ESAME DI FONDAMENTI DI INFORMATICA T-2 del 26/7/2023
Proff. E. Denti – R. Calegari – A. Molesini
Tempo a disposizione: 3h30

NOME PROGETTO ECLIPSE: CognomeNome-matricola (es. RossiMario-0000123456)
NOME CARTELLA PROGETTO: CognomeNome-matricola (es. RossiMario-0000123456)
NOME ZIP DA CONSEGNARE: CognomeNome-matricola.zip (es. RossiMario-0000123456.zip)
NOME JAR DA CONSEGNARE: CognomeNome-matricola.jar (es. RossiMario-0000123456.jar)

Si devono consegnare DUE FILE: *l'intero progetto Eclipse* e *il JAR eseguibile*

Si ricorda che compiti *non compilabili* o *palesemente lontani da 18/30* NON SARANNO CORRETTI e causeranno la verbalizzazione del giudizio "RESPINTO"

L'Ente del Turismo di Dentinia ha richiesto lo sviluppo di un'applicazione per la generazione delle previsioni meteo per una serie di località turistiche che, a partire da una serie di dati grezzi (previsioni per una specifica località e ora), produca il *bollettino meteorologico* che descriva la giornata nel suo complesso. Il bollettino dev'essere offerto in due possibili formati: *sintetico* o *dettagliato*. Il primo fa una sintesi della giornata attesa tramite una breve frase, mentre il secondo elenca anche temperatura e probabilità di pioggia a diverse ore della giornata (si vedano esempi più oltre).

DESCRIZIONE DEL DOMINIO DEL PROBLEMA

Una *previsione* indica *temperatura e probabilità di pioggia* previste per un certo *luogo* in una certa *data* a una certa *ora* (ad esempio: Ferrara, 18/07/22, 05:00, 18°, 91%).

Un *bollettino* meteo sintetizza *l'andamento complessivo* di una giornata per un certo *luogo* in una certa *data*: a tal fine, in *formato sintetico* riporta un breve *testo riassuntivo, temperatura media e probabilità di pioggia* previste globalmente per quel dato luogo in quella giornata, mentre nel *formato dettagliato* a queste informazioni si aggiungono anche le *previsioni disponibili per una serie di orari* (che non sono prefissati ma dipendono in generale dai dati disponibili per uno specifico luogo).

ESEMPIO DI BOLLETTINO SINTETICO
Previsioni per il giorno 12/07/22 a Bologna
Giornata **con piogge diffuse**, con **probabilità di pioggia del 77%** e **temperatura media di 20°C**

ESEMPIO DI BOLLETTINO DETTAGLIATO
Previsioni per il giorno 12/07/22 a Bologna
12/07/22, 00:00, Temperatura 18°, Prob. pioggia 91%
12/07/22, 02:00, Temperatura 18°, Prob. pioggia 91%
12/07/22, 05:00, Temperatura 16°, Prob. pioggia 92%
12/07/22, 08:00, Temperatura 20°, Prob. pioggia 87%
12/07/22, 11:00, Temperatura 24°, Prob. pioggia 64%
12/07/22, 14:00, Temperatura 27°, Prob. pioggia 78%
12/07/22, 17:00, Temperatura 15°, Prob. pioggia 58%
12/07/22, 20:00, Temperatura 16°, Prob. pioggia 58%
12/07/22, 23:00, Temperatura 16°, Prob. pioggia 58%
12/07/22, 23:59, Temperatura 16°, Prob. pioggia 58%
Giornata con piogge, con probabilità di pioggia del 77% e temperatura media di 20°C

Il testo riassuntivo utilizza per la giornata una delle espressioni standard "serena", "quasi serena", "con possibili piogge sparse", "variabile", "con piogge diffuse", "con piogge" o "con piogge insistenti e generalizzate", scegliendola in base alla probabilità di pioggia globale secondo la seguente scaletta:

- serena → probabilità di pioggia non superiore al 5%
- quasi serena → probabilità di pioggia non superiore al 10%
- con possibili piogge sparse → probabilità di pioggia non superiore al 25%
- variabile → probabilità di pioggia non superiore al 50%

