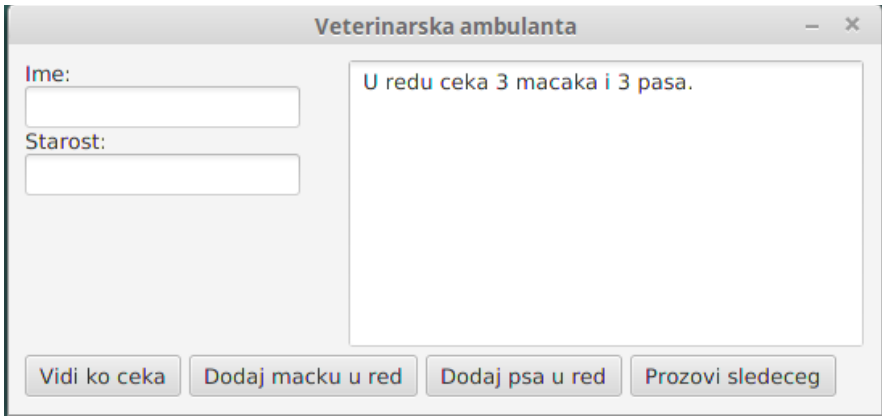
Објектно оријентисано програмирање, 03.07.2016.

Напомена: Направити на Desktop-у директоријум са именом облика **OOP_InicijaliAsistenta _Prezime_Ime** и изабрати га за *workspace* директоријум при покретању Eclipse. **Пројекат и пакет назвати исто тако!!!**
Обавезно сачекати да неко од дежурних асистената прекопира Ваш рад!
Назначен је део који треба урадити да би се освојио **праг**. Такође, кôд **не сме** имати синтаксних грешака.
Време за рад: **2,5 сата.**

Задатак (Ветеринарска амбуланта).

(***део за праг***) Написати апликацију чији графички кориснички интерфејс изгледа као на слици 1. **НИЈЕ ДОЗВОЉЕНА УПОТРЕБА SCENEBUILDER-A!** Онемогући ручни унос текста у текст-области. Клик на дугме „***Види ко чека***“ треба да омогући учитавање података из улазне датотеке *cekaonica.txt* (пример датотеке наведен поред слике 1), а затим и приказивање информације о броју паса и мачака који чекају у реду на преглед (слика 1).

Сваки ред датотеке садржи информацију о врсти љубимца (*P* - пас, *M* - мачка), имену љубимца, старости љубимца израженој у месецима (цео број) и идентификацију (ниска). Сматрати да је датотека у исправном формату. Уколико датотека не садржи никакав садржај, у текст-области приказати поруку „*Нема љубимаца у чекаоници*“.



Слика 1

```
P, Maza, 19, bc248
M, Odi, 5, z13yy
P, Dzeki, 26, acc02
P, Astra, 8, ghyzi
M, Feri, 37, z3y5x
M, Azrael, 59, 123456
```

Пример датотеке

Направити апстрактну класу ***Ljubimac*** која као чланице садржи име љубимца, старост љубимца и идентификацију, конструктор који прихвата само име и старост, конструктор који прихвата сва три податка, *get*-методе и одговарајући *toString()* метод. Класа садржи још и декларацију апстрактног метода ***String generisiIdentifikaciju();***

Из класе *Ljubimac* извести класе ***Macka*** и ***Pas***. Обе класе, поред неопходних конструктора, садрже и *toString()* метод у ком се у угластим заградама, пре свих података, наводи врста животиње (видети слику 2).

