TECNICHE DI PROGRAMMAZIONE, A.A. 2020/2021

Esempio di esame

Premessa

Questo documento presenta un esempio di possibile tema di esame, in forma testuale, con risposta. L'esempio sarà disponibile a breve anche su piattaforma exercise.

ATTENZIONE: si consiglia di non considerare le domande pubblicate come un insieme esaustivo su cui preparare l'esame. Si tratta di semplici esempi. Le risposte fornite sono puramente indicative. Il codice non viene commentato.

Come preparare l'esame

La frequenza al corso e lo svolgimento degli esercizi di laboratori dovrebbero garantire un'adeguata preparazione per l'esame. Si consideri tuttavia come, in genere, sia importate un "allenamento mirato", basato su esempi di esami, nel quale ci si prepari a:

- utilizzo della piattaforma exam/exercise, con particolare riferimento alle problematiche di gestione del tempo, all'interfaccia e alle modalità per rispondere
- comprensione del testo delle domande, abituandosi a porre l'enfasi su "quale sia il problema", "cosa si chiede"
- modalità per rispondere sia alle domande di teoria che di programmazione, evitando di perdere tempo su dettagli inutili e cercando di capire rapidamente "quali contenuti debba presentare la risposta"

In estrema sintesi, serve aver studiato e lavorato, ma serve anche "provare" qualche volta per conto proprio un esame, anche con limiti di tempo simili a quelli reali,

---

TEORIA

Esercizio 1 (5 punti)

Sia data la seguente sequenza di coppie:

3-2 3-4 5-1 7-3 5-7 9-1

dove la relazione i-j indica che il vertice i è adiacente al vertice j. Si applichi un algoritmo di on-line connectivity con quick-find, riportando a ogni passo il contenuto del vettore. I vertici siano denominati con interi compresi tra 0 e 9.

Soluzione

La tabella riportata di seguito illustra la configurazione del vettore id a ogni passo.

	0	1	2	3	4	5	6	7	8	9
	0	1	2	3	4	5	6	7	8	9
3-2	0	1	2	2	4	5	6	7	8	9
3-4	0	1	4	4	4	5	6	7	8	9
5-1	0	1	4	4	4	1	6	7	8	9
7-3	0	1	4	4	4	1	6	4	8	9
5-7	0	4	4	4	4	4	6	4	8	9
9-1	0	4	4	4	4	4	6	4	8	4

Esercizio 2 (5 punti)

Enunciato:
Si ordini in maniera ascendente mediante counting sort il seguente vettore di interi:

$2_1\ 4_1\ 3_1\ 1\ 3_2\ 5_1\ 4_2\ 2_2\ 4_3\ 5_2\ 3_3\ 9\ 8$

Domanda e formato della risposta:

- Dato il vettore da ordinare, quanto vale k?
  k =
- riportare come sequenza di interi il contenuto iniziale del vettore C delle occorrenze multiple
- a quale indice nel vettore risultato verrà memorizzata la chiave 4_2?

Risposta corretta:
k = 10
C: 0 1 2 3 3 2 0 0 1 1
all'indice 7

Esercizio 3 (5 punti)

Sia data la seguente funzione

```
int trova (int *v, int n, int s) {
  int i, j;
  for (i=0; i<n-4; i++) {
    int somma = 0;
    for (j=0; j<4; j++) {
      somma += v[i+j];
    }
    if (somma == s)
      return i;
  }
  return -1;
}
```

Si risponda alle seguenti domande:

A) Quale è la complessità di caso peggiore della funzione?

T(n) = O(.....)

B) E' possibile determinarne la complessità in termini di limite asintotico stretto?

SI/NO (dire perché)

$T(n) = \Theta(...)$

La funzione cerca, in un vettore, un intervallo di dati consecutivi avente una certa caratteristica:
C) quale caratteristica?
D) nel caso in cui il vettore contenesse più intervalli con tale caratteristica, si ritorna il primo, l'ultimo o uno a caso?

E) Supponendo che esista un vettore

```
int dati[N];
```

con N constante, quale/i di queste è una possibile chiamata della funzione:

```
1) trova (dati, N, 8);
2) trova (&dati[0], N, N);
3) trova (*dati, N, 8);
4) trova (dati+N/2, N, N);
```

Si risponda si/no a ogni alternativa, motivando la risposta.

Risposte

A) n) = O(n) lineare in quanto il ciclo interno fa un numero di iterazioni costante (indipendente da n)
B) Non si puo' definire un limite asintotico stretto in quanto il limite asintotico inferiore è $T(n) = \Omega(1)$: ad esempio si ritorna i uguale a 0.
C) Quattro interi consecutivi la cui somma sia uguale a s (si ritorna l'indice del primo)
D) Il primo, in quanto appena si trova un intervallo la funzione ritorna immediatamente
E)
    1. Si. Si cerca l'intervallo di somma 8 in tutto l vettore dati
    2. Si. Equivale al caso 1 sfruttando la dualità puntatore vettore. La somme dei 4 dati anzichè 8 è N.
    3. No. *dati non è compatibile con un puntatore a intero (il parametro formale): rappresenta invece il primo intero nel vettore, cioè dati[0])
    4. No. Il primo parametro sarebbe corretto e identificherebbe la sedonda metà vettore, ma la dimensione (N) è scorretta: potrebbe arrivare al Massimo a N/2.

