

Cognome.....

# Algoritmi e Strutture Dati

Nome.....

Prova scritta del 22 febbraio 2019

Matricola.....

TEMPO DISPONIBILE: 2 ore

Le risposte agli esercizi 1 e 2 devono essere scritte negli appositi riquadri su questo foglio (risposte scritte su altri fogli non saranno considerate). La soluzione dell'esercizio 3 va scritta su uno dei fogli di protocollo forniti. Le brutte copie NON devono essere consegnate. Ricordatevi di scrivere cognome e nome su TUTTI i fogli (inclusi quelli di brutta).

1. Considerate la funzione  $f : \{a, b, \dots, z\} \rightarrow \{0, 1, \dots, 15\}$  definita come segue:

| $k$ | $f(k)$ |
|-----|--------|
| a   | 0      |
| b   | 1      |
| c   | 2      |
| d   | 3      |
| e   | 4      |
| f   | 5      |
| g   | 6      |
| h   | 6      |
| i   | 7      |

| $k$ | $f(k)$ |
|-----|--------|
| j   | 7      |
| k   | 7      |
| l   | 8      |
| m   | 9      |
| n   | 10     |
| o   | 10     |
| p   | 11     |
| q   | 12     |
| r   | 12     |

| $k$ | $f(k)$ |
|-----|--------|
| s   | 13     |
| t   | 13     |
| u   | 14     |
| v   | 14     |
| w   | 15     |
| x   | 15     |
| y   | 15     |
| z   | 15     |

|    |  |
|----|--|
| 0  |  |
| 1  |  |
| 2  |  |
| 3  |  |
| 4  |  |
| 5  |  |
| 6  |  |
| 7  |  |
| 8  |  |
| 9  |  |
| 10 |  |
| 11 |  |
| 12 |  |
| 13 |  |
| 14 |  |
| 15 |  |

Sia  $h$  la funzione che trasforma ogni parola  $k$  sull'alfabeto  $\{a, b, \dots, z\}$  nell'intero che si ottiene applicando  $f$  al primo carattere di  $k$  e  $g$  la funzione che trasforma ogni parola  $k$  nel più piccolo numero primo maggiore o uguale alla lunghezza di  $k$ .

Esempi:  $h(\text{ratto}) = 12$ ,  $g(\text{ratto}) = 5$ ,  $h(\text{cane}) = 2$ ,  $g(\text{cane}) = 5$ .

Inserite nella tabella hash a destra, inizialmente vuota, le seguenti parole, nell'ordine indicato:

salmone tonno rombo aringa cernia aragosta totano  
ostrica

Come funzione hash utilizzate  $h$ . Per la gestione delle collisioni utilizzate l'hashing doppio mediante la funzione

$$c(k, i) = (h(k) + i \cdot g(k)) \bmod 16$$

2. La seguente sequenza di numeri, memorizzata in un array, deve essere ordinata in modo crescente:

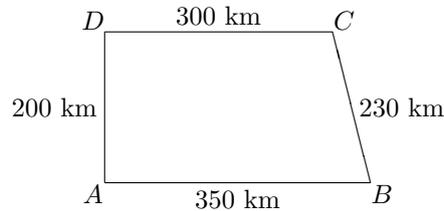
12 24 14 7 23 16 18 15

- |   |
|---|
| (a) Supponete di ordinare la sequenza mediante l'algoritmo <code>quickSort</code> , scegliendo come perno il primo elemento. Indicate il contenuto dell'array dopo avere effettuato la partizione, prima delle chiamate ricorsive di <code>quickSort</code> . |
| (b) Supponete di ordinare la sequenza mediante l'algoritmo <code>heapSort</code> . Indicate il contenuto dell'array dopo averlo trasformato in uno heap.  |
| (c) Supponete di ordinare la sequenza mediante l'algoritmo <code>insertionSort</code> . Indicate il contenuto dell'array dopo la terza iterazione del ciclo principale dell'algoritmo.  |
| (d) Supponete di dividere l'array in due parti formate rispettivamente dagli elementi nelle prime 4 e nelle ultime 4 posizioni e di applicare ad esse l'algoritmo di <code>merge</code> , senza averle ordinate. Indicate la sequenza che si ottiene.         |

**3.** In una strana nazione, da poco diventata indipendente, occorre decidere quale città sarà la capitale. Dopo lunghe discussioni, il parlamento ha stabilito il criterio bizzarro secondo cui verrà scelta la città la cui distanza media da *tutte le altre* città risulta massima. È necessario progettare un algoritmo che, dato l'elenco delle strade che collegano *direttamente* tra loro due città, con le relative lunghezze, determini quale città debba essere scelta come capitale in base al criterio indicato.

