

- Scrivere immediatamente, su ogni foglio che vi è stato consegnato, cognome, nome, numero di matricola.
- Non è consentito consultare appunti, libri, colleghi, né qualunque dispositivo elettronico, pena l'immediato annullamento della prova.
- Gli esercizi relativi al modulo A devono essere svolti su fogli differenti rispetto a quelli del modulo B.
- Tempo a disposizione: 3 ore.

### Modulo A - Esercizio 1

Si descrivano analogie e differenze nella gestione dei requisiti tra modello di processo a cascata e SCRUM.

### Modulo A - Esercizio 2

Si considerino la classe Utils e la classe di test JUnit UtilsTest riportate di seguito.

```
package it.unina.ingsw.jan22;

public class Utils {

    private int threshold;
    public Utils(int t){this.threshold = t;}

    float getNumFromString(String s){
        if(s.length() > this.threshold)
            return 1.0F;
        else
            return 0.0F;
    }
}
```

```
package it.unina.ingsw.jan22;

import org.junit.jupiter.api.*;
import static org.junit.jupiter.api.Assertions.*;

@Test
class UtilsTest {
    Utils u;

    @BeforeEach
    void setUp() {
        u = new Utils(2);
    }

    void getNumFromStringSmallerTest(String[] args) {
        assertEquals(u.getNumFromString("F"), 0.0);
    }

    void getNumFromStringLargerTest(String[] args) {
        assertEquals(u.getNumFromString("Foo"), 0.9, 0.2F);
    }
}
```

- La classe di test è sintatticamente ben formata? Se no, indicare come sarebbe possibile correggere la classe di test (N.B.: è possibile siano necessarie più correzioni).
- Al netto delle eventuali correzioni, qual è il risultato dell'esecuzione della classe di test tramite il runner di JUnit? Quanti test vengono eseguiti complessivamente? Quanti di loro falliscono?
- Quanti oggetti di tipo Utils vengono creati durante l'esecuzione della classe di test?

### Modulo A - Esercizio 3

Si vuole effettuare il deployment di un'applicazione web realizzata per supportare il turismo. L'applicazione, che è scritta in Python utilizzando il framework Flask, permette di visualizzare luoghi di interesse su una mappa e di calcolare percorsi da una certa origine a una certa destinazione che passino per il maggior numero possibile di punti di interesse. Per il calcolo dei percorsi, l'applicazione utilizza "The Open Source Routing Machine (OSRM)", un componente open source off-the-shelf scritto in C e facilmente installabile su qualsiasi sistema Linux.

Si realizzi uno schema architettonico per il deployment in public cloud AWS (o Azure) dell'applicazione web e delle sue dipendenze, indicando quali servizi utilizzare per il deployment di quali componenti. Per fare fronte in modo economicamente efficiente ai picchi di utilizzo stagionali, si richiede esplicitamente che il deployment sfrutti le possibilità di elasticità offerte dai provider di servizi public cloud. Spiegare in che modo l'architettura proposta garantisce elasticità al sistema.

### **Modulo B**

Si realizzi un'applicazione che consenta l'acquisto di videogame per diverse piattaforme (PC e console di vario genere). L'utente può registrarsi alla piattaforma ed effettuare il login, quest'ultimo è indispensabile per procedere all'acquisto di giochi. Sarà possibile effettuare una ricerca per titolo (opzionale) e/o impostare dei filtri per piattaforma, tipologia di gioco (picchiaduro, fps, gdr, ecc.), fascia di prezzo, disponibilità ed età minima. Inoltre i risultati potranno essere ordinati per rating, prezzo o titolo. Dopo aver visualizzato una scheda di riepilogo del titolo selezionato sarà possibile procedere con l'acquisto inserendo i dati del metodo di pagamento. Infine, sarà possibile accedere alla biblioteca dei titoli acquistati e, a patto di aver giocato almeno 2 ore, esprimere un giudizio che concorrerà al calcolo del rating.

- 1) Si realizzino i mockup dell'applicazione.
- 2) Tenendo conto delle interfacce, realizzare gli statechart che rappresentano le funzionalità di ricerca, acquisto e valutazione dei titoli.
- 3) Definire un piano per valutare l'usabilità del prodotto in due fasi della progettazione, la prima, astratta, basata su prototipi e la seconda in betatesting sul campo, simulando la presenza in app di un sistema di monitoraggio in background.