Anteprima di test

parte pratica

Data: Fri Jul 4 13:26:56 2014 Punteggi massimi: 27

1. Java Lista di triple - generics *(5 Punti)*

Usando i generics in **Java**, scrivi una classe **Lista3** che rappresenta una tripla di liste (o una lista di triple). Le tre liste (o i tre elementi delle triple) sono di tre tipi diversi (T, R e S ad esempio), ma sia T che R sono un comparable (cioè estende Comparable). Lista3 ha anche i seguenti metodi:

- un metoto per inserire tre valori (T R e S). I valori andranno aggiunti in fondo alla lista.

- un metodo che restituisce se esiste un elemento che ha come massimo sia il primo che il secondo elemento (T e R).

Scrivi un main in cui fai un po' di prove.

Fai un zip con la funzione export di eclipse e fai l'upload qui.

(5 Punti)

2. Ricorsione - C - countA *(12 Punti)*

Scrivi una funzione countA in C che dato in ingresso una stringa (come array di char terminata da 0) restituisce il numero di caratteri 'A' in essa.
Al solito, scrivi tre versioni: una non ricorsiva, una ricorsiva senza tail recursion e una ricorsiva con tail recursion.
Specifica esattamente i parametri che passi alla procedura, il tipo di passaggio utilizzato e il loro significato. Definisci funzioni ausiliarie di aiuto se necessario, per tenere la segnatura della funzione countA più semplice.
Scrivi anche un main di esempio in cui chiami la funzione con la string "AB" invocala in modo assegni il risultato ad una variabile locale al main AAA. Non usare alcuna altra variabile globale.
Disegna il record di attivazione per tutte e tre le versioni fino alla massima estensione del record di attivazione. Nel caso di tail recursion, spiega nel codice quali ottimizzazioni hai adottato o potresti adottare.

Leggi le istruzioni qui:

CONSEGNA IL FILE ZIP CONTENTE TUTTO QUI:

(12 Punti)

3. Dangling pointer in C++ *(4 Punti)*

Fa un esempio in **C++** in cui hai un dangling pointer (ad esempio sullo stack) di un oggetto vero e proprio e continui ad usarlo prima senza accorgerti (cioè senza alcun malfunzionamento) e poi faiin modo invece di notare qualche malfunzionamento.

Carica lo zip qui:

(4 Punti)

4. definizione di funzione in SCALA *(6 Punti)*

- Scrivi una funzione neutroPro che dati due interi a e n restituisce true se n è l'elemento neutro del prodotto con a.
- Scrivi una higher order function neutro che generalizza la ricerca del neutro per ogni operazione binaria (*,+,/,max....). L'operazione sarà un argomento della funzione. Usa il currying se riesci
- Riscrivi (con nome neutroPro2) la funzione neutroPro usando neutro. Scrivi anche la funzione neutroSomma che dati due numeri restituisce true se il secondo è neutro rispetto la somma con il primo.
- Scrivi un po' di chiamate definendo un Object e usando lo schema seguente:

```
1object prova {
2
3  def ....
4
5  def main(args: Array[String]) {
6    println( ....)
7  }
8}
```

Carica qui il progetto.

(6 Punti)