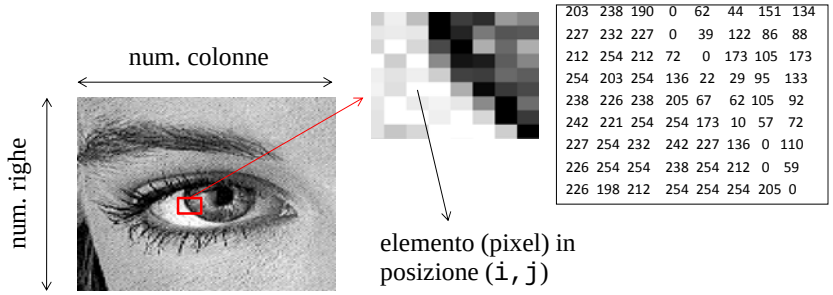
Sommario

- Immagini:
  - Formato PGM
- Tipo di dato astratto per descrivere un'immagine:
  - Memorizzazione dati
  - Operatore sovraccaricato `operator()`

Immagini

- Un'immagine è rappresentabile in memoria come una tabella di numeri.
- Ogni pixel può assumere un valore compreso tra 0 e 255. Tale valore corrisponde al valore di intensità luminosa. (0 nero – 255 bianco).



Immagini

- Le immagini sono salvate su disco in diversi formati, che differiscono fra loro a seconda del modo in cui i dati sono memorizzati su file.
- La modalità con cui vengono salvati i dati è definita all'inizio del file (*header*).
- Pertanto un file immagine sarà composto da:
  - *Header* (specifico per ogni formato): contiene informazioni sull'immagine tra cui numero di righe e numero di colonne.
  - *Dati*: valori assunti dai pixel dell'immagine. I valori possono essere salvati in formato ASCII o binario, compresso o non compresso.

Immagini: formato PGM

- Formato utilizzato per memorizzare immagini in scala di grigio.
- L'header è composto nel modo seguente:
  - Intestazione (P2 o P5) a seconda che il file sia in formato ASCII o binario.
  - Numero di colonne e numero di righe.
  - Valore massimo dei livelli di grigio.
  - Possono essere presenti commenti preceduti da #.
- I dati sono memorizzati in formato ASCII (se P2) o binario (se P5) ordinati per righe.

Immagini

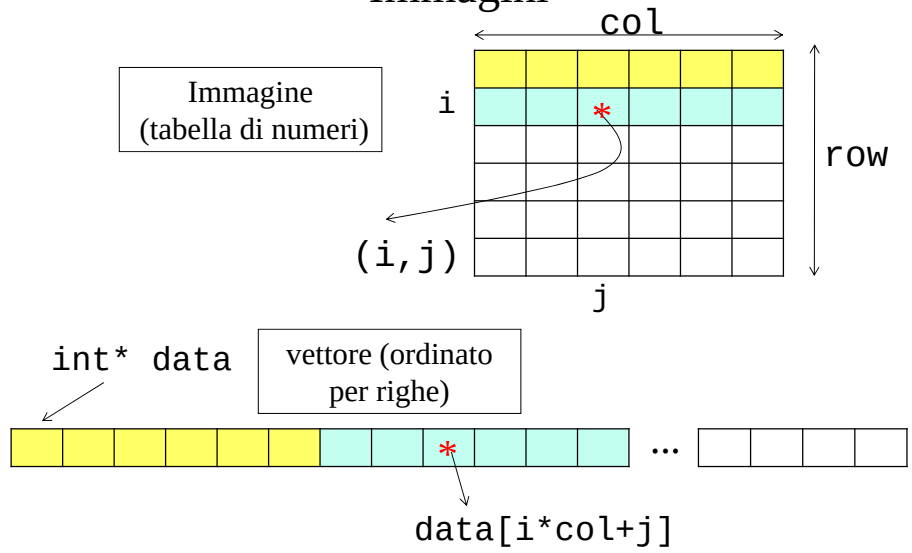
- E' possibile definire un tipo di dato astratto per descrivere un'immagine.
- I pixels dell'immagine sono memorizzati in un *vettore*, ordinati per righe.

MyImage.h

```
class MyImage{

        string type;
        int row;
        int col;
        int maxval;
        int * data;
public:

        CMyImage();
        CMyImage(int r, int c);
        CMyImage(string nomefile);
        ~CMyImage();

        bool Open(string nomefile);
        bool Save(string nomefile);

        ...
};
```

altri metodi necessari alla gestione di un'immagine

Immagini



Operatori sovraccaricati: ()

– Si può eseguire l'overloading dell'operatore di chiamata a funzione (), creando in tal modo una funzione operator alla quale è possibile passare qualsiasi tipo e numero di parametri e che restituisce un valore di qualsiasi tipo: `operator()(...)`

Operatori sovraccaricati: () - Immagine

- Si può eseguire l'overloading dell'operatore di chiamata a funzione (), per la classe MyImage per accedere ai pixels dell'immagine.

Il tipo di ritorno è un riferimento, in modo tale da poter utilizzare l'operatore sia sul *lato sinistro* sia sul *lato destro* di un assegnamento.

```
int& CMyImage::operator ()(int i, int j)
{
        return data[i*col+j];
}
```

Operatori sovraccaricati: () - Immagine

- Pertanto si potranno leggere e modificare i pixels dell'immagine nel modo seguente:

```
MyImage im1("eye.pgm");
```

– Utilizzo dell'operatore () sul *lato sinistro* di un assegnamento:

```
im1(10,15)=0;
```

– Utilizzo dell'operatore () sul *lato destro* di un assegnamento:

```
int val=im1(23,24);
```