Esercitazione di Linguaggi e Compilatori

Modulo Linguaggi

AA 2011-2012

1. Si consideri il seguente frammento di codice e si assuma la valutazione con scoping statico.

```
int x = 1; int y = 2;
  {
     void C(){
          int x = 3;
          {
            void D(){int y = 20;
                     print(x,y);
            }
            D();
          }
     }
       {
         void B(){
              int y = 4;
              {
                void D(){x=x+1;print(y);}
                C();
              }
         }
         B();
       }
  }
```

   Cosa viene stampato? Si mostri l'esecuzione fornendo l'evoluzione dello stack dei record di attivazione sino al momento in cui viene effettuato il pop del RA di `D()`.

2. Cosa sono e a cosa servono le **chiusure** (ingl: **closure/thunk**)? In Java è possibile passare come parametro un metodo ad un altro metodo? In C sono necessarie le chiusure?

3. Sia dato il seguente frammento di codice e si assuma la valutazione con scoping statico.

```
{
  int x = 1;
  void C(ref int a){
       a=a+1;
       int x = 3;
       void D(value int z){
            z = z+1;
            x=x+1;
            print(x);
       }
       D(x);
       print(x);
  }
  C(x);
  print(x);
}
```

   Cosa viene stampato? Si risponda mostrando l'evoluzione dello stack dei record di attivazione.

4. Sia σ_0 lo stato che assegna zero a tutti gli identificatori. Si calcoli lo stato σ risultante dalla computazione del seguente comando in σ_0:
   if$(x < 0)$
   then
   $x := x - 1$
   else
   $(x := 1;$
   while $(0 < x)$ **do** $x := x - 1)$.

5. Si mostri con un esempio la differenza tra *scoping statico* e *scoping dinamico* (ovvero si mostri un programma che se valutato con lo scoping

statico produce un risultato differente da quello ottenuto valutandolo con lo scoping dinamico).

6. Sia dato il seguente frammento di codice e si assuma la valutazione con scoping statico.

```
{
  int x = 4;
  {
    void f(){print(x);}
    {
       void g(h){int x=7; h();}
       {
         int x = 1; g(f);
       }
    }
  }
}
```

   Cosa viene stampato? Si mostri l'esecuzione fornendo l'evoluzione dello stack dei record di attivazione sino al momento in cui viene effettuato il pop del RA di `f()`.

7. Si consideri il seguente frammento di codice:

```
void f(int x, int y){
  x=x+1;
  y=1;
}
...
int i=1;
int[4] A={0,4,0};
f(i,A[i]);
print(A[1],A[2],i);
```

   Cosa viene stampato se i parametri sono passati per `valueresult`? Cosa viene stampato se i parametri sono passati per `name`?

8. Sia dato il seguente frammento di codice e si assuma la valutazione con scoping statico.

```
int x = 1; int y = 2;
  {
     void C(){
          int x = 3;
          {
            void C(){print(x);}
            C(); // last
          }
     }
    {
      void B(){
           int y = 4;
           {
             void D(){x=x+1;}
             C();
           }
      }
      B();
    }
}
```

   Cosa viene stampato? Si mostri l'esecuzione fornendo l'evoluzione dello stack dei record di attivazione sino al momento in cui viene effettuato il pop del RA di `C(); // last`.

9. Sia σ_0 lo stato che assegna zero a tutti gli identificatori. Si calcoli lo stato σ risultante dalla computazione del seguente comando in σ_0 :

   $x := 1;$
   (**while** $(x \geq 1)$ **do** $x := x - 1$);
   if $(x = 1)$ **then skip else** $x := 2$.

10. Sia σ_0 lo stato che assegna zero a tutti gli identificatori. Si calcoli lo stato σ risultante dalla computazione del seguente comando in σ_0 :

    $x := 1;$
    (**while** $(x \geq 1)$ **do** $x := x - 1$);
    if $(x = 0)$ **then** $y := 2;$
    $x := y;$

11. Si descriva il passaggio per nome.

12. Si mostri con un esempio che passaggio per riferimento e passaggio per value result non sono equivalenti.