SAPIENZA Università di Roma
Facoltà di Ingegneria dell'Informazione, Informatica e Statistica
Corsi di Laurea in Ingegneria Informatica ed Automatica
Corso di Progettazione del Software
Esame del **2 aprile 2022**
Tempo a disposizione: 3 ore

Requisiti. L'applicazione da progettare riguarda un videogioco basato sul gioco dell'oca. Un tabellone è caratterizzato da un link a un file immagine di background (una stringa), una sequenza non vuota di caselle e un insieme di giocatori (almeno 2). Ciascun giocatore appartiene ad un solo tabellone e ha un nome (una stringa), ed è posizionato su una casella. Le caselle appartengono ad un unico tabellone e hanno un link all'immagine della stessa (una stringa). Le caselle sono suddivise in 3 categorie: caselle normali, caselle con bonus e caselle di salto. Le caselle normali hanno un colore (una stringa). Le caselle con bonus hanno un intero che rappresenta i punti che un giocatore acquisisce fermandosi nella casella. Le caselle con salto sono associate ad un'altra casella dello stesso tabellone cosicchè quando un giocatore si ferma in una casella di salto cambia posizione saltando alla casella indicata.

Siamo interessati al comportamento dei giocatori. Un giocatore è inizialmente a *riposo*. Se riceve il comando di *si gioca* si posiziona nella prima casella del tabellone e passa allo stato *in gioco*. Quando *in gioco* riceve l'evento *lancio del dado* con payload il numero n di caselle si sposta di n caselle (se contando n si arriva all'ultima casella si ricomincia dalla prima) e se la casella di arrivo è una casella normale non fa altro, se è una casella bonus incrementa il suo punteggio secondo quanto indicato nella casella, e se è una casella di salto, salta alla casella indicata. Infine se quando *in gioco* il giocatore riceve il comando *fine* torna a *riposo*.

Siamo interessati alla seguente attività principale. L'attività prende in input un tabellone T e verifica che il numero di caselle sia almeno 50 e che le caselle delle varie categorie siano appropriate (i dettagli non interessano). Se la verifica non va a buon fine, l'attività termina segnalando in output un errore. Altrimenti concorrentemente esegue le seguenti due sottoattività: *(i)* gioco e *(ii)* analisi. La sottoattività di gioco *(i)* avvia il gioco attivando tutti i giocatori di T mandando opportuni eventi (i dettagli non interessano). Poi si mette in attesa del segnale di fine-gioco che interrompe il gioco stesso. La sottoattività di analisi *(ii)* calcola un report sul tabellone T (i dettagli non interessano). Una volta che tali sottoattività sono state completate, l'attività principale calcola i punti acquisiti da ogni giocatore durante la sottoattività di gioco, poi manda un segnale di output con i punti e con il report calcolato dalla sottoattività di analisi e infine termina.

Domanda 1. Basandosi sui requisiti riportati sopra, effettuare l'analisi producendo lo schema concettuale in UML per l'applicazione, comprensivo di: diagramma delle classi (inclusi eventuali vincoli non esprimibili in UML); diagramma stati e transizioni per la classe *Giocatore*; diagramma delle attività; specifica del diagramma stati e transizioni; segnatura dell'attività principale, sottoattività non atomiche, atomiche e segnali di input/output. Si noti che NON è richiesta la specifica delle attività. Motivare, qualora ce ne fosse bisogno, le scelte di progetto.

Domanda 2. Effettuare il progetto, illustrando i prodotti rilevanti di tale fase e motivando, qualora ce ne fosse bisogno, le scelte di progetto. In particolare definire SOLO le responsabilità sulle associazioni del diagramma delle classi.

Domanda 3. Effettuare la realizzazione, producendo un programma JAVA e motivando, qualora ce ne fosse bisogno, le scelte di progetto. In particolare, realizzare in JAVA SOLO i seguenti aspetti dello schema concettuale:

- La classe *Giocatore* con la classe *GiocatoreFired*, le classi Java per rappresentare le associazioni di cui la classe *Giocatore* ha responsabilità.
- L'attività principale e le sue eventuali sottoattività NON atomiche.