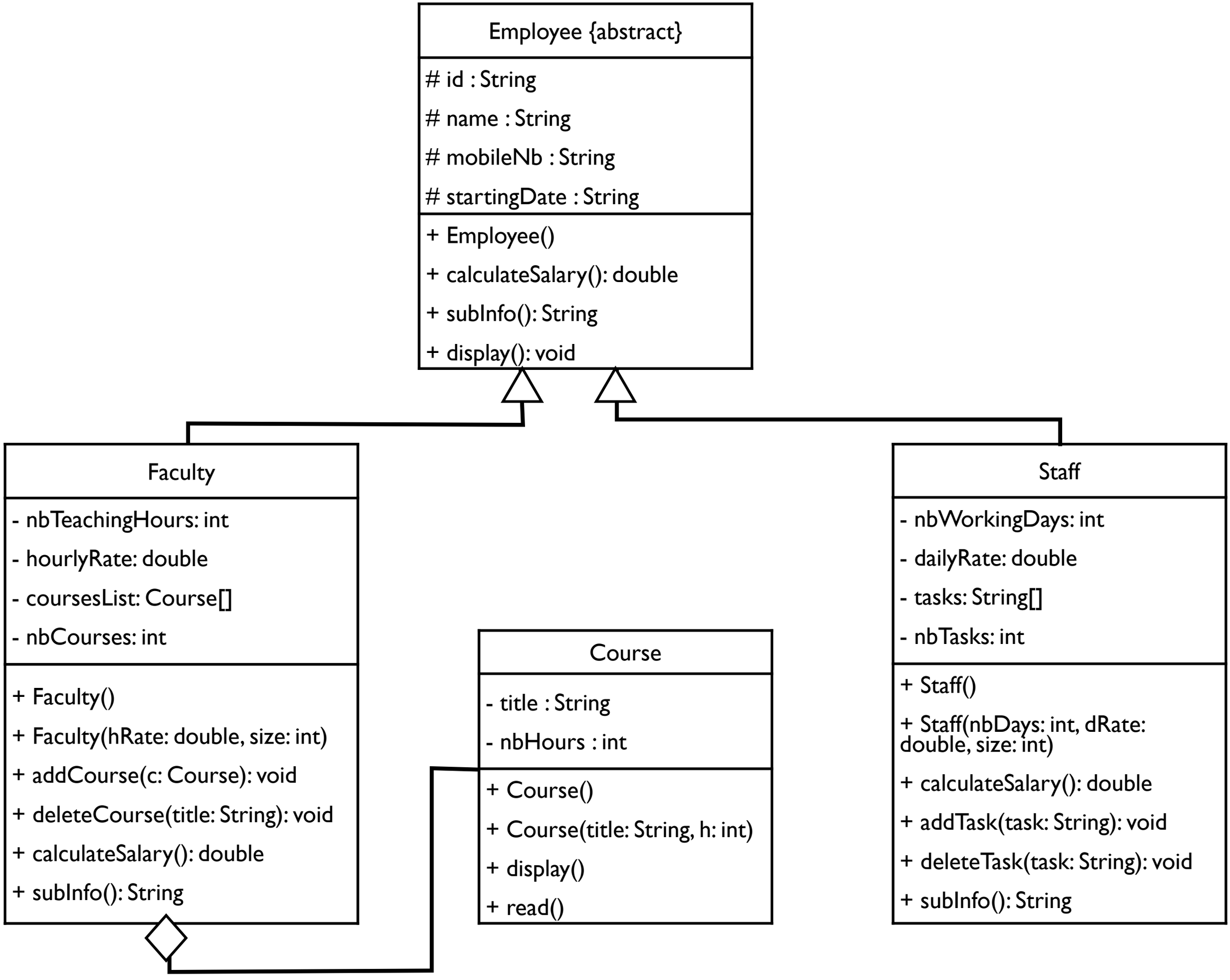
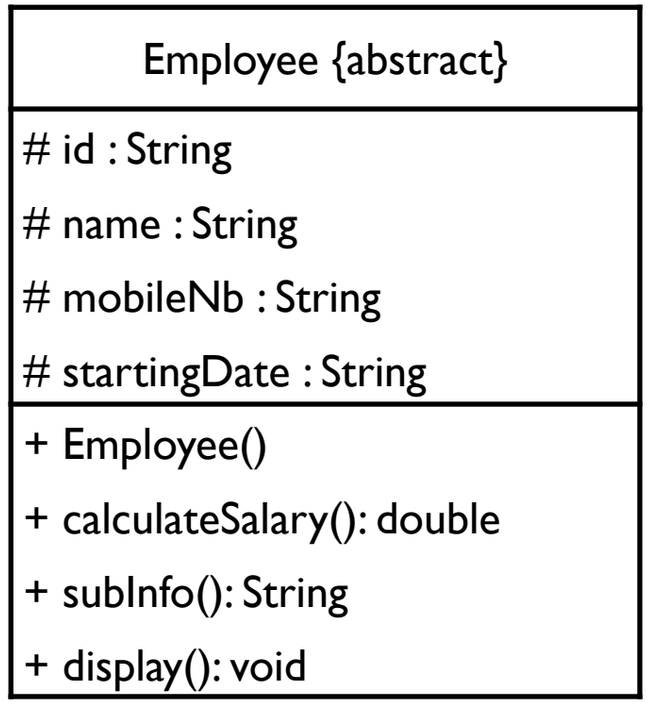
CSC 113
Tutorial 6

