PROGRAMMAZIONE II (A,B) - a.a. 2017-18
Seconda Valutazione Intermedia – 18 Dicembre 2017

Esercizio 1

Una funzione a dominio finito è una funzione che è definita solo per un numero finito di elementi. Ad esempio si consideri la seguente funzione con una sintassi nello stile di OCaML

```
let sum = fun y -> 50 + y for y in [0; 1; 2; 3; 4];;
```

La funzione `sum` è definita solamente per valori del parametro attuale che appartengono all'insieme $\{0, 1, 2, 3, 4\}$, insieme che è calcolato al momento della definizione della funzione stessa.

1. Si estenda la sintassi astratta del linguaggio didattico funzionale senza funzioni ricorsive in modo da includere tali funzioni.

   Assumendo di modellare solo funzioni unarie, come specificato anche durante lo svolgimento della verifica, e ricordandosi di non considerare la ricorsione, si ottiene la seguente soluzione

```
type exp = ...
   | DFun of ide * exp * exp list

type evT = ...
   | DFunVal of ide * exp * evT list * evT env
```

2. Si definiscano le regole OCaML dell'interprete per trattare la valutazione di dichiarazione e la chiamata di funzioni a dominio finito.

```
let rec eval (e : exp) (r : evT env) : evT = match e with
   ...
   | DFun(i, a, exL) -> let vL = (evalList exL r) in DFunVal(i, a, vL, r)
   ...
   | FunCall(f, eArg) ->
         let fClosure = (eval f r) in
            (match fClosure with
                ...
                DFunVal(arg, fBody, fDom, fDecEnv) ->
                      let v = (eval eArg r) in
                          if (check v fDom) then eval fBody (bind fDecEnv arg v)
                                            else Unbound |
                _ -> failwith("non functional value")) |
   ...
and evalList (lst : exp list) (r : evT env) : evT list = match lst with
     [ ] -> [ ] |
     e :: rest ->
         let v = (eval e r) in v :: evalList rest r |
and check (v : evT) (lst : evT list) : bool = match lst with
     [ ] -> false |
     val :: rest -> if (evTeq v val) then true else (check v rest)
and evTeq (v1 : evT) (v2 : evT) : bool = match lst with
     ... (* adeguata nozione di eguaglianza su evT *);;
```

Esercizio 2

Si consideri il seguente programma scritto in OCaML.

```
let n = 4;;

let f = (fun x -> n + x);;

let g x y = (x + y) * x in
  let rec h m n = if n < 2 then m 2
                  else g (m 2) (h (fun x -> n) (n - 2)) in
    h f 4;;
```

1. Si determini il tipo inferito dall'interprete OCaML per gli identificatori di funzione (`f`, `g` e `h`) che compaiono nel programma scelto.

```
val f : int -> int = <fun>
val g : int -> int -> int = <fun>
val h : (int -> int) -> int -> int = <fun>
```

2. Si simuli la valutazione del programma mostrando la struttura della pila dei record di attivazione.

3. Si determini il valore calcolato dal programma. `180`

Esercizio 3

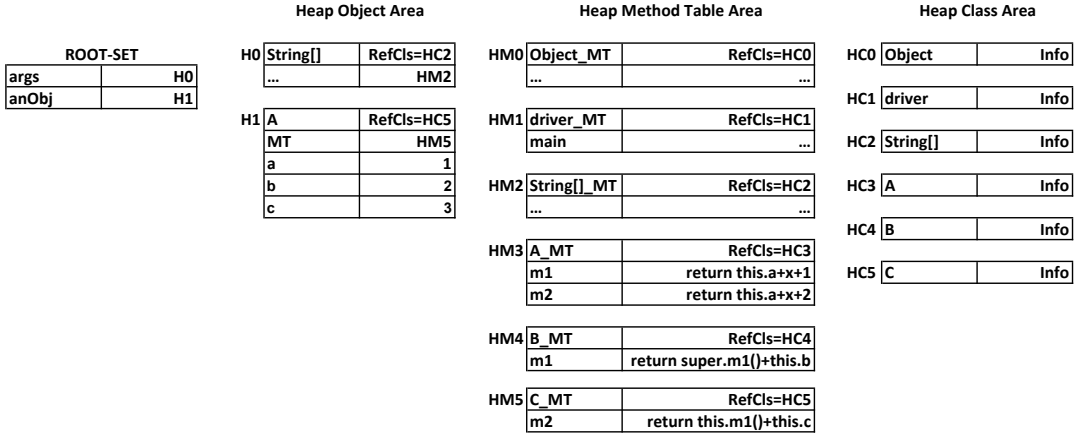
Si consideri il seguente programma scritto in una notazione Java-like.

```
class A {
  private int a = 1;
  public int m1(int x) { return a + x + 1; }
  public int m2(int x) { return a + x + 2; }
}

class B extends A {
  private int b = 2;
  public int m1(int x) { return super.m1(x) + b; }
}

class C extends B {
  private int c = 3;
  public int m2(int x) { return m1(x) + c; }
}

class driver {
  public static void main (String args[]) {
     A anObj = null;
     if (args[0] == "uno") anObj = new B();
                           else anObj = new C();
     System.out.println(anObj.m1(5) + anObj.m2(5)); //(1)
  }
}
```

1. Si descriva l'ordine di caricamento delle classi durante l'esecuzione del comando `java driver "due"`.

```
Object  driver  String[]  A  B  C
```

2. Si simuli la struttura del runtime quando l'esecuzione dell'istruzione `(1)` termina. In particolare si descriva la struttura degli oggetti sullo heap e la struttura delle tabelle dei metodi delle classi.



3. Supponendo di avere ereditarietà multipla e di estendere il programma con la seguente dichiarazione

```
class D extends B, C {
  private int d = 4;
  public int m3(int x) { return x * c; }
}
```

si descriva, motivando la risposta, la struttura della tabella dei metodi di un oggetto di tipo `D`.