Laboratorio di algoritmi e strutture dati

Docente: Violetta Lonati

Prova di laboratorio - Appello del 20 febbraio 2020 - Versione A

Indice

Note importanti

- Si legga attentamente il testo degli esercizi e le indicazioni su come svolgerli. Se ci sono dubbi sul significato delle richieste, è opportuno chiedere chiarimenti!
- I file utili allo svolgimento degli esercizi, e indicati nel testo, sono contenuti nell'archivio zip, nella cartella `allegati`.
- Gli esempi di input/output proposti nel testo sono anch'essi contenuti nell'archivio zip, in file di testo separati, nella cartella `esempi`.
- Si leggano attentamente anche le indicazioni su come preparare le risposte. Per ogni esercizio è richiesto di preparare un file: in alcuni casi si tratta di un file di testo, in altri casi di un programma in C. Per ogni esercizio viene indicato il nome con cui salvare il file; è importante rispettare questa indicazione.
- Nella prima riga di tutti i file consegnati è necessario **scrivere nome, cognome e matricola**.
- Dopo essersi autenticati, si carichino sul sito `upload.di.unimi.it` i file contenenti le risposte.

Nomi dei file da consegnare I nomi dei file devono essere i seguenti:
`es1-film.c`,
`es2-listasvolto.c`,
`es3-sorveglianza.txt`,
`es4-posate.c`,

1 Film

AlgoFlix è un sistema di streaming online di film. Il sistema memorizza ogni giorno quante persone hanno visto un certo film, usando un vettore `P` di numeri interi: `P[i]` è il numero di persone che hanno visto il film nel giorno `i`. Vogliamo calcolare il massimo numero di giorni consecutivi in cui il numero di persone che hanno visto il film risulta non decrescente. In altre parole, vogliamo calcolare la lunghezza massima tra tutti i sottovettori composti da elementi contigui di `P` che risultino ordinati in senso non decrescente. Si scriva un algoritmo per risolvere questo problema, se ne determini il costo computazionale e lo si implementi in C.

Note per la consegna. Si scriva il programma in un file di nome `es1-film.c` con il programma. Il file deve contenere un commento con l'analisi della complessità.

Il programma deve leggere da standard input un numero n seguito da una sequenza di n numeri interi, quindi deve stampare la lunghezza del più lungo sottovettore non decrescente.

Esempio di esecuzione 1

Ricevendo da standard input

```
7
1 3 2 5 1 4 7
```

il programma deve stampare

```
3
```

in quanto il più lungo sottovettore non decrescente è composto dagli ultimi tre elementi [1, 4, 7].

Esempio di esecuzione 2

Ricevendo da standard input

```
7
12 3 5 7 8 4 6
```

il programma deve stampare

```
4
```

in quanto il più lungo sottovettore non decrescente è [3, 5, 7, 8] che ha 4 elementi.

Esempio di esecuzione 3

Ricevendo da standard input

```
5
5 4 3 2 1
```

il programma deve stampare

```
1
```

perché gli unici sottovettori non decrescenti sono i singoli elementi del vettore.

2 Lista misteriosa - comprensione e scrittura di codice

Considerate le funzioni `build`, `flist` e `fstring` contenute nel file `es2-lista.c`.

Il tipo `Node` usato nelle funzioni implementa un nodo di una lista concatenata; questo tipo, come d'abitudine, ha due membri: un membro che contiene le informazioni relative all'elemento della lista e un membro che punta al prossimo elemento della lista.

Comprensione del codice. Esaminate il codice e rispondete alle seguenti domande.

1. Descrivete a parole il comportamento della funzione `build`. Cosa fa e come lo fa? In particolare descrivete:
   a) cosa restituisce la funzione `build`;
   b) quante volte viene invocata la funzione `flist`;
   c) cosa viene verificato nell'`if` a riga 9
   d) cosa fa l'assegnamento nella riga 12
2. Riassumete con una frase il comportamento della funzione `fstring`. Cosa fa e come lo fa? In particolare, come deve essere il primo argomento che si passa a tale funzione?

Scrittura di codice.