Inheritance & Abstract Classes

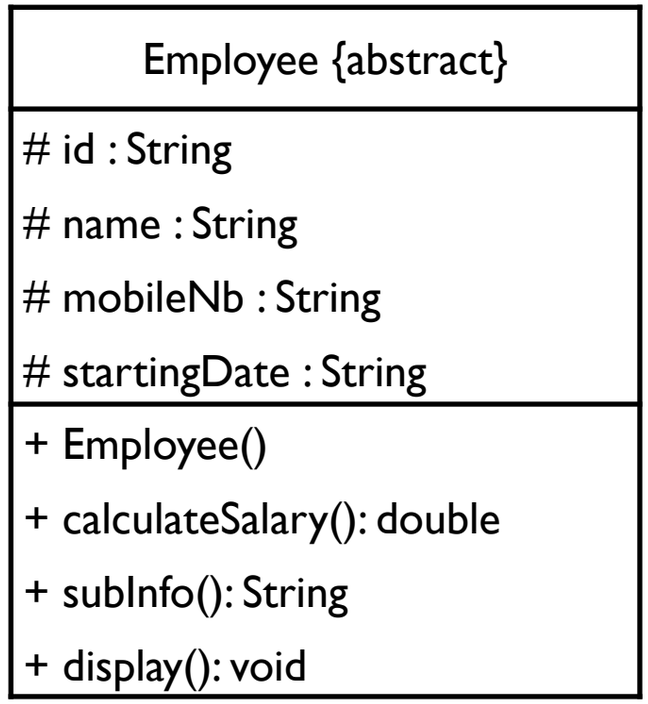

Employee {abstract}
id : String
name : String
mobileNb : String
startingDate : String
+ Employee()
+ calculateSalary(): double
+ subInfo(): String
+ display(): void
Faculty
- nbTeachingHours: int
- hourlyRate: double
- coursesList: Course[]
- nbCourses: int
+ Faculty()
+ Faculty(hRate: double, size: int)
+ addCourse(c: Course): void
+ deleteCourse(title: String): void
+ calculateSalary(): double
+ subInfo(): String
Course
- title : String
- nbHours : int
+ Course()
+ Course(title: String, h: int)
+ display()
+ read()
Staff
- nbWorkingDays: int
- dailyRate: double
- tasks: String[]
- nbTasks: int
+ Staff()
+ Staff(nbDays: int, dRate: double, size: int)
+ calculateSalary(): double
+ addTask(task: String): void
+ deleteTask(task: String): void
+ subInfo(): String

Abstract class and abstract methods

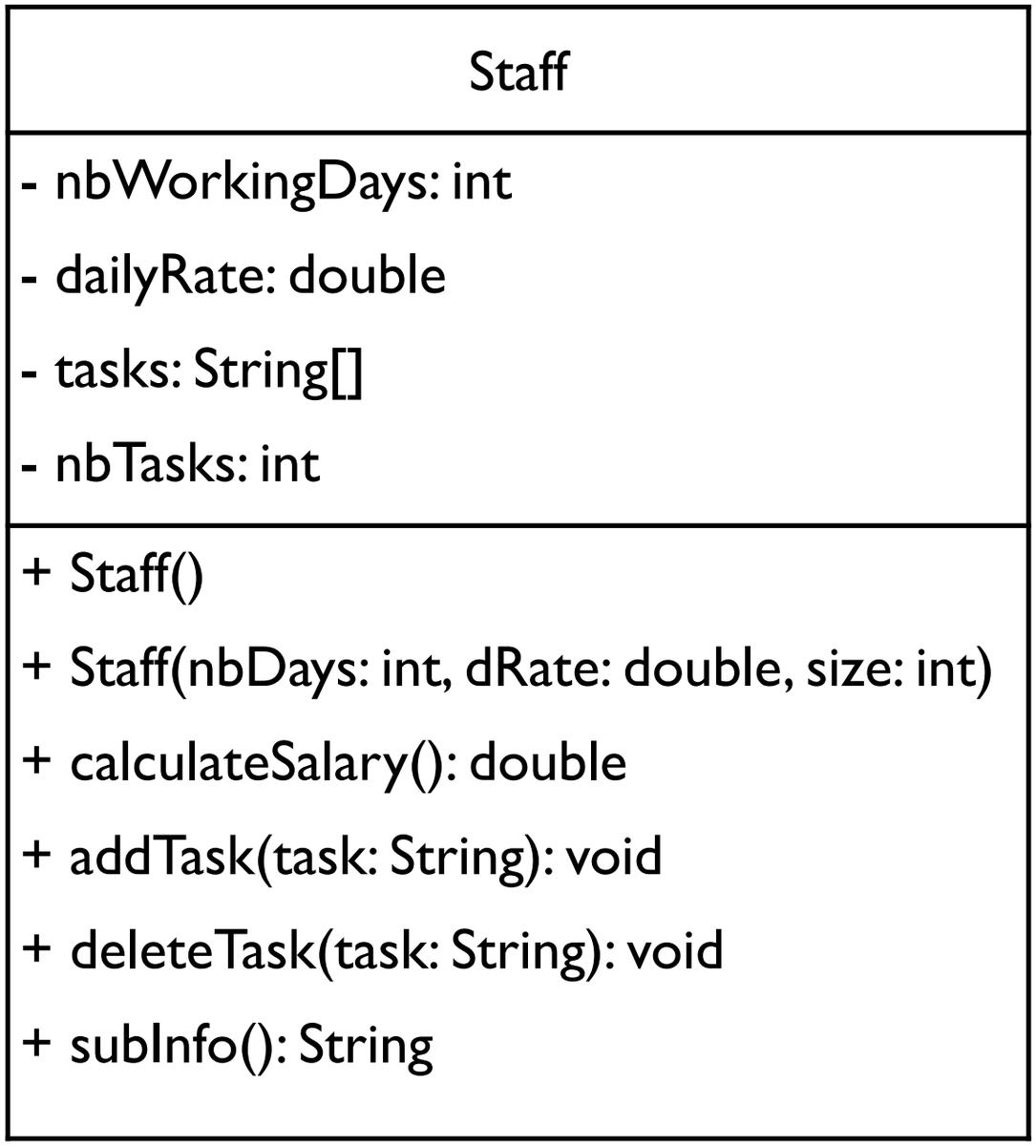



```
public abstract class Employee {

    protected String name;
    protected String id;
    protected String mobileNb;
    protected String startingDate;

    public Employee()
    {

    }

    public abstract double calculateSalary();

    public abstract String subInfo();
```