*Esempio*

Supponete che vi siano 4 città  $A, B, C, D$  con le strade schematizzate nella figura seguente. La città che in media risulta più distante da tutte le altre è la città  $B$  (distanza media 370 km; per  $A$  la distanza media dalle altre città è di 350 km; per  $C$  e  $D$  è poco più di 343 km).



*Cosa si richiede*

Indichiamo le città con  $c_1, \dots, c_n$ . Progettate un algoritmo che risolva il problema, ricevendo in ingresso una matrice  $M$  dove, per  $i, j = 1, \dots, n$ , l'elemento  $m_{i,j}$  è così definito:

$$m_{i,j} = \begin{cases} 0 & \text{se } i = j, \\ \text{lunghezza strada che collega direttamente } c_i \text{ e } c_j & \text{se tale strada esiste,} \\ \infty & \text{altrimenti.} \end{cases}$$

Risolvete i seguenti punti ***nell'ordine indicato***.

- Descrivete *sinteticamente a parole* e poi *ad alto livello in pseudocodice* un algoritmo che risolva il problema. Potete adattare uno degli algoritmi presentati nel corso (indicate esplicitamente quale algoritmo utilizzate). L'algoritmo dovrà indicare la *città con distanza media maggiore* e il *valore* di tale distanza.
- Fornite una stima del tempo di calcolo dell'algoritmo ottenuto in funzione del numero  $n$  di città.
- Indicate come l'algoritmo possa essere modificato in modo da determinare la *coppia di città più distanti tra loro*.
- Fornite una stima del tempo di calcolo utilizzato dalla parte di algoritmo descritta al punto (c) in funzione di  $n$ .

*Note*

- Le strade possono essere percorse in entrambi i sensi. Pertanto la lunghezza di una strada che collega una città  $c_i$  a una città  $c_j$  è la stessa in entrambe le direzioni.
- Nel caso al punto (a) o al punto (c) vi siano più soluzioni, è sufficiente indicarne una qualunque.
- Potete effettuare operazioni direttamente con il valore  $\infty$ . Il risultato dell'addizione e della sottrazione tra  $\infty$  e un valore finito è  $\infty$ . Il massimo e il minimo tra  $\infty$  e un valore  $x$  sono rispettivamente  $\infty$  e  $x$ .
- Non cambiate quanto stabilito nel testo dell'esercizio per i nomi della matrice di input  $M$  e i nomi  $c_1, \dots, c_n$  (e relativi indici) utilizzati per indicare le città.
- Tutte le risposte devono essere adeguatamente giustificate.

Cognome.....

# Algoritmi e Strutture Dati

Nome.....

Prova scritta del 22 febbraio 2019

Matricola.....

TEMPO DISPONIBILE: 2 ore

Le risposte agli esercizi 1 e 2 devono essere scritte negli appositi riquadri su questo foglio (risposte scritte su altri fogli non saranno considerate). La soluzione dell'esercizio 3 va scritta su uno dei fogli di protocollo forniti. Le brutte copie NON devono essere consegnate. Ricordatevi di scrivere cognome e nome su TUTTI i fogli (inclusi quelli di brutta).

