

Испит траје 3 сата.

| | | | | | | | | | | | | |
|-----------------------|---|---|---|---|---|---|---|---|---|----|----|--------|
| Задатак | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 1 | Укупно |
| Максималан број поена | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 70 | 120 |

ТЕОРИЈСКИ ДЕО

...

ПРАКТИЧНИ ДЕО

- Група ловаца на благо је дошла до информација о локацији скривеног блага! Направити програм који симулира њихову потрагу. Слика 1 илуструје како програм треба да изгледа приликом покретања. У наставку следи опис рада програма.
 - Графички кориснички интерфејс имплементирати тако да изгледа као интерфејс приказан на слици 1. Елементе прозора распоредити тако да се приликом промене његове величине компоненте аутоматски померају да задрже приказани распоред. Поље “Трагачи” (лево) је типа `QListWidget`, а поље “Мапа” (десно) је типа `QTableWidget`. Поље “Брзина игре” је типа `QDoubleSpinBox` и има опсег од 1.0 до 5.0, при чему се вредности инкрементирају у корацима од 0.1.
 - Сваког ловца потребно је представити као објекат класе `TreasureHunter`. Објекти ове класе се састоје од наредних атрибута: (1) име ловца као ниска, (2) брзина ловца као неозначен цео број, (3) број пронађених дуката као неозначен цео број, (4) пар неозначених целобројних вредности које представљају његову тренутну позицију на мапи (ред и колону). Обезбедити наредни јавни интерфејс ове класе:
 - Метод `fromQVariant` десеријализује податке о једном ловцу на благо из прослеђеног аргумента (`QVariant`). Аргумент садржи податке о свим атрибутима класе осим о иницијалном броју дуката, који је потребно поставити на 0. Подаци о реду и колони представљају иницијалну позицију ловца.
 - Метод `toString` израчунава текст са подацима о ловцу на благо у формату који је приказан на слици 2.

Дозвољено је додавање произвољних потребних поља и метода.
 - Мапу је потребно представити као објекат класе `TreasureMap`. Објекат ове класе се састоји од матрице енумератора `Tile` са могућим вредностима `Clear` (представља чисто тло), `Wall` (представља зид) и `Treasure` (представља гомилу дуката). Обезбедити наредни јавни интерфејс ове класе:
 - Метод `fromQVariant` десеријализује податке о мапи из прослеђеног аргумента (`QVariant`). Аргумент садржи матрицу целобројних вредности које је потребно конвертовати у одговарајуће вредности енумератора (0 представља чисто тло, 1 представља зид, а 2 представља дукате).

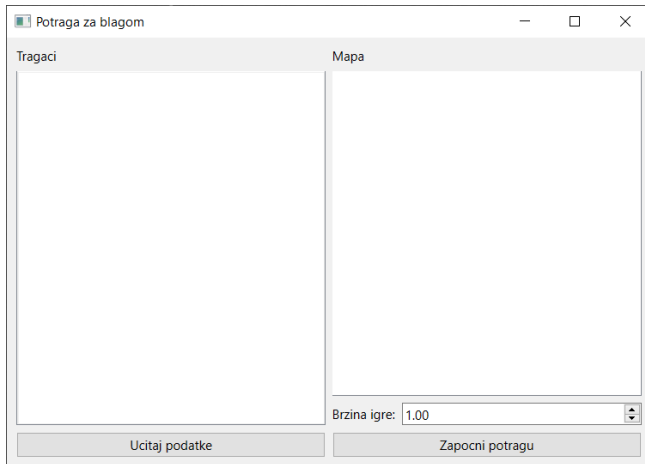
Дозвољено је додавање произвољних потребних поља и метода.
 - Кликом на дугме “Учитај податке”:
 - Уз помоћ дијалога за избор датотеке, учитати податке о ловцима на благо и мапи. Датотека је записана у JSON формату (видети пример датотеке `treasure_hunt.json`, у поставкама; број ловаца, као и величина мапе, се могу разликовати). Податке учитати као објекте класа `TreasureHunter` и `TreasureMap`.
 - Приказати ловце у `QListWidget` контроли “Трагачи” као на слици 2.
 - Приказати мапу у `QTableWidget` контроли “Мапа” као на слици 2, при чему празно поље представља чистину, поље “#” представља зид, а поље “@” представља дукате. Ловци су на мапи представљени првим карактером свог имена.

