


Algoritmi e strutture dati

ASD2024 - I Appello - Programmazione

	Gaetano Walter Fagone s282933
Iniziato	13 febbraio 2024, 12:40
Stato	Completato
Terminato	13 febbraio 2024, 12:51
Tempo impiegato	10 min. 37 secondi

Informazione

Attenzione!

Indicare quale tipologia di esame si intenda svolgere rispondendo alla domanda a scelta multipla seguente (domanda 1).

Si noti che è possibile leggere tutte o parte delle domande prima di effettuare la scelta e rispondere alla domanda 1.

Le domande dalla 2 alla 6 sono dedicate all'esame semplificato (traccia da 12pt).

In particolare:

- la domanda 2 fa parte dell'esercizio da 2 punti.
- la domanda 3 fa parte dell'esercizio da 4 punti.
- le domande 4,5,6 fanno parte dell'esercizio da 6 punti.

Le domande dalla 7 in poi sono dedicate all'esame completo (traccia da 18pt).

Un apposito separatore fa da intervallo tra le domande del compito da 12pt e le domande del compito da 18pt.

Informazione

PER ENTRAMBE LE PROVE DI PROGRAMMAZIONE (18 o 12 punti)

- se non indicato diversamente, è consentito utilizzare chiamate a funzioni standard, quali ordinamento per vettori, funzioni su FIFO, LIFO, liste, BST, tabelle di hash, grafi e altre strutture dati, considerate come librerie esterne
- gli header file riferiti dal codice dovranno essere inclusi nella versione del programma allegata alla relazione
- i modelli delle funzioni ricorsive non sono considerati funzioni standard
- consegna delle relazioni, per entrambe le tipologie di prova di programmazione: entro GIOVEDÌ 16/02/2024, alle ore 23:59, mediante caricamento su Portale
- assicurarsi di caricare l'elaborato nella [Sezione Elaborati relativa all'a.a. 2023/24](#).
- le istruzioni per il caricamento sono pubblicate sul Portale nella sezione Materiale
- QUALORA IL CODICE CARICATO CON LA RELAZIONE NON COMPILI CORRETTAMENTE, VERRÀ APPLICATA UNA PENALIZZAZIONE. Si ricorda che la valutazione del compito viene fatta, senza la presenza del candidato, sulla base dell'elaborato svolto durante l'appello. Non verranno corretti i compiti di cui non sarà stata inviata la relazione nei tempi stabiliti.

Domanda 1

Risposta non data

Punteggio max.:
1,00

Indicare quale tipologia di esame si intende svolgere.

Scegli un'alternativa:
Domanda 1

a.

18 Punti - Completo

b.

12 Punti - Semplificato

Risposta errata.

Ignorare il feedback di questa domanda.

Le risposte corrette sono: 12 Punti - Semplificato, 18 Punti - Completo

Domanda 2

Completo

Punteggio max.:
1,00

Sia dato un tipo ADT di prima classe SLIST (lista ordinata) con `Item` corrispondente alla typedef qui riportata

```
typedef struct { char name[16]; int val;} Item;
```

L'item consiste quindi in un nome (contenente solo caratteri alfabetici) a cui è associato un valore intero. Gli item in lista sono ordinati in modo crescente in base al campo name.

Scrivere una funzione SLISTmerge, avente il prototipo seguente

```
SLIST SLISTmerge(SLIST a, SLIST b);
```

I due parametri a e b sono liste ordinate. Si noti che è possibile che una stessa stringa compaia come name in più di un item sia in a che in b (o in entrambe).

La funzione genera una terza lista, i cui item contengono (nei campi name) tutte e sole le stringhe presenti in a e in b. Ma non sono possibili ripetizioni: quando una stringa compare più volte in a e/o in b, la si riporta una volta sola nel risultato, associando come campo val la somma dei valori associati alla stringa in a e/o b.

Si fornisca la definizione del tipo SLIST, del nodo in lista, e si scriva la funzione. Qualora si usi una funzione newNode, non è necessario scriverla.

