Ingegneria del Software con Laboratorio (12 crediti) – Programma a.a. 2010-11
Prova Scritta del 16 gennaio 2012

Tempo a disposizione: 3 ore (Solo parte generale: 1 ora e 30', solo UML: 1 ora e 30').

La parte generale oltre ad 1 esercizio prevede 2 domande che verranno distribuite 45' prima della scadenza per la consegna – da quel momento non sarà più consentita la consultazione del materiale.

Svolgere parte UML e parte generale su fogli separati.

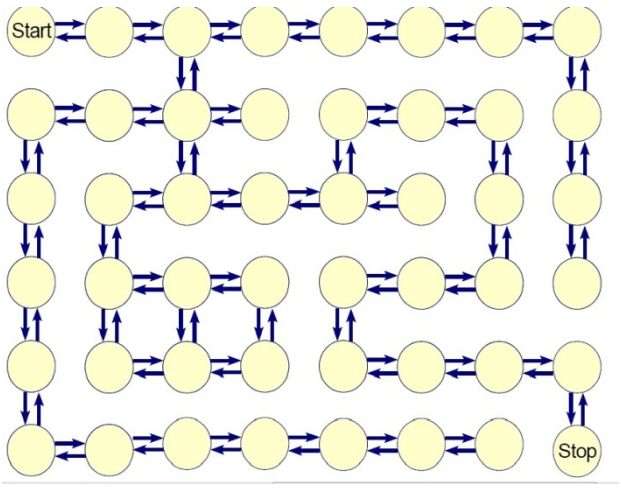
Esercizio 1 – Parte UML

Considerare il seguente testo che descrive come avviene il processo di revisione di una pubblicazione scientifica (nel seguito **paper**) da parte di una rivista. Il ricercatore manda il paper alla rivista. L'editor dopo avere ricevuto il paper lo mette nello stato **submitted** finchè non ha selezionato tre revisori, i quali hanno il compito di rivedere il paper e dare un punteggio (0=scadente, 1=sufficiente e 2=buono). Quando l'editor seleziona i revisori il paper passo nello stato **under-review**. Se il paper ottiene dai revisori tre voti maggiori o uguali a 1 il paper passa nello stato **accettato**, se ottiene almeno due 0 passa nello stato **respinto** e in tutti gli altri casi passa nello stato **major-review**. In tutti e tre i casi la notifica dei revisori e la valutazione finale (accettato, respinto o major-review) viene mandata al ricercatore. Se il paper ottiene una major-review allora il ricercatore ha tempo due mesi per sistemare il paper secondo le richieste dei revisori e risottomettere il paper per un altro ciclo di revisione. Se invece il paper viene accettato passa contemporaneamente al tecnico correttore (che corregge eventuali errori nel testo) e al tecnico grafico (che modifica/corregge le immagini). I due tecnici possono lavorare al paper in modo indipendente e in parallelo. Alla fine il paper viene passato al tecnico della stampa che lo pubblica sulla rivista.

a) Scegliere il diagramma UML più opportuno e rappresentare il workflow descritto sopra.
b) **S**crivere in pseudocodice l'algoritmo descritto sopra per attribuire la valutazione ad un paper.
c) Rappresentare tramite state machine il ciclo di vita di un paper.

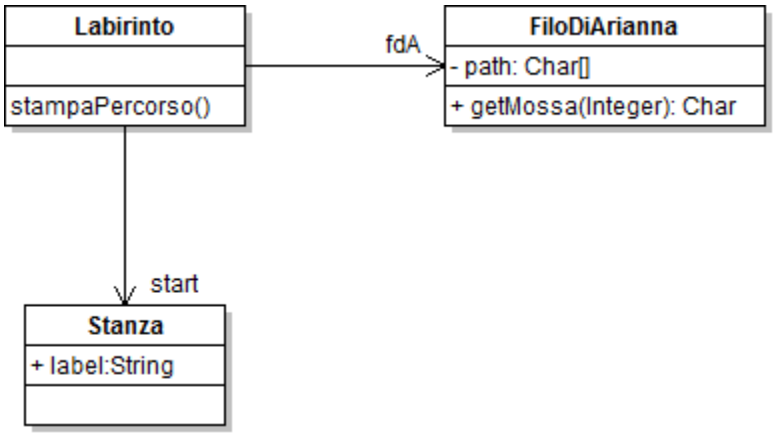
Esercizio 2 – Parte UML

Un labirinto è costruito secondo il meccanismo illustrato dal disegno dato sotto. Ogni stanza possiede quattro porte: una per tornare indietro e tre per entrare in altre stanze; alcune porte possono essere chiuse (cioè non portano ad alcuna stanza). Ogni stanza è etichettata con una stringa diversa (label). Il percorso, partendo dalla stanza contrassegnata con *Start* e arrivando alla stanza contrassegnata con *Stop*, è unico.



a) Completare il diagramma delle classi UML dato sotto che permetta la realizzazione di oggetti di tipo *Labirinto*. La porta per tornare indietro è contrassegnata con R, le altre porte con A, B, C. Per percorrere il labirinto abbiamo a disposizione una classe, connessa al *Labirinto*, chiamata *FiloDiArianna*, che ha un attributo (*path*) che contiene tutte le mosse (ovvero una sequenza di A, B, C e R), esclusa la prima che è sempre start, che costituiscono il percorso da start a stop

b) Descrivere mediante il diagramma UML che si ritiene più opportuno o in pseudocodice Java il comportamento dell'operazione *stampaPercorso*() della classe *Labirinto*. Tale operazione stampa i nomi (cioè la *label*) di tutte le porte che costituiscono il percorso da start a stop. L'operazione *getMossa*(i) restituisce l'i-esima mossa.



Esercizio 3 - Parte Generale

Considerare le due componenti (classi pseudo-Java, in cui sono omessi i cast tra oggetti e tipi base) seguenti:

```
public class Server {
  public static int alfa;
  public static bool Cerca(Lista x){
  do { if (x.info == alfa) return true;
       x = x.next;
     }
  while (x != null);
  return false;
  }
}


public class Client {
  static List MiaLista() { ... }
  public static void main(String[] args)
  {
     List s = MiaLista();
     if (s != null) {
       System.out.print("Inserisci un intero: ");
       Server.alfa = InOut.readInt();
       if (Server.Cerca(s))
          System.out.print(Server.alfa+"presente");
       else
          System.out.print(Server.alfa+"non presente");
     }
  }
```

a) Rilevare eventuali problemi delle due componenti legati alla mancata applicazione di principi di progettazione, discutendone le implicazioni.

b) Proporre una riprogettazione delle due componenti che risolva eventuali problemi rilevati in **a).**

Domanda 4 – Parte Generale

Fare riferimento al workflow descritto nell'esercizio 1. Supponendo di dovere implementare tale workflow in un applicazione Web

a) Quali individuereste come attori primari e secondari del sistema?

b) Quali domande vi piacerebbe porre per chiarire/approfondire il workflow (ci sono degli aspetti non specificati) e a chi? Scrivere almeno due domande.

c) Fornire almeno tre esempi di requisiti funzionali e tre di requisiti non funzionali per il sistema.

Domanda 5 – Parte Generale

Illustrare su un esempio a vostra scelta le principali differenze e i più significativi pro e contro di testing white box/black box. In particolare evidenziare sull'esempio almeno un caso in cui il testing black box rivela un errore e quello white box no, e viceversa.