

# Linguaggi di Programmazione: Esercitazione

Francesca Del Bonifro<sup>1</sup>

<sup>1</sup>Dipartimento di Informatica - Scienza e Ingegneria - UniBO

12/10/2020

## Implementazione di Linguaggi

# Esercizio 1

Con la notazione  $C_{L_1, L_2}^L$  indichiamo un compilatore da  $L_1$  a  $L_2$  scritto in  $L$ . Con  $I_{L_1}^L$  indichiamo un interprete scritto in  $L$  per il linguaggio  $L_1$ . Se  $P$  é un programma in  $L_1$  e  $x$  un suo dato,  $I_{L_1}^L(P, x)$  indica l'applicazione dell'interprete a  $P$  e  $x$ . Si dica se la seguente scrittura ha senso

$$I_L^L(C_{L, L_1}^L, I_{L_1}^L).$$

Se la risposta é “no”, si motivi tale fatto; se é “sí” si dica qual è il risultato ottenuto.

## Esercizio 2

Ricordando che  $I_{L_1}^{L_0}$  denota un interprete scritto in  $L_0$  che interpreta programmi scritti in  $L_1$ , e che  $C_{L_2, L_3}^{L_1}$  denota un compilatore scritto in  $L_1$  che traduce programmi scritti in  $L_2$  in equivalenti programmi scritti in  $L_3$ , affinché la seguente espressione

$$I_X^{L_0} \left( C_{L_2, L_3}^{L_1}, C_{L_1, Z}^Y \right)$$

abbia senso, quali linguaggi devono essere assegnati alle variabili  $X$ ,  $Y$  e  $Z$ ? Nel caso in cui  $Z$  assuma valore  $L_1$ , il programma calcolato serve a qualcosa?

## Esercizio 3

Con la notazione  $C_{L_1, L_2}^L$  indichiamo un compilatore da  $L_1$  a  $L_2$  scritto in  $L$ . Con  $I_{L_1}^L$  indichiamo un interprete scritto in  $L$  per il linguaggio  $L_1$ ; se  $P$  é un programma in  $L_1$  e  $x$  un suo dato,  $I_{L_1}^L(P, x)$  indica l'applicazione dell'interprete a  $P$  e  $x$ . Si dica in meno di 10 parole cosa é  $I_{L_1}^L(C_{L_1, L_2}^{L_1}, C_{L_1, L_2}^{L_1})$ .

## Esercizio 4

La seguente espressione

$$I_{L_0}^{L_1} \left( C_{L_1, L_0}^{L_0}, I_{L_0}^{L_1} \right)$$

calcola qualcosa di utile? Se rimpiazziamo, nell'espressione sopra, la seconda occorrenza di  $I_{L_0}^{L_1}$  con  $I_{L_1}^{L_0}$ , cosa otteniamo?

## Regole di Scope

# Esercizio 1

Si dica cosa viene stampato dal seguente frammento di codice scritto in uno pseudo-linguaggio che usa scoping statico e passaggio di parametri per valore. La primitiva `write(x)` permette di stampare un valore intero.

```
{int x = 2;

void pippo(value int y){
    x = x + y;
}

{ int x = 5;
  pippo(x++);
  write(x);
}

write(x); }
```

(si ricordi che un comando della forma `foo(w++)`; passa a `foo` il valore corrente di `w` e poi incrementa `w` di uno)



## Esercizio 2

Si consideri il seguente frammento di programma scritto in un pseudo-linguaggio che usi scoping dinamico e dove la primitiva `read(Y)` permette di leggere nella variabile `Y` un intero dall'input standard, mentre `write(X)` permette di stampare il valore della variabile `X`. Si dica quali sono (o qual é) i valori stampati.

# Esercizio 2

## Scoping dinamico

```
...
int X;
X = 1;
int Y;

void fie {
  foo;
  X = 0;
}

void foo {
  int X;
  X = 5;
}

read(Y);

if Y > 0 then { int X;
                X = 4;
                fie;
              }
            else { fie;
              }

write(X);
```

## Esercizio 3

Si consideri il seguente frammento di programma scritto in un pseudo-linguaggio che usi scoping dinamico e dove la primitiva `read(Y)` permette di leggere nella variabile `Y` un intero dall'input standard, mentre `write(X)` permette di stampare il valore della variabile `X`. Si dica quali sono i valori stampati.

```
int X = 0;
int Y;
void pippo() {
    X++;
}
void pluto() {
    X++;
    pippo();
}
read(Y);
if Y > 0 then { int X = 5;
                pluto();
            }
            else { pluto();
            }
write(X);
```

## Esercizio 4

Si dica cosa stampa il seguente frammento in uno pseudolinguaggio scope statico (si ricordi che l'espressione `x++` restituisce il valore della variabile `x` e successivamente incrementa `x` di uno).

```
int x = 3;
void foo(int y) {
    int x = 5;
    x = x + y;
    x = x + y;
    write(x);
    write(y);
}
foo(x++);
write(x);
```

## Esercizio 5

Si dica cosa stampa il seguente frammento in uno pseudolinguaggio con passaggio per riferimento e scope statico.

```
int x = 2;
void foo(reference int y){
    x = x+1;
    y = y+10;
    x = x+y;
    write(x);
}
{int x = 10;
  foo(x);
  write(x);
}
```

## Esercizio 6

Si dica cosa stampa il seguente frammento in uno pseudolinguaggio con passaggio per valore e scope statico.

```
int x = 3;
int y = 4;
void foo(int y, int z) {
    int x = 5;
    y = y+1;
    if (z==y) write(x);
    else write (y);
}
foo(x,x);
write(x);
write(y);
```

# Esercizio 7

Si dica cosa stampa il seguente frammento in uno pseudolinguaggio con scope dinamico

```
int x = 4;
void foo(int y) {
    int w;
    x = x + y;
    w = y;
    write(w);
    write(y);
}
{int x = 10;
  foo(x);
  write(x);
}
```

## Esercizio 8

Si dica cosa viene stampato dal seguente frammento di codice scritto in uno pseudo-linguaggio che usa scoping statico e passaggio di parametri per valore e per riferimento.

```
int x = 0;
void pippo(value int y, rif int z){
    z = x + y + z;
}
{ int x = 1;
  int y = 10;
  int z = 20;
  pippo(x++, x);
  pippo(x++, x);
  write(x);
}
write(x);
```