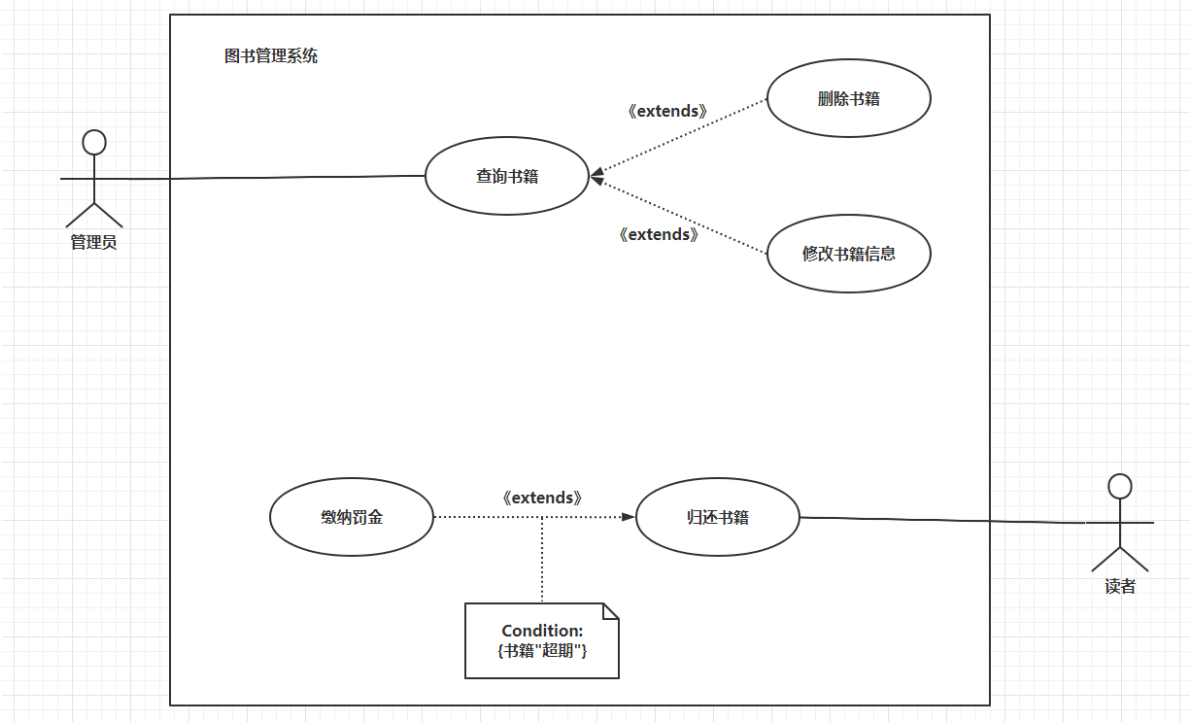
(1)管理员可进行“删除书籍”和“修改书籍信息”这两个操作，并且这两个操作在执行前都必须先进行“查询书籍”操作。

(2)读者可以“还书”，这是一个基础操作。如果读者所借书籍超期，还书时要缴纳罚金，即当书籍“超期”时，将执行“缴纳罚金”操作。

要求：画出上述系统的 UML 用例图。

> 删除书籍和修改书籍信息是对查询书籍的增强型描述
> 交纳罚金是对归还书籍的增强型描述

看不清楚图片，请戳我！



考察知识点：类图、Java语言编写类

3．某位系统分析人员针对高校学生选课系统的需求设计了分析类图的草图，如图 6.20 所示。

(1) 教务人员通过职工号和密码登录系统，录入课程信息。课程信息包括：课程编号、课程名称、授课时间、授课地点。并为每名教师排课：一名教师可以教授 0 门或多门课程，1 门课程也可以由 1 或多名教师教授。

