POLITECNICO DI TORINO

Tecniche di programmazione

TP - I appello del 3/7/2021

	ANDREA ANGELO RAINERI 280848
Iniziato	sabato, 3 luglio 2021, 11:10
Terminato	sabato, 3 luglio 2021, 12:28
Tempo impiegato	1 ora 18 min.

Informazione

Prova scritta di esame di Tecniche di Programmazione.

Primo appello 2020/2021: 3/7/2021

L'esame consta di 6 domande, per un massimo di 33 punti totali, ripartite in due parti:

- 3 domande di teoria (15 punti complessivi): domande 1, 2, 3

- 3 domande di programmazione (18 punti complessivi): domande 4, 5, 6

Si può scorrere tra le domande avanti e indietro. Non viene quindi imposta una sequenza. Le risposte alle domande di teoria e il codice C per le domande di programmazione vanno scritti negli spazi riservati, nell'editor di testo, sotto ogni domanda. Si tratta di un semplice editor di testo, senza alcuna funzionalità IDE (nessun rientro automatico, nessun run-debug, nessun completamento automatico, ecc.). Attenzione a evitare l'uso del tabulatore (tasto TAB).

Per le domande di programmazione è consentito consultare solo la documentazione disponibile a questi link (cliccare per aprire):

https://www.tutorialspoint.com/c_standard_library/index.htm

https://en.cppreference.com/w/c/language

Dopo la prova scritta, si riceverà via e-mail il file .pdf dell'esame. Affinché questo sia venga valutato, è necessario caricare nella sezione Elaborati del Portale della Didattica una relazione con auto-correzione, seguendo le istruzioni riportate in regole-esame-tp nel Portale della Didattica. La scadenza (inderogabile) è il 6 luglio 2021, 23:59.

ATTENZIONE: *in caso di difficoltà con la procedura di autenticazione, malfunzionamenti o altro, si consiglia di ritentare, al limite dopo aver fatto reboot del PC. Si ricorda che i docenti non sono in grado di fornire particolare assistenza in tali situazioni.*
Ad esame terminato e dopo aver verificato il report PDF ricevuto, qualora si ritenga di aver avuto problemi legati a malfunzionamenti o errori della piattaforma exam/respondus, è necessario scrivere un email al/alla docente, nella modalità prevista dal regolamento di ateneo. Tale email dovrebbe descrivere i problemi incontrati e fornire un eventuale dettaglio sulle parti dell'esame ritenute corrette e quelle penalizzate da errori o mancanze. Dopo aver valutato la situazione, ci potrà essere un eventuale recupero (totale o parziale), anche (se ritenuto opportuno) in forma di esame orale.

Domanda 1

Completo

Punteggio max.: 5,00

Sia data la seguente sequenza di coppie, dove la relazione i-j indica che il vertice i è adiacente al vertice j:

11-6, 4-2, 0-8, 9-3, 10-9

Si applichi un algoritmo di on-line connectivity con quickfind. I nodi sono denominati con interi tra 0 e 11.

Domanda e formato della risposta:

riportare il vettore id come sequenza di interi

- dopo il passo 1 (i passi cominciano da 1)
- dopo il passo 2
- dopo il passo 3

Riportare il vettore id come sequenza di interi

Dopo il passo 1 (i passi cominciano da 1)

0 1 2 3 4 5 6 7 8 9 10 6

Dopo il passo 2

0 1 2 3 2 5 6 7 8 9 10 6

Dopo il passo 3

8 1 2 3 2 5 6 7 8 9 10 6

Domanda 2

Completo

Punteggio max.: 5,00

Si ordini in maniera ascendente la seguente sequenza di interi mediante Shell sort con la sequenza di Knuth:

25 3 12 37 4 82 5 10 90 0 37 65 23 42 71 44

Domanda e formato della risposta:

- Data la dimensione del vettore da ordinare, da quale valore di h si parte?
  h =
  Breve motivazione:
- Riportare il vettore come sequenza di interiquando è 4-ordinato, avendo svolto i passi precedenti.

Data la dimensione del vettore da ordinare, da quale valore di h si parte?

h = 4

Breve motivazione:

Il più grande valore di h calcolato attraverso la sequenza di Knuth $h_i = 3h_{i-1} + 1$ non deve essere più grande di N/3 dove N è la dimensione del vettore da ordinare affinché le sottosequenze identificate da elementi del vettore a distanza h siano significative, ovvero costituite da almeno due elementi.

Riportare il vettore come sequenza di interiquando è 4-ordinato, avendo svolto i passi precedenti.

4 0 5 10 23 3 12 37 25 42 37 44 90 82 71 65

Domanda 3

Completo

Punteggio max.: 5,00

Dato il seguente pezzo di codice:

```
typedef struct {
    char name[10];
    char surname[10];
    float score;
} student;

student *s, vet[10];
```

Rispondere alle seguenti domande, supponendo che l'architettura su cui esegue sia a 64 bit, che le variabili di tipo float occupino 32 bit, e che sia la variabile s che il vettore vet siano stati inizializzati (anche se non viene mostrato nel codice)

A) Qual è l'occupazione di memoria in Byte della variabile s?

