Scommesse

Uno scommettitore ogni domenica esegue 5 scommesse sulle 5 partite di calcio di un campionato (assumi che il campionato ha solo 10 squadre). La vincita/perdita è regolata dalle seguenti regole:

1. se indovina il risultato (UNO, DUE o X), vince un euro;
2. se non indovina il risultato e c'è stato un pareggio (X), perde un euro;
3. se non indovina il risultato e una squadra ha vinto (UNO o DUE), perde due euro.

Lo scommettitore si è imposto delle regole per le sue scommesse:

- per ogni partita ha un budget separato; quindi la vincita/perdita di ogni scommessa incrementa/decrementa il budget riservato alla singola partita;
- se arriva ad almeno 50 euro in totale allora si ferma;
- se il budget di almeno una partita non permette di ripagare la perdita massima per quella partita, smette di scommettere.
- Naturalmente se dovesse perdere tutto, si ferma.

All'inizio lo scommettitore ha riservato 5 euro per ogni partita.

Compito

In un file word annota tutto quello che fai. Aggiungi anche gli screenshot per i punti principali (ad esempio copertura ...). Per effettuare gli screenshots usa lo strumento di cattura di windows. Anche la documentazione sarà valutata.

Sviluppa un progetto in eclipse per ognuna delle domande. Ogni progetto dovrà avere come nome il tuo COGNOME_**ES** (con ES il tipo dell'esercizio – vedi titoli degli esercizi). Puoi anche andare in parallelo (ad esempio modello NUSMV e codice Java). Metti i progetti e il documento word in una directory con nome COGNOME_NOME (che puoi usare come workspace). Alla fine crea un file zip e consegna quello.

Commenta per bene tutto il codice che scrivi.

Java

In Java scrivi una classe Scommesse che implementa il sistema sopra. Ha tre metodi:

void valuta(int[] **scommesse**, int[] **risultati**)

void valuta(int scommessa, int risultato, int partita)

boolean finito()

Il primo metodo aggiorna il budget delle scommesse chiamando il secondo metodo per tutte le partite. **scommesse** è un array che contiene le 5 scommesse e **risultati** contiene i 5 risultati. Le scommesse e i risultati sono codificati come interi (UNO -> 1, DUE -> 2, X -> 0).

Il secondo metodo aggiorna il budget per la singola partita.

Il terzo metodo restituisce se ha finito (o ha raggiunto la cifra desiderata o non riesce a pagare qualche partita)

1. ES = TESTING

Scrivere l'implementazione (se vuoi qui usa gli enumerativi per scommesse e risultati: UNO, DUE,

X) e i casi di test con Junit. Scrivi una classe di test VittoriaTest per lo scenario in cui a partire dalla configurazione iniziale il giocatore arriva ad almeno 50 euro e smette.

Esegui il controllo della copertura (istruzioni, branch, decisioni, MCDC). Se il caso sopra non è sufficiente ad avere una copertura soddisfacente, aggiungi i casi di test necessari (in una classe separata per ogni copertura).
Metti la directory test con i test separata (oltre src).
Prova con Randoop a generare casi di test e prova a valutarne la copertura.

2. ES = JML

Scrivi i contratti JML. Cerca di scrivere sia le precondizioni, che le postcondizioni dei metodi. Cerca di scrivere anche invarianti. Cerca di spostare nelle precondizioni alcuni controlli che facevi all'inizio dei metodi.
Prova i contratti JML con una classe main in cui chiami i diversi metodi.

Prova anche a modificare il codice e controlla che i contratti siano violati. Documenta bene le violazioni e le loro cause in commenti e nel file di documento.

3. ES = KEY Verifica dei programmi

Prova a verificare qualche contratto. Ricorda: togli tutti i system.out e l'uso delle librerie (ad esempio Math). Ricorda di creare un nuovo progetto e convertirlo in key. Deve apparire l'icona di key e la directory con i proofs. Evita anche di usare i float e double (non dovrebbero servire).

4. ES = NUMSV

Specificare lo scommettitore utilizzando NuSMV e le sue proprietà usando LTL e/o CTL.

(qui usa gli enumerativi per UNO, DUE, X).

Suggerimento: usa i moduli per valutare la singola partita.

Tra le proprietà desiderate (alcune potrebbero essere false) c'è:

- esiste un cammino in cui lo scommettitore riesce ad arrivare al budget massimo
- esiste uno stato in cui lo scommettitore ha il budget azzerato per tutte le partite;
- non può mai perdere tutto (arrivare a zero)

Provare se sono vere oppure no. Se sono false presentare un contro esempio e modificarle in modo che siano vere.

Aggiungi anche altre proprietà che ti sembrano importanti e commenta.

Scrivi anche alcune proprietà sia di liveness che di safety e provale. Spiegale nel documento.

5. ES = FSM (MBT con le FSM)

Considera lo scommettitore su una singola partita che smette quando arriva a 10 euro. Modella l'evoluzione del sistema con una FSM. Implementa la FSM con modelJunit.

Salva il disegno della macchina come immagine.

6. ES = COMB (Combinatorial testing)

Rappresenta in combinatorial testing la chiamata del metodo void valuta(int[] **scommesse**, int[] **risultati**).

(considera come variabili del problema le 5 scommesse e i 5 risultati)

Se riesci cerca di rappresentare nel problema anche l'incremento del budget (sulle singole partite). Introduci i vincoli opportuni.In questo modo hai anche l'oracolo.

Applica il combinatorial testing. Prova a generare la copertura pairwise. Prova a tradurre una riga dal tuo test in un test Junit per la classe Scommesse.