Programmazione 2
e Laboratorio di Programmazione

Corso di Laurea in

Informatica

Università degli Studi di Napoli “Parthenope”

Anno Accademico 2023-2024

Prof. Luigi Catuogno

1

Esercizi svolti

2

Verifica del quadrato magico

3

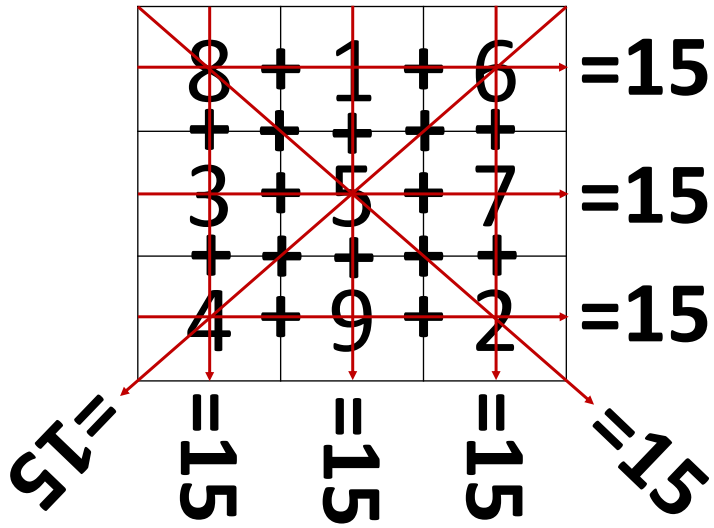
Esercizio: *Il quadrato magico…*

Un quadrato magico è una matrice di interi positivi $N \times N$ riempito con i numeri da 1 a N^2 disposti in modo tale che la somma dei numeri che occupano ciascuna riga, o colonna o diagonale, dia sempre lo stesso valore. Tale valore è costante e dipende dall'ordine del quadrato.

8	1	6
3	5	7
4	9	2

Quadrato magico di ordine 3.
Il numero magico è 15

4



5

Esercizio: *Il quadrato magico…*

- Si scriva il programma C++ che:
  - Chieda all'utente di riempire una matrice di dimensioni 3x3 con i numeri da 1 a 9 (scelti ciascuno una sola volta)
  - Dica se la matrice inserita è un quadrato magico;

Attenzione: Il programma deve essere realizzato in modo che, modificando al più due variabili (quella che indica l'ordine e quella che indica il numero magico), possa essere utilizzato per quadrati di ordine arbitrario.

6

Esercizio: *Il quadrato magico…*

4	9	2
8	1	6
3	5	7

NO

2	9	4
7	5	3
6	1	8

SI

7

Esercizio: *Il quadrato magico…*

```
#include<iostream>
using namespace std;

int main()
{
	const int ordine=3, numero=15;
	int M[ordine][ordine],i,j,sommar=0,sommac=0,sommad1=0,sommad2=0;
	bool magico=true;

	cout << "Inserisci gli elementi di M: "<<endl;
	for(i=0;i<ordine;i++)
		for(j=0;j<ordine;j++){
			cout << "M["<<i<<"]["<<j<<"]= ";
			cin >> M[i][j];
		}
```

8

Esercizio: *Il quadrato magico…*

```
6         const int ordine=3, numero=15;
7         int M[ordine][ordine],i,j,sommar=0,sommac=0,sommad1=0,sommad2=0;
```

La costante intera **ordine** la dimensione del quadrato, la costante interna **numero** indica il *numero magico* associato ai quadrati di ordine 3. L'array **M** contiene il quadrato magico da verificare.

Le variabili **sommar** e **sommac** sono usate rispettivamente per calcolare la somma degli elementi su una stessa riga e una stessa colonna del quadrato.

Le variabili **sommad1** e **sommad2** invece, sono usate per la somma degli elementi sulla diagonale principale e su quella secondaria.

