UNIVERSITÀ DEGLI STUDI DI BERGAMO | Dipartimento di Ingegneria Gestionale, dell'Informazione e della Produzione

Programmazione Avanzata

Durata: 4h
Data: 17 luglio 2024
Punteggio Massimo: 26 PUNTI.

Chiama i tuoi progetti con MATRICOLA_ESX con X il numero dell'esercizio. Per esempio, 2574188_ES1 conterrà la soluzione dell'esercizio 1 dello studente con matricola 2574188.

Consegna: crea uno zip unico e carica il file su Moodle [cartella ``2024-07-17''].

1. Funzione in C (4 punti)

Scrivi una funzione **mix_array** che dati in ingresso due array di interi array1 e array2, **restituisce** un array array3. Il contenuto delle celle di array3 in posizione pari corrisponde al DOPPIO del contenuto delle celle di array1 nelle medesime posizioni. Il contenuto delle celle di array3 in posizione dispari corrisponde al TRIPLO del contenuto delle celle di array2 nelle medesime posizioni.

Se un array è più lungo, la parte avanzata viene tagliata.
Ad esempio
mix_array([0 5 9 6 7], [9 5 7 6 2 5 8 7]) = [0 15 18 18 14]
mix_array ([8 5 1 2 9 6 5 3],[8 4 6 3 2 5 1]) = [16 12 2 9 18 15 10]

(a) Scrivi tre versioni:
- Una iterativa non ricorsiva
- Una ricorsiva senza tail call
- Una ricorsiva con tail call.

(b) Scrivi anche un main di esempio in cui chiami le funzioni con almeno gli array di cui sopra.

Non usare alcuna variabile globale. Cerca di tenere il più semplice possibile la segnatura delle funzioni ricorsive, ma se non riesci fai una funzione con segnatura semplice che chiami quella ricorsiva.

2. Record di Attivazione (4 punti)

Si consideri la seguente funzione:

```
int pow_n(int a, int ex) {
      if(ex==0) return 1 ;
      if(ex==1) return a;
      return a*(pow_n(a,(ex-1)));
} ;
```

Si mostrino i record di attivazione nello stack quando chiamata con a=5, ex=3: si crei un file .txt composto come segue (si aggiungano righe al file se necessario)

Label	Indirizzo	Contenuto	Note

3. Tipi opachi (3 punti)

Definisci il tipo opaco Set che rappresenta un insieme (NON ordinato) di interi. Nota: un insieme non contiene elementi ripetuti. Definisci l'operazione di unione. La dimensione del set deve aumentare se necessario. Implementa il set utilizzando un array. Definisci eventuali funzioni addizionali se necessario. Nota: Lo studente è libero di mantenere memorizzati il set di interi a piacimento.

costruttore: crea un set e inserisce l'insieme di interi contenuti in un array passato come argomento (facendone una copia).
union: prende due set e restituisce un set contenente l'unione degli elementi.
print: stampa il set di interi.
cancella: distrugge il set e libera la memoria
Fai un esempio in cui:

- costruisci il set con gli elementi {1,2,3} e lo stampi
- Calcola l'unione dei set contenenti gli elementi {1,4,5,7} e {8,4,3,1}
- cancelli tutti i set.

4. Inizializer list in C++ (1.5 punti)

Definire una classe Automobile. La classe Automobile ha un nome e due attributi di classe Motore e Bagagliaio. Definire una classe Automobile che ha come attributo il nome dell'auto. Sviluppare tre implementazioni diverse della classe Automobile in cui i suoi attributi sono rispettivamente il nome (memorizzato a piacere) e il Motore e Bagagliaio memorizzati mediante (a) due puntatori a una porzione di memoria nello stack, (b) due puntatori a una porzione di memoria nell'Heap, (c) due riferimenti a una porzione di memoria nell'Heap. Per ciascuno di questi tre casi discutere se (a) è possibile utilizzare l'inizializer list e (b) è possibile inizializzarlo all'interno del costruttore.

5. Smart pointers in C++ (1.5 punti)

Considerare la classe Automobile dell'esercizio 4. Crea una Automobile utilizzando un raw pointer, uno smart pointer, e uno shared smart pointer. Alla distruzione dell' Automobile devono essere distrutte anche il relativo motore e bagagliaio. Illustra e spiega le differenze relative all'utilizzo dei vari tipi di puntatori creando un metodo main e commentando opportunamente il codice.

6. Java Generics (4 punti)

Implementa la struttura dati "Set" dell'esercizio 3 in java, che però sia generica in modo che si possa memorizzare ogni tipo che estenda Object. Come tipo sottostante NON usare un array list o altre collezioni, ma utilizza un array puro array[]. Implementa i metodi dell'esercizio 2. Crea un main in cui illustri l'utilizzo dell' Set creando un set di Integer e un set di String.

7. Visitor pattern (4 punti)

Si vuole sviluppare una applicazione per gestire l'accesso allo stadio degli europei di calcio (EURO 2024). Uno spettatore ha un nome ed un cognome. Ci sono tre tipologie di spettatori: VIP, tifosi, e giornalisti. Ogni giornalista ha una testata giornalistica a cui appartiene. A differenza dei giornalisti e ai VIP (che hanno dei settori riservati), che hanno accesso alla tribuna VIP, i tifosi hanno accesso a una tribuna specifica. È possibile settare il posto assegnato ad un tifoso mediante un opportuno metodo setPosto(int tribuna, int posto).

(a) Si vuole utilizzare il pattern Visitor per
    1) Controllare se l'utente ha accesso alla tribuna VIP.
    2) Stampare nome e cognome dello spettatore.
(b) Deve essere possibile ordinare i tifosi in ordine crescente in base al loro posto ed alla loro tribuna. Utilizzare l'interfaccia Comparable e mostrare un esempio nel quale viene creata una lista di tifosi e vengono ordinati in base al loro posto e tribuna.
(c) Creare una Lista di elementi di tipo Tifoso. Discuti se è possibile assegnare questa lista ad un Lista di elementi di tipo Spettatore e presenta le relative motivazioni.

8. Programmazione funzionale (4 punti)

(a) Definire in Scala la funzione mod che dato un numero intero restituisce 1 se è pari e zero altrimetni.
(b) Definire la funzione mult che dato un numero lo raddoppia.
(c) Definire la funzione magic: Integer->Integer che data una funzione f:Integer -> Boolean e una funzione g: Integer -> Integer restituisce una funzione che applica la funzione g al valore di input se il risultato ritornato dalla funzione f è vero, altrimenti ritorna il valore di input.
(d) Definire la funzione "h" utilizzando la funzione "magic" passandogli come input le funzioni "mod" e "mult" definite in precedenza. Utilizzare la funzione "h" sui numeri interi 2 e 3.
(e) Definire una list di interi contenente gli interi 1,2,3. Applicare ad ogni elemento della lista la funzione "h". Invece di utilizzare la funzione "h" una funzione anonima
(f) Utilizzare map-reduce per sommare la lista dei valori ottenuti al punto (e).