ESAME DI FONDAMENTI DI INFORMATICA T-2 del 15/01/2013
Proff. E. Denti – G. Zannoni
Tempo a disposizione: 4 ore MAX

NB: il candidato troverà nell'archivio ZIP scaricato da Esamix anche il software "Start Kit"

NOME PROGETTO ECLIPSE: CognomeNome-matricola (es. RossiMario-0000123456)

L'istituto bancario *HappyBank* ha richiesto lo sviluppo di un'applicazione per gestire i suoi sportelli bancomat.

DESCRIZIONE DEL DOMINIO DEL PROBLEMA.

Lo sportello bancomat deve permettere due tipi di operazioni:

- il prelievo
- la visualizzazione del saldo del conto corrente.

Preliminarmente, lo sportello deve identificare l'utente tramite il numero della sua tessera bancomat (un codice numerico a 6 cifre), autenticandolo tramite codice PIN (un codice numerico a 5 cifre); successivamente, deve proporre all'utente un semplice menù a video per scegliere l'operazione e, nel caso del prelievo, l'ammontare da prelevare.

Internamente, la macchina può gestire **due tagli** di banconote: all'atto del caricamento iniziale dello sportello, si specifica quali tagli di banconote si usano e quante banconote sono caricate per ciascun taglio.

In fase di prelievo, la macchina sceglie quali e quante banconote erogare secondo il seguente algoritmo:

- per importi fino a 100 €, privilegia le banconote di piccolo taglio;
- per importi oltre i 100 €, privilegia le banconote di taglio maggiore.

In questa specifica, "privilegia" significa che la macchina tenta prioritariamente di costruire l'importo con quel taglio di banconota, usando l'altro taglio solo per completare la parte residua dell'importo; ove ciò non sia possibile, riduce progressivamente il numero di banconote di taglio prioritario, cercando di costruire l'importo in altro modo.
NB: gli esempi sottostanti illustrano nel dettaglio il senso di questa specifica in un caso concreto.

ALGORITMO: detti T1 e T2 i due tagli di banconote disponibili, con T1 prioritario, e detta S la somma richiesta, da comporre con N1 banconote di taglio T1 e N2 banconote di taglio T2, si tenta in primo luogo di costruirla con N1=S/T1 e N2=(S-N1*T1)/T2. Se così facendo l'importo è ottenibile in modo esatto (N1*T1+N2*T2=S), l'algoritmo termina; altrimenti, si riduce di 1 il numero di banconote di taglio prioritario (decrementando N1) e si ritenta il calcolo con un diverso N2, maggiore del precedente. L'algoritmo ha successo se prima o poi l'importo risulta costruibile, mentre termina con eccezione se, nonostante si sia decrementato N1 via via fino a zero, non è comunque risultato possibile sintetizzare in alcun modo l'importo richiesto.

Se l'importo chiesto dal cliente non è erogabile (perché non può essere costruito con quei tagli, o perché non ci sono più abbastanza banconote disponibili, o perché non ci sono abbastanza soldi sul conto, o pur essendoci soldi il prelievo richiesto supera comunque il prelievo massimo autorizzato), la macchina deve segnalare il problema visualizzando la dicitura *IMPORTO NON EROGABILE*.

Esempio (con bancomat caricato con banconote da 20 e 50 €)

Specifica: fino a 100 € privilegia le banconote da 20 €, mentre oltre i 100 € privilegia le banconote da 50 €

- 200 €: erogato con 4 banconote da 50 €;
- 160 €: erogato con 2 banconote da 50 € + 3 banconote da 20 € (perché il primo tentativo di usare 3 banconote da 50€ fallisce, non potendo esprimere i 10 € residui con le banconote da 20€);
- 140 €: erogato con 2 banconote da 50 € + 2 da 20 €;
- 130 €: erogato con 1 banconota da 50 € + 4 da 20 € (perché il primo tentativo di usare 2 banconote da 50€ fallisce, non potendo esprimere i 30 € residui con le banconote da 20€);
- 100 €: erogato con 5 banconote da 20 € (solo se non ce ne sono a sufficienza, si tenta con 2 da 50 €);
- 90 €: erogato con 2 banconote da 20 € + 1 da 50 €;
- 80 €: erogato con 4 banconote da 20 €;
- 30 €: importo non erogabile;
- 20 €: erogato con 1 banconota da 20 €

