TI-Nspire™ Python
Guida alla programmazione

Per sapere di più sulle tecnologie TI, consultare la guida online all'indirizzo education.ti.com/eguide.

Informazioni importanti

A meno che non sia indicato diversamente nella licenza fornita con un programma, Texas Instruments, relativamente ai programmi o ai materiali di riferimento, non rilascia alcuna garanzia, né esplicita né implicita, ivi comprese, a mero titolo esemplificativo, garanzie implicite di commerciabilità e idoneità per uno scopo particolare. Tali materiali, quindi, sono disponibili solo "così come forniti". In nessun caso Texas Instruments potrà essere ritenuta responsabile di danni speciali, collaterali, accidentali o conseguenti, collegati o riconducibili all'acquisto o all'utilizzo di tali materiali. A prescindere da qualunque tipo di azione legale eventualmente intrapresa, la responsabilità di Texas Instruments è limitata all'importo indicato nella licenza del programma. Texas Instruments, inoltre, non potrà essere ritenuta responsabile per eventuali reclami, di qualunque tipo, riguardanti l'utilizzo di tali materiali da parte di terzi.

"Python" e i loghi Python sono marchi o marchi registrati della Python Software Foundation, utilizzati da Texas Instruments Incorporated con autorizzazione della Foundation.

I prodotti reali possono differire leggermente dalle immagini pubblicate.

Sommario

Introduzione alla programmazione di Python

Utilizzando Python con i prodotti TI-Nspire™ è possibile:

- aggiungere i programmi Python ai file TNS
- creare programmi Python utilizzando i modelli
- interagire e condividere dati con altre app TI-Nspire™
- Interagire con TI-Innovator™ Hub e TI-Innovator™ Rover

L'implementazione di TI-Nspire™ Python si basa su MicroPython, un piccolo sottoinsieme della libreria standard Python 3 progettato per funzionare sui microcontrollori. L'implementazione originale di MicroPython è stata adattata per l'uso da parte di TI.

Nota: Alcune risposte numeriche possono variare dai risultati della Calcolatrice a causa delle differenze nelle implementazioni matematiche sottostanti.

Python è disponibile sui seguenti prodotti TI-Nspire™:

Palmari	Software del desktop
TI-Nspire™ CX II	TI-Nspire™ CX Premium Teacher Software
TI-Nspire™ CX II CAS	TI-Nspire™ CX CAS Premium Teacher Software
TI-Nspire™ CX II-T	TI-Nspire™ CX Student Software
TI-Nspire™ CX II-T CAS	TI-Nspire™ CX CAS Student Software
TI-Nspire™ CX II-C	
TI-Nspire™ CX II-C CAS	

Nota: Nella maggior parte dei casi, la funzionalità è identica tra le viste del palmare e del software, tuttavia è possibile riscontrare alcune differenze. Questa guida presuppone che si stia utilizzando il dispositivo palmare o la vista palmare del software.

Moduli Python

TI-Nspire™ Python include i seguenti moduli:

Moduli standard	Moduli TI
Matematica (math)	TI PlotLib (ti_plotlib)
Casuale (random)	TI Hub (ti_hub)
Matematica complessa (cmath)	TI Rover (ti_rover)
Tempo (time)	TI System (ti_system)
	TI Draw (ti_draw)
	TI Image (ti_image)

Nota: Se sono stati già creati programmi Python in altri ambienti di sviluppo Python, potrebbe essere necessario modificarli per eseguire la soluzione TI-Nspire™ Python. I moduli possono utilizzare metodi, argomenti e ordini di metodi diversi in un

programma rispetto ai moduli TI. In generale, prestare attenzione alla compatibilità quando si utilizzano versioni di Python e i moduli Python.

Quando si trasferiscono i programmi Python da una piattaforma non TI a una piattaforma TI OPPURE da un prodotto TI a un altro, ricordare che:

- I programmi che utilizzano funzionalità di lingua fondamentali e librerie standard (matematica, casuale, ecc.) possono essere trasferiti senza modifiche.
- I programmi che utilizzano librerie specifiche della piattaforma come matplotlib per moduli PC o TI richiederanno modifiche prima di poter essere eseguiti su una piattaforma diversa. Ciò può essere vero anche tra le piattaforme TI.

Come per qualsiasi versione di Python, sarà necessario includere le importazioni per utilizzare le funzioni, i metodi o le costanti contenuti in un determinato modulo. Ad esempio, per eseguire la funzione cos() dal modulo di matematica, utilizzare i seguenti comandi:

```
>>>from math import *
>>>cos(0)
1.0
```

Per un elenco dei menu con le relative voci e descrizioni, consultare la sezione Mappa dei menu.

Installazione di un programma Python come modulo

Per salvare il programma Python come modulo:

- Nell'Editor, selezionare **Azioni > Installa come Modulo Python**.
- Nella Shell, selezionare **Strumenti > Installa come Modulo Python**.

Dopo la selezione, si verifica quanto segue:

- è controllata la sintassi Python.
- Il file è salvato e spostato nella cartella PyLib.
- Compare una finestra di dialogo che conferma l'installazione del file come modulo.
- Il file viene chiuso e il modulo è pronto per l'uso.
- Il nome del modulo verrà aggiunto al menu **Altri moduli** con una voce di menu **da <module> importazione ***.

Se si prevede di condividere questo modulo con altri, è consigliabile seguire queste linee guida:

- Archiviare un solo modulo per ogni file TNS.
- Il nome del modulo corrisponde al nome del file TNS (ad es., il modulo "my_ program" si trova nel file "my_program.tns").
- Aggiungere una pagina Note prima dell'editor Python che descriva l'intento del modulo, la versione e le funzioni.
- Utilizzare la funzione ver() per visualizzare il numero di versione del modulo.

- (Facoltativo) Aggiungere una funzione di guida per visualizzare l'elenco dei metodi nella funzione.

Spazi di lavoro di Python

Esistono due aree di lavoro per la programmazione di Python: L'Editor di Python e la Shell di Python.

Editor di Python	Shell di Python
• Creare, modificare e salvare i programmi Python • Evidenziazione e rientro automatico della sintassi • Prompt in linea per guidare gli argomenti delle funzioni • Suggerimenti per la visualizzazione dell'intervallo di valori validi • Il tasto var elenca le variabili e le funzioni utente globali definite nel programma corrente • Scelte rapide da tastiera	• Esegui programmi Python • Comodo per testare piccoli frammenti di codice • Interazione con la cronologia della Shell per selezionare input e output precedenti per il riutilizzo • Il tasto var elenca le variabili utente globali definite nell'ultimo programma eseguito nel problema specificato

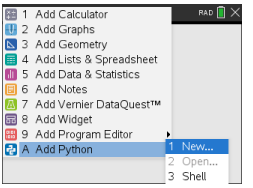
Nota: È possibile aggiungere più programmi e Shell Python a un problema.

Editor di Python

L'Editor di Python consente di creare, modificare e salvare i programmi di Python.

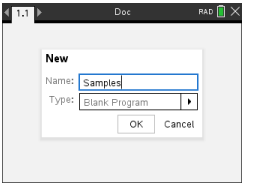
Aggiunta di una pagina dell'Editor di Python

Per aggiungere una nuova pagina dell'Editor di Python nel problema corrente, premere menu e selezionare **Aggiungi Python > Nuovo**.

