# Programmare in Processing - Parte 2

## Valori

Sappiamo come utilizzare le funzioni per far effettuare al computer le operazioni che vogliamo. Spesso le funzioni necessitano di parametri. Quando chiamiamo una funzione, per fornirgli i parametri utilizziamo i **valori**.
I **valori** sono i numeri che passiamo a una funzione attraverso i parametri.

Es. **ellipse(50, 75, 30, 20)**

In questa linea di codice 50, 70, 30 e 30 sono i **valori** che assumono i parametri x, y, width(larghezza) e height (altezza) nella funzione ellipse per far disegnare al computer un'ellisse.

## Operatori

A livello base i computer sono dei velocissimi calcolatori e calcolano numeri modificando valori con gli **operatori** aritmetici +, -, *, e /
Se combiniamo due valori con un operatore ne otteniamo un terzo che possiamo usare come qualsiasi altro valore. Quindi possiamo riscrivere la linea di codice precedente sostituendo ai valori la combinazione di due valori con un operatore:

```
ellipse(10+40, 300-225, 3*10, 40/2);
```

Questa linea di codice ci dice esattamente la stessa cosa della versione precedente: 50, 70, 30 e 30 sono i **valori** che facciamo assumere ai parametri x, y, width(larghezza) e height (altezza) nella funzione ellipse con la differenza che questi valori li otteniamo con valori combinati con operatori.
L'uso degli operatori diventa indispensabile quando dobbiamo cambiare dei valori in istruzioni che si ripetono in tante volte. Facciamo un esempio. Questo codice permette di disegnare un bersaglio che riempie la finestra di display:

```
size(200, 200);
fill(255, 0, 0);
ellipse(100, 100, 200, 200);
fill(255, 255, 255);
ellipse(100, 100, 150, 150);
fill(255, 0, 0);
ellipse(100, 100, 100, 100);
```

Se cambiamo la dimensione della finestra, per esempio la facciamo diventare 400x400, per avere sempre i cerchi centrati nella finestra dobbiamo cambiare il codice in molti posti e calcolare le nuove posizioni e le nuove dimensioni.

```
size(400, 400);
fill(255, 0, 0);
ellipse(200, 200, 400, 400);
fill(255, 255, 255);
ellipse(200, 200, 300, 300);
fill(255, 0, 0);
ellipse(200, 200, 200, 200);
```


Dobbiamo eseguire i seguenti calcoli:

- Per trovare le coordinate del centro dei cerchi: il valore della larghezza e il valore dell'altezza della finestra devono essere divisi per 2.
- Per il primo cerchio: i valori di larghezza e altezza devono essere identici a quelli corrispondenti alle dimensioni della finestra.
- Per il secondo cerchio: valori uguali al 75% dei valori delle dimensioni della finestra.
- Per il terzo cerchio: valori uguali al 50% dei valori delle dimensioni della finestra.

Cosa succede se vogliamo ulteriormente cambiare le dimensioni della finestra?

- Un modo di procedere, non ancora ottimale, potrebbe essere di impostare le operazioni matematiche e farle fare direttamente al computer

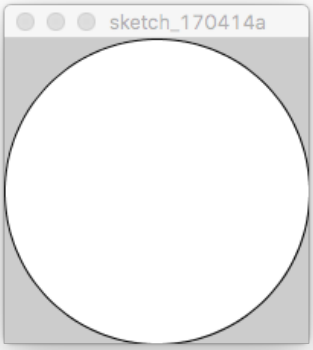
```
size(400, 400);
fill(255, 0, 0);
ellipse(400/2, 400/2, 400, 400);
fill(255, 255, 255);
ellipse(400/2, 400/2, 400*.75, 400*.75);
fill(255, 0, 0);
ellipse(400/2, 400/2, 400/2, 400/2);
```

## Variabili

A questo punto se vogliamo cambiare nuovamente le dimensioni della finestra, dobbiamo sostituire i nuovi valori a tutti i 400. Possiamo provare utilizzando 150 come nuovo valore per larghezza ed altezza della finestra.
Questa procedura ci permette di non dover nuovamente effettuare i calcoli matematici ma utilizzando Processing abbiamo la possibilità di fare di meglio. Non sarebbe bello se Processing tenesse traccia dei valori per altezza e larghezza al posto nostro?
Questa cosa è possibile utilizzando le **variabili**, che non sono altro dei nomi (o contenitori in memoria) che contengono valori.

Una **variabile** può essere utilizzata ovunque sia presente un **valore,** inclusi i casi in cui vengono utilizzati gli operatori.
Alcune variabili sono autodefiniti da Processing. Per esempio le variabili **width** e **heigh** contengono rispettivamente la larghezza e l'altezza della finestra di lavoro.
Per esempio per disegnare un cerchio che riempie la finestra basta il seguente codice:

```
size(200, 200);
ellipse(width/2, height/2, width, height);
```




Questo programma funziona perché sappiamo che la dimensione della finestra è 200x200 e possiamo impostare il centro del cerchio nel mezzo della finestra [100,100] utilizzando le variabili **width** e **height**.

Questo sistema diventa sempre più utile via via che la dimensione del programma aumenta.
Proviamo ad esempio a riscrivere il codice del bersaglio utilizzando le variabili:

```
size(150, 150)
fill (255, 0, 0)
ellipse(width/2, height/2, width, height)
fill (255, 255, 255)
ellipse(width/2, height/2, width *0.75, height * 0.75)
fill (255, 0, 0)
ellipse(width/2, height/2, width/2, height/2)
```

# ESERCIZI

Provare a riscrivere gli esercizi della precedente lezione utilizzando le variabili width e height, in modo da ottenere un programma che funziona per qualunque dimensione della finestra. E' utile provare a disegnare prima la figura su un pezzo di carta in modo da visualizzare i rapporti della figura rispetto alle dimensioni della finestra.

Scrivere il programma in modo che utilizzi colori differenti in base alla dimensione della finestra

Siate creativi utilizzando come colore il risultato della funzione random() che genera numeri casuali (aiutarsi con la guida online di Processing)