PROGRAMMAZIONE II (A,B) - a.a. 2014-15
Appello Straordinario — Novembre 2015

Esercizio 1

Si consideri il tipo di dati astratto modificabile `BoxOffice` che intende rappresentare i dati relativi alla vendita di biglietti per un evento. L'interfaccia Java presentata di seguito rappresenta una descrizione parziale dell'astrazione `BoxOffice`.

```
public interface BoxOffice {

  /* restituisce la capienza dell'evento */
  public int getCapacity();

  /* restituisce il numero di biglietti ancora disponibili */
  public int ticketAvailable();

  /* restituisce il costo del singolo biglietto */
  public int getPrice();

 /* richiesta di acquisto di biglietti da parte di un rivenditore autorizzato */
  public boolean sellTicket(int num, String seller);

  /* richiesta di biglietti omaggio da parte di un rivenditore autorizzato */
  public boolean getTicket(int num, String seller);
}
```

1. Assumendo di adottare una strategia di programmazione difensiva, si completi la specifica del tipo di dato `BoxOffice`, definendo le clausole REQUIRES, MODIFIES e EFFECTS di ogni metodo, incluso i costruttori, indicando le eccezioni eventualmente lanciate e se sono checked o unchecked.

2. Si definiscano la struttura di implementazione concreta, la funzione di astrazione e l'invariante di rappresentazione.

3. Si fornisca l'implementazione del metodo `sellTicket` e si dimostri che l'implementazione preserva l'invariante di rappresentazione.

4. Supponiamo di estendere l'astrazione `BoxOffice`. Si definisca la classe `BrockeredBoxOffice` che estende `BoxOffice` in modo tale che un rivenditore autorizzato possa richiedere un numero massimo di biglietti (comprensivo di quelli omaggio, lo stesso numero per ogni rivenditore). Si fornisca la specifica e l'implementazione del tipo di dati `BrockeredBoxOffice` e si indichi, motivando la risposta, se è soddisfatto il principio di sostituzione.

Esercizio 2

Si consideri il seguente programma OCAML

```
let rec repeat n a =
  if n = 0 then [] else a :: (repeat (n-1) a);;

let rec transform f l =
  begin match l with
  | [] -> []
  | h :: ls -> f(h) @ (transform f ls)
  end;;

let myfun n =
  let addN x = [] in
    let addL x = repeat 2 x in
      if n < 0 then addN else addL;;

transform (myfun 5) [1;2];;
```

1. Si descriva lo stato dello stack dei record di attivazione nella simulazione della valutazione del programma.

Esercizio 3

Si consideri il seguente frammento di codice Java

```
void start() {
  A a = new A();      /* Linea 1 */
  B b = new B();      /* Linea 2 */
  a.s(b);             /* Linea 3 */
  b = null;           /* Linea 4 */
  a = null;           /* Linea 5 */
  System.out.println("start completed"); /* Linea 6 */
}
```

1. Si indichi, motivando la risposta, quando l'oggetto `b` creato alla linea 2 diventa garbage.