1. Considerate la funzione  $f : \{a, b, \dots, z\} \rightarrow \{0, 1, \dots, 15\}$  definita come segue:

| $k$ | $f(k)$ |
|-----|--------|
| a   | 0      |
| b   | 1      |
| c   | 2      |
| d   | 3      |
| e   | 4      |
| f   | 5      |
| g   | 6      |
| h   | 6      |
| i   | 7      |

| $k$ | $f(k)$ |
|-----|--------|
| j   | 7      |
| k   | 7      |
| l   | 8      |
| m   | 9      |
| n   | 10     |
| o   | 10     |
| p   | 11     |
| q   | 12     |
| r   | 12     |

| $k$ | $f(k)$ |
|-----|--------|
| s   | 13     |
| t   | 13     |
| u   | 14     |
| v   | 14     |
| w   | 15     |
| x   | 15     |
| y   | 15     |
| z   | 15     |

|    |  |
|----|--|
| 0  |  |
| 1  |  |
| 2  |  |
| 3  |  |
| 4  |  |
| 5  |  |
| 6  |  |
| 7  |  |
| 8  |  |
| 9  |  |
| 10 |  |
| 11 |  |
| 12 |  |
| 13 |  |
| 14 |  |
| 15 |  |

Sia  $h$  la funzione che trasforma ogni parola  $k$  sull'alfabeto  $\{a, b, \dots, z\}$  nell'intero che si ottiene applicando  $f$  al primo carattere di  $k$  e  $g$  la funzione che trasforma ogni parola  $k$  nel più piccolo numero primo maggiore o uguale alla lunghezza di  $k$ .

Esempi:  $h(\text{ratto}) = 12$ ,  $g(\text{ratto}) = 5$ ,  $h(\text{cane}) = 2$ ,  $g(\text{cane}) = 5$ .

Inserite nella tabella hash a destra, inizialmente vuota, le seguenti parole, nell'ordine indicato:

sardina trota rombo astice cefalo aragosta seppia  
nasello

Come funzione hash utilizzate  $h$ . Per la gestione delle collisioni utilizzate l'hashing doppio mediante la funzione

$$c(k, i) = (h(k) + i \cdot g(k)) \bmod 16$$

2. La seguente sequenza di numeri, memorizzata in un array, deve essere ordinata in modo crescente:

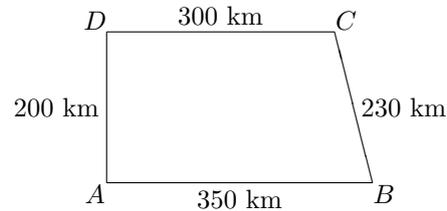
13 25 15 8 24 17 19 16

- |   |
|---|
| (a) Supponete di ordinare la sequenza mediante l'algoritmo <code>quickSort</code> , scegliendo come perno il primo elemento. Indicate il contenuto dell'array dopo avere effettuato la partizione, prima delle chiamate ricorsive di <code>quickSort</code> . |
| (b) Supponete di ordinare la sequenza mediante l'algoritmo <code>heapSort</code> . Indicate il contenuto dell'array dopo averlo trasformato in uno heap.  |
| (c) Supponete di ordinare la sequenza mediante l'algoritmo <code>insertionSort</code> . Indicate il contenuto dell'array dopo la terza iterazione del ciclo principale dell'algoritmo.  |
| (d) Supponete di dividere l'array in due parti formate rispettivamente dagli elementi nelle prime 4 e nelle ultime 4 posizioni e di applicare ad esse l'algoritmo di <code>merge</code> , senza averle ordinate. Indicate la sequenza che si ottiene.         |

**3.** In una strana nazione, da poco diventata indipendente, occorre decidere quale città sarà la capitale. Dopo lunghe discussioni, il parlamento ha stabilito il criterio bizzarro secondo cui verrà scelta la città la cui distanza media da *tutte le altre* città risulta massima. È necessario progettare un algoritmo che, dato l'elenco delle strade che collegano *direttamente* tra loro due città, con le relative lunghezze, determini quale città debba essere scelta come capitale in base al criterio indicato.

*Esempio*

Supponete che vi siano 4 città  $A, B, C, D$  con le strade schematizzate nella figura seguente. La città che in media risulta più distante da tutte le altre è la città  $B$  (distanza media 370 km; per  $A$  la distanza media dalle altre città è di 350 km; per  $C$  e  $D$  è poco più di 343 km).