Final display calls subInfo

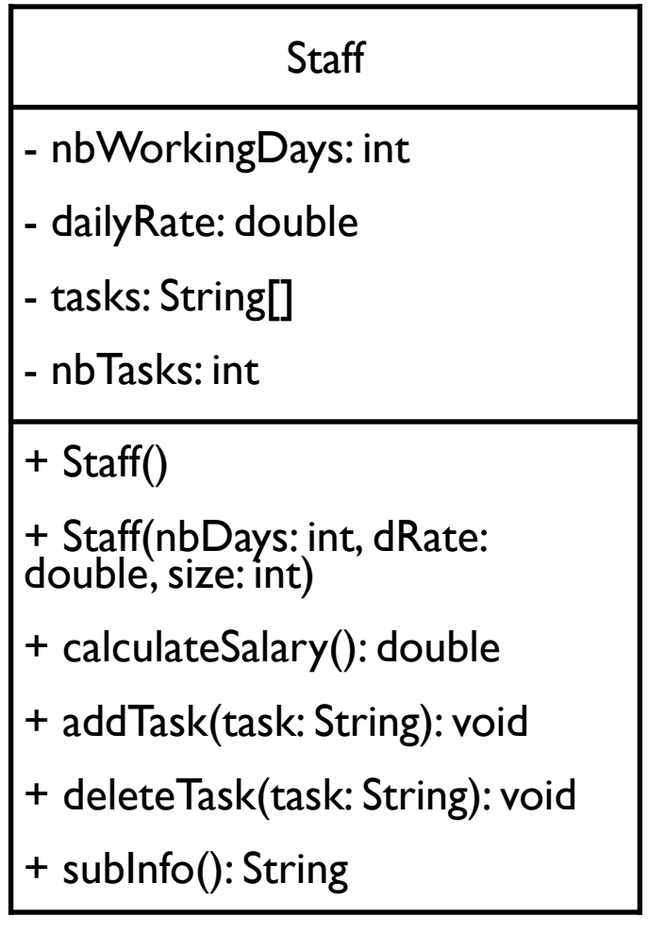



```
public final void display()
    {
        System.out.println("Name: "+name);
        System.out.println("ID: "+id);
        System.out.println("Mobile: "+mobileNb);
        System.out.println("Starting date: "+startingDate);
        System.out.println(subInfo());
}
```


Staff
- nbWorkingDays: int
- dailyRate: double
- tasks: String[]
- nbTasks: int
+ Staff()
+ Staff(nbDays: int, dRate: double, size: int)
+ calculateSalary(): double
+ addTask(task: String): void
+ deleteTask(task: String): void
+ subInfo(): String

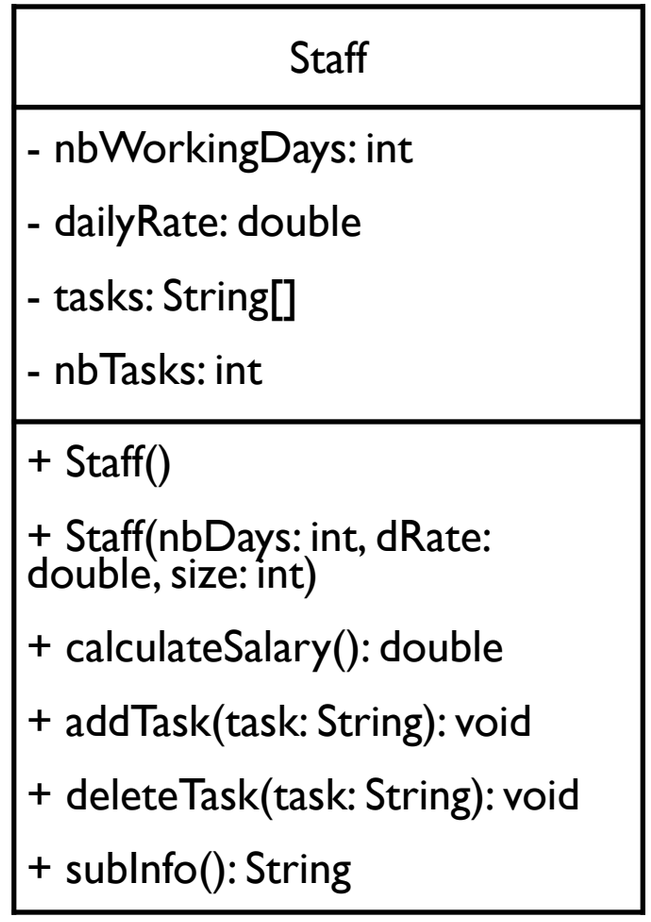
Declare variables and default constructor

```
public class Staff extends Employee{

    private int nbWorkingDays;
    private double dailyRate;
    private String[] tasks;
    private int nbTasks;

    public Staff()
    {

    }
```



