Cognome e Nome _____

Matricola _____

# Programmazione e Algoritmi 2 A.A. 2022/2023

## Appello del 7 giugno 2023

---

**Esercizio 1 [8 punti]**

Si considerino le seguenti classi.

```
class A {
      protected int n;
      public A (int n) {
            this.n=n;
      }
      public int metodo(A a) {
            return n + 10;
      }
      public int metodo(B b) {
        return n + 20;
      }
}
```

```
class B extends A {
      public B(int x, int y) {
            super (x+y);
      }
      public int metodo(A a) {
            return n + 30;
      }
      public int metodo(B b) {
            return n + 40;
      }
}
```

Si dica cosa viene stampato a video dal seguente codice:

```
A a = new A(2);
B b = new B(5, 3);
A ab = new B(15, 2);
System.out.println(a.metodo(b));
System.out.println(b.metodo(ab));
System.out.println(ab.metodo(ab));
```

Si giustifichi la risposta mostrando in particolare:

- le firme associate a tempo di compilazione e a tempo di esecuzione ad ogni chiamata di metodo
- l'evoluzione della memoria nelle parti stack ed heap.

---

**Esercizio 2 [6 punti]**

Si consideri la seguente sequenza di numeri interi:

**[5, 10, 20, 16, 9, 3, 27, 33, 1]**

1. **[3 punti]** Mostrare **passo-passo l'evoluzione dell'albero binario di ricerca** che si ottiene, a partire dall'albero vuoto, inserendo le chiavi nell'ordine indicato.
2. **[3 punti]** Mostrare **passo-passo l'evoluzione del min-heap** che si ottiene, a partire dall'heap vuoto, inserendo le prime sette chiavi (da 5 a 27) della sequenza nell'ordine indicato. Dopo aver inserito tali chiavi, effettuare **due estrazioni** del minimo mostrando l'heap che si ottiene dopo ogni estrazione, e successivamente inserire le ultime due chiavi (33 e 1) della sequenza.

**Esercizio 3 [10 punti]**

*Si vogliono gestire, in Java, gli spettacoli di un cinema/teatro con una sala avente capienza di 524 posti.*
Si assuma, senza scrivere nulla, l'esistenza di una classe **Persona** con

- una variabile di istanza **nome** (tipo String, private);
- una variabile di istanza **cognome** (tipo String, private);

e i seguenti metodi di istanza:

- un costruttore che crea un oggetto dati **nome e cognome**;
- metodi pubblici accessori **`getNome()`** e **`getCognome()`**;
- metodo **`equals`** che restituisce *true* se e solo se due oggetti della classe Persona hanno uguale il nome e uguale il cognome.

Si assuma, senza scrivere nulla, l'esistenza di una classe astratta **Spettacolo** con

- una variabile di istanza **spettatori** (tipo Array di Persona, private); si tenga conto non è detto che gli spettatori presenti (laddove siano meno della lunghezza dell'array) siano tutti memorizzati in posizioni consecutive dell'array.
- una variabile di istanza **titolo** (tipo String, private);
- una variabile di istanza **giorno** (tipo int, private), che rappresenta il giorno della data dello spettacolo contando i giorni trascorsi dal 1° gennaio 2000.

e, tra gli altri, i seguenti metodi di istanza:

- un costruttore che crea uno spettacolo dato **titolo** e **giorno**, con l'array di spettatori di lunghezza ***524*** contenente *null* in tutte le posizioni;
- metodo di accesso **`public int getGiorno()`**.
- metodo di accesso **`public String getTitolo()`**.
- Metodo di accesso **`public Persona getSpettatore(int i)`**, che restituisce lo spettatore in posizione **i** (con **i** che può variare da 0 a 523); il metodo restituisce **null** se nella posizione specificata non è presente alcuno spettatore, o se la posizione non è una posizione valida.

Si assuma, senza scrivere nulla, che la classe astratta **Spettacolo** è estesa dalle sottoclassi (non astratte) **Proiezione** e **Rappresentazione.** Si tenga conto del fatto che la classe **Proiezione** ha una variabile di istanza **durata** (tipo int, private) che contiene la durata in minuti della proiezione, mentre la classe **Rappresentazione** ha una variabile di istanza **cast** (tipo ArrayList<Persona>, private) che contiene la lista degli attori coinvolti nella rappresentazione. Per tutte e due le sottoclassi sono presenti i costruttori (che prendono in input anche la durata e l'ArrayList del cast, rispettivamente) che richiamino opportunamente il costruttore della classe **Spettacolo** e sono presenti metodi di accesso pubblici per le variabili di istanza introdotte (**durata** e **cast**).