1. Definite, in C, un tipo `Node` coerentemente a come viene usato nelle funzioni.
2. Scrivete una funzione `fstring_iter` che produca lo stesso risultato di `fstring` ma senza usare la ricorsione e senza usare le funzioni di `string.h`.
3. Partendo dalle funzioni `build` e `flist`, scrivete delle varianti `build2` e `flist2` in modo che le istruzioni

   ```
   char word[20] = "";
   printf( "%s\n", fstring( parola, build2( "hello") ) );
   ```

   producano in output la stringa

   ```
   hellohellhelheh
   ```

Note per la consegna. Salvate il file allegato con nome `es2-listasvolto.c`. Scrivete le risposte alle domande di comprensione del codice in un commento all'inizio del file. Completate il file come specificato sopra: il file dovrà contenere, oltre alle funzioni già fornite, anche le funzioni `fstring_iter`, `build2` e `flist2`. Il programma consegnato **non deve contenere** alcuna funzione `main` (anche se naturalmente è opportuno fare dei test durante la scrittura del programma!!!)

3 Videosorveglianza - modellazione

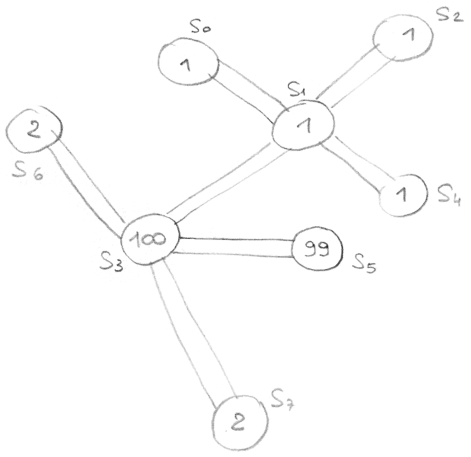
Una catena di supermercati deve proteggere i propri depositi di merce con adeguati sistemi di videosorveglianza.

Ogni *deposito D* è costituito da un insieme (finito) di *stanze*, tra cui un'unica *stanza di ingresso*. Ci sono poi dei *corridoi* che possono collegare due stanze fra loro (ma non c'è necessariamente un corridoio per ogni coppia di stanze); due stanze collegate da un corridoio si dicono *attigue*. Una qualunque stanza S è collegata con la stanza di ingresso da un unico *percorso*; questo percorso attraversa in ordine alcune stanze, chiamiamole $T_0, T_1, \ldots T_k$, tutte distinte e tali che T_0 è la stanza di ingresso, T_k è la stanza S e, per ogni $i = 0, 1, \ldots k-1$, le stanze T_i e T_{i+1} sono collegate da un corridoio.

Una stanza può essere *sorvegliata direttamente* collocandovi una videocamera. Una stanza attigua a una sorvegliata direttamente risulta essere *sorvegliata indirettamente*. Un *impianto* di videosorveglianza è costituito dal posizionamento di videocamere nelle stanze in modo tale che ogni stanza sia sorvegliata direttamente o indirettamente; per questioni economiche non è permesso collocare una videocamera in una stanza sorvegliata indirettamente; è invece possibile che una stanza sia sorvegliata indirettamente da più di una videocamera.

A ogni stanza S è associato un numero intero positivo che denota il valore $v(S)$ della merce depositata in S. Il valore di un impianto è definito come la somma del valore della merce depositata nelle e stanze *in cui è posizionata una videocamera*. Un impianto è *ottimale* per il deposito D se ha valore massimo (ossia, ha valore maggiore o uguale al valore di qualsiasi altro impianto per D). Si noti che in genere per un deposito possono progettarsi più impianti ottimali. Un corridoio risulta *buio* in un impianto se entrambe le stanze che collega non sono sorvegliate direttamente.

Esempio Il deposito D è costituito dalle stanze $S_0, S_1, S_3, \ldots S_7$. Quattro corridoi collegano le stanze S_0, S_2, S_3 e S_4 con la stanza S_1. Altri tre corridoi collegano le stanza S_5, S_6 e S_7 alla stanza S_3. La stanza S_3 contiene merce per valore pari a 100; la stanza S_5 contiene merce per valore pari a 99; le stanze S_0, S_1, S_2 e S_4 contengono ciascuna merce per valore pari a 1; le stanze S_6, S_7 contengono ciascuna merce per valore pari a 2. Il deposito è illustrato nella seguente figura:



La stanza d'ingresso è la stanza S_0. La stanza S_6 è collegata alla stanza S_0 dal percorso S_0, S_1, S_3, S_6.

Posizionando tre telecamere nelle stanze S_2, S_3 e S_4 non si ha un impianto di sorveglianza, poiché la stanza S_0 non sarebbe sorvegliata né direttamente né indirettamente; aggiungendo una telecamera nella stanza S_0 si ha un impianto di valore 103. Un altro impianto si ottiene mettendo 4 telecamere nelle stanze S_1, S_5, S_6 e S_7; in questo caso il valore dell'impianto è 104 e non è possibile aggiungere altre telecamere. Il deposito ottimale di valore 106 si ottiene posizionando le telecamere nelle stanze $S_0, S_2, S_4, S_5, S_6, S_7$. C'è un solo corridoio buio, quello tra S_1 e S_3.