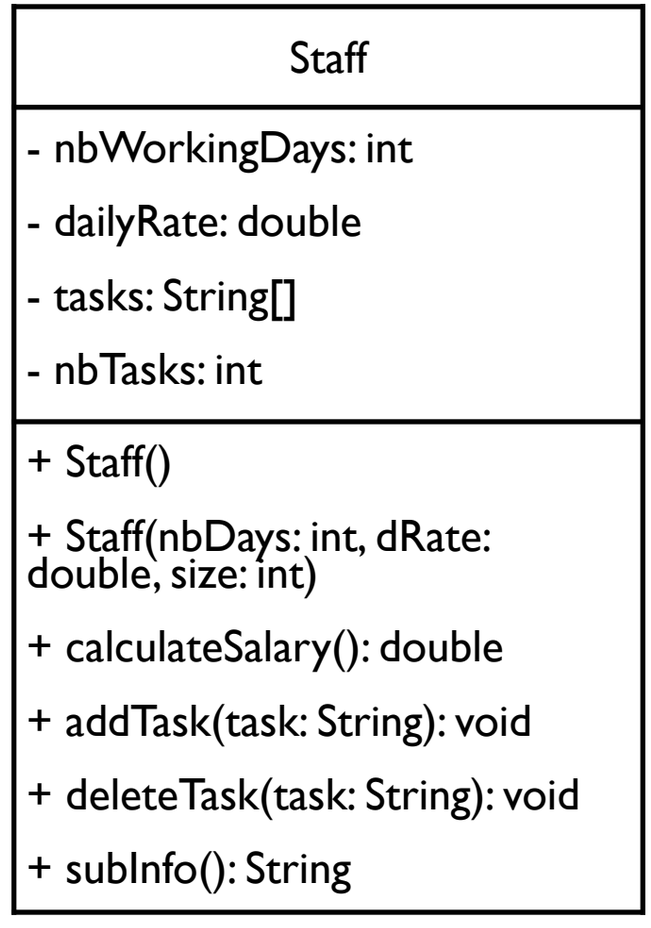
Copy Constructor

```
public Staff(int nbDays, double dRate, int size)
    {
        this.dailyRate = dRate;
        this.nbWorkingDays = nbDays;
        this.tasks = new String[size];

    }
```



Add task

```
public void addTask(String t)
    {
        if(nbTasks < tasks.length)
        {
            this.tasks[nbTasks++] = t;
            System.out.println("Add operation successful.");
            return;
        }
        System.out.println("Error: Courses list is full.");
}
```



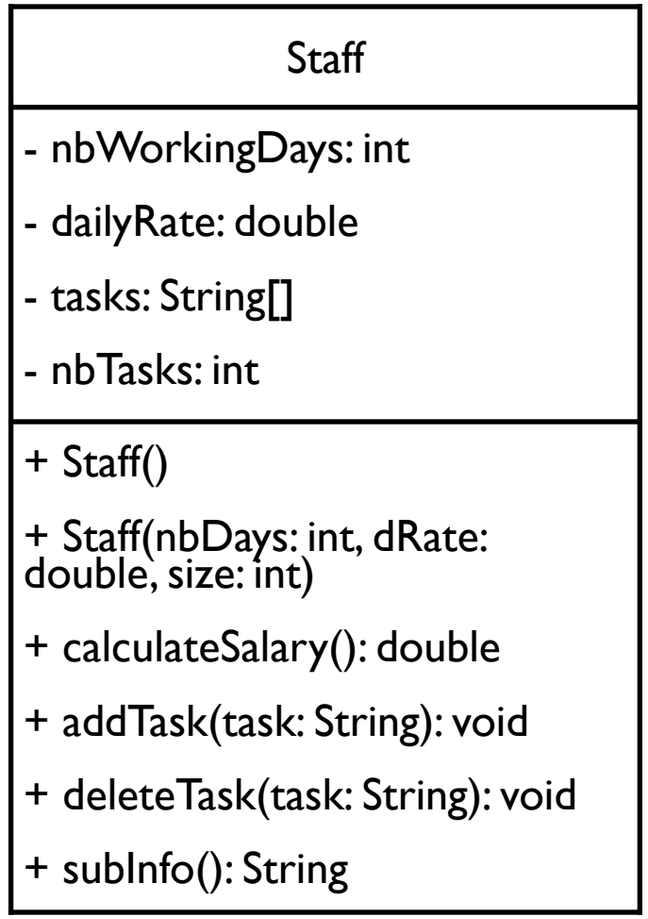
Delete

```
public void deleteTask(String task)
    {
        int i = search(task);

        if(i==-1)
        {
            System.out.println("Error: Course does not exist.");
            return;
        }

        tasks[i] = tasks[--nbTasks];
        tasks[nbTasks] = null;
        System.out.println("Delete operation successful.");
        return;
}
```

Staff
- nbWorkingDays: int - dailyRate: double - tasks: String[] - nbTasks: int
+ Staff() + Staff(nbDays: int, dRate: double, size: int) + calculateSalary(): double + addTask(task: String): void + deleteTask(task: String): void + subInfo(): String

Search is a private method for internal use only.

