Софийски Университет "Климент Охридски"
Факултет по Математика и Информатика

Финален изпит No. 1

Курс: **Приложно Обектно Ориентирано Програмиране с Java, част 1**
Преподавател: д-р. Е. Кръстев **Студент** :
Дата: **Време за работа: 120 min**

Инструкции: Изпълнете следното задание за обектно ориентирано програмиране и предайте в своя акаунт в Мудъл пълния набор от файлове на IntelliJ проекта, създаден за решаване на програмата. Пълен набор от точки се присъжда за пълно и коректно решение на всички подзадачи.

Оценки:

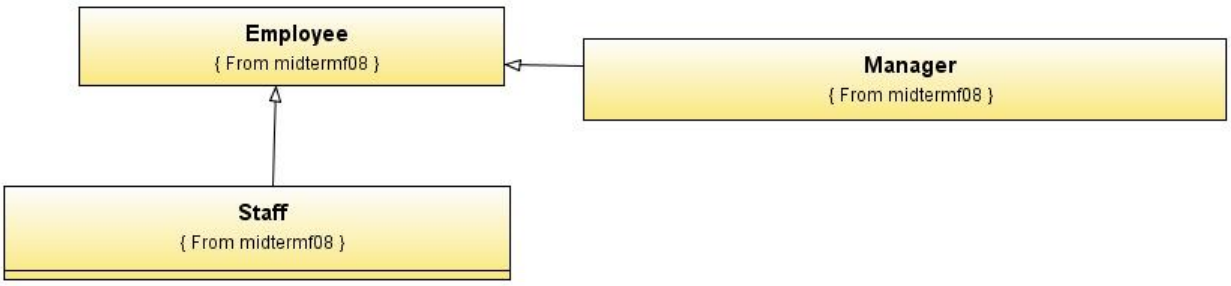
2	**от 0 до 54 точки**
3	**от 55 до 64 точки**
4	**от 65 до 74 точки**
5	**от 75 до 84 точки**
6	**от 85 до 100 точки**

Задача 1 (100 точки)

Приложете следните принципи на Обектно ориентираното програмиране на **Java**:

- **hiding of information**
- **software reuse**
- **inheritance**
- **polymorphism**

при намиране решението на следната **задача. Напишете *конзолно приложение*** на **Java** посредством, което **менажерът** на даден отдел **назначава личния състав** за своя отдел. При това се приема, че менажерът (`Manager`) и членовете (`Staff`) на личният състав в отдела му са служители (`Employee`) **според дадената UML диаграма** по- долу. За **описание на списъка от членове на личния състав** ще използваме `class ArrayList` от package `java.util`, който позволява **добавяне на обекти** (`Staff`) **в списък** с **метода** си `add(Staff staffmember)` и **четене** на елемент `Staff` от списъка с **пореден номер** `k` посредством **метода** си `get(k)`. **Празен списък** се създава с **конструктора по подразбиране** на `class ArrayList`



При реализацията на **събитието по назначаването** на член от състава се използва следното **бизнес правило**:

- задава се принадлежност на служителя към отдела на менажера,
- фиксира се новата му работна заплата и
- съответния служител се добавя в списъка на личния състав на отдела.

За **по- голяма гъвкавост при реализацията на бизнес правилата** използвайте ***callback*** и ***closure*** в **конзолно приложение** за **обработка на това събитие** като **извършете следните действия**

1. **Дефинирайте *class Employee***, който **има** *име* ***name*** **и** *заплата* ***salary*****, а също има:**
   - **GET и SET методи за данните на този клас**. . По подразбиране ***name*** и ***salary*** имат стойности " No Name" и ***1000***.
   - **Конструктор за общо ползване**
   - метод
     public String toString()
     за този клас – връща ***String***, съдържащ текущите стойности на *име* ***name*** **и** *заплата* ***salary,*** **форматирана с два знака** след десетичната запетая

     Точки:8

2. **Дефинирайте *class Staff***, **който е *Employee*** и **има** текстово описание ***workAt*** на ***принадлежност на служител към отдел.*** Нека този клас **също има:**
   - **GET и SET методи за данните на този клас**. Когато ***workAt*** е ***null***, то ***workAt*** приема подразбираща се стойност „***Candidate***"
   - **Конструктор за общо ползване**
   - метод
     public String toString()
     за този клас – връща ***String*** , съдържащ текущите стойности на всичките *данни* на инстанцията от ***class Staff***

     Точки: 8

3. **Дефинирайте *class Manager***, **който наследява *Employee*** и **има** текстово описание на ***отдела manageDeptName*****,** управляван от този менажер**.** Нека този клас **също има:**

   GET и SET методи за данните на този клас. Когато ***manageDeptName*** е ***null*** се хвърля изключение ***java.security.InvalidParameterException***
   - **Конструктор за общо ползване**
   - метод
     public String toString()
     за този клас – връща ***String*** , съдържащ текущите стойности на всичките *данни* на инстанцията от ***class Manager***