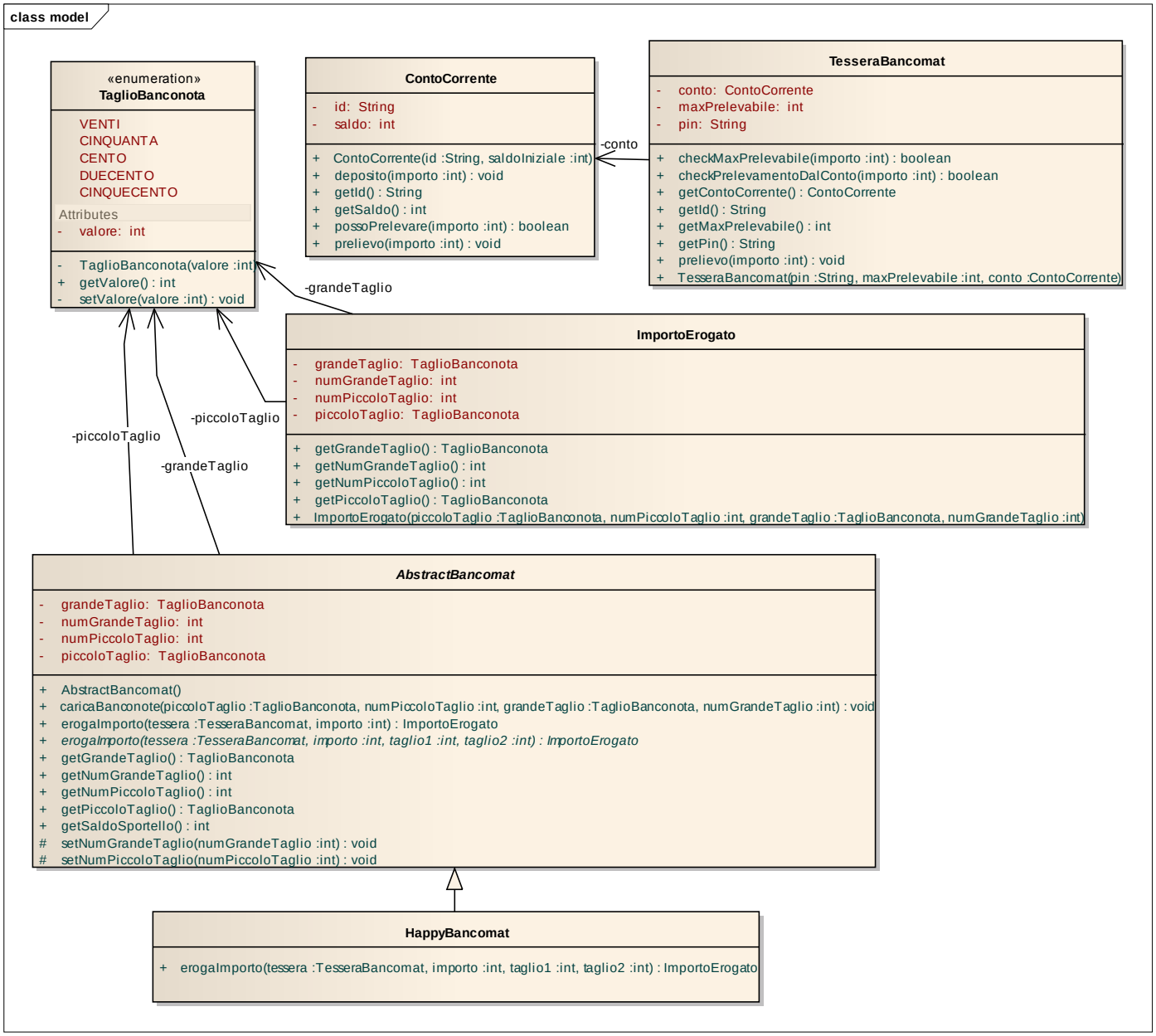
Il sistema si appoggia ai seguenti file, il cui formato è specificato più oltre:

1) BancomatConfiguration.txt contiene la configurazione iniziale dello sportello (quali banconote e quante);
2) TessereBancomat.txt contiene i dati delle tessere bancomat abilitate all'uso dello sportello, specificando per ciascuna il numero di conto corrente, il pin, e il prelievo massimo;
3) ContiCorrenti.txt mantiene l'elenco dei conti correnti, specificando per ogni cliente la disponibilità sul conto.