9

Esercizio: *Il quadrato magico…*

```
1  #include<iostream>
2  using namespace std;
3
4  int main()
5  {
6         const int ordine=3, nu     15;
7         int M[ordine][ordine]  ,j,sommar=
8         bool magico=true;
9
10        cout << "Inserisci gli elementi di M:     d1;
11        for(i=0;i<ordine;i++)
12              for(j=0;j<ordine;j++){
13                    cout << "M["<<i<<"]["<<j<<"]= ";
14                    cin >> M[i][j];
15              }
```

Variabile booleana che conterrà il responso. Inizialmente, assumiamo che il quadrato che sarà fornito è magico (magico=true)

Input su matrici «da manuale». Due cicli annidati, uno (variabile guida i) itera sulle righe, l'altro (j) itera sulle colonne.

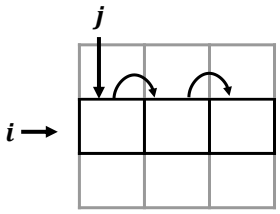
10

Esercizio: *Il quadrato magico*

```
16          for(i=0;i<ordine;i++){
17               sommar=0;
18               sommac=0;
19                  for(j=0;j<ordine;j++){
20                            sommar=sommar+M[i][j];
21                            sommac=sommac+M[j][i];
22                  }
23                  if (sommac!=numero||sommar!=numero) {
24                        magico=false;
25                        break;
26                  }
27          }
 …  …
```

Calcoliamo la somma degli elementi sulla riga **i**-*esima*. Ogni volta azzeriamo la variabile **sommar;**

Si procede aggiungendo di volta in volta l'elemento della colonna **j (per j da 0 a ordine)** che sta sulla riga corrente;



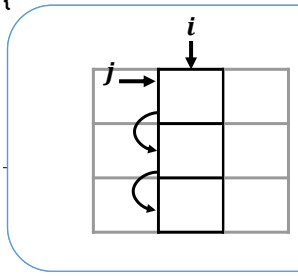
11

Esercizio: *Il quadrato magico*

```
16          for(i=0;i<ordine;i++){
17               sommar=0;
18               sommac=0;
19                  for(j=0;j<ordine;j++){
20                            sommar=sommar+M[i][j];
21                            sommac=sommac+M[j][i];
22                  }
23                  if (sommac!=numero||sommar!=numero) {
24                        magico=false;
25                        break;
26                  }
27          }
 …  …
```

Se invertiamo gli indici, possiamo calcolare contemporaneamente anche la somma degli elementi sulla colonna **i**-*esima* nello stesso modo.

Si procede aggiungendo di volta in volta l'elemento della riga **j (per j da 0 a ordine)** della colonna corrente;



12

Esercizio: *Il quadrato magico…*

```
16          for(i=0;i<ordine;i++){
17              sommar=0;
18              sommac=0;
19                 for(j=0;j<ordine;j++){
20                            sommar=sommar+M[i][j];
21                            sommac=sommac+M[j][i];
22                 }
23                 if (sommac!=numero||sommar!=numero) {
24                        magico=false;
25                        break;
26                 }
27          }
 …  …
```

Se uno dei due totali dovesse risultare diverso dal numero magico, abbiamo finito: **`magico`** passa a **`false`** e terminiamo.

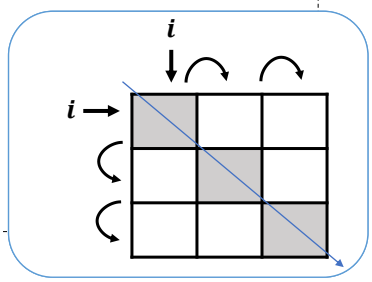
13

Esercizio: *Il quadrato magic*

Se righe e colonne sono OK, magico è ancora true, quindi passiamo a calcolare le somme sulle diagonali

```
28          if(magico){
29                 for(i=0;i<ordine;i++) {
30                        sommad1=sommad1+M[i][i];
31                        sommad2=sommad2+M[i][ordine-1-i]
32                 }
33
34                 if (sommad1!=numero || sommad2!=numero)
35                        magico=false;
36          }
37          cout << "il quadrato e' magico? ";
38          if(magico)
39                 cout << "Si!"<<endl;
40          else
41                 cout << "No!"<<endl;
42  }
```

Sulla diagonale principale, gli elementi hanno coordinate uguali. Quindi, per **`i`** da **0** a **3**, sommiamo gli elementi **`M[i][i]`**.