Esempio:

a) ("roma", 7), ("torino", 4), ("zagabria", 5)

b) ("roma", 3), ("torino", 3), ("torino", 2), ("venezia", 10)

Risultato: ("roma", 10), ("torino", 9), ("venezia", 10), ("zagabria", 5)

Testo della risposta Domanda 2

Attenzione! Si consiglia l'uso degli spazi al posto delle tabulazioni per l'indentazione del codice, dal momento che il carattere TAB viene utilizzato per la navigazione della pagina da parte della piattaforma.

```
SLIST SLISTmerge(SLIST a, SLIST b) {  
  
}
```

Domanda 3

Completo

Punteggio max.:
1,00

Sia dato un BST di interi (ADT di prima classe). Si scriva la funzione che genera un vettore (da allocare dinamicamente, senza riallocazione) di puntatori ai nodi del BST. Il vettore di puntatori deve essere ordinato secondo questo criterio: profondità crescente e, a pari profondità, valori (dell'intero contenuto nei nodi) crescenti.

La funzione deve essere richiamabile come

```
pnodes = BSTlevelizedNodes (b, &n);
```

Dove b è il BST, pnodes il (vettore) risultato e n il numero di nodi

E' richiesta la definizione dei tipo BST e del nodo.

Non è ammesso l'uso di funzioni di libreria.

Testo della risposta Domanda 3

Attenzione! Si consiglia l'uso degli spazi al posto delle tabulazioni per l'indentazione del codice, dal momento che il carattere TAB viene utilizzato per la navigazione della pagina da parte della piattaforma.

```
//scrivere qui le definizioni dei tipi
```

```
//scrivere qui la funzione
```

```
... BSTlevelizedNodes (...) {  
  
}
```

.

Domanda 4

Completo

Punteggio max.:
1,00

E' dato un elenco di parole senza duplicati (vettore di stringhe).

Le stringhe in elenco posso essere utilizzate per generare stringhe più lunghe mediante concatenazione. Una concatenazione è una stringa generata dalla sequenza ordinata di tutte o parte delle stringhe in elenco prese al più una volta.

Si scriva una funzione bestConcat, che, dato l'elenco, generi la stringa più lunga ottenibile mediante concatenazione, rispettando un vincolo: date due parole consecutive nella sequenza, se la prima termina con vocale, la seconda non può iniziare con vocale. Se la prima termina con consonante, la seconda non può iniziare con consonante. Ai fini della bontà del risultato la lunghezza si misura in numero di caratteri (quindi non di parole).

Nota bene: le due domande a seguire NON sono facoltative. Assicurarsi di rispondere a tutte le richieste.

Testo della risposta Domanda 4

Attenzione! Si consiglia l'uso degli spazi al posto delle tabulazioni per l'indentazione del codice, dal momento che il carattere TAB viene utilizzato per la navigazione della pagina da parte della piattaforma.

```
//scrivere qui il codice...
```

```
char *bestConcat (char **parole, int nparole) {  
}
```

Domanda 5

Risposta non data

Punteggio max.:
1,00

Si giustifichi la scelta del modello combinatorio adottato.

Testo della risposta Domanda 5

Domanda 6

Risposta non data

Punteggio max.:
1,00

Si descrivano i criteri di pruning adottati o il motivo della loro assenza.

Testo della risposta Domanda 6

Informazione

PAGINA DI INTERMEZZO

La prova da 12 punti termina prima di questa pagina di intermezzo.

La prossima domanda rappresenta l'inizio della prova da 18 punti.

Informazione

Descrizione del problema

E' dato un grafo non orientato. NV indica il numero di vertici e NE indica il numero di archi. I nodi sono numerati da 0 a NV-1. A ogni nodo è associato un nome (una stringa di al più 20 caratteri alfabetici, terminatore compreso) e un valore (un numero intero). Il nome non è univoco, cioè più nodi possono essere associati allo stesso nome.

Un cammino nel grafo è quindi associato a:

- Una stringa determinata dalla concatenazione dei nomi associati ai nodi nel cammino
- Un valore complessivo, dato dalla somma dei valori dei nodi nel cammino.

Richieste del problema

A seguire una sintesi delle richieste del problema. Per ogni richiesta si troverà una domanda dedicata nelle sezioni a seguire con una descrizione più dettagliata per le richieste.