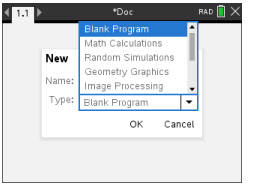


È possibile creare un programma vuoto o selezionare un modello.

Programma vuoto



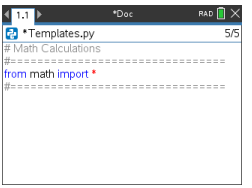
Modello



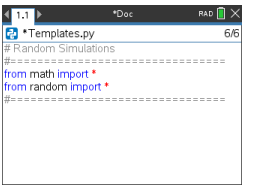
Dopo aver creato il programma, viene visualizzato l'Editor di Python. Se è stato selezionato un modello, le istruzioni di importazione necessarie vengono aggiunte automaticamente (vedere di seguito).

Nota: è possibile avere più programmi in un unico file TNS proprio come con le altre app. Se il programma Python è destinato ad essere utilizzato come modulo, il file TNS può essere salvato nella cartella PyLib. Questo modulo può essere quindi utilizzato in altri programmi e documenti.

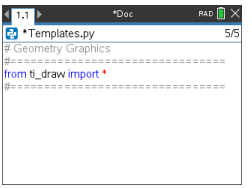
Calcoli matematici



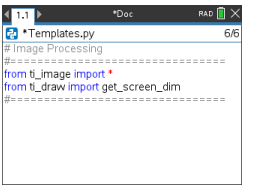
Simulazioni casuali



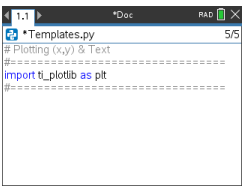
Grafici Geometria



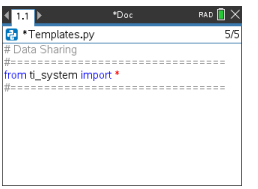
Elaborazione delle immagini



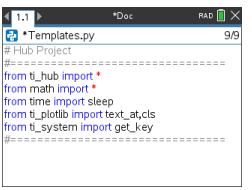
Tracciamento (x,y) e Testo



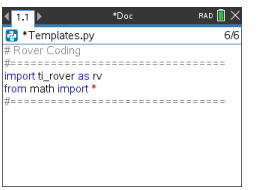
Condivisione dei dati



Progetto TI-Innovator Hub



Codifica di TI-Rover



Apertura di un programma Python

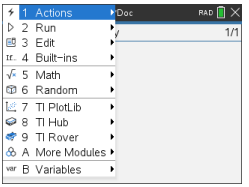
Per aprire un programma Python esistente, premere doc e selezionare **Inserisci > Aggiungi Python > Apri**. Verrà visualizzato un elenco di programmi salvati nel file TNS.

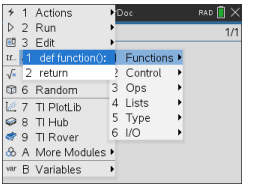
Se la pagina dell'Editor utilizzata per creare il programma è stata eliminata, il programma è ancora disponibile nel file TNS.

Lavorare nell'Editor Python

Premendo menu viene visualizzato il menu Strumenti documento. Con queste opzioni di menu è possibile aggiungere, spostare e copiare blocchi di codice per il programma.

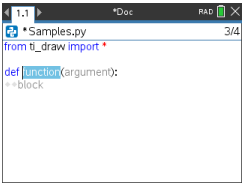
Menu Strumenti del documento



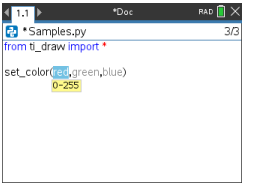


Gli elementi selezionati dai menu del modulo aggiungeranno automaticamente un modello di codice all'Editor con i prompt in linea per ogni parte della funzione. È possibile spostarsi da un argomento a quello successivo premendo tab (avanti) o shift+tab (indietro). I suggerimenti o gli elenchi pop-up appariranno quando saranno disponibili per permettere di selezionare i valori corretti.

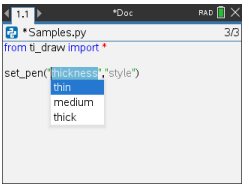
Prompt in linea



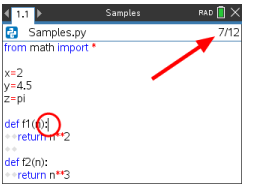
Suggerimenti



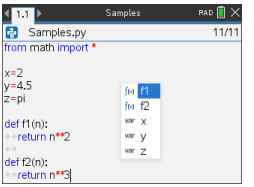
Elenchi pop-up



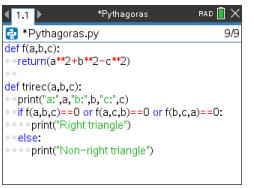
I numeri a destra del nome del programma riflettono il numero di riga corrente del cursore e il numero totale di righe del programma.



Le funzioni e le variabili globali definite nelle righe al di sopra della posizione corrente del cursore possono essere inserite premendo [var] e selezionando l'elenco.



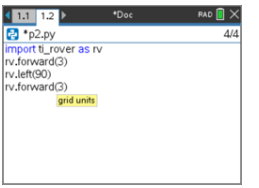
Man mano che si aggiunge il codice al programma, l'Editor visualizza parole chiave, operatori, commenti, stringhe e rientri in diversi colori per facilitare l'identificazione dei diversi elementi.



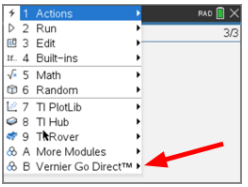
Le voci di menu incollate vengono posizionate sulla riga corrente o su quella successiva in base al contesto.

Prima della versione 6.2.0

Versione 6.2.0 e successive

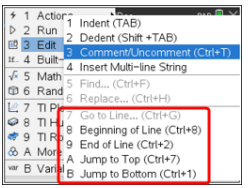


Il menu principale visualizza il menu del modulo in base al programma che si sta modificando, consentendo un accesso più rapido alle opzioni. Viene visualizzato solo un modulo alla volta.

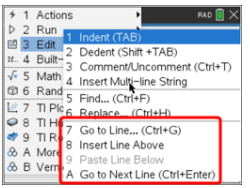


Il menu Modifica visualizza le voci di menu e i tasti di scelta rapida di uso più frequente.

Prima della versione 6.2.0



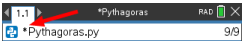
Versione 6.2.0 e successive



Salvataggio ed esecuzione dei programmi

Al termine del programma, premere menu e selezionare **Esegui > Verifica sintassi & Salva**. Ciò consentirà di controllare la sintassi del programma Python e di salvarla nel file TNS.

Nota: se sono presenti modifiche non salvate nel programma, verrà visualizzato un asterisco accanto al nome del programma.



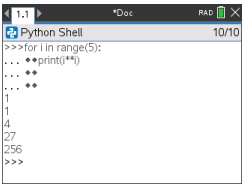
Per eseguire il programma, premere menu e selezionare **Esegui > Esegui**. Verrà eseguito il programma corrente nella successiva pagina della Shell di Python o un nuovo programma se la pagina successiva non è una Shell.