B) Qual è l'occupazione in Byte del vettore vet?

C) Dire se e quali delle seguenti righe di codice sono corrette. In caso contrario, giustificare la risposta.

1. s++;
2. s=vet;
3. vet+=2;
4. s=*vet;

A) Qual è l'occupazione di memoria in Byte della variabile s?

8 (Architettura a 64bit, s è un puntatore)

B) Qual è l'occupazione in Byte del vettore vet?

65 (52 bit per struct student)

C) Dire se e quali delle seguenti righe di codice sono corrette. In caso contrario, giustificare la risposta.

1. s++; OK

2. s=vet; OK

3. vet+=2; NO, la variabile che identifica un vettore è un puntatore al primo elemento del vettore, ma n on è possibile modificarlo essendo definito da C come una costante.

4. s=*vet; NO, s è un puntatore a tipo struct student e non può dunque assumere il valore del primo el emento del vettore vet. Sarebbe stato corretto far puntare s al primo elemento di vet (riga 2)

Domanda 4

Completo

Punteggio max.: 4,00

Si scriva una funzione C avente il seguente prototipo

int strFindSimilar(char str1[], char str2[]);

La funzione, date due stringhe str1 e str2, cerca in str1 la prima sotto-stringa “simile” a str2, dove due stringhe sono dette simili se:

hanno la stessa lunghezza

il numero caratteri corrispondenti uguali è (strettamente) maggiore della metà della lunghezza di str2 (>strlen(str2)/2)

Se tale sotto-stringa esiste, se ne ritorna l’indice del primo carattere. Se non esiste, ritorna -1.

Esempio di esecuzione:

*strFindSimilar("FifthOfNovember", "September") ritorna 6, visto che la sotto-stringa che inizia al carattere 6 ("fNov**ember**") ha 5 caratteri uguali alla stringa "Sept**ember**";*

strFindSimilar("Seventh", "September") ritorna -1;

*strFindSimilar("JustF**o**r**Ex**ample", "n**o**t**Ex**") ritorna 4, visto che la sotto-stringa che inizia al carattere 4 ("F**o**r**Ex**") ha 3 caratteri uguali (su 5) alla stringa "n**o**t**Ex**".*

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.
Sono possibili, se necessarie o utili, altre funzioni oltre a quella/e richiesta/e

```
#include <string.h>

int strFindSimilar(char str1[], char str2[]) {
    int len1 = strlen(str1), len2 = strlen(str2);
    int i, j, cnt;

    // Cerco in ogni sottostringa di len2 in str1 caratteri uguali con str2
    for (i = 0; i < len1 - len2; i++) {
        cnt = 0;
        for (j = 0; j < len2; j++) {
            if (str1[i+j] == str2[j])
                cnt++;
        }
        // Uscita anticipata dalla funzione appena viene identificata la prima sottostringa valida
        if (cnt > len2/2)
            return cnt;
    }

    // Caso finale: nessuna sottostringa valida identificata
    return -1;
}
```

Domanda 5

Completo

Punteggio max.: 6,00

Si scriva una funzione C che, data una matrice quadrata di float, trovi la diagonale avente valor medio massimo. La funzione abbia prototipo:

float matrMaxDiag(float M[DIM][DIM], int n);

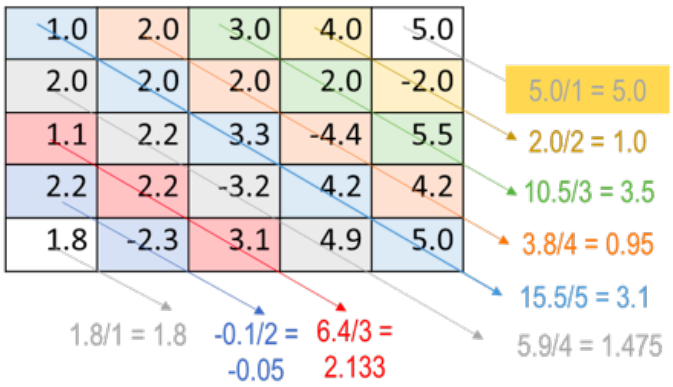
dove DIM è la dimensione della matrice, mentre n (n <= DIM) è il numero di righe e colonne effettivamente usato.

Per ogni diagonale va calcolata la media aritmetica delle sue caselle. Tra queste va calcolato e ritornato il valor medio massimo.

