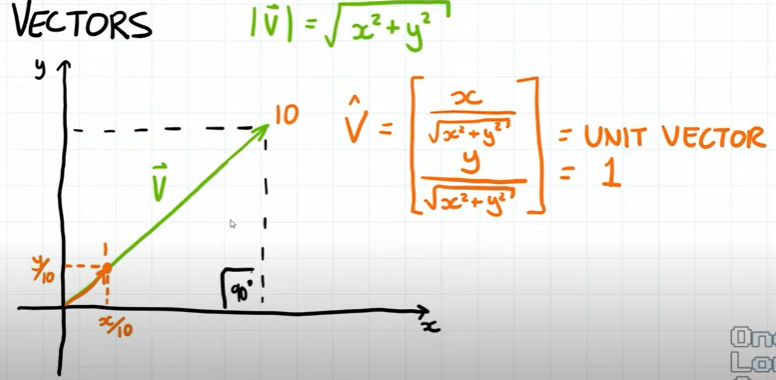
**ESSENTIAL MATHEMATICS**

**UNITARY VECTOR – NORMALIZING**

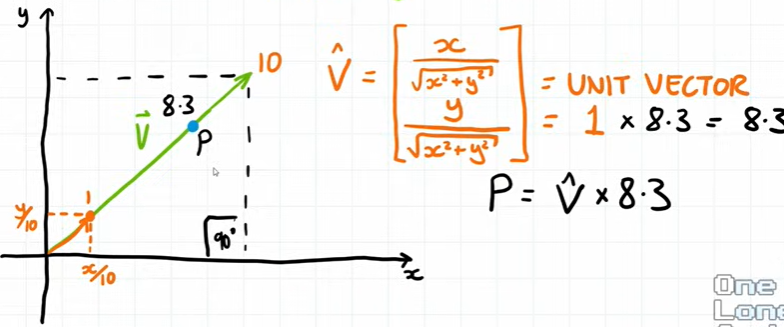
Ogni vettore può essere “normalizzato” – reso di lunghezza (o magnitudo) = 1 – calcolando la lunghezza  
di tale vettore attraverso il teorema di pitagora e dividendo le due componendi x ed y per tale lunghezza



Ricordiamo che le componenti fondamentali di un vettore sono **lunghezza** e **direzione**. Attraverso la   
normalizzazione possiamo “eliminare” la componente lunghezza per tenere conto solo della direzione.

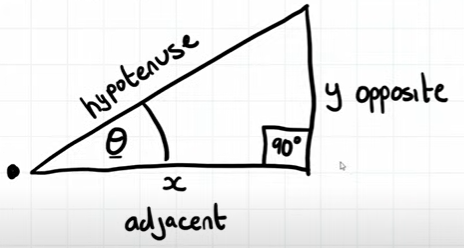
• Supponiamo di voler **trovare un punto lungo la direzione del mio vettore che dista un dato valore   
 dall’origine**:

Quello che dovrò fare sarà semplicemente moltiplicare il mio vettore unitario per tale distanza



**ANGLES**

**GRADI**



**Formule in angoli**

NOTA: In molti linguaggi di programmazione invece di troviamo

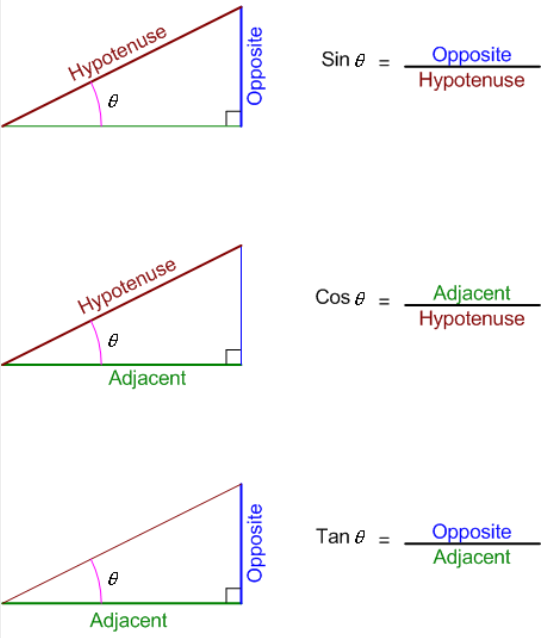
Bisogna però fare attenzione ad esempio che x sia diversa da zero, per questo  
spesso di fa uso della funzione **atan2** che esegue tutta una serie di controlli

**RADIANTS**  
• Solitamente i programmi non elaborano questo tipo di informazioni in termini di “angoli” ma di **radianti**.