Nota: con l'esecuzione del programma viene controllata automaticamente la sintassi e viene salvato il programma.

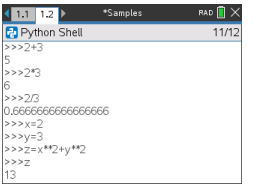
Shell di Python

La Shell di Python è l'interprete che esegue i programmi Python, altri pezzi di codice Python o semplici comandi.

Codice di Python

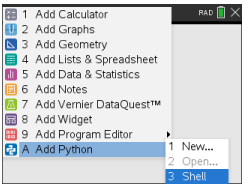


Comandi semplici

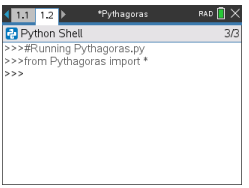


Aggiunta di una pagina della Shell di Python

Per aggiungere una nuova pagina della Shell di Python nel problema corrente, premere menu e selezionare **Aggiungi Python > Shell**.



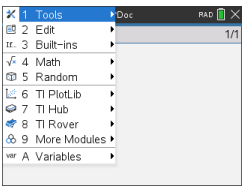
La Shell di Python può anche essere avviata dall'Editor di Python eseguendo un programma premendo menu e selezionando **Esegui > Esegui**.

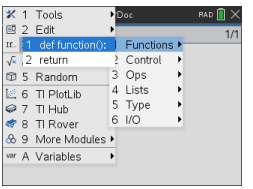


Lavorare nella Shell di Python

Premendo menu viene visualizzato il menu Strumenti documento. Con queste opzioni di menu è possibile aggiungere, spostare e copiare blocchi di codice.

Menu Strumenti del documento

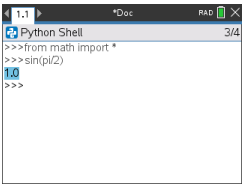




Nota: Se si utilizza un metodo da uno dei moduli disponibili, assicurarsi di eseguire prima un'istruzione del modulo di importazione, come in qualsiasi ambiente di codifica Python.

L'interazione con l'output della Shell è simile all'app Calcolatrice dove è possibile selezionare e copiare gli input e gli output precedenti per l'utilizzo in altre aree della Shell, dell'Editor o in altre app.

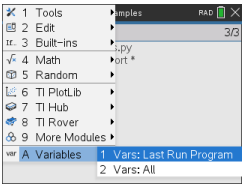
Premere la freccia verso l'alto per selezionare, quindi copiare e incollare nella posizione desiderata



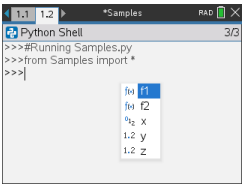
Le funzioni e le variabili globali dell'ultimo programma eseguito possono essere inserite premendo var o ctrl+L ed effettuando una selezione dall'elenco oppure premere menu e selezionare **Variabili > Var: Ultimo programma eseguito**.

Per effettuare una scelta da un elenco di funzioni e variabili globali sia dall'ultimo programma eseguito che da qualsiasi modulo importato, premere menu e selezionare **Variabili > Var: Tutti**.

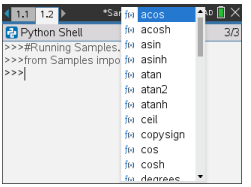
Menu Variabili



Variabili dell'ultimo programma eseguito

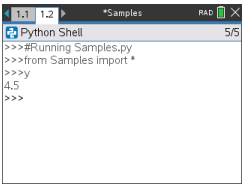


Tutte le variabili

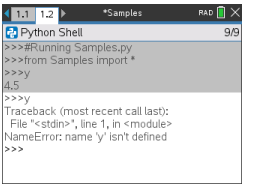


Tutte le pagine della Shell di Python nello stesso problema condividono lo stesso stato (definizioni variabili definite dall'utente e importate). Quando si salva o si esegue un programma Python in quel problema, o si preme menu e si seleziona **Strumenti > Reinizializza shell**, la cronologia della Shell mostrerà uno sfondo grigio indicante che lo stato precedente non è più valido.

Prima di salvare o reinizializzare



Dopo il salvataggio o la reinizializzazione



Nota: L'opzione menu **Strumenti > Cancella cronologia** consente di cancellare la schermata di qualsiasi attività precedente nella Shell, tuttavia le variabili sono ancora disponibili.

Messaggi

Durante una sessione Python possono essere visualizzati errori e altri messaggi informativi. Se viene visualizzato un errore nella Shell quando viene eseguito un programma, verrà visualizzato il numero di riga del programma. Premere ctrl menu e selezionare **Vai all'Editor di Python**. Nell'Editor, premere menu quindi selezionare **Modifica > Vai alla riga**. Inserire il numero di riga e premere enter. Il cursore verrà visualizzato sul primo carattere della riga in cui si è verificato l'errore.

Interruzione di un programma in esecuzione

Durante l'esecuzione di una funzione o di un programma, il puntatore visualizzato è ⧖ "occupato".

▶ Per arrestare la funzione o il programma:

- Windows®: Premere il tasto **F12**.
- Mac®: Premere il tasto **F5**.
- Palmare: Premere il tasto on.

Mappa dei menu di Python

Questa sezione elenca tutti i menu e gli elementi dei menu per l'Editor o la Shell di Python e una breve descrizione per ciascuno di essi.

Nota: Per gli elementi di menu che dispongono di tasti di scelta rapida, gli utenti Mac® devono sostituire ⌘ (**Cmd**) ogni volta che viene utilizzato **Ctrl**. Per un elenco completo dei tasti di scelta rapida del software e del palmare TI-Nspire™, vedere la eGuide per la tecnologia TI-Nspire™.

Menu Azioni

Nota: Si applica solo all'Editor.

Voce	Descrizione
Nuovo	Apre la finestra di dialogo **Nuovo** in cui inserire un nome e selezionare un tipo per il nuovo programma.
Apri (Open)	Apre un elenco di programmi disponibili nel documento corrente.
Crea copia (Create Copy)	Apre la finestra di dialogo **Crea copia** in cui è possibile

Voce	Descrizione
	salvare il programma corrente con un altro nome.
Rename; Rinomina	Apre la finestra di dialogo **Rinomina** in cui è possibile rinominare il programma corrente.
Close (Chiudi)	Chiude il programma corrente.
Impostazioni	Apre la finestra di dialogo **Impostazioni** in cui è possibile modificare la dimensione carattere sia per l'Editor che per la Shell.
Installa come modulo Python	Controlla la sintassi Python del file TNS corrente e lo sposta nella cartella PyLib.

Menu Esegui

Nota: Si applica solo all'Editor.

Voce	tasti di scelta rapida	Descrizione
Esegui	Ctrl+R	Controlla la sintassi, salva il programma e lo esegue in una shell Python.
Controlla sintassi e salva	Ctrl+B	Controlla la sintassi e salva il programma.
Vai alla shell	N/A	Passa alla Shell relativa al programma corrente o apre una nuova pagina Shell accanto all'Editor.

Menu Strumenti

Nota: Si applica solo alla Shell.