- con piogge diffuse → probabilità di pioggia non superiore al 65%
- con piogge → probabilità di pioggia non superiore all'80%
- con piogge insistenti → probabilità di pioggia superiore all'80%

Temperatura media e probabilità di pioggia globali di una data giornata si ottengono calcolando una *opportuna media pesata* delle singole previsioni orarie. Più precisamente:

- se non esiste una previsione per le ore 00:00, la si crea uguale alla prima previsione disponibile per la giornata
- se non esiste una previsione per le ore 23:59, la si crea uguale all'ultima previsione disponibile per la giornata
- per ogni coppia di previsioni consecutive, si determinano temperatura e probabilità di pioggia medie: tali valori si pesano con un peso pari alla durata dell'intervallo in questione, ossia alla distanza temporale fra le due previsioni considerate.

ESEMPIO DI CALCOLO
Data la sequenza di previsioni per il giorno 12/07/22 a Bologna:
12/07/22, 02:00, Temperatura 18°, Prob. pioggia 91%
12/07/22, 05:00, Temperatura 16°, Prob. pioggia 92%
12/07/22, 08:00, Temperatura 20°, Prob. pioggia 87%
12/07/22, 11:00, Temperatura 24°, Prob. pioggia 64%
12/07/22, 14:00, Temperatura 27°, Prob. pioggia 78%
12/07/22, 17:00, Temperatura 15°, Prob. pioggia 58%
12/07/22, 20:00, Temperatura 16°, Prob. pioggia 58%
12/07/22, 23:00, Temperatura 16°, Prob. pioggia 58%
si introducono in primo luogo due previsioni extra per le ore 00:00 e 23:59, rispettivamente uguali alla prima e all'ultima della sequenza. Dopo di che si calcolano le seguenti medie:
dalle ore 00:00 alle ore 02:00 (ovvero per 02:00 ore), Temperatura 18°, Prob. pioggia 91%
dalle ore 02:00 alle ore 05:00 (ovvero per 03:00 ore), Temperatura 17°, Prob. pioggia 91.5%
dalle ore 05:00 alle ore 08:00 (ovvero per 03:00 ore), Temperatura 18°, Prob. pioggia 90.5%
dalle ore 08:00 alle ore 11:00 (ovvero per 03:00 ore), Temperatura 22°, Prob. pioggia 75.5%
dalle ore 11:00 alle ore 14:00 (ovvero per 03:00 ore), Temperatura 25.5°, Prob. pioggia 71%
dalle ore 14:00 alle ore 17:00 (ovvero per 03:00 ore), Temperatura 21°, Prob. pioggia 68%
dalle ore 17:00 alle ore 20:00 (ovvero per 03:00 ore), Temperatura 15.5°, Prob. pioggia 58%
dalle ore 20:00 alle ore 23:00 (ovvero per 03:00 ore), Temperatura 16°, Prob. pioggia 58%
dalle ore 23:00 alle ore 23:59 (ovvero per 00:59 ore), Temperatura 16°, Prob. pioggia 58%
La temperatura media e le probabilità di pioggia globali si ottengono infine semplicemente facendo la media pesata di tali valori medi, utilizzando come peso la durata dei rispettivi intervalli – ovvero:
T = (18° x 2h + 17° x 3h + 18° x 3h + … + 16° x 0h59m) / 23h59m = 19.9° → 20° (arrotondamento al più vicino)
P = (91% x 2h + 91.5% x 3h + 90.5% x 3h + … + 58% x 0h59m) / 23h59m = 77.1% → 77% (arrotondamento al più vicino)

Il file di testo Previsioni.txt, descritto più oltre, contiene i dati di una serie di previsioni puntuali per diverse città.