Parte 1 (punti: 18)

Dati (namespace happybank.model) *(punti: 10)*

Il modello dei dati deve essere organizzato secondo il diagramma UML più sotto riportato.



SEMANTICA:

a) la classe ***ContoCorrente*** (fornita nello start kit) rappresenta un conto corrente ed è caratterizzata da un identificatore univoco e un saldo; tale classe è dotata di utili operazioni quali ***possoPrelevare***() che indica se sia possibile effettuare il prelievo dell'importo passato come parametro e ***prelievo***() che effettua il prelievo dell'importo passato come parametro (se ***possoPrelevare***() restituisce false, invocare ***prelievo***() con tale importo provoca eccezione);

b) la classe ***TesseraBancomat*** (fornita nello start kit) rappresenta una tessera bancomat ed è caratterizzata da un conto corrente (un'istanza della classe ***ContoCorrente***), dal codice PIN e dall'importo massimo prelevabile; offre operazioni come ***checkPrelevamentoDalConto***() per verificare se sia possibile prelevare l'importo passato come parametro dal conto su cui si appoggia la tessera, ***checkMaxPrelevabile***() per verificare se l'importo passato come parametro è inferiore al massimale previsto dalla tessera, e ***prelievo***() per effettuare il prelievo dell'importo passato come parametro; ovviamente, se uno qualsiasi dei metodi *check* restituisce false per un certo importo, tentare di fare un prelievo dello stesso importo provocherà necessariamente eccezione;

c) l'enumerativo ***TaglioBanconota*** (fornito) rappresenta i tagli di banconote con cui il bancomat può essere caricato;

d) la classe ***ImportoErogato*** (fornita) rappresenta l'importo erogato dal bancomat nella forma "quali e quante banconote dei due tagli", ossia le due coppie <taglio piccolo, numero di banconote di taglio piccolo>, <taglio grande, numero di banconote di taglio grande> necessarie a comporre l'importo richiesto;

e) la classe ***AbstractBancomat*** (fornita) rappresenta uno sportello bancomat che può essere caricato con banconote di due tagli diversi, uno più piccolo e uno più grande, in modo da avere adeguata flessibilità nella composizione degli importi. La classe espone i seguenti metodi pubblici:

- metodi getter: ***getSaldoSportello*** (restituisce il totale in cassa, ossia il valore totale delle banconote), ***getPiccoloTaglio*** e ***getNumPiccoloTaglio*** (restituiscono rispettivamente il valore del taglio piccolo e il numero di tali banconote caricate), ***getGrandeTaglio*** e ***getNumGrandeTaglio*** (restituiscono rispettivamente il valore del taglio grande e il numero di tali banconote caricate);
- metodi setter: ***setNumPiccoloTaglio*** e ***setNumGrandeTaglio*** sono metodi protected che consentono di impostare i quantitativi di banconote rimanenti a valle di una selezione (v. dopo, algoritmo di selezione delle banconote);
- ***caricaBanconote*** prende in ingresso il taglio della banconota di piccolo taglio, il numero di tali banconote, il taglio della banconota di grande taglio e il numero di tali banconote e carica il bancomat con le banconote specificate; in caso di problemi sui parametri (valori nulli e/o inconsistenti), lancia una ***IllegalArgumentException***;
- ***erogaImporto/2*** prende in ingresso il numero della tessera bancomat e l'importo richiesto ed eroga tale importo distinguendo la politica di scelta delle banconote (preferenza a quelle di piccolo taglio per importi fino a 100€, a quelle di grande taglio oltre i 100€) in base all'importo richiesto. **Da notare** che questo metodo **non** contiene l'algoritmo di scelta banconote, che è invece delegato al metodo astratto ***erogaImporto/4*** (descritto sotto), usato dal metodo concreto ***erogaImporto/2***.