*Cosa si richiede*

Indichiamo le città con  $c_1, \dots, c_n$ . Progettate un algoritmo che risolva il problema, ricevendo in ingresso una matrice  $M$  dove, per  $i, j = 1, \dots, n$ , l'elemento  $m_{i,j}$  è così definito:

$$m_{i,j} = \begin{cases} 0 & \text{se } i = j, \\ \text{lunghezza strada che collega direttamente } c_i \text{ e } c_j & \text{se tale strada esiste,} \\ \infty & \text{altrimenti.} \end{cases}$$

Risolvete i seguenti punti ***nell'ordine indicato***.

- Descrivete *sinteticamente a parole* e poi *ad alto livello in pseudocodice* un algoritmo che risolva il problema. Potete adattare uno degli algoritmi presentati nel corso (indicate esplicitamente quale algoritmo utilizzate). L'algoritmo dovrà indicare la *città con distanza media maggiore* e il *valore* di tale distanza.
- Fornite una stima del tempo di calcolo dell'algoritmo ottenuto in funzione del numero  $n$  di città.
- Indicate come l'algoritmo possa essere modificato in modo da determinare la *coppia di città più distanti tra loro*.
- Fornite una stima del tempo di calcolo utilizzato dalla parte di algoritmo descritta al punto (c) in funzione di  $n$ .

*Note*

- Le strade possono essere percorse in entrambi i sensi. Pertanto la lunghezza di una strada che collega una città  $c_i$  a una città  $c_j$  è la stessa in entrambe le direzioni.
- Nel caso al punto (a) o al punto (c) vi siano più soluzioni, è sufficiente indicarne una qualunque.
- Potete effettuare operazioni direttamente con il valore  $\infty$ . Il risultato dell'addizione e della sottrazione tra  $\infty$  e un valore finito è  $\infty$ . Il massimo e il minimo tra  $\infty$  e un valore  $x$  sono rispettivamente  $\infty$  e  $x$ .
- Non cambiate quanto stabilito nel testo dell'esercizio per i nomi della matrice di input  $M$  e i nomi  $c_1, \dots, c_n$  (e relativi indici) utilizzati per indicare le città.
- Tutte le risposte devono essere adeguatamente giustificate.

Cognome.....

# Algoritmi e Strutture Dati

Nome.....

Prova scritta del 22 febbraio 2019

Matricola.....

TEMPO DISPONIBILE: 2 ore

Le risposte agli esercizi 1 e 2 devono essere scritte negli appositi riquadri su questo foglio (risposte scritte su altri fogli non saranno considerate). La soluzione dell'esercizio 3 va scritta su uno dei fogli di protocollo forniti. Le brutte copie NON devono essere consegnate. Ricordatevi di scrivere cognome e nome su TUTTI i fogli (inclusi quelli di brutta).

1. Considerate la funzione  $f : \{a, b, \dots, z\} \rightarrow \{0, 1, \dots, 15\}$  definita come segue:

| $k$ | $f(k)$ |
|-----|--------|
| a   | 0      |
| b   | 1      |
| c   | 2      |
| d   | 3      |
| e   | 4      |
| f   | 5      |
| g   | 6      |
| h   | 6      |
| i   | 7      |

| $k$ | $f(k)$ |
|-----|--------|
| j   | 7      |
| k   | 7      |
| l   | 8      |
| m   | 9      |
| n   | 10     |
| o   | 10     |
| p   | 11     |
| q   | 12     |
| r   | 12     |

| $k$ | $f(k)$ |
|-----|--------|
| s   | 13     |
| t   | 13     |
| u   | 14     |
| v   | 14     |
| w   | 15     |
| x   | 15     |
| y   | 15     |
| z   | 15     |

|    |  |
|----|--|
| 0  |  |
| 1  |  |
| 2  |  |
| 3  |  |
| 4  |  |
| 5  |  |
| 6  |  |
| 7  |  |
| 8  |  |
| 9  |  |
| 10 |  |
| 11 |  |
| 12 |  |
| 13 |  |
| 14 |  |
| 15 |  |

Sia  $h$  la funzione che trasforma ogni parola  $k$  sull'alfabeto  $\{a, b, \dots, z\}$  nell'intero che si ottiene applicando  $f$  al primo carattere di  $k$  e  $g$  la funzione che trasforma ogni parola  $k$  nel più piccolo numero primo maggiore o uguale alla lunghezza di  $k$ .