TEMPO STIMATO PER SVOLGERE L'INTERO COMPITO: 2h15 – 3h

PARTE 1 – Modello dei dati: Punti 11 [TEMPO STIMATO: 55-65 minuti]

PARTE 2 – Persistenza: Punti 13 [TEMPO STIMATO: 60-75 minuti]

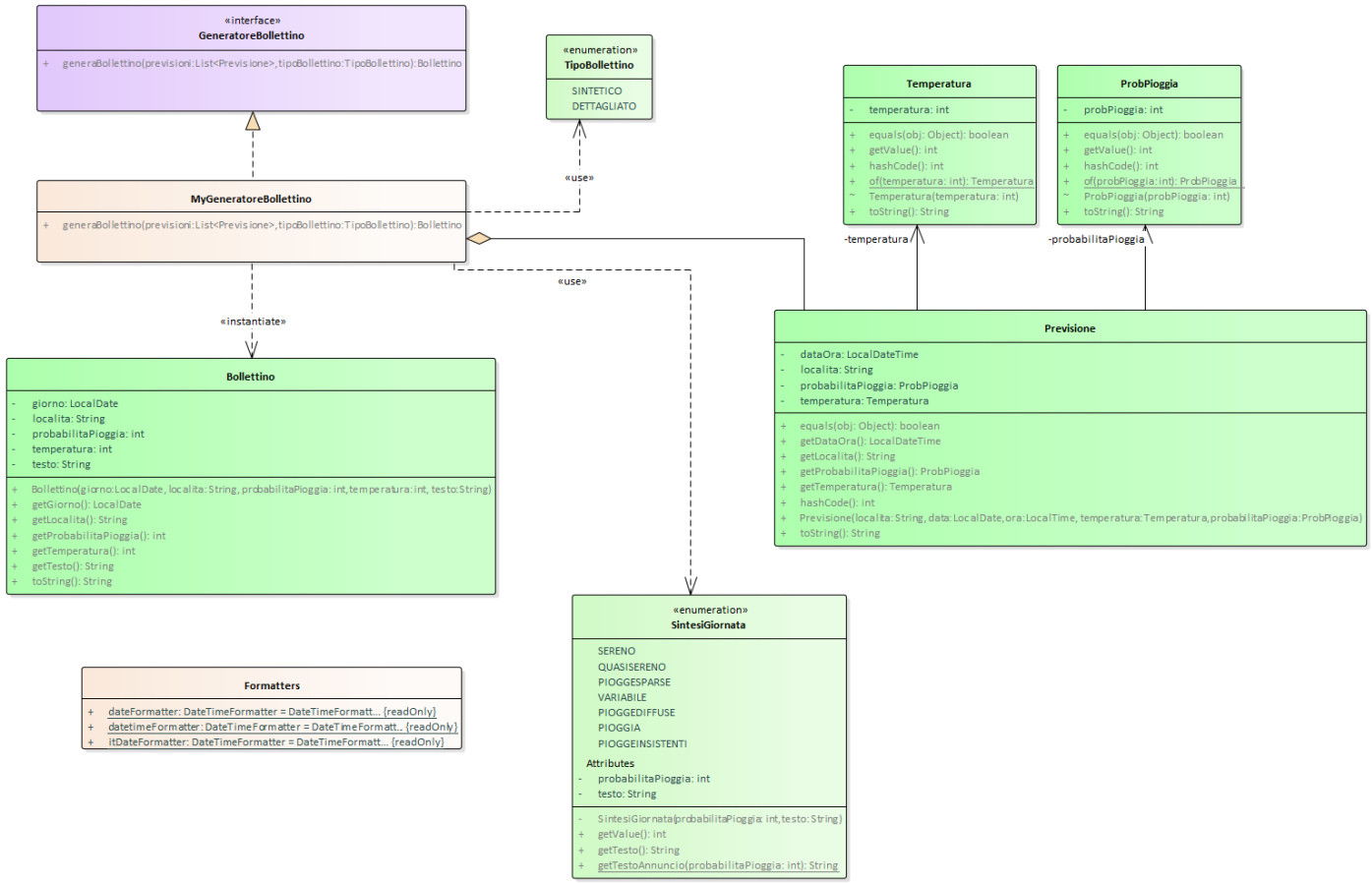
PARTE 3 – Grafica: Punti 6 [TEMPO STIMATO: 20-40 minuti]