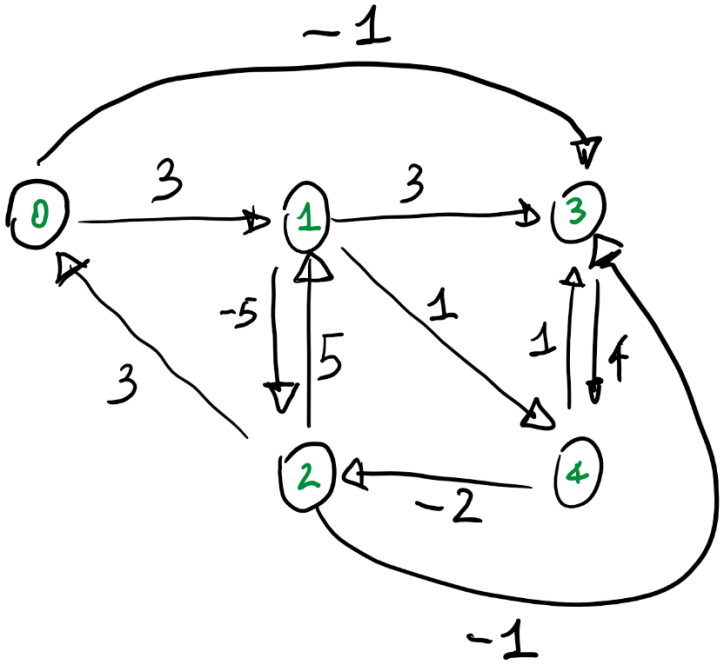
Si scriva una classe **CinemaTeatro** con

- una variabile di istanza **spettacoli** (tipo ArrayList<Spettacolo>, private);

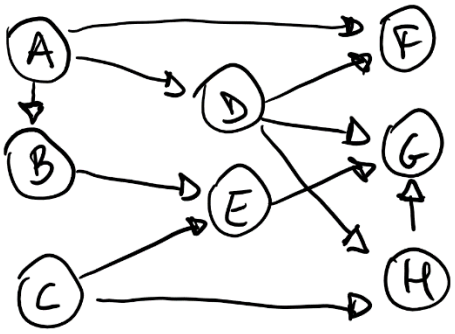
Implementare i seguenti metodi di istanza:

- un costruttore senza parametri che crea un ArrayList **spettacoli** vuoto.
- un metodo **`public ArrayList<Persona> estraiTuttiICast ()`** che restituisce un ArrayList contenente, senza ripetizioni, tutte le persone presenti nel cast di tutte le rappresentazioni dell'ArrayList **spettacoli**.
- un metodo **`public ArrayList<Spettacolo> vistoDa (Persona p)`** che restituisce tutti gli spettacoli contenuti nell'ArrayList **spettacoli** nel cui pubblico sia presente la persona **p**.

**Esercizio 4 [10 punti]**




a. **[5 punti]** Mostrare una possibile esecuzione **passo-passo** dell'algoritmo di **Bellman-Ford** per calcolare i cammini minimi a partire dalla sorgente **0.** Dire, giustificando la risposta, se il grafo possiede cicli di peso negativo.




b. **[5 punti]** Calcolare un ordinamento topologico del DAG in figura, usando il metodo della visita in profondità ed evidenziando per ogni nodo i timestamp di inizio e fine visita.

**Regole per lo svolgimento della prova scritta**:

- Per svolgere il compito si hanno a disposizione **120** minuti.
- Scrivere **subito** nome, cognome, matricola su OGNI FOGLIO (**compreso questo**).
- Durante la prova scritta **non** è possibile abbandonare l'aula.
- Non è ammesso **per nessun motivo** comunicare in qualsiasi modo con altre persone
- È possibile consultare appunti, libri e dispense.
- Qualsiasi strumento elettronico di calcolo o comunicazione (telefoni cellulari, calcolatrici, palmari, computer, etc…) deve essere **completamente disattivato** e **depositato in vista sulla cattedra**
- Mettere in vista sul banco un valido documento di identità.