Voce	tasti di scelta rapida	Descrizione
Riesegui ultimo programma	Ctrl+R	Riesegue l'ultimo programma relativo alla Shell corrente.
Vai all'Editor di Phyton	N/A	Apre la pagina Editor relativa alla Shell corrente.
Esegui	N/A	Apre un elenco di programmi disponibili nel documento corrente. Dopo la selezione, viene eseguito il programma scelto.
Cancella Cronologia	N/A	Cancella la cronologia nella Shell corrente, ma non reinizializza la

Voce	tasti di scelta rapida	Descrizione
		Shell.
Reinizializza shell	N/A	Ripristina lo stato di tutte le pagine Shell aperte nel problema corrente. Tutte le variabili definite e le funzioni importate non sono più disponibili.
dir()	N/A	Visualizza l'elenco delle funzioni nel modulo specificato quando viene utilizzato dopo l'istruzione di importazione.
Da importazione PROGRAMMA *	N/A	Apre un elenco di programmi disponibili nel documento corrente. Dopo la selezione, l'istruzione di importazione viene incollata nella Shell.
Installare come modulo Python	N/A	Abilitato solo per i moduli in formato binario. Sposta il file TNS corrente nella cartella PyLib.

Menu Modifica

Nota: Ctrl+A seleziona tutte le righe di codice o l'output per tagliare o eliminare (solo Editor) oppure copiare e incollare (Editor e Shell).

Voce	tasti di scelta rapida	Descrizione
Rientro	TAB*	Esegue il rientro del testo sulla riga corrente o sulle righe selezionate. * Se sono presenti dei prompt in linea incompleti, TAB consentirà di passare al prompt successivo.
Elimina rientro	Maiusc+TAB**	Elimina il rientro del testo sulla riga corrente o sulle righe selezionate. ** Se sono presenti dei prompt in linea incompleti, Maiusc+TAB consentirà di passare al prompt precedente.

Voce	tasti di scelta rapida	Descrizione
Commento/Rimuovi commento	Ctrl+T	Aggiunge/rimuove il simbolo di commento a/dall'inizio della riga corrente.
Inserisci stringa a più righe	N/A	(Solo Editor) Inserisce un modello di stringa a più righe.
Trovare	Ctrl+F	(Solo Editor) Apre la finestra di dialogo **Trova** e ricerca la stringa inserita nel programma corrente.
Sostituire	Ctrl+H	(Solo Editor) Apre la finestra di dialogo **Sostituisci** e ricerca la stringa inserita nel programma corrente.
Go to Line (Vai alla riga)	Ctrl+G	(Solo Editor) Apre la finestra di dialogo **Vai alla riga** e passa alla riga specificata nel programma corrente.
Inizio della riga	Ctrl+8	Sposta il cursore all'inizio della riga corrente.
Fine della riga	Ctrl+2	Sposta il cursore alla fine della riga corrente.
Vai all'inizio	Ctrl+7	Sposta il cursore all'inizio della prima riga del programma.
Vai alla fine	Ctrl+1	Sposta il cursore alla fine dell'ultima riga del programma.
Cancella una riga	Ctrl+Canc	Cancella la riga nel punto in cui si trova il cursore.

Menu integrati

funzioni

Voce	Descrizione
def function():	Definisce una funzione dipendente sulle variabili specificate.
return	Definisce il valore prodotto da una funzione.

Strutture di controllo (Control)

Voce	Descrizione
if..	Istruzione condizionale.
if..else..	Istruzione condizionale.
if..elif..else..	Istruzione condizionale.
for index in range(size):	Esegue l'iterazione su un intervallo.
for index in range(start,stop):	Esegue l'iterazione su un intervallo.
for index in range(start,stop,step):	Esegue l'iterazione su un intervallo.
for index in list:	Esegue l'iterazione degli elementi dell'elenco.
while..	Esegue le istruzioni in un blocco di codice fino a quando una condizione non restituisce False.
elif:	Istruzione condizionale.
else:	Istruzione condizionale.

Ops

Voce	Descrizione
x=y	Imposta il valore della variabile.
x==y	Incolla l'operatore di confronto uguale a (==).
x!=y	Incolla l'operatore di confronto diverso da (!=).
x>y	Incolla l'operatore di confronto maggiore di (>).
x>=y	Incolla l'operatore di confronto maggiore o uguale a (>=).
x<y	Incolla l'operatore di confronto minore di (<).
x<=y	Incolla l'operatore di confronto minore o uguale a (<=).

Voce	Descrizione
e	Incolla l'operatore logico and (e).
—Oppure—	Incolla l'operatore logico or (o).
not	Incolla l'operatore logico not (non).
Vero	Incolla il valore booleano True.
Falso	Incolla il valore booleano False.

Liste

Voce	Descrizione
[]	Incolla le parentesi ([]).
list() (Differenza in una lista)	Converte la sequenza nel tipo "lista".
len()	Restituisce il numero di elementi della lista.
max() (Massimo)	Restituisce il valore massimo nella lista.
min() (Minimo)	Restituisce il valore minimo nella lista.
.append()	Il metodo aggiunge un elemento a una lista.
.remove()	Il metodo rimuove la prima istanza di un elemento da una lista.
range(start,stop,step)	Restituisce un insieme di numeri.
for index in range(start,stop,step)	Utilizzato per iterare un intervallo.
.insert()	Il metodo aggiunge un elemento alla posizione specificata.
.split()	Il metodo restituisce una lista con elementi separati da un delimitatore specificato.
sum() (Somma)	Restituisce la somma degli elementi di una lista.
sorted()	Restituisce una lista ordinata.
.sort()	Il metodo ordina una lista esistente.

Tipo

Voce	Descrizione
int()	Restituisce una parte intera.
float()	Restituisce un valore mobile.

Voce	Descrizione
round(x,ndigits)	Restituisce un numero a virgola mobile arrotondato al numero di cifre specificato.
str()	Restituisce una stringa.
complex()	Restituisce un numero complesso.
type()	Restituisce il tipo di oggetto.

I/O

Voce	Descrizione
print()	Visualizza l'argomento come stringa.
input()	Chiede l'immissione da parte dell'utente.
eval()	Valuta un'espressione rappresentata come stringa.
.format()	Il metodo formatta la stringa specificata.

Menu Matematica

Nota: Quando si crea un nuovo programma che utilizza questo modulo, si consiglia di utilizzare il tipo di programma **Calcoli matematici**. In questo modo verranno importati tutti i moduli pertinenti.

Voce	Descrizione
from math import *	Importa tutti i metodi (funzioni) dal modulo matematico.
fabs()	Restituisce il valore assoluto di un numero reale.
sqrt() (Radice quadrata)	Restituisce la radice quadrata di un numero reale.
exp() (e alla potenza)	Restituisce e**x.
pow(x,y)	Restituisce x elevato alla potenza di y.
log(x, base)	Restituisce il $\log_{base}(x)$. log(x) senza base restituisce il logaritmo naturale x.
fmod(x,y)	Restituisce il valore del modulo di x e y. Utilizzare quando x e y sono mobili.
ceil()	Restituisce il numero intero più piccolo maggiore o uguale a un numero reale.
floor() (Arrotondato per difetto)	Restituisce il numero intero più grande minore o uguale a un numero reale.
trunc()	Tronca un numero reale a un intero.

Voce	Descrizione
frexp()	Restituisce una coppia (y,n) dove x == y * 2**n.

Costante

Voce	Descrizione
e	Returns value for the constant e.
pi	Returns value for the constant pi.