Esempi:  $h(\text{ratto}) = 12$ ,  $g(\text{ratto}) = 5$ ,  $h(\text{cane}) = 2$ ,  $g(\text{cane}) = 5$ .

Inserite nella tabella hash a destra, inizialmente vuota, le seguenti parole, nell'ordine indicato:

sgombro tonno razza aringa cefalo anguilla totano  
nasello

Come funzione hash utilizzate  $h$ . Per la gestione delle collisioni utilizzate l'hashing doppio mediante la funzione

$$c(k, i) = (h(k) + i \cdot g(k)) \bmod 16$$

2. La seguente sequenza di numeri, memorizzata in un array, deve essere ordinata in modo crescente:

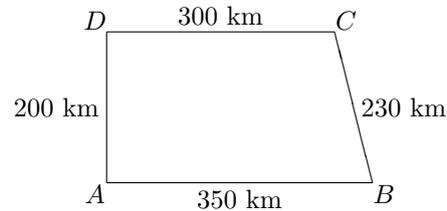
14 26 16 9 25 18 20 17

- |   |
|---|
| (a) Supponete di ordinare la sequenza mediante l'algoritmo <code>quickSort</code> , scegliendo come perno il primo elemento. Indicate il contenuto dell'array dopo avere effettuato la partizione, prima delle chiamate ricorsive di <code>quickSort</code> . |
| (b) Supponete di ordinare la sequenza mediante l'algoritmo <code>heapSort</code> . Indicate il contenuto dell'array dopo averlo trasformato in uno heap.  |
| (c) Supponete di ordinare la sequenza mediante l'algoritmo <code>insertionSort</code> . Indicate il contenuto dell'array dopo la terza iterazione del ciclo principale dell'algoritmo.  |
| (d) Supponete di dividere l'array in due parti formate rispettivamente dagli elementi nelle prime 4 e nelle ultime 4 posizioni e di applicare ad esse l'algoritmo di <code>merge</code> , senza averle ordinate. Indicate la sequenza che si ottiene.         |

**3.** In una strana nazione, da poco diventata indipendente, occorre decidere quale città sarà la capitale. Dopo lunghe discussioni, il parlamento ha stabilito il criterio bizzarro secondo cui verrà scelta la città la cui distanza media da *tutte le altre* città risulta massima. È necessario progettare un algoritmo che, dato l'elenco delle strade che collegano *direttamente* tra loro due città, con le relative lunghezze, determini quale città debba essere scelta come capitale in base al criterio indicato.

*Esempio*

Supponete che vi siano 4 città  $A, B, C, D$  con le strade schematizzate nella figura seguente. La città che in media risulta più distante da tutte le altre è la città  $B$  (distanza media 370 km; per  $A$  la distanza media dalle altre città è di 350 km; per  $C$  e  $D$  è poco più di 343 km).



*Cosa si richiede*

Indichiamo le città con  $c_1, \dots, c_n$ . Progettate un algoritmo che risolva il problema, ricevendo in ingresso una matrice  $M$  dove, per  $i, j = 1, \dots, n$ , l'elemento  $m_{i,j}$  è così definito:

$$m_{i,j} = \begin{cases} 0 & \text{se } i = j, \\ \text{lunghezza strada che collega direttamente } c_i \text{ e } c_j & \text{se tale strada esiste,} \\ \infty & \text{altrimenti.} \end{cases}$$

Risolvete i seguenti punti ***nell'ordine indicato***.

- Descrivete *sinteticamente a parole* e poi *ad alto livello in pseudocodice* un algoritmo che risolva il problema. Potete adattare uno degli algoritmi presentati nel corso (indicate esplicitamente quale algoritmo utilizzate). L'algoritmo dovrà indicare la *città con distanza media maggiore* e il *valore* di tale distanza.
- Fornite una stima del tempo di calcolo dell'algoritmo ottenuto in funzione del numero  $n$  di città.
- Indicate come l'algoritmo possa essere modificato in modo da determinare la *coppia di città più distanti tra loro*.
- Fornite una stima del tempo di calcolo utilizzato dalla parte di algoritmo descritta al punto (c) in funzione di  $n$ .