Strutture dati e letture

Definire opportune strutture dati per rappresentare i dati del problema e tutte le strutture dati ausiliarie ritenute opportune per la risoluzione dei problemi di verifica e di ricerca.

Definire inoltre la funzione di lettura secondo il formato del file descritto nelle domande a seguire. Si chiede di utilizzare l'ADT GRAPH proposto nel corso, adattandolo al problema.

Problema di verifica

Si scriva una funzione checkString che, dato il grafo e una stringa, verifichi se la stringa può rappresentare la concatenazione dei nomi associati ai nodi su un cammino nel grafo

Problema di ottimizzazione

Si scriva la funzione bestPath, che, dato il grafo e un numero M, determini il cammino (eventualmente ciclico, ma con nomi ripetuti nel cammino al più M volte) di valore massimo, dove il valore è la somma dei valori associati a ogni vertice nel cammino.

Esempio: se M valesse 2, significa che ogni nome può comparire al più 2 volte in un cammino: se il nome caratterizzasse due vertici, i vertici potranno comparire in un cammino entrambi, una volta sola, oppure solo uno dei due, fino a due volte.

Si tenga conto di un ulteriore vincolo: date due parole consecutive nella sequenza, se la prima parola termina con vocale, la seconda non può iniziare con vocale, se la prima termina con consonante, la seconda non può iniziare con consonante. E' sufficiente che il cammino venga stampato.

Domanda **7**

Completo

Punteggio max.:
1,00

Strutture dati e acquisizione

Scrivere qui la definizione e implementazione delle strutture dati repute necessarie a modellare le informazioni del problema, quali il grafo ed eventuali dati aggiuntivi, e la funzioni di acquisizione dei dati stessi. Si chiede di utilizzare l'ADT GRAPH proposto nel corso, con opportune modifiche per adattarlo al problema. Non è necessario realizzare completamente la GRAPHload, ma è sufficiente indicare le modifiche e scrivere in modo esplicito le eventuali aggiunte.

Il grafo è acquisito da un file testo nel formato

NV NE

<Elenco nodi (terne numero nome valore)>

<Elenco archi (coppie di numeri)>

Si puo' assumere che i nodi siano ordinati con identificatore (numero) crescente.

Esempio

4 6

0 sale 4

1 pepe 7

2 origano 6

3 cannella 8

4 peperoncino 5

5 origano 8

0 1

1 3

2 3

2 4

4 5

0 5

Testo della risposta Domanda 7

Attenzione! Si consiglia l'uso degli spazi al posto delle tabulazioni per l'indentazione del codice, dal momento che il carattere TAB viene utilizzato per la navigazione della pagina da parte della piattaforma.

...

Domanda 8

Completo

Punteggio max.:
1,00

Problema di verifica

Si scriva una funzione `checkString` che, dato il grafo e una stringa, verifichi se la stringa può rappresentare la concatenazione dei nomi associati ai nodi su un cammino nel grafo.

Testo della risposta Domanda 8

Attenzione! Si consiglia l'uso degli spazi al posto delle tabulazioni per l'indentazione del codice, dal momento che il carattere TAB viene utilizzato per la navigazione della pagina da parte della piattaforma.

...

Domanda 9

Risposta non data

Punteggio max.:
1,00

Problema di ricerca e ottimizzazione

Si scriva la funzione `bestPath`, che, dato il grafo e un numero M , determini il cammino (eventualmente ciclico, ma con nomi ripetuti nel cammino al più M volte) di valore massimo, dove il valore è la somma dei valori associati a ogni vertice nel cammino. Esempio: se M valesse 2, significa che ogni nome può comparire al più 2 volte in un

cammino: se il nome caratterizzasse due vertici, i vertici potranno comparire in un cammino entrambi, una volta sola, oppure solo uno dei due, fino a due volte.

Si tenga conto di un ulteriore vincolo: date due parole consecutive nella sequenza, se la prima parola termina con vocale, la seconda non può iniziare con vocale, se la prima termina con consonante, la seconda non può iniziare con consonante. E' sufficiente che il cammino venga stampato.

Testo della risposta Domanda 9