(2) 学生通过学号和密码登录系统，选择课程。系统要求 1 门课程至少要有 10 名学生选课才能开课，1 名学生至少要选择 1 门课程。

(3) 教师通过职工号和密码登录系统，获取课程的学生名单。

(4) 课程结束后，教师通过该系统录入学生成绩。

(5) 学生可通过该系统查询自己的课程成绩。

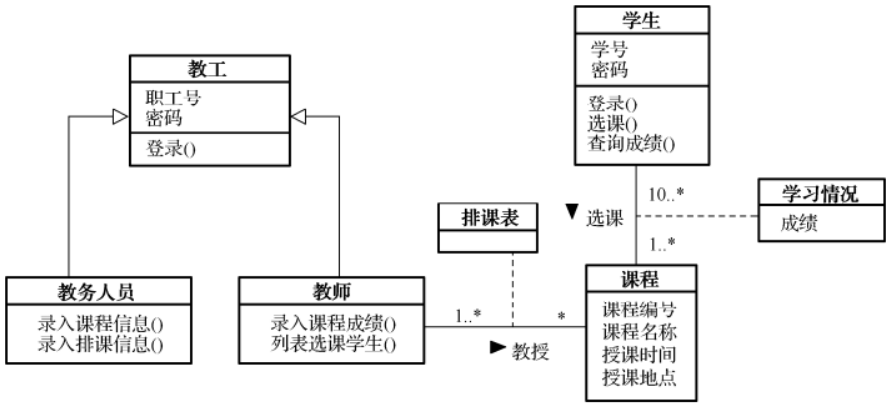
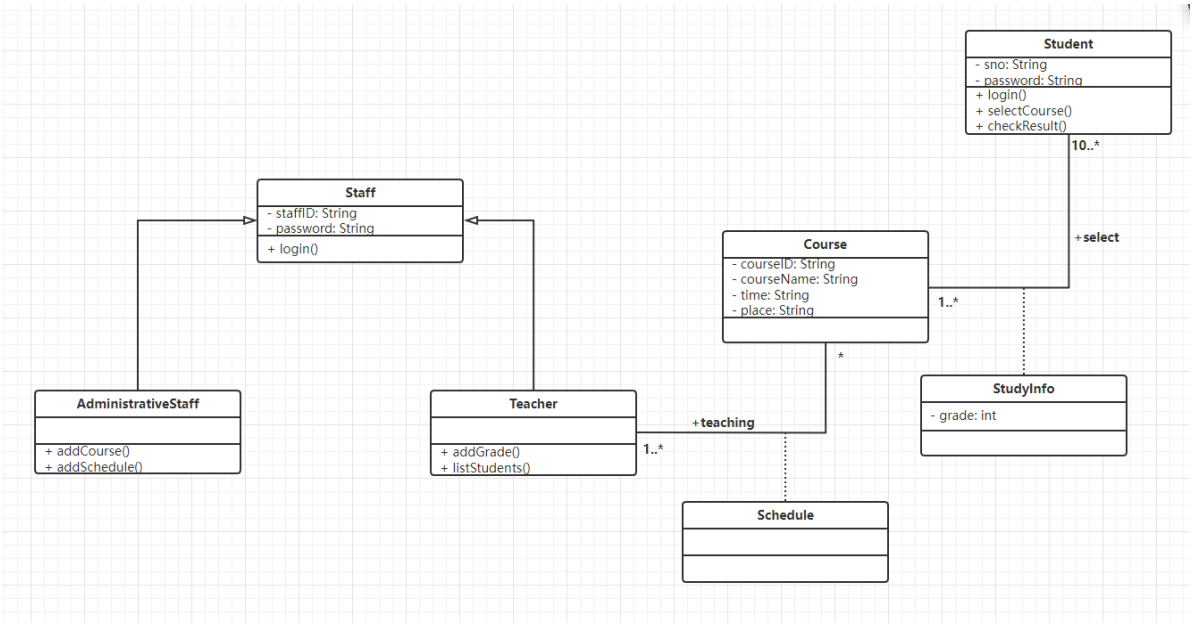


图 6.20　类图草图

请基于图 6.20 进行细化和完善，尽可能补充需要的信息，完成该系统的设计类图，并尝试使用 Java 或 C++语言编写该系统的代码框架。

参考解答：
类图：

看不清楚图片，请戳我！



Java语言编写该系统代码框架
【注：有些方法仅凭题干无法补全，因此做第二小问的时候，只需掌握Java面向对象的基本操作即可，构造方法、get、set方法、一些普通方法即可】

```
// 学生类
public Class Student{
    private String sno;
    private String password;

    // 无参构造方法
    public Student(){

    }
```

```
    // 有参构造方法
    public Student(String sno,String password){
        this.sno=sno;
        this.password=password;
    }

    // get方法
    public String getSno(){
        return this.sno;
    }
    public String getPassword(){
        return this.password;
    }

    // set方法
    public void setSno(String sno){
        this.sno=sno;
    }
    public void setPassword(String password){
        this.password=password;
    }

    // 登录方法
    public boolean login(String sno,String password){
        if(sno.equals("xxxxxx")&&password.equals("123456"))
            return true;
        else
            return false;
    }

    // 选课方法
    public boolean selectCourse(String courseID){

    }

    // 查成绩方法
    public int checkResult(String courseID){

    }
}
```

```
// 教工类
public Class Staff{
    private String staffID;
    private String password;

    // 无参构造方法
    public Staff(){

    }

    // 有参构造方法
    public Staff(String staffID,String password){
        this.staffID=staffID;
        this.password=password;
    }
```

```
    // get方法
    public String getStaffID(){
        return this.staffID;
    }
    public String getPassword(){
        return this.password;
    }

    // set方法
    public void setStaffID(String staffID){
        this.staffID=staffID;
    }
    public void setPassword(String password){
        this.password=password;
    }

    // 登录方法
    public boolean login(String staffID,String password){
        if(staffID.equals("xxxxx")&&password.equals("123456"))
            return true;
        else
            return false;
    }
}
```

```
// 教务人员类
public Class AdministriveStaff extends Staff{

    public boolean addCourse(){

    }

    public boolean addSchedule(){

    }
}
```

```
// 教师类
public Class Teacher extends Staff{

    public boolean addGrade(){

    }

    public List listStudents(){

    }
}
```

```
// 课程类
pulic Class Course{
    private String courseID;
    private String courseName;
    private String time;
    private String place;
```

```
    // 无参构造方法
    public Course(){

    }

    // 有参构造方法
    public Course(String courseID,String courseName,String time,String place){
        this.courseID=courseID;
        this.courseName=courseName;
        this.time=time;
        this.place=place;
    }

    // get方法
    public String getCourseID(){
        return this.courseID;
    }
    public String getCourseName(){
        return this.courseName;
    }
    public String getTime(){
        return this.time;
    }
    public String getPlace(){
        return this.place;
    }

    // set方法
    public void setCourseID(String courseID){
        this.courseID=courseID;
    }
    public void setCourseName(String courseName){
        this.courseName=courseName;
    }
    public void setTime(String time){
         this.time=time
    }
    public void setPlace(String place){
        this.place=place;
    }
}
```

```
// 成绩类
public Class StudyInfo{
    private int grade;

    // 无参构造方法
    public StudyInfo(){

    }

    // 有参构造方法
    public StudyInfo(int grade){
        this.grade=grade;
    }
```

```
    // get方法
    public int getGrade(){
        return this.grade;
    }

    // set方法
    public void setGrade(int grade){
        this.grade=grade;
    }
}
```

```
// 排课表类
public Class Schedule{

}
```