f) la classe ***HappyBancomat*** **(da realizzare)** concretizza ***AbstractBancomat*** implementando il metodo protected ***erogaImporto/4*** incapsulando la logica di composizione dell'importo tramite banconote di diverso taglio descritta nel "Dominio del problema". A tal fine oltre al numero della tessera bancomat e all'importo richiesto, questo metodo riceve in ingresso due ulteriori argomenti (il taglio da usare per primo e il taglio da usare per secondo) e costruisce un'istanza di ***ImportoErogato*** appropriata; lancia ***ImportoNonErogabileException*** (fornita) se:

- l'importo richiesto supera il massimo prelevabile;
- l'importo richiesto non è disponibile sul conto;
- l'importo richiesto non è componibile con i tagli presenti
- il numero di banconote disponibili in uno dei due tagli non consente di soddisfare la richiesta.

Lo Start Kit contiene anche i test (da includere nel progetto) per verificare il funzionamento di queste classi.

Persistenza (namespace happybank.persistence) *(punti 8)*

Come già anticipato, il file di testo BancomatConfiguration.txt contiene la configurazione iniziale dello sportello bancomat della banca, in una sola riga. Più esattamente tale riga contiene, separati da spazi, i due tagli delle banconote con cui il bancomat viene caricato con la corrispondente quantità, nel formato *TaglioMinore x Quantità + TaglioMaggiore x Quantità*

```
ESEMPIO DEL FILE
BancomatConfiguration.txt
20 x 400 + 50 x 500
```

400 banconote da 20€ + 500 banconote da 50 €

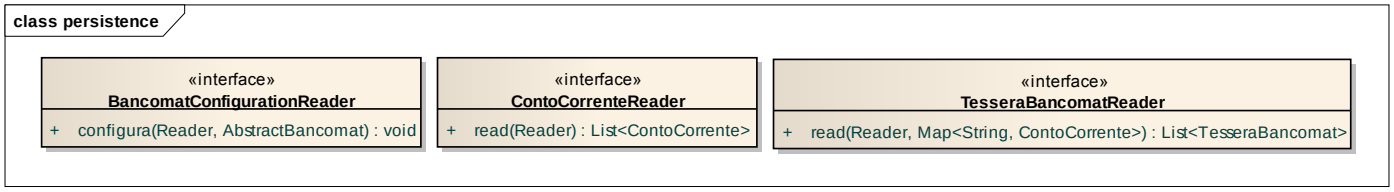
Il file di testo TessereBancomat.txt contiene i dati delle tessere bancomat conosciute, una per riga, specificando per ognuno nell'ordine il numero di conto corrente, il pin, e il prelievo massimo:

```
ESEMPIO DEL FILE TessereBancomat.txt
1234567 02026 250
3214576 94715 400
...
```

Infine, il file di testo ContiCorrenti.txt contiene lo stato dei conti correnti, specificando per ogni conto – identificato tramite un proprio identificatore – la disponibilità attuale sul conto:

```
ESEMPIO DEL FILE ContiCorrenti.txt
1234567 6145
3214576 1700
...
```

Sono fornite le tre interfacce ***BancomatConfigurationReader***, ***TesseraBancomatReader*** e ***ContiCorrentiReader*** che dichiarano i metodi per leggere e/o scrivere i tre file. Le tre classi ***MyBancomatConfigurationReader***, ***MyTesseraBancomatReader*** e ***MyContiCorrentiReader*** (fornita nello start kit) implementano tali interfacce effettuando i necessari controlli sul formato del file e lanciando ***BadFileFormatException*** (fornita) quando opportuno.



MyBancomatConfigurationReader **(da realizzare)** dichiara un metodo ***configura*** che prende in ingresso un ***Reader*** da cui leggere e un ***HappyBancomat*** da configurare mediante opportuna chiamata al metodo ***caricaBanconote***.

MyTesseraBancomatReader **(da realizzare)** dichiara un metodo ***read*** a due argomenti, un ***Reader*** da cui leggere e un oggetto ***Map*** usato dall'algoritmo di caricamento per ottenere gli oggetti ***ContoCorrente*** a partire dai numeri di conto corrente.