*Note*

- Le strade possono essere percorse in entrambi i sensi. Pertanto la lunghezza di una strada che collega una città  $c_i$  a una città  $c_j$  è la stessa in entrambe le direzioni.
- Nel caso al punto (a) o al punto (c) vi siano più soluzioni, è sufficiente indicarne una qualunque.
- Potete effettuare operazioni direttamente con il valore  $\infty$ . Il risultato dell'addizione e della sottrazione tra  $\infty$  e un valore finito è  $\infty$ . Il massimo e il minimo tra  $\infty$  e un valore  $x$  sono rispettivamente  $\infty$  e  $x$ .
- Non cambiate quanto stabilito nel testo dell'esercizio per i nomi della matrice di input  $M$  e i nomi  $c_1, \dots, c_n$  (e relativi indici) utilizzati per indicare le città.
- Tutte le risposte devono essere adeguatamente giustificate.

Cognome.....

# Algoritmi e Strutture Dati

Nome.....

Prova scritta del 22 febbraio 2019

Matricola.....

TEMPO DISPONIBILE: 2 ore

Le risposte agli esercizi 1 e 2 devono essere scritte negli appositi riquadri su questo foglio (risposte scritte su altri fogli non saranno considerate). La soluzione dell'esercizio 3 va scritta su uno dei fogli di protocollo forniti. Le brutte copie NON devono essere consegnate. Ricordatevi di scrivere cognome e nome su TUTTI i fogli (inclusi quelli di brutta).

1. Considerate la funzione  $f : \{a, b, \dots, z\} \rightarrow \{0, 1, \dots, 15\}$  definita come segue:

| $k$ | $f(k)$ |
|-----|--------|
| a   | 0      |
| b   | 1      |
| c   | 2      |
| d   | 3      |
| e   | 4      |
| f   | 5      |
| g   | 6      |
| h   | 6      |
| i   | 7      |

| $k$ | $f(k)$ |
|-----|--------|
| j   | 7      |
| k   | 7      |
| l   | 8      |
| m   | 9      |
| n   | 10     |
| o   | 10     |
| p   | 11     |
| q   | 12     |
| r   | 12     |

| $k$ | $f(k)$ |
|-----|--------|
| s   | 13     |
| t   | 13     |
| u   | 14     |
| v   | 14     |
| w   | 15     |
| x   | 15     |
| y   | 15     |
| z   | 15     |

|    |  |
|----|--|
| 0  |  |
| 1  |  |
| 2  |  |
| 3  |  |
| 4  |  |
| 5  |  |
| 6  |  |
| 7  |  |
| 8  |  |
| 9  |  |
| 10 |  |
| 11 |  |
| 12 |  |
| 13 |  |
| 14 |  |
| 15 |  |

Sia  $h$  la funzione che trasforma ogni parola  $k$  sull'alfabeto  $\{a, b, \dots, z\}$  nell'intero che si ottiene applicando  $f$  al primo carattere di  $k$  e  $g$  la funzione che trasforma ogni parola  $k$  nel più piccolo numero primo maggiore o uguale alla lunghezza di  $k$ .

Esempi:  $h(\text{ratto}) = 12$ ,  $g(\text{ratto}) = 5$ ,  $h(\text{cane}) = 2$ ,  $g(\text{cane}) = 5$ .

Inserite nella tabella hash a destra, inizialmente vuota, le seguenti parole, nell'ordine indicato:

spigola trota razza astice cernia anguilla seppia  
ostrica

Come funzione hash utilizzate  $h$ . Per la gestione delle collisioni utilizzate l'hashing doppio mediante la funzione