**Conversione**

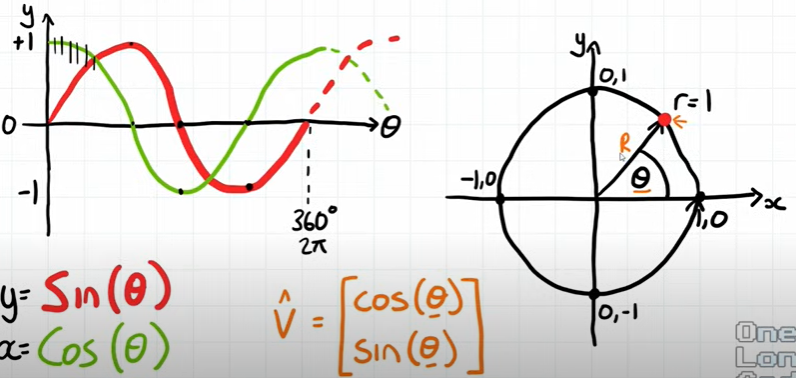
**Formule**

Un modo per memorizzare le relazione che si creano fra ipotenusa ed assi è l’acronimo **SOH CAH TOA**



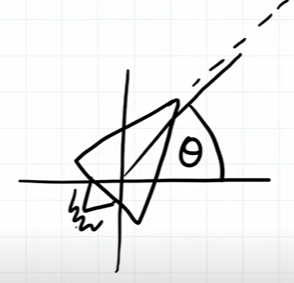
**SIN & COS**

Possiamo descrivere un vettore unitario in termini di seno e coseno



Questo risulta utile nella programmazione di un gioco: Supponiamo di avere un’astronave e di conoscere   
di cui conosciamo solo l’angolo rispetto all’asse cartesiano

Attraverso le coordinate polari posiamo trasformare questo angolo in un “vettore direzione” lungo il quale  
viaggerà l’astronave. Sarà inoltre semplice ruotare l’astronave in quanto basterà aumentare o diminuire   
l’angolo

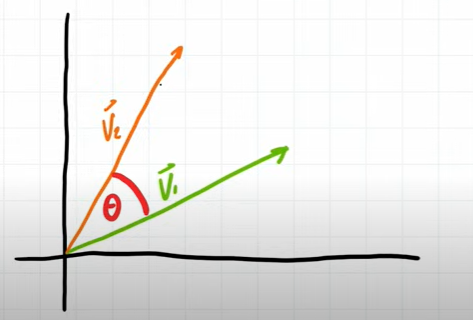


Passare da angoli a vettori può essere utili a seconda dei casi: Ruotare un oggetto verrà più facile in termini   
di angoli mentre muoverlo in una certa direzione richiederà l’uso di vettori

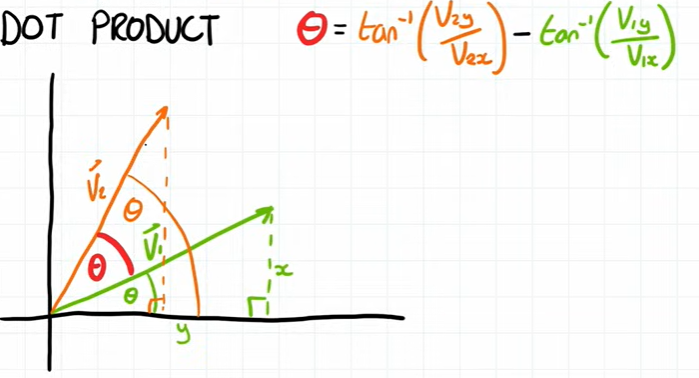
NOTA: Sin, Cos, tan, sqrt risultano **pesanti in termini computazionali**. Mentre i calcoli richiesti dai vettori  
 risultano più leggeri (risultano quindi preferibili)

**DOT PRODUCT**Dato il discorso sul costo computazionale, risulta utile il “dot product” che ci permette di lavorare sugli   
angoli in vermini di vettori.

Supponiamo di avere due vettori sul piano e di voler calcolare l’angolo che formano fra loro

****  
Cominciamo rappresentando questo vettori con vettori unitari in quanto il Dot Product sarà dato  
dalla somma del prodotto delle singole componenti dei vettori.

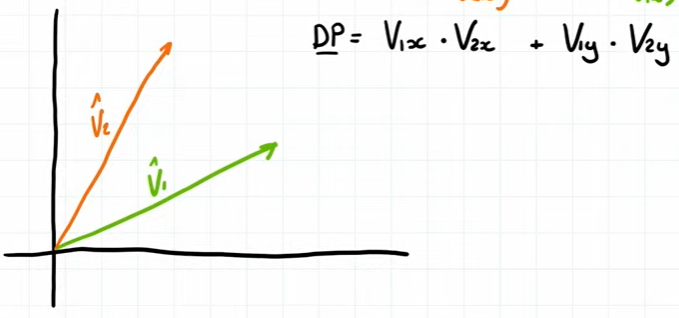
Vediamo prima il metodo classico per calcolarlo:



Come vediamo ci sono diversi calcoli da fare ed il costo computazionale sarebbe elevato.

• C’è un altro modo per risolvere questo problema. Ricordiamo che un vettore è composto da una direzione ed una lunghezza. In questa situazione la lunghezza risulta però irrilevante perché al variare di quest’ultima non cambia l’angolo fra i due vettori.

Possiamo quindi considerare i vettori unitari ed il dot product sarà dato dalla somma del prodotto delle singole componenti dei vettori



Questo sarà un risultato scalare (un numero) e non un vettore. Vediamo perché.

• Consideriamo un singolo vettore e che anche il nostro asse x sia un vettore unitario:

minuto 26:16