# *Fondamenti di Programmazione B*

# Prova di programmazione (Java)

# Classe Agenda

1) *(a)* Realizzare in Java una classe di nome Data, con tre attributi, giorno, mese, anno, di tipo intero, e con i seguenti costruttori e metodi public:

* costruttore con tre parametri, giorno, mese, anno. N.B. controllare che *1 ≤* giorno *≤ 31*, *1 ≤* mese *≤ 12*, anno ≥ 0, e in caso contrario sollevare un'eccezione "data non valida";
* metodo precede(d) che restituisce true se questa data (this) è precedente alla data d;
* metodo toString() che converte questa data (this) nella stringa "giorno/mese/anno".

*(b)* Realizzare una classe di nome DataConOrario, derivata da Data, con un nuovo attributo ora (di tipo intero) che rappresenta l’ora del giorno. La classe offre in particolare i seguenti costruttori e metodi public:

* costruttore con quattro parametri, giorno, mese, anno, ora (utilizzare obbligatoriamente il costruttore con tre parametri della classe Data). N.B. controllare che *0 ≤* ora *≤ 23* e in caso contrario sollevare un'eccezione "orario non valido";
* metodo precede(d) che restituisce true se questa DataConOrario è precedente alla DataConOrario d (utilizzare obbligatoriamente il metodo precede di Data);
* metodo toString() che converte questa DataConOrario nella stringa “giorno/mese/anno – h:ore” (utilizzare obbligatoriamente il metodo toString di Data).

*(c)* Realizzare una classe di nome Agenda che rappresenta una sequenza di oggetti di tipo DataConOrario (max. *100*). La classe offre i seguenti costruttori e metodi public:

* costruttore Agenda(): crea una sequenza vuota;
* metodo add(a): aggiunge l'oggetto a di tipo DataConOrario a questa Agenda; nel caso non ci sia spazio sufficiente solleva un'eccezione SpazioNonSufficiente;
* metodo toString(): restituisce la stringa che rappresenta in modo opportuno i dati di tutti gli elementi della Agenda.

Implementare la sequenza tramite un array di 100 elementi di tipo DataConOrario.

2) Scrivere un programma principale di prova che: crea due oggetti DataConOrario di nome A e B e li inizializza con due date con orario (giorno, mese, anno, ora) fornite dall’utente tramite std input; crea un oggetto Agenda di nome D ed inserisce A e B in D (tramite il metodo add), aggiungendo prima la data più vecchia e poi quella più recente (utilizzare il metodo precede di DataConOrario); infine stampa D su std output (tramite il metodo println).

Prevedere anche la “cattura” e gestione all’interno del main delle eventuali eccezioni generate dal programma. N.B. Definire (obbligatoriamente) le eccezioni come nuove classi, estendendo in modo opportuno le classi di libreria per le eccezioni non controllate.

Aggiungere le dichiarazioni necessarie ad inserire tutte le classi realizzate nel package esameFP