Esercizi di Linguaggi di Programmazione

Luca Bedogni

April 14, 2015

Considerare un linguaggio che ammette il passaggio per nome. Dire cosa stampa il seguente codice.

```
int k = 2;
int A[5];
A = {1, 2, 4, 7, 3}
void swap(int name x, int name y) {
  int temp = x;
  x = y;
  y = temp;
  write(A);
}
swap(k,A[k]);
```

Dire cosa stampa il seguente codice.

```
#include <iostream>
#include <string.h>

using namespace std;

void swap(int a, int b) {
int t = a;
   a = b;
   b = t;
}

void swapVR(int &a, int &b) {
  int t = a;
  a = b;
  b = t;
}

int main() {
  int v = 5;
  int l[5] = {1, 2, 3, 4, 5};

  int t1 = v;
  int t2 = l[0];
```

```
  swapVR(t1, t2);

  v = t1;
  l[0] = t2;
  cout << "Value-Result: " << v << "   " << l[0] << endl;

  swap(t1, t2);
  cout << "Value       : " << t1 << "   " << t2 << endl;

  swapVR(t1, t2);
  cout << "Reference   : " << t1 << "   " << t2 << endl;
}
```

Dire cosa stampa il seguente codice.

```
#include <stdio.h>

int fie(int w) {
    int x = 0;
    printf("%d\n", (w++) + (x++) );
    printf("%d\n", (w++) + (x++) );
}

int main() {
    int x = 1;
    fie(x);
    printf("%d\n", x);
}
```

Dire cosa stampa il seguente codice.

```
#include <stdio.h>

int fie(int w, int z) {
    int x = 0;
    printf("%d\n", (w++) + (x++) + ++z);
    printf("%d\n", (w++) + (x++) + ++z);
}

int main() {
    int x = 1;
    int A[8];
    int i;
    for (i = 0; i < 8; i++)
        A[i]=i;

    fie(x, A[x]);
    printf("%d\n", x);
}
```

Dire cosa stampa il seguente codice con scoping statico/dinamico, deep binding/shallow binding.

```
int u = 42;
int v = 69;
int w = 17;

void add( int z ){
  u = v + u + z
}

void bar( function h ){
  int u = w;
  h(v)
}

void foo( int x, int w ){
  int v = x;
  bar(add)
}

foo(u,13)
print(u)
```

Considerare un linguaggio che ammette il passaggio per nome. Dire cosa stampa il seguente codice.

```
int x = 10
int y = 0
void foo(name int v, name int z) {
  v = 4
  z++
  if (z == v)
    write(x)
  else
    write(y)
}
foo(y,y)
foo(x,y)
```

Dire cosa stampa il seguente codice.

```
#include <iostream>
#include <string.h>
using namespace std;

void foo(int &y) {
    cout << y << endl;
    y = 3;
```

```
    cout << y << endl;
}

int main() {
    int x = 0;
    cout << x << endl;
    foo(x);
    cout << x << endl;
}
```

Considerare un linguaggio che ammette il passaggio per valore risultato. Dire cosa stampa il seguente codice.

```
int X[10];
int i = 1;
X[0] = 5;
X[1] = 5;
X[2] = 5;
void foo (value-result int Y,J) {
  X[J] = J - 1;
  write(Y);
  J++;
  X[J] = J;
  write(Y);
}
write(X[i]);
foo(X[i],i);
write(X[i]);
```

Considerare un linguaggio che ammette il passaggio per nome e per riferimento. Dire cosa stampa il seguente codice.

```
int a = 1
int f (reference int x) {
  int a = 15
  int g (reference int x) {
    return f(x) + a
  }
  if (x == 0) {
    return 1
  } else {
    x = x - 1
    a=a-1
    return g(x) + a
  }
}
write (f(a))
```

L'esecuzione del seguente frammento di codice su una certa implementazione risulta nella stampa dei valori 4 e 1.

```
int W[10];
int x = 4;
for (int i=0, i<10, i++)
  W[i]=i;
void foo(int x; int y) {
  x = x+1;
  y=1;
}
foo (x, W[x])
write (W[4])
write (W[5])
```

Si fornisca una possibile spiegazione.

Considerare un linguaggio che ammette il passaggio per nome e per riferimento. Dire cosa stampa il seguente codice.

```
int n = 0;
procedure p(value/name int k) {
  write(k);
  n = n + 1;
  write(k);
}

p(n);
```

Don Knut example.

```
def A(k, x1, x2, x3, x4, x5):
  def B():
    nonlocal k
    k -= 1
    return A(k, B, x1, x2, x3, x4)
  if k<=0:
    return x4()+x5()
  else:
    return B()


def uno():
    return 1
def menouno():
    return -1
def zero():
    return 0

print(A(3, uno, menouno, menouno, uno, zero))
```