Trigonometria

Voce	Descrizione
radians()	Converte l'angolo in gradi a radianti.
degrees()	Converte l'angolo in radianti in gradi.
sin() (Seno)	Restituisce il seno dell'argomento in radianti.
cos() (Coseno)	Restituisce il coseno dell'argomento in radianti.
tan() (Tangente)	Restituisce la tangente dell'argomento in radianti.
asin()	Restituisce l'arcoseno dell'argomento in radianti.
acos()	Restituisce l'arcocoseno dell'argomento in radianti.
atan()	Restituisce l'arcotangente dell'argomento in radianti.
atan2(y,x)	Restituisce l'arcotangente di y/x in radianti.

Menu Casuale

Nota: Quando si crea un nuovo programma che utilizza questo modulo, si consiglia di utilizzare il tipo di programma **Simulazioni casuali**. In questo modo verranno importati tutti i moduli pertinenti.

Voce	Descrizione
from random import *	Importa tutti i metodi dal modulo casuale.
random()	Restituisce un numero a virgola mobile da 0 a 1,0.
uniform(min,max)	Restituisce un numero casuale x (mobile) in modo tale che min <= x <= max.
randint(min,max)	Restituisce un numero intero casuale tra min e max.

Voce	Descrizione
choice(sequence)	Restituisce un elemento casuale da una sequenza non vuota.
randrange(start,stop,step)	Restituisce un numero casuale dall'inizio alla fine per passo.
seed()	Inizializza il generatore di numero casuali.

Menu TI PlotLib

Nota: Quando si crea un nuovo programma che utilizza questo modulo, si consiglia di utilizzare il tipo di programma **Tracciamento (x,y) e Testo**. In questo modo verranno importati tutti i moduli pertinenti.

Voce	Descrizione
import ti_plotlib as plt	Importa tutti i metodi (funzioni) dal modulo ti_plotlib nello spazio dei nomi "plt". Di conseguenza, tutti i nomi di funzione incollati dai menu saranno preceduti da "plt".

Imp

Voce	Descrizione
cls()	Cancella l'area di tracciamento.
grid(x-scale,y-scale,"style")	Visualizza una griglia utilizzando la scala specificata per gli assi x e y.
window(xmin,xmax,ymin,ymax)	Definisce la finestra di tracciamento mappando l'intervallo orizzontale specificato (xmin, xmax) e l'intervallo verticale (ymin, ymax) nell'area di plottaggio assegnata (pixel).
auto_window(x-list,y-list)	Ridimensiona automaticamente la finestra di tracciamento per adattare gli intervalli di dati all'interno di x-list e y-list specificati nel programma prima di auto_window().
axes("mode")	Visualizza gli assi nella finestra specificata nell'area di tracciamento.
labels("x-label","y-label",x,y)	Visualizza le etichette "x-label" e "y-label" sugli assi del tracciato nelle posizioni x e y della riga.
title("title")	Visualizza il "titolo" (title) centrato sulla riga superiore della finestra.
show_plot()	Visualizza il disegno in buffer. Le funzioni use_buffer() e show_plot() sono utili

Voce	Descrizione
	nei casi in cui la visualizzazione di più oggetti sullo schermo potrebbe causare ritardi (non necessarie nella maggior parte dei casi).
use_buffer()	Attiva un buffer fuori schermo per accelerare il disegno.

Punteggiatura

Voce	Descrizione
color(red,green,blue)	Imposta il colore per tutti i grafici/tracciati seguenti.
cls()	Cancella l'area di tracciamento.
show_plot()	Esegue la visualizzazione del tracciato come impostato nel programma.
scatter(x-list,y-list,"mark")	Traccia una sequenza di coppie ordinate da (x-list,y-list) con lo stile di contrassegno specificato.
plot(x-list,y-list,"mark")	Traccia una linea utilizzando coppie ordinate da x-list e y-list specificati.
plot(x,y,"mark")	Traccia un punto utilizzando le coordinate x e y con lo stile di contrassegno specificato.
line(x1,y1,x2,y2,"mode")	Traccia un segmento di linea da (x1,y1) a (x2,y2).
lin_reg(x-list,y-list,"display")	Calcola e disegna il modello di regressione lineare, ax+b, di x-list,y-list.
pen("size","style")	Imposta l'aspetto di tutte le seguenti linee fino alla successiva esecuzione di pen().
text_at(row,"text","align")	Visualizza il "testo" (text) nell'area di tracciamento con l'"allineamento" (align) specificato.

Proprietà

Voce	Descrizione
xmin	Variabile specificata per gli argomenti della finestra definiti come plt.xmin.
xmax	Variabile specificata per gli argomenti della finestra definiti come plt.xmax.
ymin	Variabile specificata per gli argomenti della finestra definiti come plt.ymin.
ymax	Variabile specificata per gli argomenti della finestra definiti come plt.ymax.

Voce	Descrizione
m	Dopo aver eseguito plt.linreg() in un programma, i valori calcolati di pendenza, m e intercetta, b, sono memorizzati in plt.m e plt.b.
b	Dopo aver eseguito plt.linreg() in un programma, i valori calcolati di pendenza, a e intercetta, b, sono memorizzati in plt.m e plt.b.

Menu TI Hub

Nota: Quando si crea un nuovo programma che utilizza questo modulo, si consiglia di utilizzare il tipo di programma **Progetto Hub**. In questo modo verranno importati tutti i moduli pertinenti.

Voce	Descrizione
from ti_hub import *	Importa tutti i metodi dal modulo ti_hub.

Dispositivi integrati Hub > Output a colori

Voce	Descrizione
rgb(red,green,blue)	Imposta il colore del LED RGB.
blink(frequency,time)	Imposta la frequenza e la durata del lampeggiamento per il colore selezionato.
off()	Spegne il LED RGB.

Dispositivi integrati Hub > Output luce

Voce	Descrizione
on()	Accende il LED.
off()	Spegne il LED.
blink(frequency,time)	Imposta la frequenza e la durata del lampeggiamento per il LED.

Dispositivi integrati Hub > Output audio

Voce	Descrizione
tone(frequency,time)	Riproduce un tono della frequenza specificata per il tempo specificato.

Voce	Descrizione
note("note",time)	Riproduce la nota specificata per il tempo specificato. La nota è specificata utilizzando il nome della nota e un'ottava. Ad esempio: A4, C5. I nomi delle note sono C, CS, D, DS, E, F, FS, G, GS, A, AS e B. I numeri dell'ottava variano da 1 a 9 (compresi).
tone(frequency,time,tempo)	Riproduce un tono della frequenza specificata per il tempo e il ritmo specificati. Il ritmo definisce il numero di bip al secondo che vanno da 0 a 10 (compresi).
note("note",time,tempo)	Riproduce la nota specificata per il tempo e il ritmo specificati. La nota è specificata utilizzando il nome della nota e un'ottava. Ad esempio: A4, C5. I nomi delle note sono C, CS, D, DS, E, F, FS, G, GS, A, AS e B. I numeri dell'ottava variano da 1 a 9 (compresi). I numeri del ritmo variano da 0 a 10 (compresi).