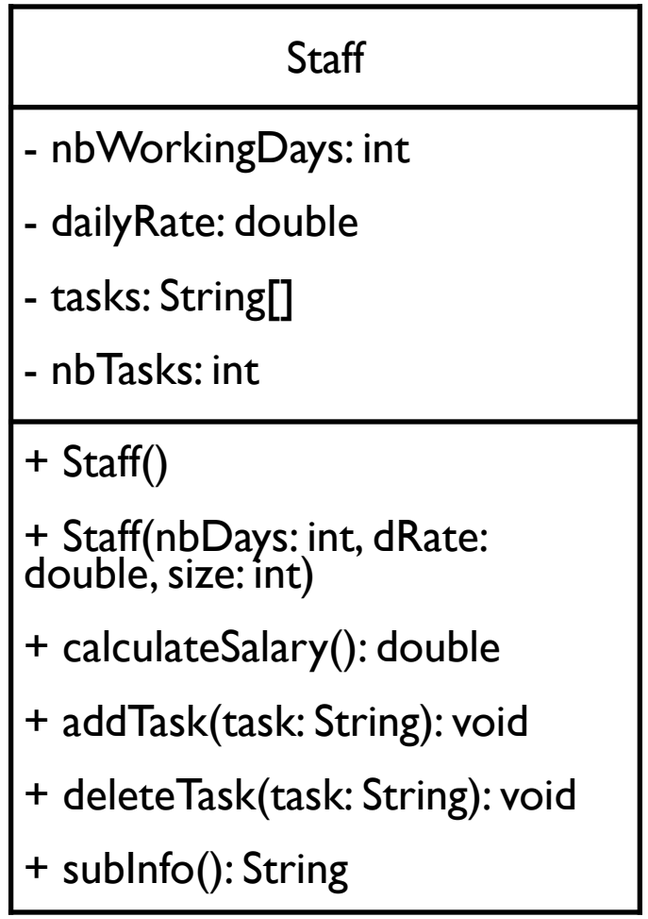
```
private int search(String title)
    {
        int i;

        for(i=0; i< tasks.length; i++)
        {
            if(tasks[i] != null)
                if(tasks[i].equals(title))
                    return i;
        }
        return -1;
}
```