JAVAFX – Parametri run configuration nei LAB

```
--module-path="C:\applicativi\moduli\javafx-sdk-19\lib"
--add-modules=javafx.controls
```

Parte 1 – Modello dei dati (punti: 11)

Package: meteodent.model **[TEMPO STIMATO: 60-75 minuti]**



SEMANTICA:

a) L'enumerativo ***TipoBollettino*** (fornito) definisce semplicemente le due costanti DETTAGLIATO e SINTETICO

b) L'enumerativo ***SintesiGiornata*** (fornito) definisce le sette costanti da SERENO a PIOGGEINSISTENTI, accoppiandole ai rispettivi valori-soglia come definiti nel Dominio del Problema e alla corrispondente stringa descrittiva, recuperabili tramite i due accessor *getValue* e *getTesto*. Il metodo *getTestoAnnuncio* completa l'implementazione, restituendo una stringa compiutamente descrittiva.

c) Le due classi ***Temperatura*** e ***ProbPioggia*** (fornite) incapsulano ciascuna un valore intero, dandogli semantica. In ossequio al pattern factory, il costruttore non è pubblico: la costruzione deve quindi avvenire attraverso il metodo statico *of*. L'accessor *getValue* e implementazioni standard per *toString*, *equals* e *hashcode* completano l'implementazione.

d) La classe ***Previsione*** (fornita) rappresenta una previsione intesa come specificato nel Dominio del Problema (i suoi dati corrispondono a quelli di una riga del file da leggere nella persistenza). Il costruttore verifica gli argomenti in ingresso, lanciando ***IllegalArgumentException*** con adeguato messaggio d'errore in caso di incoerenze. Appositi accessor consentono di recuperare le singole proprietà, *ivi inclusa la data e l'ora sotto forma di LocalDateTime*. Sono incluse opportune implementazioni di *equals*, *toString* e *hashcode*.

e) La classe ***Bollettino*** (fornita) rappresenta un bollettino inteso come specificato nel Dominio del Problema. Il costruttore verifica gli argomenti in ingresso, lanciando ***IllegalArgumentException*** con adeguato messaggio d'errore in caso di incoerenze. Appositi accessor consentono di recuperare le singole proprietà. È inclusa un'idonea implementazione di *toString*.

f) La classe ***Formatters*** (da realizzare) definisce i tre formattatori usati in tutta l'applicazione: in particolare, *itDateFormatter* è un formattatore per data secondo lo standard italiano ma con l'anno su quattro cifre,

mentre *dateFormatter* e *datetimeFormatter* sono, rispettivamente, formattatori per data/ data e ora secondo lo standard italiano in formato SHORT.

g) L'interfaccia ***GeneratoreBollettino*** (fornita) dichiara il metodo *generaBollettino,* che genera un ***Bollettino*** a partire da una lista di ***Previsione*** e dal ***TipoBollettino*** desiderato. Da specifica di progetto, il metodo deve lanciare ***IllegalArgumentException*** con adeguato messaggio d'errore nei seguenti casi:

- Lista previsioni vuota o nulla
- Le previsioni nella lista non si riferiscono tutte alla stessa località
- Le previsioni nella lista non si riferiscono tutte alla stessa data
- Vi sono due o più previsioni per lo stesso orario

h) La classe ***MyGeneratoreBollettino*** (da realizzare) implementa ***GeneratoreBollettino*** in ossequio alle specifiche sopra elencate e a quanto previsto dal Dominio del Problema. NB: l'arrotondamento finale della temperatura e probabilità di pioggia deve avvenire all'intero più vicino (metodo *Math.round*).