Dispositivi integrati Hub > Ingresso luminosità

Voce	Descrizione
measurement()	Legge il sensore LUMINOSITÀ (livello di luce) integrato e restituisce una lettura. L'intervallo predefinito è compreso tra 0 e 100. Ciò può essere modificato utilizzando la funzione range().
range(min,max)	Imposta l'intervallo per mappare le letture dal sensore del livello di luce. Se entrambi mancano o sono impostati sul valore Nessuno, viene impostato l'intervallo di luminosità predefinito da 0 a 100.

Aggiungi dispositivo di input

Questo menu contiene un elenco dei sensori (dispositivi di input) supportati dal modulo ti_hub. Tutti gli elementi del menu incollano il nome dell'oggetto e prevedono l'utilizzo di una variabile e una porta con il sensore. Ciascun sensore ha un metodo measurement() che restituisce il valore del sensore.

Voce	Descrizione
DHT (Umidità e temperatura digitali)	Restituisce un elenco composto dai valori correnti di temperatura, umidità, tipo di sensore e ultimo stato di lettura in cache.
Ranger	Restituisce la misurazione della distanza corrente dal ranger ultrasonico specificato. • **measurement_time()** - Restituisce il tempo impiegato dal segnale ultrasonico per raggiungere l'oggetto (il "tempo di volo").
Livello luce	Restituisce il livello di luminosità dal sensore di livello di luce esterno (luminosità).
temperatura	Restituisce la lettura della temperatura dal sensore di temperatura esterno. La configurazione predefinita supporta il sensore di temperatura Seeed nelle porte IN 1, IN 2 o IN 3. Per utilizzare il sensore di temperatura TI LM19 dal pacchetto basetta di TI-Innovator™ Hub, modificare la porta sul pin BB in uso e utilizzare un argomento opzionale "TIANALOG". Esempio: mylm19=temperature("BB 5","TIANALOG")
Umidità	Restituisce la lettura del sensore di umidità.
Magnetico	Rileva la presenza di un campo magnetico. Il valore soglia per determinare la presenza del campo viene impostato mediante la funzione trigger(). Il valore predefinito della soglia è 150.
App Vernier	Legge il valore dal sensore analogico Vernier specificato nel comando. Il comando supporta i seguenti sensori Vernier: • **temperatura** - Sensore di temperatura in acciaio inossidabile. • **livello di luce** - Sensore del livello di luce TI. • **pressione** - Sensore di pressione gas originale • **pressione** - Sensore di pressione gas più recente.

Voce	Descrizione
	• **pH** - Sensore di pH. • **forza10** - Impostazione ±10 N, sensore a doppia forza. • **forza50** - Impostazione ±50 N, sensore a doppia forza. • **accelerometro** - Accelerometro Low-G. • **generico** - Consente l'impostazione di altri sensori non supportati direttamente sopra e l'uso dell'API calibrate() sopra per impostare coefficienti dell'equazione.
Ingresso analogico	Supporta l'uso di dispositivi generici a ingresso analogico.
Input digitale	Restituisce lo stato corrente del pin digitale collegato all'oggetto DIGITALE, oppure lo stato in cache dell'ultimo valore di uscita digitale impostato sull'oggetto.
Potenziometro	Supporta un sensore di potenziometro. L'intervallo del sensore può essere modificato dalla funzione range().
Termistore	Legge i sensori del termistore. I coefficienti predefiniti sono progettati in modo da corrispondere al termistore incluso nel pacchetto basetta di TI-Innovator™ Hub, in caso di utilizzo con un resistore fisso da 10 KΩ. È possibile configurare un nuovo insieme di coefficienti di calibrazione e resistenza di riferimento per il termistore utilizzando la funzione calibrate().
Sonorità	Supporta i sensori di intensità del suono.
Ingresso colori	Fornisce le interfacce per un sensore di ingresso colori con collegamento I2C. Il pin bb_port viene utilizzato in aggiunta alla porta I2C per controllare il LED sul sensore dei colori. • **color_number():** Restituisce un valore da 1 a 9 che rappresenta il colore rilevato dal sensore. I numeri rappresentano i colori per la seguente mappatura:

Voce	Descrizione
	1: Rosso 2: Verde 3: Blu 4: Ciano 5: Magenta 6: Giallo 7: Nero 8: Bianco 9: Grigio • **red():** Restituisce un valore da 0 a 255 che rappresenta l'intensità del livello di colore ROSSO rilevato. • **green():** Restituisce un valore da 0 a 255 che rappresenta l'intensità del livello di colore VERDE rilevato. • **blue():** Restituisce un valore da 0 a 255 che rappresenta l'intensità del livello di colore BLU rilevato. • **gray():** Restituisce un valore da 0 a 255 che rappresenta il livello di grigio rilevato, dove 0 è nero e 255 è bianco.
Porta BB	Fornisce supporto per l'utilizzo di tutti i pin della porta 10 BB come porta di ingresso/uscita digitale combinata. Le funzioni di inizializzazione presentano un parametro opzionale "mask" che consente l'uso del sottoinsieme dei 10 pin. • **read_port():** Legge i valori correnti sui pin di ingresso della porta BB. • **write_port(value):** Imposta i valori dei pin di uscita sul valore specificato, dove il valore è compreso tra 0 e 1023. Tenere presente che il valore viene regolato anche rispetto al valore della maschera nell'operazione var=bbport(mask), se è stata fornita una maschera.
Tempo Hub	Consente l'accesso al timer in millisecondi interno.

Voce	Descrizione
TI-RGB Array	Fornisce le funzioni per la programmazione di TI-RGB Array. La funzione di inizializzazione accetta un parametro "LAMP" opzionale per attivare una modalità ad alta luminosità per TI-RGB Array che richiede un alimentatore esterno. • **set(led_position, r,g,b):** Imposta una specifica posizione led_position (0-15) sul valore r,g,b specificato, dove r,g,b sono valori da 0 a 255. • **set(led_list,red,green,blue):** Imposta i LED definiti in “led_list” sul colore specificato per “rosso”, “verde”, “blu”. La “led_list” è un elenco Python che include indici dei LED da 0 a 15. Ad esempio, il set ([0,2,4,6,15], 0, 0, 255) imposterà i LED 0, 2, 4, 6 e 15 su blu. • **set_all(r,g,b):** Imposta tutti i LED RGB nell’array sullo stesso valore r,g,b. • **all_off():** Disattiva tutti valori RGB nell’array. • **measurement():** Restituisce il disegno corrente approssimativo che l’array RGB utilizza da TI-Innovator™ in milliampere. • **pattern(pattern):** Utilizzando il valore dell’argomento come valore binario nell’intervallo compreso tra 0 e 65535, attiva i pixel dove sarebbe stato generato un valore 1 nella rappresentazione. I LED vengono attivati in ROSSO con il valore del livello pwm di 255. • **pattern(value,red,green,blue):** Imposta i LED definiti dal “modello” sul colore specificato per “rosso”, “verde”, “blu”.

Aggiungi dispositivo di output

Questo menu contiene un elenco dei dispositivi di output supportati dal modulo ti_hub. Tutti gli elementi del menu incolleranno il nome dell’oggetto e prevedono l’utilizzo di una variabile e una porta con il dispositivo.

Voce	Descrizione
LED	Funzioni per il controllo dei LED collegati esternamente.

