Cognome e Nome:_____ Matr.:_____

linguaggi di programmazione – A
15 giugno 2022

Esercizio 1.A – rispondere, in maniera concisa, alle seguenti domande (12 punti)

1. Nella valutazione degli argomenti delle funzioni, in cosa differiscono Haskell e Scheme?

2. Che tipo di dati vengono memorizzati nell'heap?

3. Descrivere in quali situazione si ha un deadlock.

4. Quali sono le principali caratteristiche della programmazione orientata agli oggetti?

5. Cosa sono i tipi enumerazione?

6. Nella programmazione concorrente, cosa sono i semafori?

7. Quali sono le caratteristiche del passaggio dei parametri per riferimento?

8. Cos'è una dangling reference?

9. Cos'è il Java bytecode.

10. Che tipo di linguaggi vengono riconosciuti dagli analizzatori sintattici?

11. In un linguaggi di programmazione, cosa s'intende per binding?

12. Presentare un esempio di polimorfismo ad hoc.

13. Cosa si intende per valutatine corto-circuito un'operazione booleana?

14. Cosa si intende per effetto collaterale nella valutazione di un'espressione?

15. Cosa distingue l'iterazione determinata da quella indeterminata?

16. Cos'è la chiusura di Kleene.

17. Cosa distingue un linguaggio funzionale puro da un linguaggio funzionale non puro?

18. Cosa sono i moduli?

1. Cognome e Nome:_____ Matr.:_____

– 15 giugno 2022– A

Esercizio 2.A – Grammatiche (4 punti)

1. Date le seguenti regole di produzione:

$$S ::= \epsilon \mid aS \mid aSbS$$
$$P ::= \epsilon \mid aPbP$$

siano $L(S)$ e $L(P)$ i linguaggi generati dalle grammatiche con simboli iniziali S e P

- Si diano le stringhe di $L(S)$ e $L(P)$ di lunghezza ≤ 4;
- si caratterizzino le stringhe di $L(S)$ e $L(P)$;
- si dica se la grammatiche con simbolo iniziale S e P sono ambigue (mostrando un testimone dell'ambiguità oppure argomentando opportunamente sulla non ambiguità);
- si stabilisca se il linguaggio $L(S)$ è regolare (argomentando opportunamente la risposta), e nel caso, lo si descriva in linguaggio mediante un espressione regolare.

Esercizio 3.A – Stack di attivazione (6 punti)

1. Si mostri l'evoluzione dello stack di attivazione e dell'output dei due frammenti di programma seguenti. Si ipotizzi che il linguaggio C-like abbia scoping statico, assegnamento che calcola prima l-value e poi r-value, valutazione delle espressioni da sinistra a destra, valutazione degli argomenti da sinistra a destra, e indici vettori inizianti da 0. Inoltre, per il secondo frammento di codice, mostrare l'evoluzione del display.

```
char s[10] = "abcdefghij";
int i = 1;
void foo(ref int j, val char c)
{
  write(s[j++]);
  s[++i] = s[++j]
  write(s[--j], c);
}
write(foo(i, s[i--]));
write(i);
```

```
int x=1, y=2;
void Q(ref int r, val int v){
  v++;
  r = r + v + y;
  write(r, v);
};
void G (ref int y, int R(ref int, val int)){
  R(y, x);
  write(y);
};
G(y, Q));
write(x, y);
```

1. Cognome e Nome:_____ Matr.:_____

– 15 giugno 2022– A

Esercizio 4.A – Haskell (7 punti)

1. Scrivere le seguenti funzioni Haskell:

   - una funzione che in una lista di booleani conta il numero di elemeniti uguali a True presenti nella lista;
   - una funzione che data una lista di elementi ed un valore conta il numero di volte che il valore appare nella lista;
   - una funzione che data una lista determina il numero di massimo di ripetizioni, non necessariamente consecutive, di una stesso valore nella lista.

   Si supponga che gli elementi della lista siano confrontabili ma non ordinabili, ossia il tipo degli elementi della lista appartenga alla type class 'Eq' ma non alla typeclass 'Ord'.

   Giocando sul fatto di usare o meno: le funzioni 'fold', la ricorsione di coda o le funzione precedentemente definite, fornire un'implentazione alternativa per ciascuna delle funzioni precedenti.

   Si definisca il tipo di ogni funzione definita.

Esercizio 5.A – Eccezioni (2 punti)

1. Descrivere il comportamento del seguente programma con ognuno dei diversi meccanismi di passaggio dei parametri: valore, riferimento, valore-risultato, nome.

```
{ int y=2
  void f(___ int x) throws E() {
    x = x + y;
    throw new E();
    x = x + y;
  }
  try{ f(y) } catch (E) {y++};
  write (y);
}
```

Esercizio 6.A – Sistema di assegnazione di tipo (3 punti)

1. Nel sistema di tipi per il linguaggio F1, costruire la derivazione di tipo per la seguente espressione:

   `\z :  Bool * Nat .  if (first z) then (second z) + 1 else (second z) - 1`

   Nota: i primi passi di derivazione, a partire dagli assiomi, se elementari o ripetizioni di derivazioni già fatte, possono essere omessi.