Anteprima di test

parte al PC

Data: Fri Feb 14 16:32:18 2014 Punteggi massimi: 32

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

Definisci una classe Persona che ha un campo età.

Usando i generics in **Java**, scrivi una classe **MyList<T>** che rappresenta una liste di persone (o sottoclassi di persona) ed estende ArrayList<T> (non è detto che T vada bene, è solo per esempio). **MyList** ha anche i seguenti metodi:

- un metodo per inserire una persona;

- un metodo che restituisce la persona più giovane;

Se definisci una sottoclasse di Persona, Studente, cosa devi fare per poter definire qualcosa come **MyList<Studente> classe;?**

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

Crea un progetto in eclipse con nome COGNOME_NOME_JAVA. Per la consegna produci un file zip usando la funzione export archive file di eclipse. Lo zip abbia nome COGNOME_NOME_JAVA.zip. Consegna il file ZIP qui:

(5 Punti)

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

Scrivi una funzione strLunPari in C che dato in ingresso una stringa (come array di char terminata da 0) restituisce (come booleano) se la sua lunghezza è pari (senza usare funzioni ausiliarie).
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 strLunPari 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 globale PARI. 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:

Crea un progetto in eclipse con nome COGNOME_NOME_C. Per la consegna produci un file zip usando la funzione export archive file di eclipse. Lo zip abbia nome COGNOME_NOME_C.zip. Consegna il file ZIP qui:

(12 Punti)

3. Tipi opachi in C *(4 Punti)*

Fa un esempio in **C** in cui usi un puntatore a **tipo opaco** per realizzare un implementazione di una Astronave. Aggiungi qualche campo (ad esempio nome) e qualche metodo (ad esempio viaggia).

Crea un progetto in eclipse con nome COGNOME_NOME_OP. Per la consegna produci un file zip usando la funzione export archive file di eclipse. Lo zip abbia nome COGNOME_NOME_OP.zip. Consegna il file 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}
```

Crea un progetto in eclipse con nome COGNOME_NOME_SCALA. Per la consegna produci un file zip usando la funzione export archive file di eclipse. Lo zip abbia nome COGNOME_NOME_SCALA.zip. Consegna il file ZIP qui:

(6 Punti)

5. Testing *(5 Punti)*

Data le seguente funzione in Java

```
1// uno è minore se ha meno di 18 anni (o 20 per gli uomini)
2public static boolean isMinor(int eta, char gender) {
3  if (gender !='M' && gender !='F')
4     throw new RuntimeException();
5  if ( eta <= 18 || (eta <=20 && gender ='M'))
6  return true;
7   else
8  return false;
9}
```

Trova i casi di test per la copertura delle istruzioni, delle decisioni, delle condizioni e l'MCDC.

Scrivi ogni caso di di test come metodo di test (@Test) JUnit, oppure come metodo static a parte che chiami in un main. Contolla in ogni caso (usando assertTrue/False o assert) il valore restituito. Commenta con esattezza però per ogni caso di test a quale copertura serve. Usa dei nomi significativi (tipo testIstruzioni1,...). Puoi procedere in modo incrementale, cioè aggiungere solo i casi che ti mancano per una certa copertura, ma devi spiegarlo.

Crea un progetto in eclipse con nome COGNOME_NOME_TESTING. Per la consegna produci un file zip usando la funzione export archive file di eclipse. Lo zip abbia nome COGNOME_NOME_TESTING.zip. Consegna il file ZIP qui:

(5 Punti)