Voce	Descrizione
RGB	Supporto per il controllo dei LED RGB esterni.
TI-RGB Array	Fornisce le funzioni per la programmazione di TI-RGB Array.
Altoparlante	Funzioni per supportare un altoparlante esterno con TI-Innovator™ Hub. Le funzioni sono equivalenti a quelle per il “suono” sopra riportate.
Potenza	Funzioni per il controllo della potenza esterna con TI-Innovator™ Hub. • **set(value):** Imposta il livello di potenza al valore specificato, compreso tra 0 e 100. • **on():** Imposta il livello di potenza su 100. • **off():** Imposta il livello di potenza su 0.
Servo continuo	Funzioni per il controllo dei servomotori continui. • **set_cw(speed,time):** Il servo girerà in senso orario alla velocità specificata (0-255) e per la durata in secondi specificata. • **set_ccw(speed,time):** Il servo girerà in senso antiorario alla velocità specificata (0-255) e per la durata in secondi specificata. • **stop():** Arresta il servo continuo.
Uscita analogica	Funzioni per l’uso di dispositivi generici a ingresso analogico.
Motore a vibrazione	Funzioni per il controllo dei motori a vibrazione. • **set(val):** Imposta l’intensità del motore a vibrazione su "val" (0-255). • **off():** Spegne il motore a vibrazione. • **on():** Accende il motore a vibrazione al livello più alto.
Relè	Controlla le interfacce per il controllo dei relè. • **on():** Imposta il relè sullo stato ON (ATTIVAZIONE). • **off():** Imposta il relè sullo stato ON (DISATTIVAZIONE).
Servo	Funzioni per il controllo dei servo motori. • **set_position(pos):** Imposta la posizione di estensione del servo entro un intervallo da -90 a +90. • **zero():** Imposta l’estensione del servo nella posizione zero.
Onda quadra	Funzioni per generare un’onda quadra. • **set(frequency,duty,time):** Imposta l’onda quadra di uscita con un ciclo di lavoro predefinito del 50% (se non è specificato) e una frequenza di uscita specificata da "frequence". La frequenza può essere compresa tra 1 e 500 Hz. Il ciclo di lavoro, se specificato, può essere compreso tra 0 e 100%. • **off():** Disattiva l’onda quadra.

Voce	Descrizione
Output digitale	Interfacce per il controllo di un'uscita digitale. • **set(val):** Imposta l'uscita digitale sul valore specificato da "val" (0 o 1). • **on():** Imposta lo stato dell'uscita digitale su alto (1). • **off():** Imposta lo stato dell'uscita digitale su basso (0).
Porta BB	Fornisce le funzioni per la programmazione di TI-RGB Array. Vedere i dettagli sopra.

Comandi

Voce	Descrizione
sleep(seconds)	Sospende l'esecuzione del programma per un numero di secondi specificato. Importato dal modulo 'tempo' (time).
text_at(row,"text","align")	Visualizza il "testo" (text) specificato nell'area di tracciamento con l'"allineamento" (align) specificato. Parte del modulo ti_plotlib.
cls()	Cancella la schermata della Shell per il tracciamento. Parte del modulo ti_plotlib.
while get_key() != "esc":	Esegue i comandi nel loop "while" finché non viene premuto il tasto "esc".
get_key()	Restituisce una stringa che rappresenta il tasto premuto. Il tasto '1' restituisce "1", 'esc' restituisce "esc" e così via. Quando viene richiamato senza alcun parametro - get_key() - restituisce immediatamente. Quando viene richiamato con un parametro - get_key (1) - attende finché non viene premuto un tasto. Parte del modulo ti_system.

Porte

Sono le porte di ingresso e uscita disponibili su TI-Innovator™ Hub.

Voce
OUT 1
OUT 2
OUT 3
IN 1
IN 2
IN 3
BB 1
BB 2
BB 3
BB 4
BB 5
BB 6
BB 7
BB 8
BB 9
BB 10
Porta I2C

Menu TI Rover

Nota: Quando si crea un nuovo programma che utilizza questo modulo, si consiglia di utilizzare il tipo di programma **Codifica Rover**. In questo modo verranno importati tutti i moduli pertinenti.

Voce	Descrizione
import ti_rover as rv	Importa tutti i metodi (funzioni) dal modulo ti_rover nello spazio dei nomi "rv". Di conseguenza, tutti i nomi di funzione incollati dai menu saranno preceduti da "rv".

Drive

Voce	Descrizione
forward(distance)	Sposta Rover in avanti per la distanza specificata nelle unità della griglia.
backward(distance)	Sposta Rover all'indietro per la distanza specificata nelle

Voce	Descrizione
	unità della griglia.
left(angle_degrees)	Gira Rover verso sinistra dell'angolo specificato in gradi.
right(angle_degrees)	Gira Rover verso destra dell'angolo specificato in gradi.
stop()	Arresta immediatamente qualsiasi movimento corrente.
stop_clear()	Arresta immediatamente qualsiasi movimento corrente e cancella tutti i comandi in sospeso.
resume()	Riprende l'elaborazione dei comandi.
stay(time)	Rover rimane in posizione per il tempo specificato in secondi (facoltativo). Se non viene specificato alcun intervallo di tempo, Rover rimane in posizione per 30 secondi.
to_xy(x,y)	Sposta Rover nella posizione iniziale (x,y) sulla griglia virtuale.
to_polar(r,theta_degrees)	Sposta Rover nella posizione in coordinate polari (r, theta) sulla griglia virtuale. L'angolo è specificato in gradi.
to_angle(angle,"unit")	Gira Rover dell'angolo specificato nella griglia virtuale. L'angolo è relativo a un angolo zero che punta verso l'asse x nella griglia virtuale.

Guida > Guida con opzioni

Voce	Descrizione
forward_time(time)	Sposta Rover in avanti per il tempo specificato.
backward_time(time)	Sposta Rover all'indietro per il tempo specificato.
forward(distance,"unit")	Sposta Rover in avanti alla velocità predefinita per la distanza specificata. La distanza può essere specificata in unità di griglia, metri o giri di ruota.
backward(distance,"unit")	Sposta Rover all'indietro alla velocità predefinita per la distanza specificata. La distanza può essere specificata in unità di griglia, metri o giri di ruota.
left(angle,"unit")	Gira Rover verso sinistra dell'angolo specificato.

Voce	Descrizione
	L’angolo può essere in gradi, radianti o gradi centesimali.
right(angle,"unit")	Gira Rover verso destra dell’angolo specificato. L’angolo può essere in gradi, radianti o gradi centesimali.
forward_time(time,speed,"rate")	Sposta Rover in avanti per il tempo specificato, alla velocità specificata. La velocità può essere specificata in unità di griglia, metri o giri di ruota.
backward_time(time,speed,"rate")	Sposta Rover all’indietro per il tempo specificato, alla velocità specificata. La velocità può essere specificata in unità di griglia, metri o giri di ruota.
forward(distance,"unit",speed,"rate")	Sposta Rover in avanti per la distanza specificata, alla velocità specificata. La distanza può essere specificata in unità di griglia, metri o giri di ruota. La velocità può essere specificata in unità di griglia, metri o giri di ruota.
backward(distance,"unit",speed,"rate")	Sposta Rover all’indietro per la distanza specificata, alla velocità specificata. La distanza può essere specificata in unità di griglia, metri o giri di ruota. La velocità può essere specificata in unità di griglia, metri o giri di ruota.