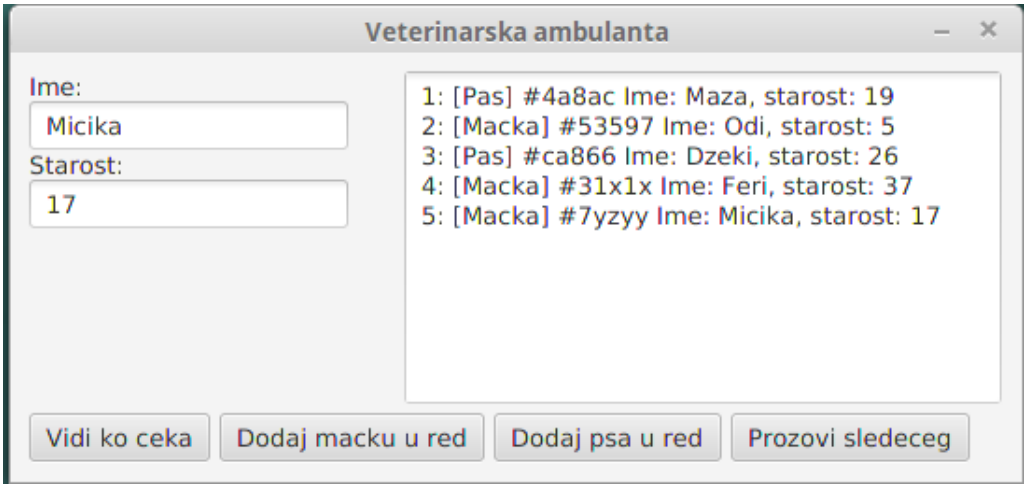
Затим написати класу ***Red***. Класа као чланицу садржи двоструко-повезану листу објеката класе *Ljubimac*. Направити конструктор који инстанцира ову листу. Класа још треба да садржи имплементације метода ***void dodajURed(Ljubimac ljubimac)*** и ***Ljubimac prozoviSledeceg().***

Метод ***dodajURed()*** додаје кућног љубимца на крај реда за чекање, док метод ***prozoviSledeceg()*** враћа и уклања првог љубимца који се налази у реду. Уколико ниједан љубимац не чека у реду, метод треба да врати ***null***.

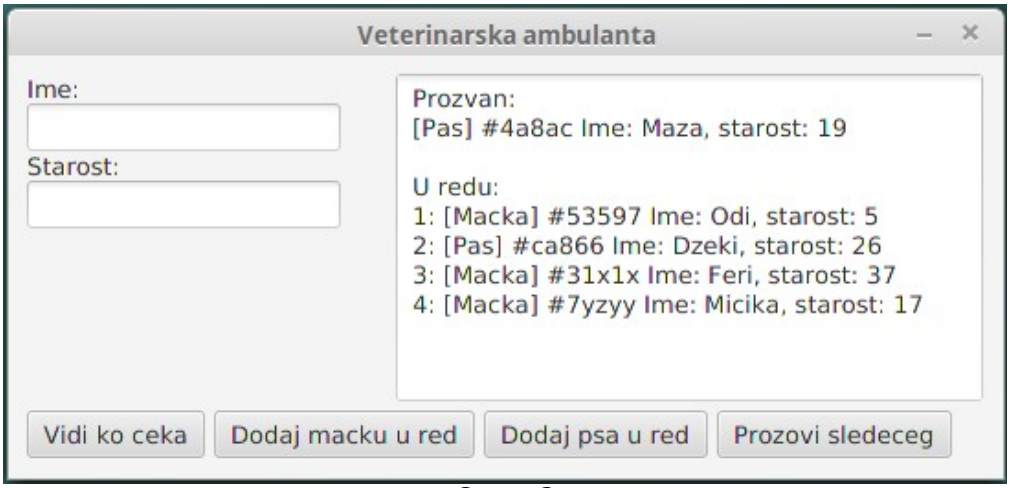
Поред дела за праг, након учитавања садржаја датотеке, у ред додати љубимце чији су подаци наведени у датотеци. *Љубимац се додаје у ред ако и само ако му је идентификација исправна.* У обе класе имплементирати статички метод ***boolean validnaIdentifikacija(String id)*** који проверава валидност идентификације. Идентификација је исправна уколико садржи тачно пет карактера, за пса из скупа вредности *{a, b, c, 2, 4, 6, 8}*, а за мачку из скупа вредности *{x, y, z, 1, 3, 5, 7, 9}*. Имплементирати још и одговарајући *toString()* метод који исписује ред у облику датом на слици 2.

Притискањем дугмета „***Додај мачку у ред***“, прави се објекат класе *Macka* са подацима наведеним у текстуалним пољима, и додаје се на крај реда за чекање. Аналогно, дугме „***Додај пса у ред***“ додаје пса на крај реда. Претпоставити да ће се подаци увек исправно уносити. Приликом додавања новог љубимца у ред, доделити му идентификацију позивом конкретне имплементације метода ***generisiIdentifikaciju()*** који креира ниску која садржи пет карактера из дозвољеног скупа вредности. Након додавања новог љубимца у ред, приказује се цео садржај реда. Поред сваког љубимца потребно је навести и његов редни број (слика 2).

Притискањем дугмета „***Прозови следећег***“, уклања се љубимац који је први у реду за чекање и приказују се његови подаци. Поред љубимца који више не чека, приказати и нови садржај реда (слика 3).



Слика 2



Слика 3