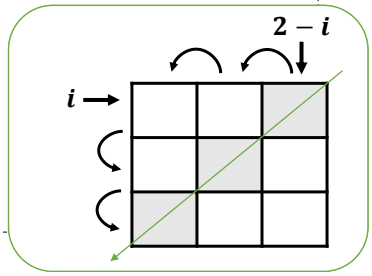
14

Esercizio: *Il quadrato magic*

Se righe è colonne sono OK, magico è ancora true, quindi passiamo a calcolare le somme sulle diagonali

Sulla diagonale secondaria, gli elementi hanno coordinate «opposte» Quindi, per **i** da **0** a **3**, sommiamo gli elementi **M[i][3-1-i]**.

```
28        if(magico){
29              for(i=0;i<ordine;i++) {
30                    sommad1=sommad1+M[i][i];
31                    sommad2=sommad2+M[i][ordine-1-i];
32              }
33
34              if (sommad1!=numero || sommad2!=numero)
35                    magico=false;
36        }
37        cout << "il quadrato e' magico? ";
38        if(magico)
39              cout << "Si!"<<endl;
40        else
41              cout << "No!"<<endl;
42  }
```



15

Esercizio: *Il quadrato magico…*

Se uno dei due totali dovesse risultare diverso dal numero magio, abbiamo finito: **magico** passa a **false**.

```
28        if(magico){
29              for(i=0;i<ordine;i++) {
30                    sommad1=sommad1+M[i][i];
31                    sommad2=sommad2+M[i][ordine-1-i];
32              }
33
34              if (sommad1!=numero || sommad2!=numero)
35                    magico=false;
36        }
37        cout << "il quadrato e' magico? ";
38        if(magico)
39              cout << "Si!"<<endl;
40        else
41              cout << "No!"<<endl;
42  }
```

16

Esercizio: *Il quadrato magico…*

Modificando opportunamente i parametri **`ordine`** e **`numero`**, è possibile utilizzare il programma per verificare qualsiasi quadrato. Il numero magico funzione dall'ordine.

Si modifichi il programma per eseguire la verifica di un quadrato di ordine 5. In questo caso, il numero magico è 65.

17	24	1	8	15
23	5	7	14	16
4	6	13	20	22
10	12	19	21	3
11	18	25	2	9

17

Calcolare il numero magico…

Il *numero magico* di un quadrato dipende solo dal suo ordine è può essere calcolato con la seguente formula (dove n è l'ordine del quadrato):