Parte 2 – Persistenza (punti: 13)

Package: meteodent.persistence **[TEMPO STIMATO: 55-65 minuti]**

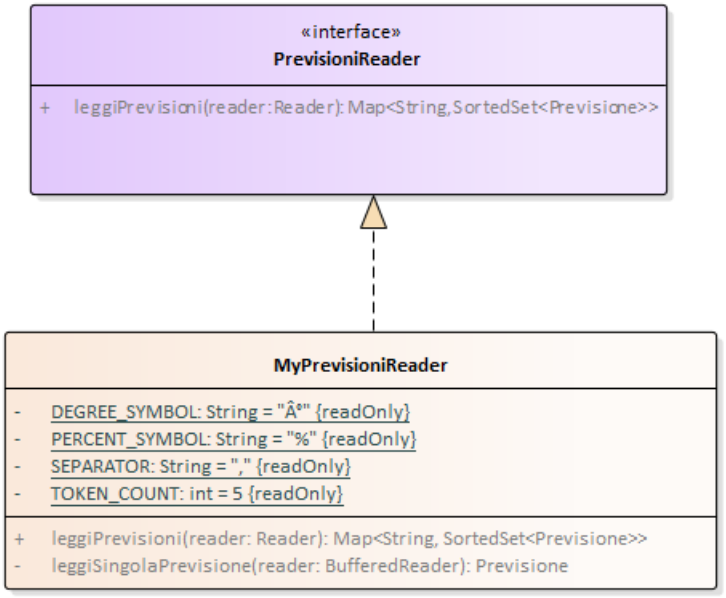
Il file di testo previsioni.txt contiene una serie di previsioni, una per riga, riferite a più località ed eventualmente per diverse date: in ogni riga, gli elementi sono separati da virgole.

Ogni riga specifica cinque elementi:

- la località (una stringa)
- la data a cui la previsione si riferisce, nel classico formato GG/MM/AA
- l'orario a cui la previsione si riferisce, nel classico formato HH:MM
- la temperatura prevista, costituita da un numero intero (eventualmente anche negativo) subito seguito, senza spazi intermedi, dal carattere '**°**'
- la probabilità di pioggia prevista, costituita da un numero intero (ovviamente compreso fra 0 e 100) subito seguito, senza spazi intermedi, dal carattere '**%**'

ESEMPIO

```
Bologna, 12/07/22, 02:00, 18°, 91%
Bologna, 12/07/22, 05:00, 16°, 92%
Bologna, 12/07/22, 08:00, 20°, 87%
Bologna, 12/07/22, 11:00, 24°, 64%
...
Ferrara, 18/07/22, 05:00, 18°, 91%
Ferrara, 18/07/22, 15:00, 16°, 92%
Ferrara, 18/07/22, 22:00, 20°, 87%
Reggio Emilia, 11/07/22, 02:00, -1°, 21%
Reggio Emilia, 11/07/22, 10:00, -1°, 32%
Reggio Emilia, 11/07/22, 18:00, 6°, 50%
...
```



SEMANTICA:

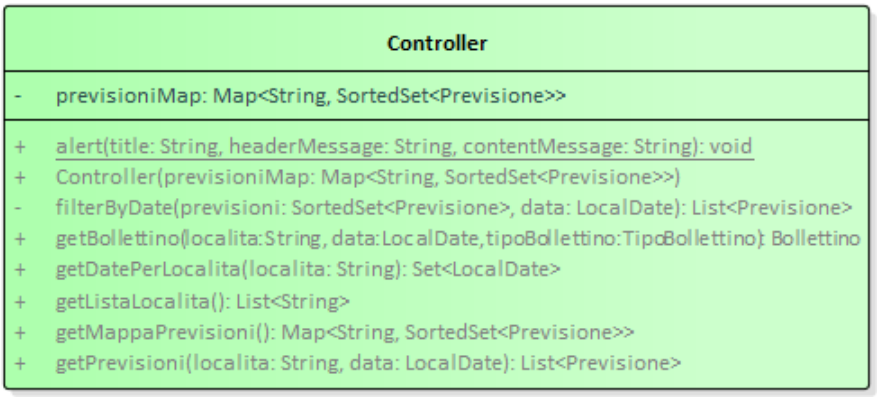
a) L'interfaccia ***PrevisioniReader*** (fornita) dichiara il metodo *leggiPrevisioni* che carica da un apposito Reader (già aperto) i dati necessari, restituendo una mappa che associa a ogni località (*stringa*) l'insieme ordinato delle previsioni che la riguardano (*SortedSet<Previsione>*), ordinato in senso crescente per data e ora. Il metodo lancia:
   - ***IllegalArgumentException*** con opportuno messaggio d'errore in caso di argomento (reader) nullo;
   - ***BadFileFormatException*** con messaggio d'errore appropriato in caso di problemi nel formato del file (mancanza/eccesso di elementi, errori nel formato delle date/orari, percentuali assurde, formato errato nelle temperature e nelle probabilità di pioggia, etc.);
   - una ***IOException*** in caso di altri problemi di I/O.