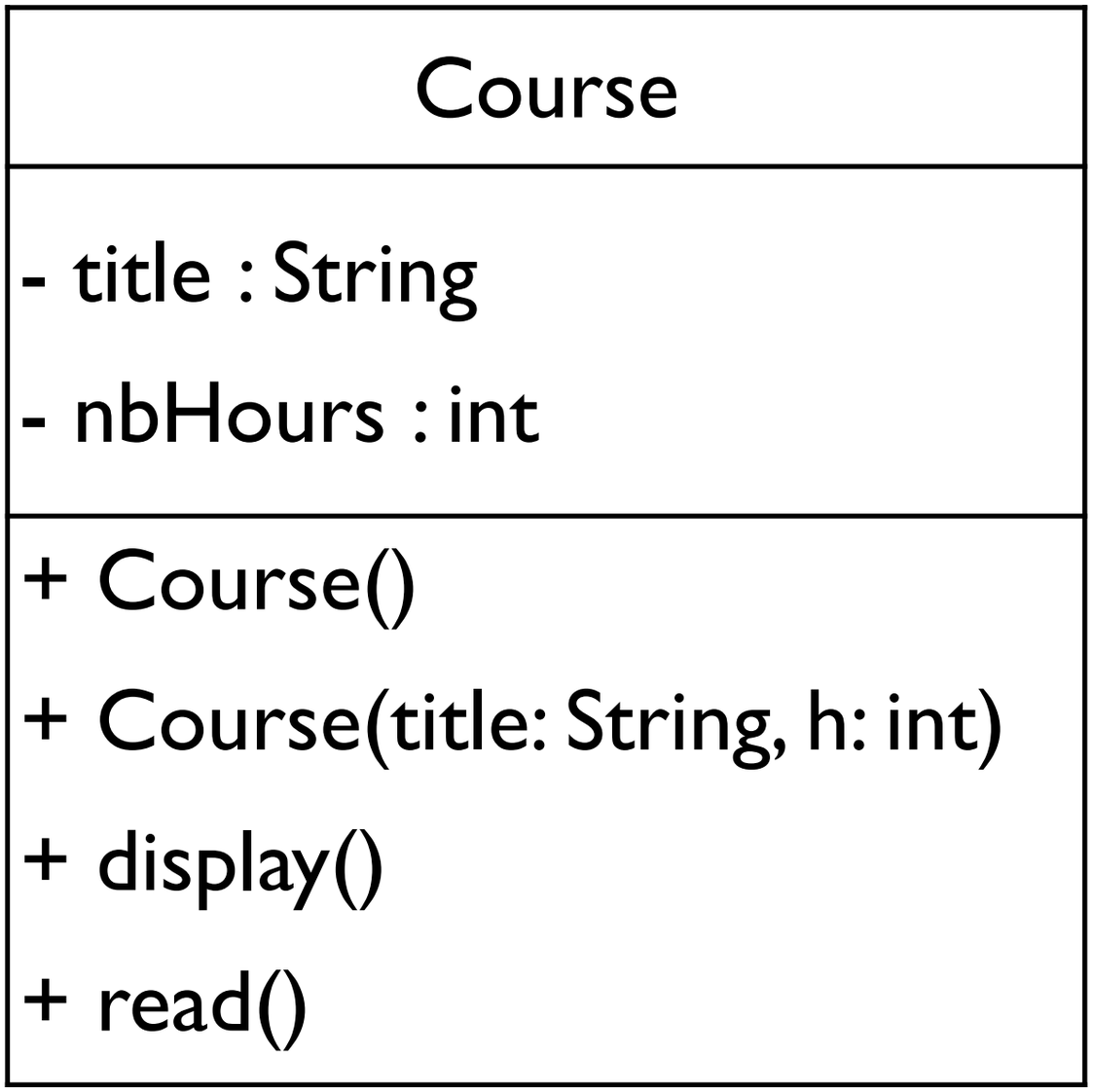


CalculateSalary and subInfo implementation

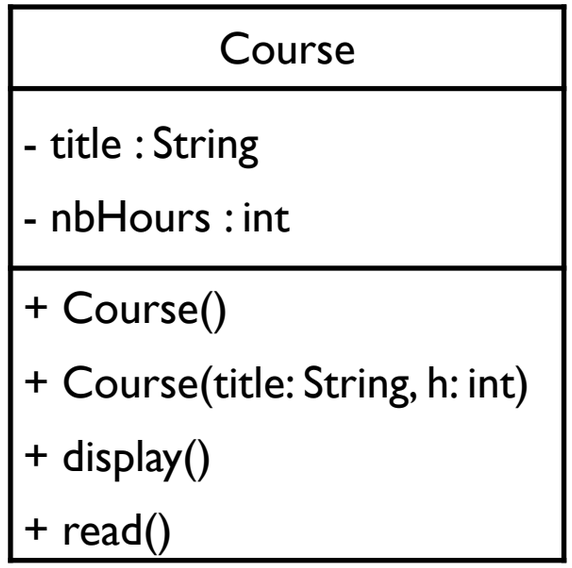
```
public double calculateSalary()
    {
        return nbWorkingDays * dailyRate;
    }
```

```
public String subInfo()
    {
        return "Working days : " + nbWorkingDays +
         "\nDaily rate: "+ dailyRate+
         "\n# of tasks: " + nbTasks;

    }
```



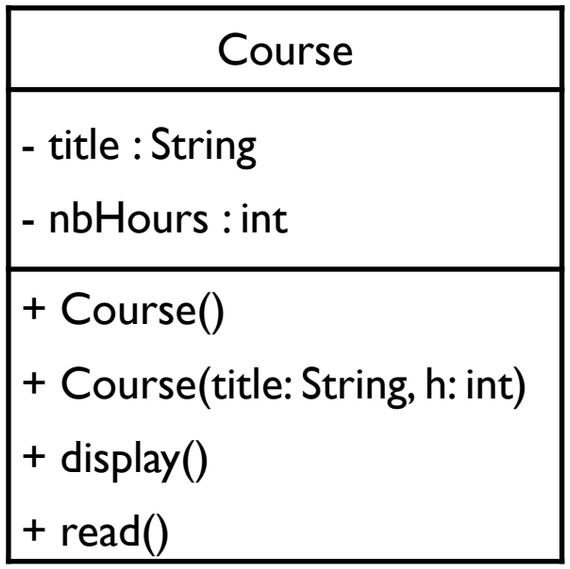

Course
- title : String
- nbHours : int
+ Course()
+ Course(title: String, h: int)
+ display()
+ read()

Declare variables and default constructor and copy constructor



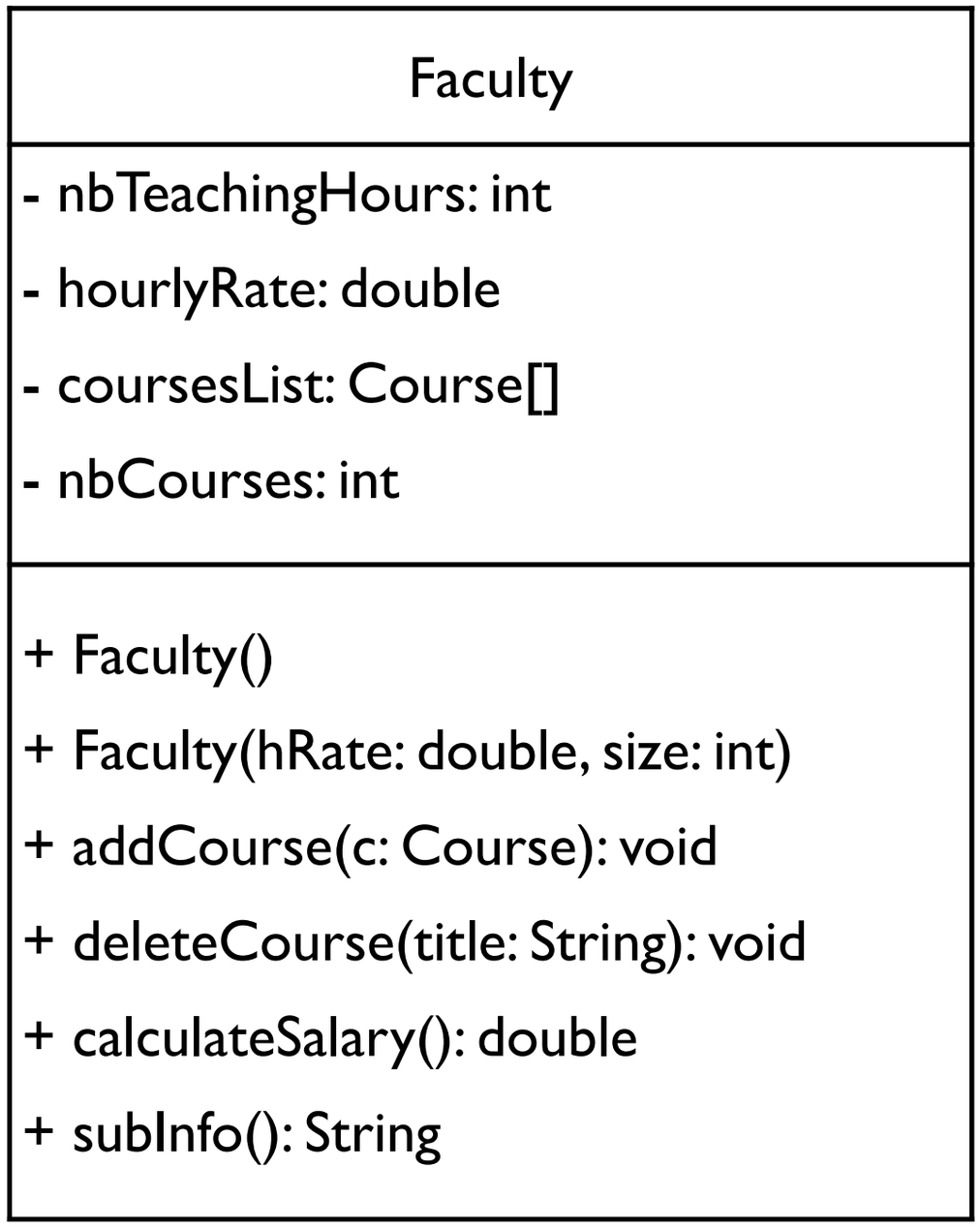

```
public class Course {
    private String title;
    private int nbHours;

    public Course()
    {

    }

    public Course(String title,int h)
    {
        this.title = title;
        nbHours = h;
    }
```