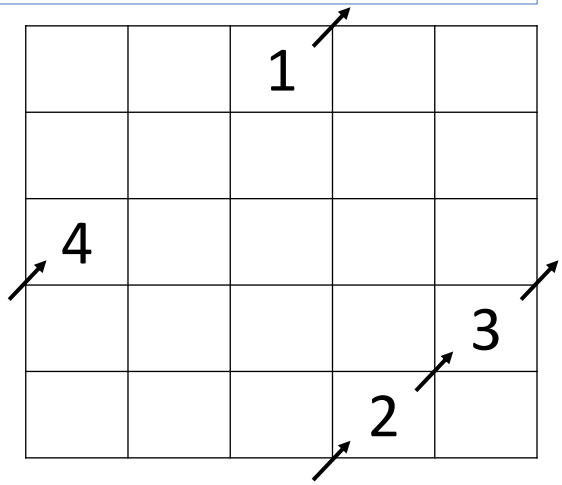
$$M(n) = \frac{1}{2}n(n^2 + 1)$$

17	24	1	8	15
23	5	7	14	16
4	6	13	20	22
10	12	19	21	3
11	18	25	2	9

18

Algoritmo risolutore per ordine *dispari*

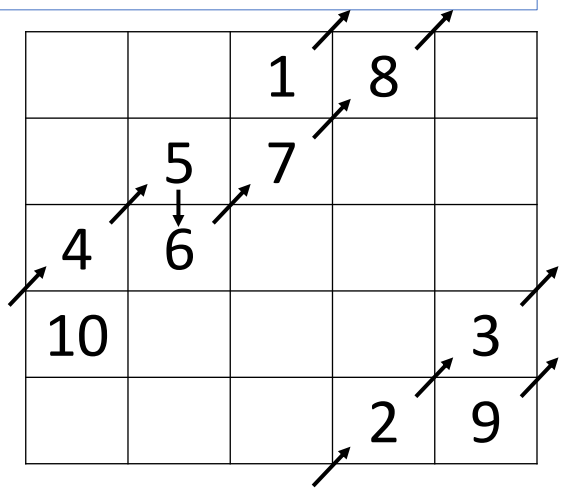
1) Cominciare posizionando 1 nella cella centrale della prima riga
2) Aggiungere nell'ordine gli altri numeri occupando la cella a destra nella riga superiore
   a) se si supera il bordo della tabella, si «sbuca» dall'altra parte (il famigerato «effetto Pac Man»)



19

Algoritmo risolutore per ordine *dispari*

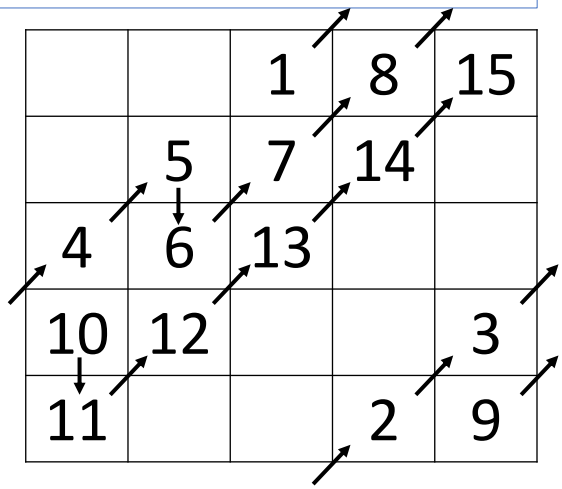
1) Cominciare posizionando 1 nella cella centrale della prima riga
2) Aggiungere nell'ordine gli altri numeri occupando la cella a destra nella riga superiore
   a) se si supera il bordo della tabella, si «sbuca» dall'altra parte (il famigerato «effetto Pac Man»)
   b) Se la cella di sopra a destra è già occupata, allora proseguire la numerazione con quella subito sotto la cella corrente
3) Proseguire fino al riempimento del quadrato.



20

Algoritmo risolutore per ordine *dispari*

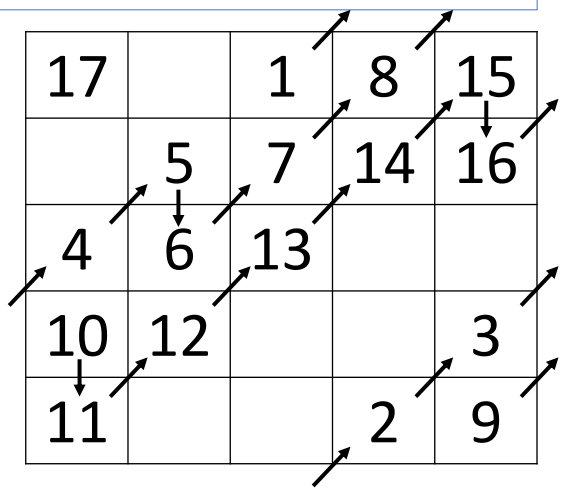
1) Cominciare posizionando 1 nella cella centrale della prima riga
2) Aggiungere nell'ordine gli altri numeri occupando la cella a destra nella riga superiore
   a) se si supera il bordo della tabella, si «sbuca» dall'altra parte (il famigerato «effetto Pac Man»)
   b) Se la cella di sopra a destra è già occupata, allora proseguire la numerazione con quella subito sotto la cella corrente
3) Proseguire fino al riempimento del quadrato.



21

Algoritmo risolutore per ordine *dispari*

1) Cominciare posizionando 1 nella cella centrale della prima riga
2) Aggiungere nell'ordine gli altri numeri occupando la cella a destra nella riga superiore
   a) se si supera il bordo della tabella, si «sbuca» dall'altra parte (il famigerato «effetto Pac Man»)
   b) Se la cella di sopra a destra è già occupata, allora proseguire la numerazione con quella subito sotto la cella corrente
3) Proseguire fino al riempimento del quadrato.



22