b) La classe ***MyPrevisioniReader*** (**da realizzare**) implementa ***PrevisioniReader*** secondo le specifiche sopra descritte.

Parte 3 (punti: 6)

Package: meteodent.controller *(punti 0)*

Il Controller è organizzato secondo il diagramma UML seguente:



SEMANTICA:

La classe ***Controller*** (fornita) riceve in fase di costruzione la mappa delle previsioni, che viene mantenuta nel suo stato interno. È fornita un'ampia serie di metodi per recuperare tutte le informazioni utili:

- La mappa previsioni (*getMappaPrevisioni*)

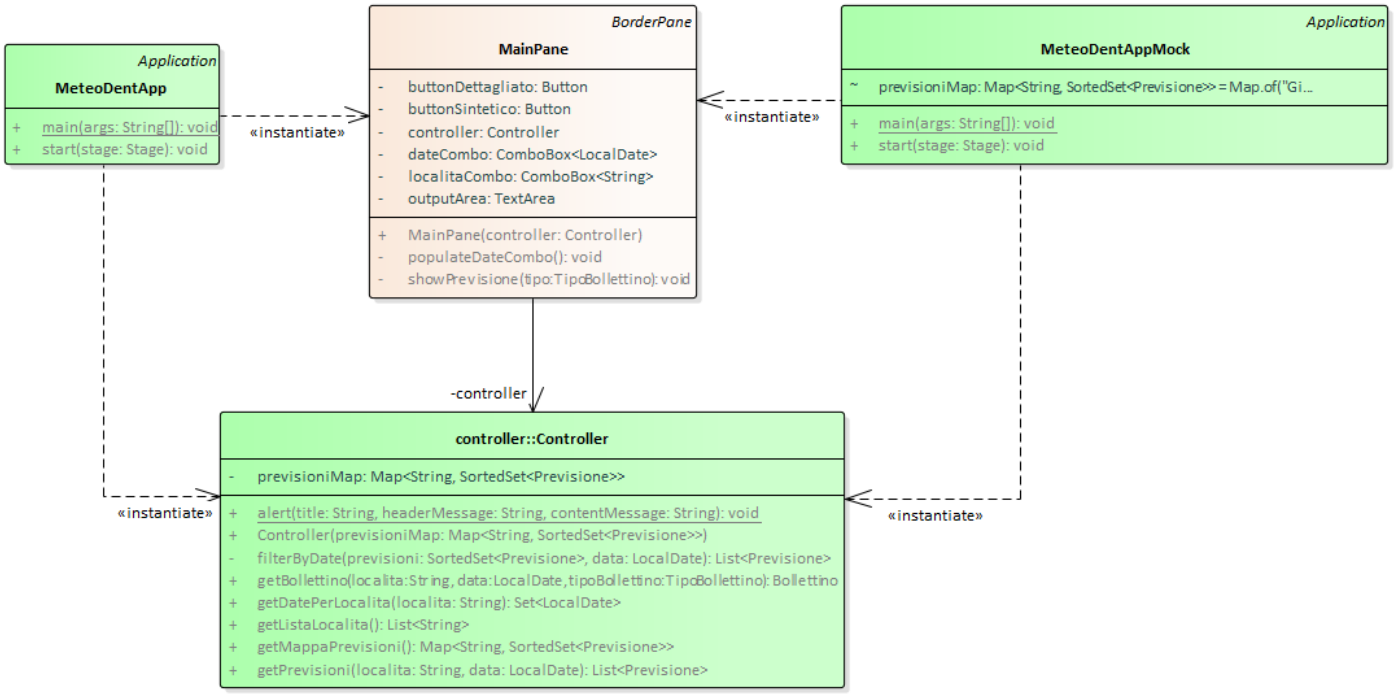
- La lista delle località (*getListaLocalita*)
- l'insieme delle date per le quali esistono previsioni per una data località (*getDatePerLocalita*)
- La lista delle previsioni per una data località e un certo giorno (*getPrevisioni*)
- Il bollettino per un certo giorno, a un certo orario, in una data località (*getBollettino*)