Esempio di esecuzione (la printf non è richiesta, la fa il main):

1.0	2.0	3.0	4.0	5.0
2.0	2.0	2.0	2.0	-2.0
1.1	2.2	3.3	-4.4	5.5
2.2	2.2	-3.2	4.2	4.2
1.8	-2.3	3.1	4.9	5.0

```
Max diag average: 5.0
```



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.
Sono possibili, se necessarie o utili, altre funzioni oltre a quella/e richiesta/e

```
float matrMaxDiag(float M[DIM][DIM], int n) {
    int i, j;
    float maxAvg, avgs[DIM] = {0};

    // Somma di tutti gli elementi sulle diagonali e memorizzazione nel vettore avgs
    for (i = 0; i < n; i++) {
     for (j = 0; j < n; j++) {
            avgs[i-j+n] += M[i][j];
        }
    }

    // Calcolo delle medie
    for (i = 0; i < n*2 - 1; i++) {
        if ((i+1) <= n)
            avgs[i] /= i+1;
        else
            avgs[i] /= (n - (i+1)%n);
    }

    // Identificazione massimo
    maxAvg = avgs[0];
    for (i = 1; i < n*2 - 1; i++) {
        if (maxAvg < avgs[i])
            maxAvg = avgs[i];
    }

    return maxAvg;

}
```

Domanda 6

Completo

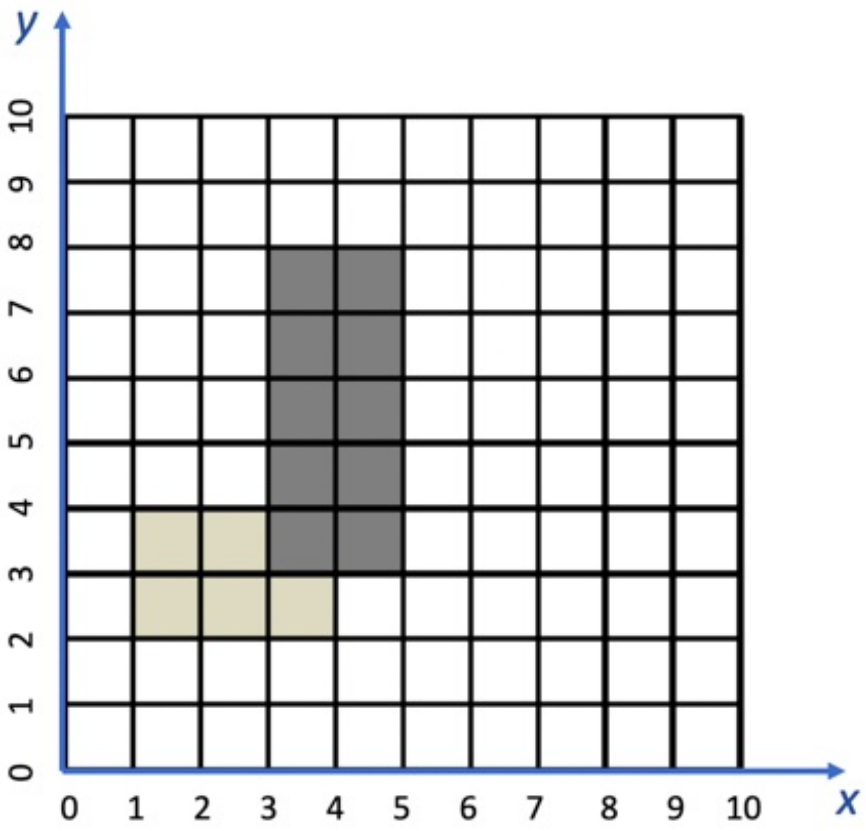
Punteggio max.: 8,00

Si consideri una regione quadrata nel primo quadrante del piano Cartesiano, con coordinata (0,0) per l'angolo in basso a sinistra e (100,100) per l'angolo in alto a destra. Un file di testo contiene un elenco di rettangoli, aventi lati paralleli agli assi del piano cartesiano. Ogni rettangolo viene descritto in una riga del file attraverso le coordinate x ed y del suo vertice in basso a sinistra e del suo vertice in alto a destra, riportate in questo ordine usando spazi come caratteri separatori. Le coordinate sono numeri interi inclusi tra 0 e 100, estremi inclusi. Si scriva una funzione:

int areaIntersection(FILE * fp);

che riceve come parametro un puntatore al file (già aperto) e ritorna l'area totale delle regioni che si trovano all'intersezione tra due o più rettangoli.

Esempio di esecuzione: (per facilitare le comprensione, si riporta una regione più piccola, con coordinate incluse nell'intervallo 0…10).



Se assumiamo che il il file di esempio contenga 1 2 4 4 sulla prima riga (rettangolo più in basso) e 3 3 5 8 nella seconda riga (rettangolo più in alto), l'area totale ritornata dalla funzione, per l'intersezione, è 1.

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.
Sono possibili, se necessarie o utili, altre funzioni oltre a quella/e richiesta/e

```
#include <stdio.h>
#define DIM 100

int areaIntersection(FILE * fp) {
    int x1, y1, x2, y2, i, j, area = 0;
    int M[DIM][DIM] = {0};

    // Lettura file
    while (fscanf(fp, "%d %d %d %d", &x1, &y1, &x2, &y2) == 4) {
        for (j = x1; j < x2; j++) {
            for (i = DIM - y1 - 1; i > DIM - y2 - 1; i--) {
                // Verifica intersezione
                if (M[i][j] == 1)
                    area++;
                else
                    M[i][j] = 1;
            }
        }
    }

    return area;
}
```