SAPIENZA Università di Roma
Facoltà di Ingegneria dell'Informazione, Informatica e Statistica
Corso di Laurea in Ingegneria Informatica ed Automatica
Corso di Progettazione del Software
Esame del **29 gennaio 2020**
Tempo a disposizione: 3 ore

Requisiti. L'applicazione da progettare riguarda una parte di un videogioco di biliardo robotizzato. Un gioco è costituito da almeno due giocatori e un tavolo. Ciascun giocatore ha un nome (una stringa) ed un insieme non vuoto di palle robotizzate. Un tavolo è rettangolare ed è caratterizzato da una lunghezza e una larghezza ed un insieme non vuoto di buche, ciascuna in una posizione del tavolo (determinate da coordinate cartesiane). Ciascuna buca appartiene esclusivamente ad un tavolo e può essere prenotata da una palla. Ciascuna palla robotizzata appartiene ad esattamente un giocatore, ha un nome e una posizione corrente sul tavolo (in coordinate cartesiane) e può muoversi per cambiare posizione. Le palle sono divise in "pesanti" e "leggere" e quelle leggere hanno un colore (una stringa), quelle pesanti invece non hanno un colore perchè sono cromate. Tutti i movimenti sono caratterizzati dalle leggi della fisica, ma i dettagli per questo compito non interessano. Tuttavia la massa delle palle leggere è trascurabile rispetto a quelle pesanti e in una collisione queste ultime NON modificano la propria traiettoria (mentre quelle leggere sì).

Siamo interessati al comportamento delle palle. Una palla è inizialmente *inattiva*. Se riceve dal proprio giocatore il comando *vai in buca* con payload la buca in cui deve andare, chiede alla buca di prenotarla mettendosi in *attesa* della risposta. Se la buca gli risponde *prenotata* (questo significa che era libera e ora è prenotata per la palla in questione), allora si mette in *movimento* verso la buca; altrimenti torna *inattiva* segnalando al proprio giocatore il fallimento. Quando la buca gli dice che l'ha *raggiunta*, mette come propria posizione corrente la posizione della buca, e diviene di nuovo *inattiva*. Se mentre è in *movimento* riceve un evento *collisione* con payload un'altra palla, se lei stessa è una palla pesante e l'altra palla è leggera, ignora l'evento; altrimenti si mette in uno stato *temporaneo* e quando si *ferma* (un evento esterno con payload la posizione), modifica la propria posizione corrente nella nuova e si mette nello stato di *inattiva*.

Siamo interessati alla seguente attività principale. L'attività prende in input un gioco G e verifica che il numero di palle pesanti sia al massimo il 10% del numero di palle leggere. 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, palle e buche di G 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 i colori delle palle leggere presenti nel gioco e per ciascuno il numero di palle di quel colore e prepara un report con questi dati (una stringa). Una volta che tali sottoattività sono state completate, l'attività principale manda un segnale di output con il report calcolato nella sottoattività di analisi e 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 vincoli non esprimibili in UML); diagramma stati e transizioni per la classe *Palla*; 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, con i relativi motivi (molteplicità, operazioni, requisiti).

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 `Palla` con classe `PallaFired`, le eventuali sottoclassi, e le classi JAVA per rappresentare le *associazioni* di cui la classe `Palla` ha responsabilità.
- L'*attività principale* e le sue eventuali sottoattività NON atomiche.