Rispondete ai seguenti punti.

1. Modellate la situazione usando una struttura dati opportuna, chiarendo come sia collegata alla situazione descritta; in particolare, utilizzando una terminologia appropriata. descrivete come sono rappresentate le stanze, i corridoi e gli impianti, e definite che cosa sono i corridoi bui.

2. Progettate e descrivete un algoritmo che, dato un deposito e un insieme di sue stanze, stabilisce se tale insieme definisce un impianto.

3. Progettate e definite un algoritmo che determina dove posizionare le telecamere di un impianto. L'impianto che si ottiene col vostro algoritmo è ottimale? Se sì, spiegate perché, altrimenti fornite un esempio in cui l'algoritmo non fornisce un impianto ottimale.

4. Progettate e descrivete un algoritmo che determina dove posizionare le telecamere per avere un **impianto ottimale**.

Note per la consegna Scrivete le risposte in un file di testo con nome `es3-sorveglianza.txt`.

4 Posate - facoltativo

In una fabbrica di vasellame, un robot a due bracci è usato per confezionare scatole di posate. In ogni scatola c'è una coppia di posate: un coltello (K) e una forchetta (F). Il robot riceve su un nastro una sequenza di posate; se necessario, il robot può appoggiare le posate su una pila, operando secondo le regole seguenti.

- Se entrambi i bracci sono liberi, il robot prende il prossimo oggetto con il braccio sinistro.
- Se il robot ha già qualcosa nel braccio sinistro, ma ha il braccio destro libero, allora controlla se sulla cima della pila c'è un oggetto dell'altro tipo e in questo caso lo afferra con il braccio destro, altrimenti afferra col braccio destro il prossimo oggetto sul nastro.
- Se il robot ha in mano due oggetti di tipo diverso, li mette in una scatola.
- Se il robot ha in mano due oggetti dello stesso tipo, mette sulla cima della pila quello che ha nel braccio destro.

Il robot conclude con successo il suo lavoro se riesce a processare tutti gli oggetti sul nastro e, quando questi sono finiti, la pila ed entrambi i bracci sono vuoti.

Scrivete una funzione `robot` che legge da standard input una sequenza di lettere `F` e `K` e restituisce 0 se il robot conclude con successo il suo lavoro su un nastro che contiene questa sequenza di posate, altrimenti restituisce 1. Ad esempio, la funzione restituirà 0 se riceve come input la sequenza `FFFKKK`, la sequenza `FKFKFK` o la sequenza `FKKF`, ma restituirà 1 se riceve come input la sequenza `FFF` o la sequenza `FKFKK`.

La funzione deve inoltre stampare la sequenza delle operazioni eseguite dal robot, una per riga, secondo questa sintassi:

- `sx A` significa che ha preso dal nastro, col braccio sinistro, l'oggetto di tipo `A` (simmetricamente per l'operazione `dx A`);
- `sx push A` significa che ha messo sulla cima della pila l'oggetto di tipo `A` che aveva nel braccio sinistro (simmetricamente per l'operazione `dx push A`);

- `sx pop A` significa che ha preso col braccio sinistro un oggetto di tipo `A` dalla cima della pila (simmetricamente per l'operazione `dx pop A`);
- `+1` significa che ha messo una coppia di posate nella scatola liberando entrambi i bracci.

Infine, quando finisce la sequenza di input, la funzione deve stampare anche:

- il numero n di coppie di posate che è riuscito a comporre;
- il tipo di stoviglia che tiene il braccio sinistro (o `Z` se il braccio è vuoto);
- la sequenza (eventualmente vuota) di posate che sono rimaste sulla pila.

Nota per lo svolgimento. Anche se l'algoritmo da implementare usa una pila, non è necessario implementare la pila come struttura dati astratta e mantenerne in memoria l'intero contenuto.

Note per la consegna. Scrivete la funzione `robot` in un file chiamato `es4-posate.c`; il file deve anche contenere il seguente `main`, che invoca la funzione `robot` e ne stampa il valore restituito:

```
int main ( void ) {
  printf( "%d\n", robot() );
}
```

Esempio di esecuzione Ricevendo da standard input la sequenza di caratteri `KKFFFKFF` il programma deve stampare il seguente output output:

```
sx K
dx K
dx push K
dx F
+1
sx F
dx pop K
+1
sx F
dx K
+1
sx F
dx F
dx push F
3 F F
1
```