考察知识点：PAD图

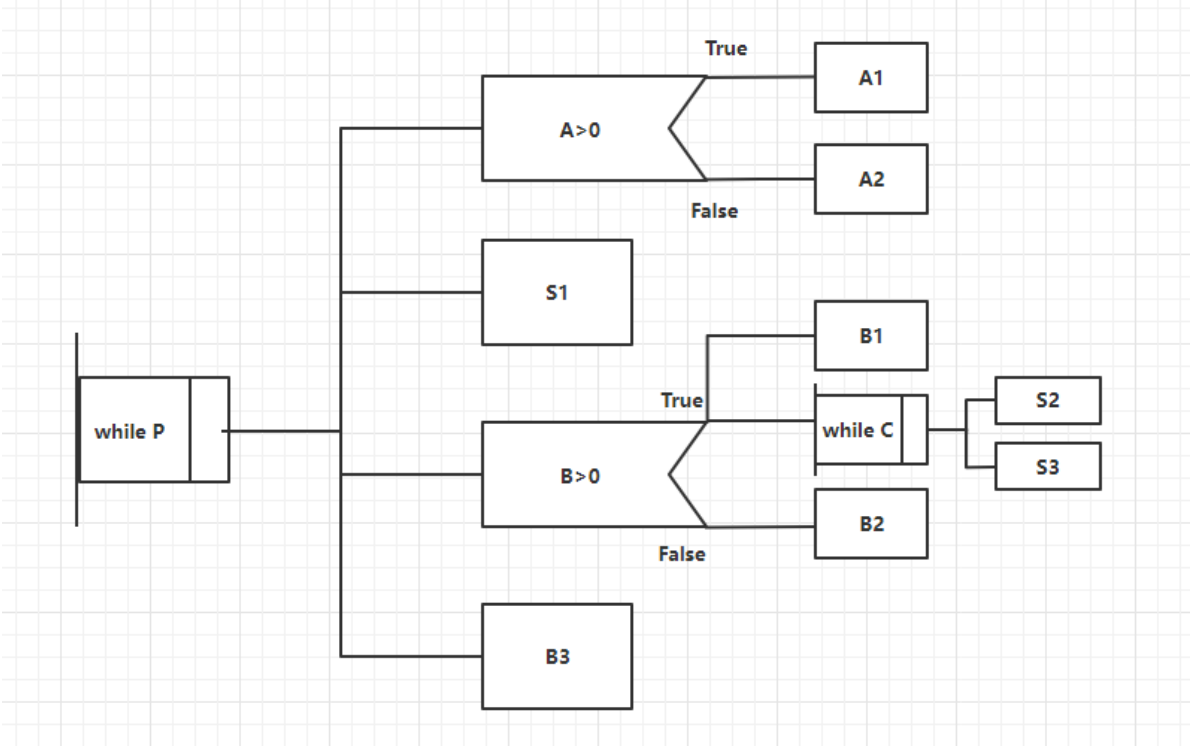
6. 画出下面用 PDL 写出的程序的 PAD 图。

```
while P do
    if A>0
        then A1
        else A2
    end if;
    S1;
    if B>0
        then B1;
            while C do
                S2; S3;
            end while;
        else B2;
```

软件工程

```
        end if;
        B3;
    end while;
```

参考解答：


True
A1
A>0
A2
False
S1
B1
True
while C
S2
while P
B>0
S3
B2
False
B3