Lo Start Kit contiene anche i test (da includere nel progetto) per verificare il funzionamento di queste classi.

Parte 2 (punti: 12)

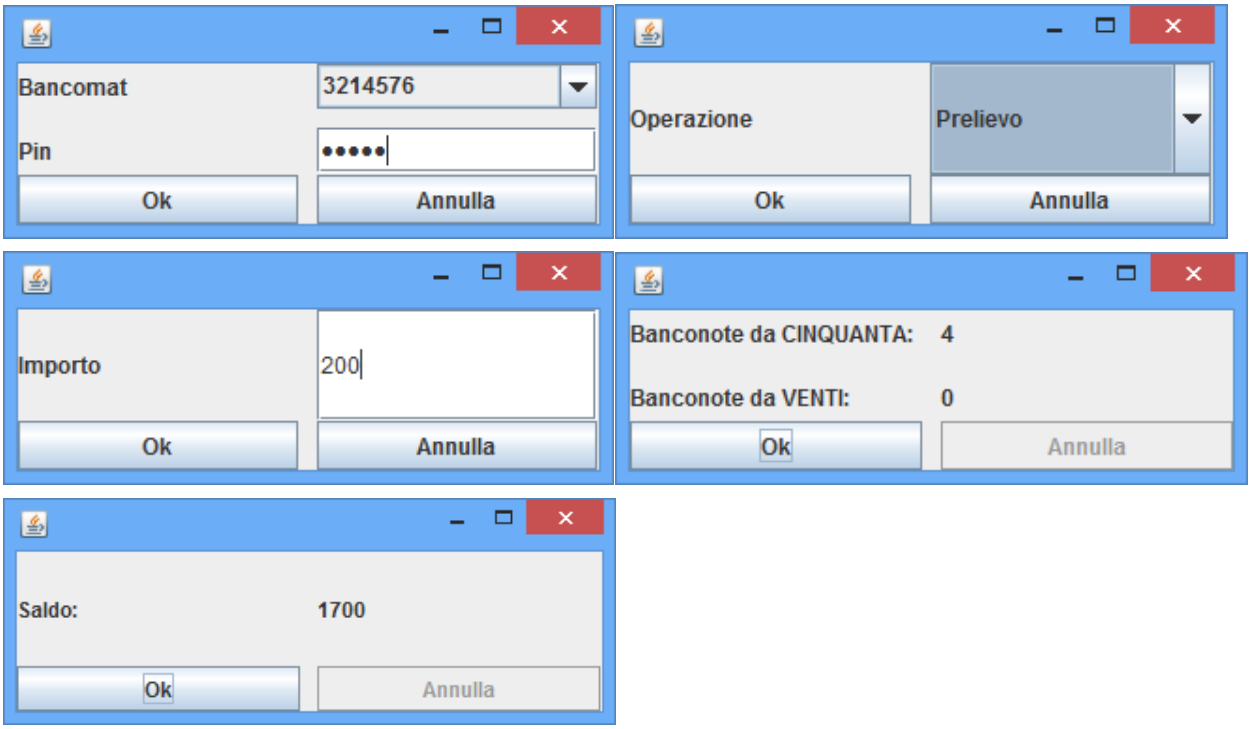
Controller e Interfaccia Utente(namespace happybank.ui) *(punti 12)*

La classe ***Controller*** (fornita nello start kit) governa l'interfaccia utente decidendo quali viste aprire e fornendo i dati per popolarle; i metodi pubblici sono quelli di cui all'elenco che segue:

- ***loadData***: invocato dal main, provvede al caricamento dell'insieme dei dati;
- ***inizio***: mostra la vista che consente di selezionare una tessera bancomat e di validare il PIN;
- ***checkPin***: prende in ingresso l'id della tessera (del conto corrente collegato) e il PIN inserito e ne verifica la validità; in caso di successo, memorizza la tessera bancomat validata come tessera corrente e mostra la vista che consente di selezionare un'operazione;