Корисник може више пута изнова учитавати податке.
 - Кликом на дугме “Започни потрагу”:
 - Уколико су претходно учитани подаци о ловцима на благо и мапи, започиње се са потрагом. У супротном, кликом на дугме се не дешава ништа.
 - Онемогућити употребу контрола “Учитај податке”, “Започни потрагу” и “Брзина игре”.
 - Потрага се спроводи тако што се за сваког ловца на благо покреће по нит.
 - Након покретања, свака од нити понавља наредни низ корака, све док се на мапи и даље налази бар једна гомила дуката:
 - Неки ловци су бржи од других, па је нит потребно успавати на $\frac{100}{BrzinaLovca \times BrzinaIgre} \times 1000$ милисекунди.
 - Ловац бира насумично слободно поље (на којем није зид или неки други ловац) поред њега (лево, десно, горе или доле), те прелази на њега. Уколико у неком кораку не постоји такво слободно поље, прескочити тај корак. Кретање ловца приказати на мапи.

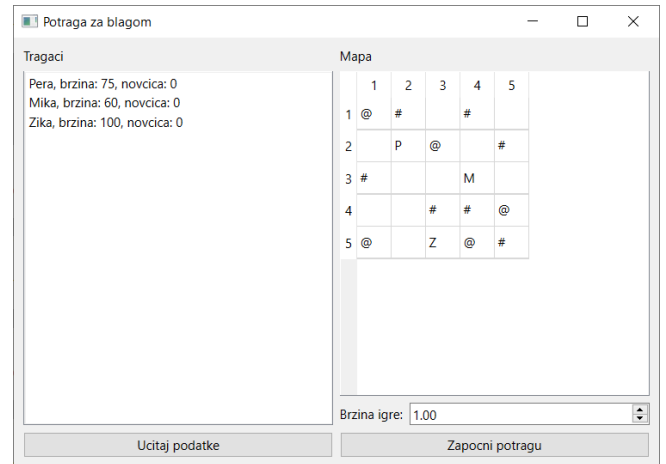
- У случају да је по преласку на ново поље ловац стао на гомилу дуката, том ловцу је потребно додати насумичан број од 1 до 5 дуката и уклонити ту гомилу са мапе.

Све нити морају да конкурентно оперишу над истим подацима.

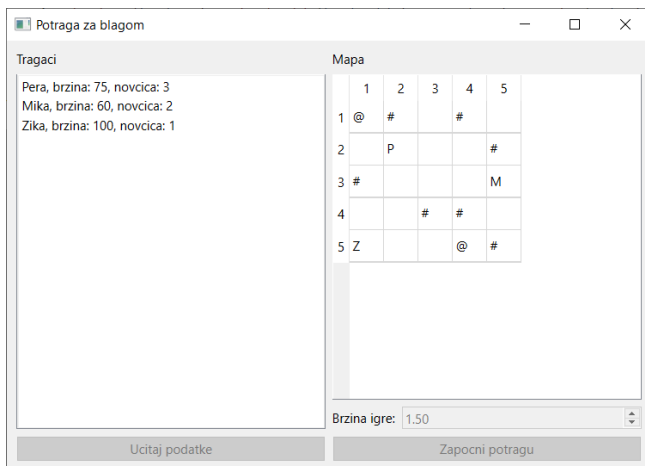
- (е) По свакој извршеној акцији ловаца, главна нит треба да ажурира поља “Трагачи” и “Мапа”, на начин на који је то приказано на слици 3. По завршетку потраге за благом (сви дукати су пронађени и уклоњени са мапе), потребно је омогућити све претходно искључене контроле, као и приказати информације о ловцу који је пронашао највећи број дуката, као што је приказано на слици 4. У случају да је више таквих ловаца, изабрати произвољног.
- (ж) Осигурати се да не долази до проблема у конкурентном окружењу и водити рачуна о раду са динамичким ресурсима.



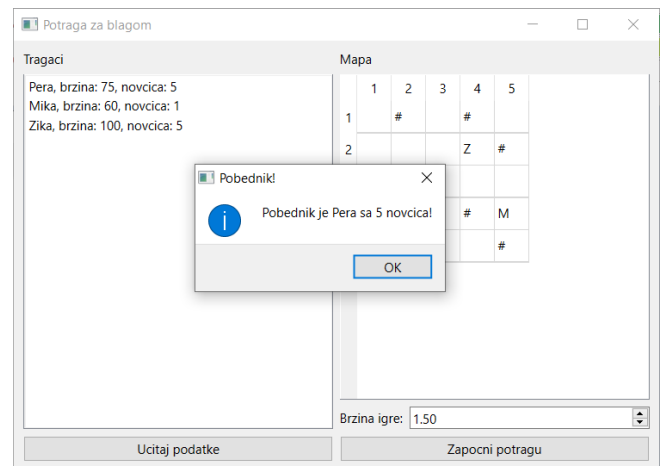
Слика 1: Приказ графичког интерфејса.



Слика 2: Приказ након учитавања података.



Слика 3: Приказ у току рада нити.



Слика 4: Приказ по завршетку рада нити.