

Cognome e Nome _____

Matricola _____

Appello del 19 giugno 2024

Esercizio 1 [8 punti]

Si considerino le seguenti classi.

```
class A {
    protected int n;

    public A(int n) {
        this.n = n;
    }

    public A(int n, int m) {
        this(n*m);
    }

    1 public int metodo(A a) {
        n = n - a.n;
        return n;
    }

    2 public int metodo(B b) {
        n = n + b.n;
        return n;
    }
}
```

```
class B extends A {
    public B(int n) {
        this(n, n-1);
    }

    public B(int n, int m) {
        super(n-1, m);
    }

    3 public int metodo(A a) {
        a.n = n + a.n;
        return a.n;
    }

    4 public int metodo(B b) {
        b.n = n + b.n;
        return n;
    }
}
```

Si dica cosa viene stampato a video dal seguente codice:

```
A a = new A(5, 7);
B b = new B(2);
A ab = new B(3, 4);
System.out.println(a.metodo(ab));
System.out.println(b.metodo(b));
System.out.println(ab.metodo(a));
System.out.println(ab.metodo(ab));
```

Si giustifichi la risposta individuando in particolare le firme associate a tempo di compilazione e a tempo di esecuzione ad ogni chiamata di metodo (riempire le caselle della tabella con **1, 2, 3, 4** o **ERRORE** se non è possibile identificare nessun metodo):

	Tempo di compilazione	Tempo di esecuzione
a. metodo (ab)		
b. metodo (b)		
ab. metodo (a)		
ab. metodo (ab)		

Esercizio 2 [5 punti]

Mostrare **passo-passo l'esecuzione dell'heap-sort** per ordinare l'array **[7, 3, 80, 12, 9, 17, 2]** in modo non decrescente (compresa la fase iniziale di **build-max-heap**).

Esercizio 3 [9 punti]

Si vogliono gestire, in Java, i movimenti associati al credito di SIM di telefonia mobile.

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

- una variabile di istanza **idSim** (tipo String, private), che contiene il numero telefonico associato alla SIM a cui il movimento si riferisce;
- una variabile di istanza **tempo** (tipo long, private), che rappresenta l'istante del movimento espresso in secondi trascorsi dalle ore 00:00 del 1° gennaio 2000.

e i seguenti metodi di istanza:

- un costruttore che crea un movimento dati campi **idSim** e **tempo**;
- metodo di accesso **public String getIdSim()**.
- metodo di accesso **public long getTempo()**.
- metodo pubblico astratto **public abstract double getImporto()** che restituisce l'importo in Euro del movimento. In particolare, se il movimento è un movimento di spesa viene restituito un valore minore o uguale a zero, mentre se è un movimento di ricarica credito viene restituito un valore positivo.

La classe astratta **Movimento** è estesa dalle sottoclassi (non astratte) **Chiamata**, **Sms** e **Ricarica**.

Si tenga conto del fatto che la classe **Chiamata** ha:

- una variabile di istanza **durata** (tipo int, private) che contiene la durata in secondi della chiamata;
- una variabile di istanza **destinatario** (tipo String, private), che contiene il numero chiamato.

La classe **Sms** deve avere:

- una variabile di istanza **destinatario** (tipo String, private), che contiene il destinatario dell'SMS.

La classe **Ricarica** deve avere:

- una variabile di istanza **importo** (tipo double, private), che contiene l'importo della ricarica e deve valere almeno 0,01.

Si assuma, senza scrivere nulla, che tutte le sottoclassi abbiano i costruttori (che prendono in input gli opportuni parametri e che richiamino opportunamente il costruttore della classe **Movimento**), i metodi di accesso per tutti i campi e l'implementazione del metodo **double getImporto()**.

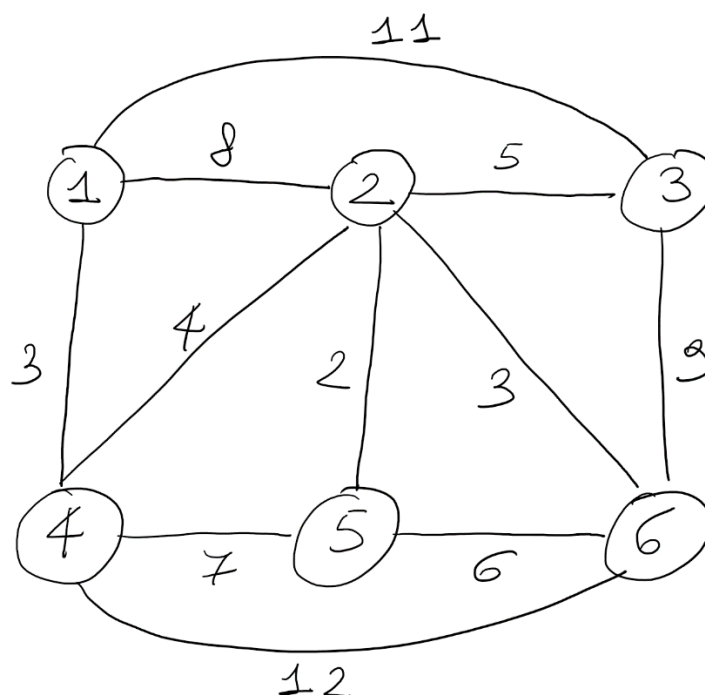
Si assuma, senza scrivere nulla, l'esistenza di una classe **Gestore** con una variabile di istanza **movimenti** (tipo **ArrayList<Movimento>**, private), contenenti tutte le chiamate, gli sms e le ricariche effettuate da tutte le SIM del gestore.

Si aggiungano alla classe **Gestore** i seguenti metodi:

- un metodo **public double getCredito (String idSim)** che restituisce il credito residuo attualmente presente sulla Sim **idSim**.
- un metodo **public double getTotaleRicariche (String idSim)** che restituisce il totale delle ricariche effettuate sulla Sim **idSim**.
- un metodo **public int contoSMS (String idSim1, String idSim2)** che restituisce il numero di tutti gli SMS intercorsi tra **idSim1** e **idSim2** (in entrambe le direzioni).

Esercizio 4 [12 punti]

Si consideri il grafo non diretto in figura.



- a. [6 punti] Mostrare una possibile esecuzione **passo-passo** dell'algoritmo di **Dijkstra** per i cammini minimi da singola sorgente, a partire dal nodo sorgente **6**. Ad ogni passo, bisogna mostrare il contenuto della coda con priorità utilizzata dall'algoritmo e del vettore dei padri.
Che costo complessivo ha il cammino minimo dal nodo 6 al nodo 1?
- b. [6 punti] Mostrare una possibile esecuzione **passo-passo** dell'algoritmo di **Prim** per il minimo albero ricoprente, a partire dal nodo sorgente **1**. Ad ogni passo, bisogna mostrare il contenuto della coda con priorità utilizzata dall'algoritmo e del vettore dei padri.
Che costo complessivo ha il minimo albero ricoprente?

Regole per lo svolgimento della prova scritta:

- Per svolgere il compito si hanno a disposizione **150** 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.