|  |
| --- |
| ***Teme:***   * + ***Raba objekto, metode***     - ***Konstruktorji***     - ***Nastavljalci, vračalci***     - ***Diagrami primerov rabe – implementacija primera rabe***     - ***Diagram aktivnosti – implementacija diagrama aktivnosti***     - ***Razredni diagram - realizacija kode iz opisa***     - ***Sekvenčni diagram*** |

|  |
| --- |
| **Splošna navodila:**  **Ulomek je urejen par(števec,imenovalec), pri čemer je v naši aplikaciji vedno predstavljen v svoji okrajšani obliki. V vaji implementiramo osnovne operacije nad ulomki.**  **Pomožni vir za tiste, ki se snovi ne spomnite najbolje:**  **Ulomek, Wikipedija,** [**http://sl.wikipedia.org/wiki/Ulomek**](http://sl.wikipedia.org/wiki/Ulomek)**, maj 2015**  **Specifikacija:**  **Želeli bi spisati program, ki izvede demonstracijo osnovnih operacij nad ulomki. Način rabe podaja naslednji digram (use-case, rabe):**    **(demo vključuje demonstracijo vseh štirih metod)**  **Struktura ulomka je predpisana z diagramom spodaj (class, razredni):**    **pri implementaciji morate upoštevati vse elemente diagrama, vključno z dostopnostnimi kvalifikatorji lastnosti in metod.**  **Za razumevanje delovanja je popisana še aktivnost (activity) v postopku seštevanja ulomkov:**      **in pa del postopka pri množenju, ki omogoča zadostiti potrebi po tem, da je ulomek vedno zapisan v normalizirani (okrajšani) obliki (Sequence, sekvenčni diagram):**    **opomba:**  **znotraj definicije razreda se nahaja metoda public String toString(). Ta metoda je posebna, vedno je prisotna v objektu. Vrača predstavitev objekta v obliki niza. Dejansko, če objekt skušate izpisati z metodo System.out.println(objekt), se uporabi ta metoda. V našem primeru v telo dodate samo vrstico kode System.out.println("{"+stevec+","+imenovalec+"}"); ki ulomek enostavno vizualizira kot {4,3} kar predstavlja ulomek 4/3** |

**Naloga 1 [[1]](#endnote-1)**

Realizirajte zastavljeno. Privzeti konstruktor naj generira ulomek {0,1}, ostale ulomke/dvočlenike generirate z poljubnimi dvemi (hm, celimi) števili iz intervala od [-1000 ,1000].

**Naloga 2**

Razmislite: ali je smiselno da sta števec in imenovalec neceli vrednosti ? Popravite lastnosti in metode, če smiselnost najdete, sicer ostanite pri realizaciji naloge 1.

**Naloga 3[[2]](#endnote-2)**

Popravite vizualizacije ulomkov, da se bodo izrisovali, kot ste tega navajeni. V splošnem naj bi košček demonstracije postopka seštevanja bil videti kot:

Uporabite JavaFX. V razred dodajte metodo, ki bo ustrezno vizualizira posamezen ulomek/dvočlenik, …

1. V bistvu realizirate ali nalogo 1 ali nalogo 2 v odvisnosti od tega, kako se boste odločili glede na besedilo naloge 2. [↑](#endnote-ref-1)
2. Neobvezna naloga [↑](#endnote-ref-2)