     Точки: 8

4. Нека **отдел Личен състав** се представя от ***class HRdepartment***. Този отдел **има** менажер ***manager*** (***Manager*** тип) и **има** списък ***staff*** (***ArrayList*** тип), чиито елементи са **членовете *Staff* на личния състав** (*за работа с* ***ArrayList*** *виж* ***уводната част*** *на теста).* Нека този клас **също има:**
   - **GET и SET метод за данната *manager* на този клас** . Когато ***manager*** е ***null*** се хвърля изключение ***java.security.InvalidParameterException***
   - **Конструктор за общо ползване с единствен аргумент от тип** ***Manager*** *(при създаване на нов* ***HRdepartment*** ***да се отчете****, че списъка* ***staff*** *е празен* ***ArrayList****)*
   - метод
     public String toString()

за този клас – връща ***String*** , съдържащ текущите стойности на всичките *данни* на инстанцията от ***class HRdepartment***

Точки: 8

5. Нека **отдел Личен състав** да задава **бизнес правилата** при **назначаване на нови членове** на личния състав от своя менажер. За целта създайте **модел за обработка** на събитието ***staffAppoint***, чрез използване на ***callback***. Като начало **опишете обекта на събитието** с ***class StaffAppointEventArgs*** като приемете, че **при назначаване на нови служители** менажерът **изпраща** на **отдел Личен състав референция** към **обекта** от тип ***Staff, който е назначен заедно със*** **стойността на заплатата** (тип ***double***) ***, която е договорена при назначаването.***

Точки: 10

6. **Създайте** ***interface StaffAppointHandler***, който да **декларира метода за обработка** на **обекта** ***StaffAppointEventArgs*** на **събитието** ***StaffAppoint***. За определеност, **нека** този **метод се именува** ***addStaff***

Точки: 4

7. **Добавете референция** ***staffAppoint*** от тип ***StaffAppointHandler*** към **източника на събитието**,който е представен с ***class Manager*****. Добавете също в** ***class Manager*** **:**

   - **метод** ***addStaffAppointHandler*****,** позволяващ **инициализация** на **референцията** ***staffAppoint*****.**
   - **метод** ***void onStaffAppoint (Staff member, double newStaffMemberSalary)*** за **"*изстрелване*" на** събитието ***staffAppoint***

Точки: 14

8. **Създайте недостъпно** описание на **бизнес правилата** по назначаване на личен състав във **вътрешен клас** на ***class HRdepartment***. Нека този **вътрешен клас наследява** ***StaffAppointHandler*** и **има**:

   a) **име** на отдел ***appointDepartment***, за който се прилагат правилата. Когато ***appointDepartment*** е ***null*** се хвърля изключение ***java.security.InvalidParameterException*** (3 точки)

   b) **конструктор**, който инициализира ***appointDepartment*** (3 точки)

   c) **използва** обекта на събитието ***StaffAppointEventArgs,*** за да **реализира бизнес правилата по отношение на** ***Staff*** **обекта в** ***StaffAppointEventArgs***, посредством метода ***addStaff*** на ***interface StaffAppointHandler*** , а именно (5 точки):

      - **задава се принадлежност** на ***Staff*** **обекта** към отдела ***appointDepartment*** , за където е назначението му, а и за където важат бизнес правилата

- **<u>задава</u>** се нова **работна заплата** на ***Staff*** **обекта (***виж* ***void onStaffAppoint() в т.7)***
- съответния **служител се <u>добавя в списъка</u>** на личния състав ***staff*** (***ArrayList*** тип) на ***HRdepartment*** отдела.

d) Предефинира метода ***toString***() на клас ***Object*** във вътрешния клас, така че този метод да връща **резултата от изпълнението метода** ***toString***() **на външния клас,** към който е добавено **името** на отдел ***appointDepartment*** (5 точки)

Точки: 16

9. Напишете метод

public StaffAppointHandler getAppointHandler()

в ***class HRdepartment,*** който **създава обект от вътрешния клас**, посредством **текущата** стойност на ***manageDeptName*** от ***manager*** референцията в ***HRdepartment (менажерът на отдела е този, който назначава служителите в отдела си)***

Точки: 3

10. **Създайте** ***class AppointTest*** за тестване на модела в ***static main()*** **метода** на този клас:
    a) **Създайте обект** ***boss*** **от клас** ***Manager*** , **работещ** в отдел ***"Invoices"*** (***име*** *и* ***заплата*** *по избор на студента*)
    b) **Създайте масив** ***candidates*** **от два обекта** от клас ***Staff***- кандидати за назначаване (*име, отдел за назначаване и заплата по избор на студента*)
    c) **Изведете на стандартен изход данните** на **обекта от клас** ***Manager*** и ***Staff*** **кандидатите** за назначаване
    d) **Създайте обект от клас** ***HRdepartment***, където менажер е обектът ***boss***
    e) **Създайте референция** ***sah*** **към обект от вътрешния клас** на ***HRdepartment***, капсулиращ *бизнес правилата* за назначение **<u>и регистрирайте</u>** този обект с метода ***addStaffAppointHandler*** в ***boss*** инстанцията **за обработка на събитието** ***staffAppoint***
    f) **Нека менажерът** ***boss*** да назначи (метод ***onStaffAppoint***) на работа ***Staff*** обектите- елементи на масива ***candidates,*** съответно със **заплати** 1300 и 1500.
    g) **Накрая изведете на стандартен изход резултата от изпълнението на метода** ***sah.toString()*****, а също текстовото описание** на елементите на масива ***candidates*** със съответния им метод ***toString*****().**

Точки: 21