Infine, il metodo statico *alert*, utilizzabile anche dal MainPane, consente di far comparire all'utente, ove occorra, una finestra di dialogo con opportuno messaggio d'errore.

Package: meteodent.ui **[TEMPO STIMATO: 20-40 minuti]** *(punti 6)*

La classe ***MeteodentApp*** (fornita) costituisce l'applicazione JavaFX che si occupa di aprire i file, creare il controller e incorporare il ***MainPane***. Per consentire di collaudare la GUI anche in assenza / in caso di malfunzionamento della parte di persistenza, è possibile avviare l'applicazione mediante la classe ***MeteodentAppMock***.

L'interfaccia utente è illustrata nelle figure seguenti e segue il modello sotto illustrato*:*



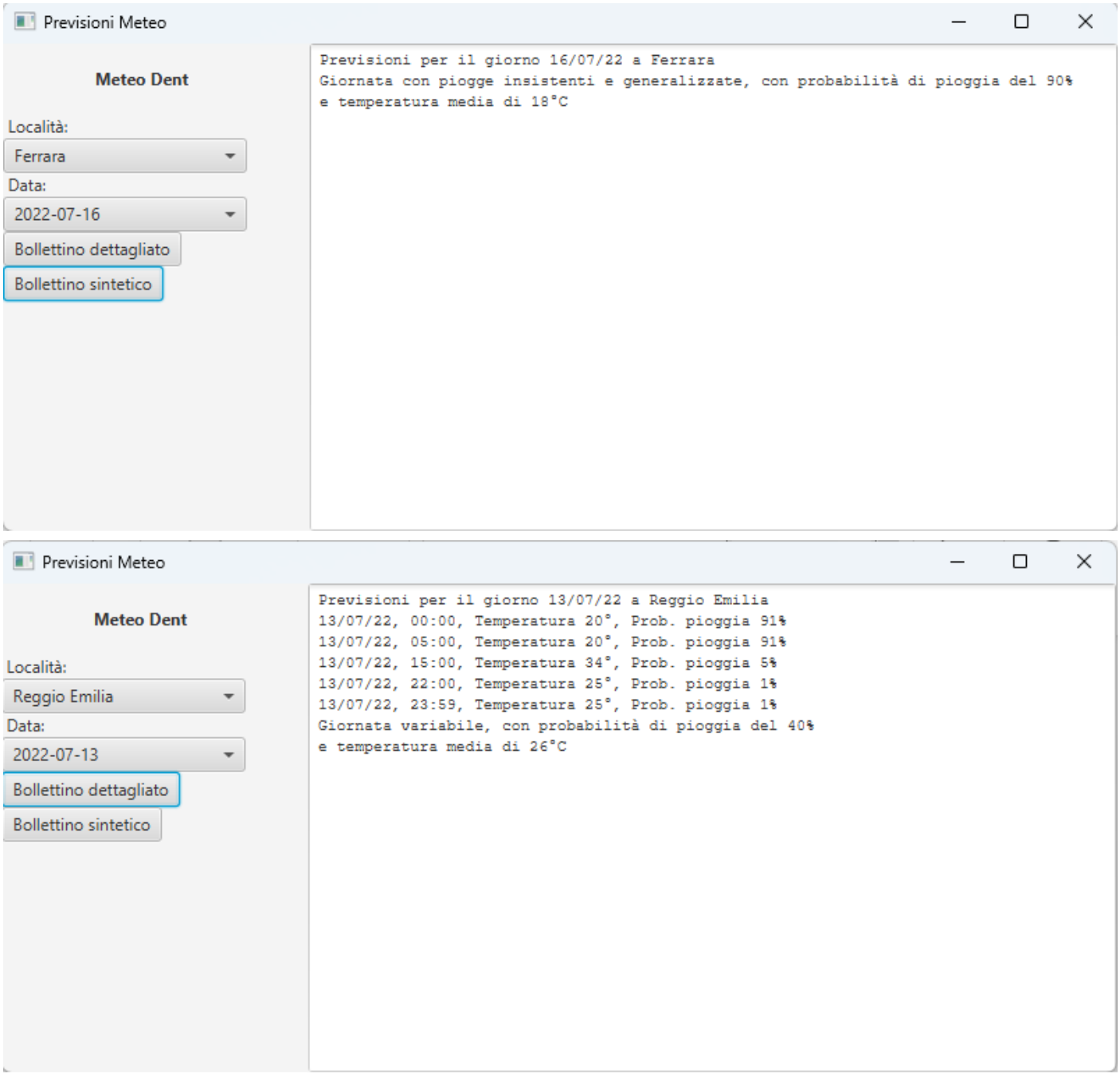
L'interfaccia grafica si presenta come segue:

Fig. 1: la situazione iniziale della GUI, con combo tipologie vuota, prima di qualunque interazione con l'utente.

- a sinistra una combo elenca tutte le località disponibili: di seguito, un'altra combo, *popolata dinamicamente in base alla località selezionata nella prima*, elenca le date per le quali esistono previsioni per la località scelta; subito sotto, due pulsanti consentono di produrre e visualizzare il bollettino, nei due formati possibili
- a destra, un'area di testo (inizialmente vuota e non scrivibile dall'utente) mostra i risultati.

Comportamento:

- la selezione di una località dalla prima combo deve causare il ri-popolamento della seconda combo con tutte e sole le date per le quali esistono previsioni per la località scelta
- la scelta di una di tali date dalla seconda combo NON produce effetti immediati
- la pressione di uno dei due pulsanti causa invece immediatamente la generazione del bollettino richiesto e la sua visualizzazione nell'area a lato.



Figg. 2 / 3.

Il MainPane è fornito *parzialmente realizzato*: è presente quasi tutta l'impostazione strutturale, mentre sono da completare la configurazione di alcuni componenti e la gestione degli eventi.

In particolare, ***MainPane*** estende ***BorderPane*** e prevede:

1) a sinistra, una ***VBox*** per le varie combo, etichette e pulsanti
2) a destra, una ***VBox*** con la sola area di output.

La **parte da completare** riguarda:

1) l'aggancio dei gestori eventi rispettivamente alla prima combo e ai due pulsanti
2) il popolamento della combo delle date (metodo privato *populateDateCombo*), che deve recuperare la località di interesse, recuperare tramite il controller le date che la riguardano, e preimpostare il primo item come selezione di default
3) la logica di gestione dell'evento (metodo privato *showPrevisione*), che deve recuperare dalle due combo i dati necessari e far generare al ***Controller*** il ***Bollettino*** del tipo richiesto, mostrandone l'output sull'area.

Cose da ricordare

- salva costantemente il tuo lavoro: l'informatica a volte può essere "subdolamente ostile"..
- in particolare: se ora compila e stai per fare modifiche, salva la versione attuale (non si sa mai)

Checklist di consegna

- Hai fatto un **JAR eseguibile**, che contenga cioè l'indicazione del main?
- Hai controllato che **si compili** e **ci sia tutto**? [NB: non includere il PDF del testo]
- Hai **rinominato** IL PROGETTO, lo ZIP e il JAR esattamente come richiesto?
- Hai **chiamato** la cartella del progetto esattamente come richiesto?
- Hai fatto un **unico file ZIP (NON .7z, rar o altri formati)** contenente l'intero progetto? In particolare, ti sei assicurato di aver incluso tutti i file .java (e non solo i .class)?
- **Hai consegnato DUE file distinti, ossia lo ZIP col progetto e il JAR eseguibile?**
- Su EOL, hai **premuto** il tasto "CONFERMA" per inviare il tuo elaborato?