Display and read




```
public void display()
    {
        System.out.print("title: " + title + " Hours: "+nbHours);
    }
```

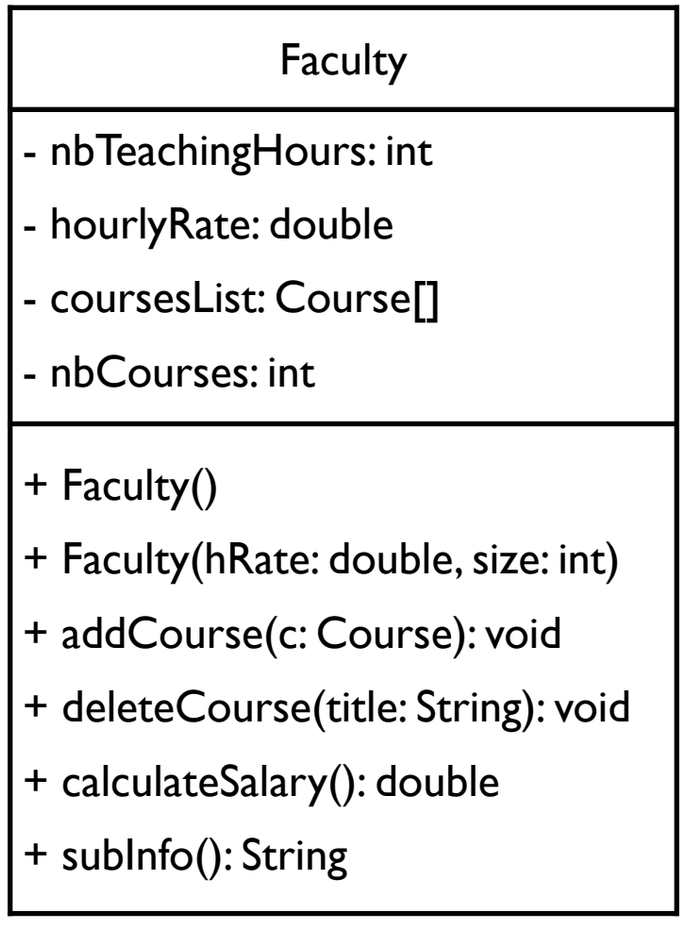
```
public void read()
    {
        Scanner s = new Scanner(System.in);
        System.out.println("Enter couse name: ");
        title = s.next();
        System.out.println("Enter number of hours: ");
        nbHours = s.nextInt();
    }
```


Faculty
- nbTeachingHours: int
- hourlyRate: double
- coursesList: Course[]
- nbCourses: int
+ Faculty()
+ Faculty(hRate: double, size: int)
+ addCourse(c: Course): void
+ deleteCourse(title: String): void
+ calculateSalary(): double
+ subInfo(): String