- ***selezionaOperazione***: prende in ingresso una stringa che rappresenta l'operazione selezionata dall'utente e, in base a questa, decide quale vista mostrare;
- ***preleva***: prende in ingresso l'importo da prelevare, effettua il prelevamento e in caso di successo mostra le banconote prelevate (mediante un'apposita vista, vengono mostrati tagli e quantità delle banconote);
- ***showMessage***: consente di mostrare una dialog con un messaggio;
- ***getIdTessereBancomat***: restituisce l'elenco dei numeri di conto corrente – gli stessi identificatori individuano univocamente anche una tessera bancomat;
- ***getOperazioniDisponibili***: recupera l'elenco delle operazioni disponibili dallo sportello bancomat (restituisce un array di stringhe);
- ***getSaldoCorrente***: restituisce il saldo della tessera bancomat correntemente selezionata .

L'interfaccia utente deve essere simile (non necessariamente identica) agli esempi mostrati in figura.



La classe ***Program*** (non mostrata nel diagramma UML, ma fornita nello start kit) contiene il main di partenza dell'intera applicazione.

La classe ***MyMainView*** (fornita nello start kit), che implementa l'interfaccia ***MainView***, realizza la finestra principale ed è sostanzialmente un contenitore di altre viste.

Tutte le viste sono derivate da ***JPanel*** e contengono una pulsantiera con due pulsanti (Ok e Annulla); tale pulsantiera è fornita nello start kit ed è realizzata dalla classe ***OkCancelButtonPanel***. La pressione di un bottone sulla pulsantiera scatena il classico ***ActionEvent*** a cui agganciarsi tramite l'usuale metodo ***addActionListener***; per discriminare il bottone premuto si suggerisce di recuperare il nome dell'azione ad esso associata tramite il metodo ***getActionCommand***, che restituisce la stringa "**Ok**" in caso di pressione del bottone Ok o "**Cancel**" in caso di pressione del bottone Annulla.

Le viste usate dal controller sono le seguenti:

- ***SelezionaBancomatControllaPin*** **(da realizzare):** mostrata dal metodo ***Controller.inizio,*** consente di selezionare una tessera bancomat mediante opportuna combo, e di introdurre il relativo PIN; si suggerisce di utilizzare un ***JPasswordField*** per l'inserimento del PIN e di usare il metodo ***Controller.checkPin*** per convalidarlo (alla pressione del bottone Ok);

- ***SelezionaOperazione*** (fornita nello start kit): mostrata dal metodo ***Controller.checkPin*** in caso di validazione positiva, mostra una combo con cui scegliere l'operazione da eseguire (prelievo o saldo); alla pressione del bottone Ok viene invocato il metodo ***Controller.selezionaOperazione***, mentre alla pressione del bottone Annulla viene invocato il metodo ***Controller.inizio***;
- ***MostraSaldo*** (fornita nello start kit): mostrata dal metodo ***Controller.selezionaOperazione*** quando il parametro vale "Saldo", mostra il saldo del conto collegato con la tessera corrente; il bottone Annulla è disabilitato e la pressione del bottone Ok invoca il metodo ***Controller.inizio*** in modo da ricominciare da capo;
- ***SelezionaImporto*** **(da realizzare):** mostrata dal metodo ***Controller.selezionaOperazione*** quando il parametro vale "Prelievo", consente all'utente di inserire l'importo da prelevare mediante un opportuno ***JTextField***; alla pressione del bottone Ok il contenuto del ***JTextField*** viene convertito in un intero e (nel caso in cui la conversione abbia avuto successo) viene invocato il metodo ***Controller.preleva*** con l'importo da prelevare; nel caso di conversione fallita, viene mostrato un messaggio che invita l'utente ad inserire un valore numerico (usare il metodo ***Controller.showMessage***). In caso di pressione del bottone Annulla, occorre invece invocare il metodo ***Controller.inizio***().
- ***ErogazioneImporto*** (fornita nello start kit): mostrata dal metodo ***Controller.preleva*** qualora l'importo richiesto sia effettivamente erogabile, mostra il taglio ed il numero delle banconote prelevate; anche in questo caso il bottone Annulla è disabilitato e la pressione del bottone Ok riporta all'inizio.