PROGRAMMAZIONE

Esercizio 4 (6 punti)

Sia data un vettore *V* di *N* interi. Si scriva una funzione C che visualizzi tutti i sottovettori di dimensione massima formati da celle contigue contenenti dati dispari. La funzione abbia prototipo:

```
void maxOdd(int v[], int N);
```

Esempio: dato il vettore 1 3 7 1 0 1 9 3 1 0, i sottovettori di dimensione massima contenenti dati dispari sono
1 9 3 1 e 1 3 7 1.

```
int maxOdd(int v[], int n) {
  int lung, j, k;
  int daStampare, trovato;

  trovato = 0;
  for(lung=n; !trovato && lung>0; lung--) {
    /* per ogni lunghezza decrescente dei sottovettori
       non minore della massima sinora trovata */
    for(j=0; j<=n-lung; j++) {
      /* per ogni possibile indice di inizio di un sottovettore */
      daStampare = 1;
      for(k=j; daStampare && k<j+lung; k++) {
        /* verifica di validita' del sottovettore */
        if (v[k]%2==0) { /* se pari */
          daStampare = 0; /* sottovettore non valido */
        }
      }
      if (daStampare) {
        trovato = 1; /* imposta il flag che non
                        lascera' iterare su una lunghezza minore */
        printf("sottovettore massimo:");
        for(k=j; k<j+lung; k++)
          printf(" %d");
        printf("\n");
        /* saltiamo il sottointervallo appena controllato */
        j = k+1;
      }
    }
  }
  return 0;
}
```

Esercizio 5 (4 punti)

Sia data una stringa *S* di caratteri. Si scriva una funzione C che conti quante sottostringhe di *n* caratteri in essa contenute hanno 2 vocali. Si ricorda che in C non esiste una funzione (quindi va realizzata) che determini se un carattere sia costante o vocale La funzione abbia prototipo:

```
void countAndPrint(char str[], int n);
```

Esempio: se S=”forExample” e n= 4, le sottostringhe di lunghezza 4 con 2 vocali sono 4 e sono “forE”, “orEx”, “rExa” e “Exam”.

Possibile risposta (attanzione: non è l’unica possibile; si tratta di una proposta)

```
// funzione che determina se un carattere è vocale
int isvowel(char c) {
  char vowels[] = "aeiouAEIOU";
  int i, nv=strlen(vowels);
  for(i=0;i<nv;i++)
    if (c == vowels[i])
      return 1;
  return 0;
}

void countAndPrint(char str[], int n) {
  int i, j, len = strlen(str);
  for(i=0;i<len-n;i++) {
    int cnt = 0;
    for(j=i;j<i+n;j++) {
      if (isvowel(str[j])) cnt++;
    }
    if (cnt == 2) {
      printf("%s\n", n, str+i);
    }
  }
}
```

Esercizio 6 (8 punti)

Sia dato un vettore `V` di `N` interi i cui elementi rappresentano, in formato compresso, una sequenza di numeri che deve essere inserita in una matrice `M` di interi di `r` righe e `c` colonne, secondo la strategia row-major, cioè per righe. Il vettore contiene quindi, in sequenza, i dati di tutte le righe, cercando di compattare i dati ripetuti.
Per decodificare la sequenza, gli interi del vettore devono essere considerati a coppie (`v[0]`, `v[1]`), (`v[2]`, `v[3]`), (`v[4]`, `v[5]`), etc. Detti v[i] e v[i+1] due dati ripetuti (con i indice pari), v[i] rappresenta l’intero, v[i+1] le ripetizioni, cioè quante volte v[i] va ripetuto sulla stessa riga della matrice `M` (o sulla riga successiva, se la riga corrente termina).

Si scriva una funzione C con il seguente prototipo che riempia la matrice e ne stampi il contenuto:

```
int buildMatrix(int V[], int N, int M[MAXR][MAXC], int nr, int
                                                    nc);
```

La funzione verifichi anche che il contenuto del vettore `V` sia corretto, cioè compatibile con le dimensioni della matrice `M` (già esistente) in modo tale da riempire tutte le sue caselle (per nr righe e nc colonne): il valore di ritorno sia 1 per indicare corretto e 0 per indicare non corretto

Esempio. Siano `nr` = 3, `nc` = 5, `N` = 14 e `V` = (1, 2, 17, 2, 3, 1, 8, 4, 6, 1, 7, 3, 5, 2): la matrice `M` avrà il seguente contenuto:

$$M = \begin{pmatrix} 1 & 1 & 17 & 17 & 3 \\ 8 & 8 & 8 & 8 & 6 \\ 7 & 7 & 7 & 5 & 5 \end{pmatrix}$$

Possibile risposta (attanzione: non è l'unica possibile; si tratta di una proposta)

```
int buildMatrix(int V[], int N, int M[MAXR][MAXC], int nr, int nc) {
  int i, j, k, x, y;
  k=0;
  for(i=0;i<N;i=i+2) {
    int val = V[i];
    int rip = V[i+1];
    for(j=0;j<rip;j++) {
      x = k / nc;
      y = k % nc;
      M[x][y] = val;
      k++;
    }
  }
  return k==nr*nc;
}
```