Declare variables and default constructor and copy constructor

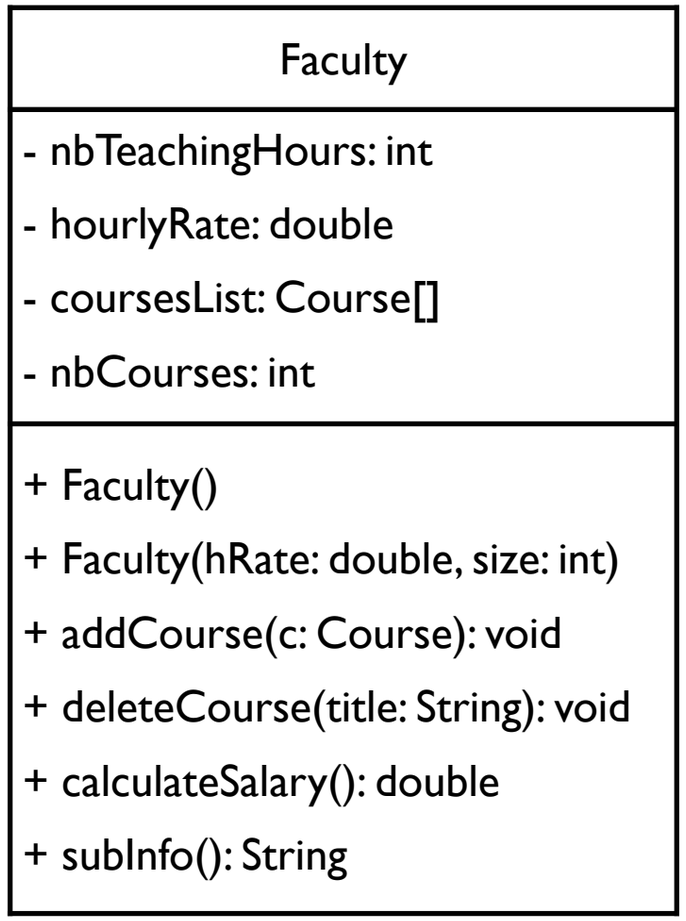
```
public class Faculty extends Employee{
    private int nbTeachingHours;
    private double hourlyRate;
    private Course[] coursesList;
    private int nbCourses;
```

```
    public Faculty()
    {

    }

    public Faculty(double hRate, int size)
    {
        this.hourlyRate = hRate;
        this.coursesList = new Course[size];
        nbTeachingHours = 0;
    }
```



addCourse should add the course and also add the hours.

```
public void addCourse(Course c)
    {
        if(nbCourses < coursesList.length)
        {
            coursesList[nbCourses++] = c;
            nbTeachingHours += c.getNbHours();
            System.out.println("Add operation successful.");
            return;
        }
        System.out.println("Error: Courses list is full.");
}
```



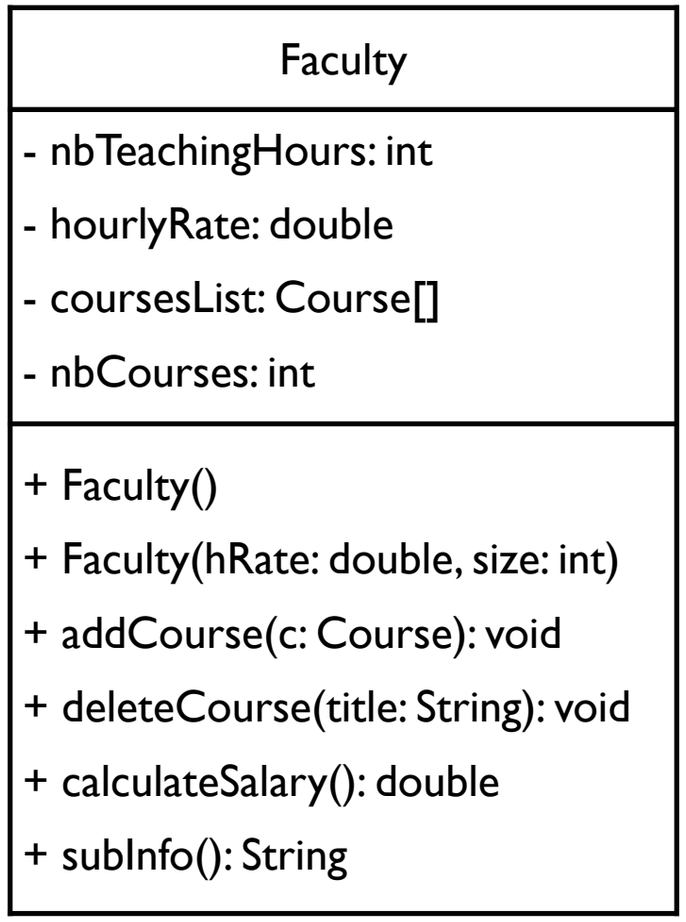
deleteCourse should delete the given course title and subtract the hours.

```
public void deleteCourse(String title)
    {
        int i = search(title);

        if(i==-1)
        {
            System.out.println("Error: Course does not exist.");
            return;
        }

        nbTeachingHours -= coursesList[i].getNbHours();
        coursesList[i] = coursesList[--nbCourses];
        coursesList[nbCourses] = null;
        System.out.println("Delete operation successful.");
        return;
}
```

Faculty
- nbTeachingHours: int - hourlyRate: double - coursesList: Course[] - nbCourses: int
+ Faculty() + Faculty(hRate: double, size: int) + addCourse(c: Course): void + deleteCourse(title: String): void + calculateSalary(): double + subInfo(): String

Search is a private method for internal use only.

```
private int search(String title)
    {
        int i;

        for(i=0; i< coursesList.length; i++)
        {
            if(coursesList[i] != null)
                if(coursesList[i].getTitle().equals(title))
                    return i;
        }
        return -1;
}
```



CalculateSalary and subInfo implementation

```
public double calculateSalary()
    {
        return nbTeachingHours * hourlyRate;
    }
```

```
public String subInfo()
    {
        return "Teaching Hours: " + nbTeachingHours +
         "\nHourly rate: "+ hourlyRate+
         "\n# of courses: " + nbCourses;
    }
```

Faculty
- nbTeachingHours: int - hourlyRate: double - coursesList: Course[] - nbCourses: int
+ Faculty() + Faculty(hRate: double, size: int) + addCourse(c: Course): void + deleteCourse(title: String): void + calculateSalary(): double + subInfo(): String