

Laboratorio di programmazione e Informatica 1

- A.A. 2018-2019 -

Prova di laboratorio: pre-appello - 30 maggio 2019

ISTRUZIONI: La prima riga di ogni programma C deve contenere il proprio nome e cognome. Tutti i programmi devono essere strutturati in funzioni e completi di commenti che spieghino il procedimento. Creare una cartella $\langle \text{CognomeNome} \rangle$ e copiare all'interno i file sorgente dei programmi (.C o .cpp). La cartella andrà copiata nella pennetta del docente.

ATTENZIONE! Non saranno valutati programmi che non passano la fase di compilazione. Si consiglia pertanto di "mettere sotto commento" le parti di programma che danno errore in compilazione.

Tempo a disposizione: 2 ore e mezza.

ESERCIZI

Esercizio 1 (17 punti) Relazioni Transitive

Una *relazione binaria* R su un insieme A è un sottoinsieme del prodotto cartesiano $A \times A$. Conveniamo di scrivere, come sempre, $R(x, y)$ per indicare che la coppia (x, y) è nella relazione R , cioè $(x, y) \in R$. Una matrice quadrata di interi M_R di dimensione $n \times n$ contenente valori 0, 1 può rappresentare una relazione binaria R su un insieme finito di n elementi $A = \{a_0, \dots, a_{n-1}\}$ stipulando che $M_R[i][j] = 1$ se e solo se $R(a_i, a_j)$ vale. Ricordiamo che R è transitiva se $\forall x, y, z, R(x, y) \vee R(y, z) \Rightarrow R(x, z)$.

Vista sulla matrice, per tutte le terne di indici, ogni qualvolta nella matrice di input si abbia $M_R[i][j] = 1$ e $M_R[j][k] = 1$ allora anche $M_R[i][k] = 1$.

Esempio: Consideriamo le seguenti matrici:

	a_0	a_1	a_2	a_3
a_0	1	1	1	1
a_1	1	1	1	1
a_2	0	0	0	0
a_3	0	0	1	0

	a_0	a_1	a_2	a_3
a_0	0	1	0	0
a_1	1	0	1	1
a_2	0	0	0	0
a_3	0	0	1	0

Nella matrice a sinistra la relazione è transitiva mentre la relazione a destra non lo è. Infatti se consideriamo gli elementi a_0, a_1, a_2 vale che $R(a_0, a_1)$ e $R(a_1, a_2)$ ma non $R(a_0, a_2)$. I punti 5. 6. sono opzionali e valgono ulteriori 3 punti.

- Scrivere una funzione `int Transitiva(int M[][SIZE], int *n1, int *n2, int *n3)` che, prende in input la matrice corrispondente alla relazione, e verifica se la relazione è transitiva (restituisce 1 oppure 0). Nel caso in cui non sia transitiva, le variabili $n1$, $n2$ ed $n3$ dovranno contenere una tripla di elementi per cui non vale la proprietà.
- Scrivere un programma in C che:
 1. Genera una matrice 15x15 a valori interi random tra 0,1.

2. Stampa la matrice ben formattata sullo schermo.
3. Utilizza la funzione `Transitiva` e scrive se la matrice generata corrisponde ad una relazione transitiva o, nel caso contrario scrive sullo schermo una terna di elementi x, y, z per cui non vale la proprietà transitiva.
4. Modifica la matrice M in modo che la relazione sia valida per le tre coppie (x, y) , (y, z) , (x, z) .
5. (opz) Ripete i punti 3. e 4. finché la matrice non rappresenti una relazione transitiva.
6. (opz) Ristampa la matrice, scrivendo quante modifiche sono state apportate alla matrice originale.

Anche i punti 1) e 2) devono essere svolti con delle funzioni. I punti 5. e 6. sono opzionali e valgono ulteriori 3 punti.

Esercizio 2 (13 punti) *Transitività nelle parole: il lucchetto*

Nel linguaggio enigmistico un lucchetto è un gioco caratterizzato dallo schema $XY + YZ \Rightarrow XZ$. In pratica si trovano due parole w_1 e w_2 in cui l'ultima parte (un suffisso) di w_1 è uguale alla prima parte (un prefisso) di w_2 . Eliminando le parti in comune e leggendo il restante si ottiene la terza parola w_3 che si chiama "lucchetto".

Per esempio:

CANE/NERO = CARO;
 VELA/LANA = VENA;
 CRISTALLO/STALLONE = CRINE;
 RISCHIO/SCHIOCCO = RICCO,
 MAGGIO/GIOIA = MAGIA
 CARATO/ATOMI = CARMI
 CAST/TERME = CASERME
 MATCH/CHADOR = MATADOR.

- Scrivere una funzione `int lunghe( char s[] )` che restituisce la lunghezza della stringa s ;
- Scrivere una funzione `int suff_pref( char s[], char t[], int n )` che restituisce 1 se il suffisso di lunghezza n della stringa s è uguale al prefisso di lunghezza n della stringa t e restituisce 0 in caso siano diversi. (Essenzialmente la stringa s deve essere controllata a partire dalla posizione $\text{lunghe}(s)-n$).
- Scrivere un programma che legge due stringhe e utilizzando la funzione precedente verifica se le due stringhe possono creare un lucchetto.
- (opzionale +3 punti) Il programma scrive un lucchetto tra le due parole (che ovviamente non deve necessariamente avere un senso come parola in italiano) .