$$c(k, i) = (h(k) + i \cdot g(k)) \bmod 16$$

2. La seguente sequenza di numeri, memorizzata in un array, deve essere ordinata in modo crescente:

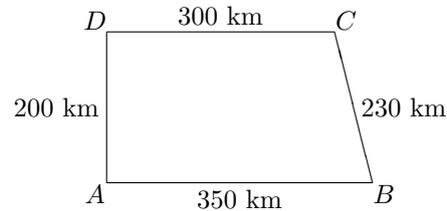
15 27 17 10 26 19 21 18

- |   |
|---|
| (a) Supponete di ordinare la sequenza mediante l'algoritmo <code>quickSort</code> , scegliendo come perno il primo elemento. Indicate il contenuto dell'array dopo avere effettuato la partizione, prima delle chiamate ricorsive di <code>quickSort</code> . |
| (b) Supponete di ordinare la sequenza mediante l'algoritmo <code>heapSort</code> . Indicate il contenuto dell'array dopo averlo trasformato in uno heap.  |
| (c) Supponete di ordinare la sequenza mediante l'algoritmo <code>insertionSort</code> . Indicate il contenuto dell'array dopo la terza iterazione del ciclo principale dell'algoritmo.  |
| (d) Supponete di dividere l'array in due parti formate rispettivamente dagli elementi nelle prime 4 e nelle ultime 4 posizioni e di applicare ad esse l'algoritmo di <code>merge</code> , senza averle ordinate. Indicate la sequenza che si ottiene.         |

**3.** In una strana nazione, da poco diventata indipendente, occorre decidere quale città sarà la capitale. Dopo lunghe discussioni, il parlamento ha stabilito il criterio bizzarro secondo cui verrà scelta la città la cui distanza media da *tutte le altre* città risulta massima. È necessario progettare un algoritmo che, dato l'elenco delle strade che collegano *direttamente* tra loro due città, con le relative lunghezze, determini quale città debba essere scelta come capitale in base al criterio indicato.

*Esempio*

Supponete che vi siano 4 città  $A, B, C, D$  con le strade schematizzate nella figura seguente. La città che in media risulta più distante da tutte le altre è la città  $B$  (distanza media 370 km; per  $A$  la distanza media dalle altre città è di 350 km; per  $C$  e  $D$  è poco più di 343 km).



*Cosa si richiede*

Indichiamo le città con  $c_1, \dots, c_n$ . Progettate un algoritmo che risolva il problema, ricevendo in ingresso una matrice  $M$  dove, per  $i, j = 1, \dots, n$ , l'elemento  $m_{i,j}$  è così definito:

$$m_{i,j} = \begin{cases} 0 & \text{se } i = j, \\ \text{lunghezza strada che collega direttamente } c_i \text{ e } c_j & \text{se tale strada esiste,} \\ \infty & \text{altrimenti.} \end{cases}$$

Risolvete i seguenti punti ***nell'ordine indicato***.

- Descrivete *sinteticamente a parole* e poi *ad alto livello in pseudocodice* un algoritmo che risolva il problema. Potete adattare uno degli algoritmi presentati nel corso (indicate esplicitamente quale algoritmo utilizzate). L'algoritmo dovrà indicare la *città con distanza media maggiore* e il *valore* di tale distanza.
- Fornite una stima del tempo di calcolo dell'algoritmo ottenuto in funzione del numero  $n$  di città.
- Indicate come l'algoritmo possa essere modificato in modo da determinare la *coppia di città più distanti tra loro*.
- Fornite una stima del tempo di calcolo utilizzato dalla parte di algoritmo descritta al punto (c) in funzione di  $n$ .

*Note*

- Le strade possono essere percorse in entrambi i sensi. Pertanto la lunghezza di una strada che collega una città  $c_i$  a una città  $c_j$  è la stessa in entrambe le direzioni.
- Nel caso al punto (a) o al punto (c) vi siano più soluzioni, è sufficiente indicarne una qualunque.
- Potete effettuare operazioni direttamente con il valore  $\infty$ . Il risultato dell'addizione e della sottrazione tra  $\infty$  e un valore finito è  $\infty$ . Il massimo e il minimo tra  $\infty$  e un valore  $x$  sono rispettivamente  $\infty$  e  $x$ .
- Non cambiate quanto stabilito nel testo dell'esercizio per i nomi della matrice di input  $M$  e i nomi  $c_1, \dots, c_n$  (e relativi indici) utilizzati per indicare le città.
- Tutte le risposte devono essere adeguatamente giustificate.