Corso di Paradigmi di Programmazione
Prova scritta del 26 giugno 2012.

Tempo a disposizione: ore 2.
Svolgere gli esercizi 1-4 e 5-8 su due fogli differenti.

1. Si consideri la seguente grammatica $G = (\{S, T\}, \{a, b\}, S, P)$ dove P è dato dalle seguenti produzioni:

$$\begin{array}{lcl} S & ::= & SaS \mid TbT \mid T \\ T & ::= & TaT \mid S \mid a \end{array}$$

   Si mostri che G è ambigua.

2. Con la notazione $\mathcal{C}^{L}_{L_1,L_2}$ indichiamo un compilatore da L_1 a L_2 scritto in L. Con $\mathcal{I}^{L}_{L_1}$ indichiamo un interprete scritto in L per il linguaggio L_1; se P è un programma in L_1 e x un suo dato, $\mathcal{I}^{L}_{L_1}(P, x)$ indica l'applicazione dell'interprete a P e x. Si dica se la valutazione di $\mathcal{I}^{L}_{L_1}(\mathcal{C}^{L_1}_{L_1,L}, \mathcal{I}^{L_1}_{L})$ produce un qualche risultato motivando la risposta.

3. Si consideri il seguente frammento in uno pseudolinguaggio con modello delle variabili a riferimento e garbage collector mediante contatori dei riferimenti; se `OGG` è un generico oggetto nello heap, indichiamo con `OGG.cont` il suo contatore (nascosto).

```
class C{int n; C next;}
C v;                                                          2
C foo(){
   C p = new C();      // oggetto OGG1                        4
   p.next = new C();   // oggetto OGG2
   C q = new C();      // oggetto OGG3                        6
   q.next = p.next;
   v = p.next;                                                8
   return p.next;
   }                                                          10
C r = foo();
```

   Si dica quali sono i valori dei contatori dei riferimenti dei tre oggetti dopo l'esecuzione della linea 6 e della linea 9.

4. In uno pseudolinguaggio con eccezioni (`try/catch`) si incontra il seguente blocco di codice:

```
void ecc(int para1)  {
   try {throw new X();} catch (X) {write(1);}
  }
void f (int para2) {
  if (para2 == 1) {ecc(1);}
  try { ecc(2);} catch (X) {write{2);}
  }

void main () {
    try {f(1);} catch (X) {write(3);}
    try {f(2);} catch (X) {write(4);}
}
```

   Si dica cosa viene stampato all'esecuzione di `main()`.

5. Si consideri il seguente frammento in uno pseudolinguaggio con tipi statici, dove `f` è una certa funzione di due argomenti:

```
int i,j;
float y,z;
y = f(i,j);
z = f(y,i);
```

   Si fornisca (i) una possibile intestazione per la funzione `f` e (ii) le ipotesi che occorre fare sul sistema di tipi dello pseudolinguaggio affinché l'intestazione data in (i) sia corretta.

6. Facendo riferimento a uno o più linguaggi di programmazione reali, si fornisca un esempio di due tipi di dato che sono equivalenti strutturalmente ma non per nome. Si forniscano quindi due esempi di tipi che sono compatibili ma non equivalenti.

7. Cosa stampa il seguente frammento di codice in un linguaggio con scope statico e passaggio per riferimento ? L'ordine di valutazione delle espressioni influenza il risultato? Perchè?

```
{int x = 1;
int A[5];
int foo (){return x++};
for (int i=0, i<5, i++)
   A[i]=i;
int f(reference int a){
   int x = 5;
   a = x+a;
   a = x+a;
   return a;
   }
write (f(A[foo()+x]) + A[x] + x);
}
```

8. Sono date le seguenti definizioni di funzione:

```
int fact_falso (int n){
    if (n == 0) return 1;
    else return fact_falso(n+1);
}
int fact_vero (int n){
     if (n == 0) return 1;
    else return n*fact_vero(n-1);

}
```

   Una certa implementazione del linguaggio si comporta nel modo seguente. Alla chiamata `fact_falso(1)`, non risponde, rimanendo in un loop infinito. Alla chiamata `fact_vero(-1)` risponde dopo qualche tempo con `Stack overflow during evaluation`, abortendo l'esecuzione. Si dia una spiegazione motivata di questi due fatti.