Input

Voce	Descrizione
ranger_measurement()	Legge il sensore di distanza a ultrasuoni sulla parte anteriore di Rover, restituendo la distanza corrente in metri.
color_measurement()	Restituisce un valore da 1 a 9, indicando il colore predominante "visto" dal sensore di input colore Rover. 1 = rosso 2 = verde 3 = blu 4 = ciano

Voce	Descrizione
	5 = magenta 6 = giallo 7 = nero 8 = grigio 9 = bianco
red_measurement()	Restituisce un valore compreso tra 0 e 255 che indica il livello di rosso percepito visualizzato dal sensore di input colore.
green_measurement()	Restituisce un valore compreso tra 0 e 255 che indica il livello di verde percepito visualizzato dal sensore di input colore.
blue_measurement()	Restituisce un valore compreso tra 0 e 255 che indica il livello di blu percepito visualizzato dal sensore di input colore.
gray_measurement()	Restituisce un valore compreso tra 0 e 255 che indica il livello di grigio percepito visualizzato dal sensore di input colore.
encoders_gyro_measurement()	Restituisce un elenco di valori che contengono i conteggi dell'encoder delle ruote sinistra e destra nonché l'orientamento corrente del giroscopio.
gyro_measurement()	Restituisce un valore che rappresenta la lettura corrente del giroscopio, compresa la deriva, in gradi.
ranger_time()	Restituisce il tempo impiegato dal segnale ultrasonico dal ranger TI-Rover per raggiungere l'oggetto (il "tempo di volo").

Output

Voce	Descrizione
colore_rgb(r,g,b)	Imposta il colore del LED RGB di Rover su specifici valori di rosso, verde e blu.
color_blink(frequency,time)	Imposta la frequenza e la durata del lampeggiamento per il colore selezionato.
color_off()	Spegne il LED RGB di Rover.
motor_left(speed,time)	Imposta la potenza del motore sinistro sul valore specificato per la durata specificata.

Voce	Descrizione
	La velocità rientra nell'intervallo da -255 a 255 con 0 che indica l'arresto. I valori della velocità positivi indicano la rotazione in senso antiorario, mentre i valori della velocità negativi indicano il senso orario. Il parametro temporale opzionale, se specificato, ha un intervallo valido da 0,05 a 655,35 secondi. Se non specificato, viene utilizzato un valore predefinito di 5 secondi.
motor_right(speed,time)	Imposta la potenza del motore sinistro sul valore specificato per la durata specificata. La velocità rientra nell'intervallo da -255 a 255 con 0 che indica l'arresto. I valori della velocità positivi indicano la rotazione in senso antiorario, mentre i valori della velocità negativi indicano il senso orario. Il parametro temporale opzionale, se specificato, ha un intervallo valido da 0,05 a 655,35 secondi. Se non specificato, viene utilizzato un valore predefinito di 5 secondi.
motors("ldir",left_val,"rdir",right_val,time)	Imposta le ruote sinistra e destra ai livelli di velocità specificati, per un periodo di tempo opzionale in secondi. I valori della velocità (left_val, right_ val) sono compresi nell'intervallo da 0 a 255 con 0 che indica l'arresto. I parametri ldir e rdir specificano la rotazione in senso orario o in senso antiorario delle rispettive ruote. Il parametro temporale opzionale, se specificato, ha un intervallo valido da 0,05 a 655,35 secondi. Se non specificato, viene utilizzato un valore predefinito di 5 secondi.

Percorso

Voce	Descrizione
waypoint_xythdrn()	Legge x-coord, y-coord, tempo, orientamento, distanza percorsa, numero di giri di ruota, numero di comando del waypoint corrente. Restituisce un elenco con tutti questi valori come elementi.
waypoint_prev	Legge x-coord, y-coord, tempo, orientamento, distanza percorsa, numero di giri di ruota, numero di comando del waypoint precedente.
waypoint_eta	Restituisce il tempo stimato per raggiungere un waypoint.
path_done()	Restituisce un valore di 0 o 1 a seconda che Rover si stia muovendo (0) o che abbia arrestato tutti i movimenti (1).
pathlist_x()	Restituisce un elenco di valori X dall'inizio al valore corrente di waypoint X incluso.
pathlist_y()	Restituisce un elenco di valori Y dall'inizio al valore corrente di waypoint Y incluso.
pathlist_time()	Restituisce una lista del tempo in secondi dall'inizio al valore temporale corrente di waypoint incluso.
pathlist_heading()	Restituisce una lista degli orientamenti dall'inizio al valore di orientamento corrente di waypoint incluso.
pathlist_distance()	Restituisce una lista delle distanze dall'inizio al valore di distanza corrente di waypoint incluso.
pathlist_revs()	Restituisce una lista del numero di giri effettuati dall'inizio al valore dei giri corrente di waypoint incluso.
pathlist_cmdnum()	Restituisce una lista di numeri di comando per il percorso.
waypoint_x()	Restituisce la coordinata x del waypoint corrente.
waypoint_y()	Restituisce la coordinata y del waypoint corrente.
waypoint_time()	Restituisce il tempo impiegato nello spostamento dal waypoint precedente a quello corrente.
waypoint_heading()	Restituisce l'orientamento assoluto del waypoint corrente.
waypoint_distance()	Restituisce la distanza percorsa tra il waypoint precedente e quello corrente.
waypoint_revs()	Restituisce il numero di giri necessari per spostarsi tra il waypoint precedente e quello corrente.

Impostazioni

Voce	Descrizione
units/s	Opzione per la velocità in unità di griglia al secondo.
m/s	Opzione per la velocità in metri al secondo.
revs/s	Opzione per la velocità in giri di ruota al secondo.
palmari	Opzione per la distanza in unità di griglia.
m	Opzione per la distanza in metri.
revs	Opzione per la distanza in giri di ruota.
Gradi	Opzione per la rotazione in gradi.
radianti	Opzione per la rotazione in radianti.
gradians	Opzione per la rotazione in gradi centesimali.
clockwise	Opzione per specificare la direzione della ruota.
counter-clockwise	Opzione per specificare la direzione della ruota.

Comandi

Questi comandi sono la raccolta di funzioni da altri moduli e dal modulo TI Rover.

Voce	Descrizione
sleep(seconds)	Sospende l'esecuzione del programma per un numero di secondi specificato. Importato dal modulo tempo (time).
text_at(row,"text","align")	Visualizza il "testo" (text) nell'area di tracciamento con l'"allineamento" (align) specificato. Importato dal modulo ti_plotlib.
cls()	Cancella la schermata della Shell per il tracciamento. Importato dal modulo ti_plotlib.
while get_key() != "esc":	Esegue i comandi nel loop "while" finché non viene premuto il tasto "esc".
wait_until_done()	Sospende l'esecuzione del programma finché Rover non completa il comando corrente. Questo è un modo utile per sincronizzare i comandi non Rover con il movimento di Rover.
while not path_done()	Esegue i comandi nel loop "while" fino a quando Rover non arresta tutti i movimenti.

Voce	Descrizione
	La funzione path_done() restituisce un valore di 0 o 1 a seconda che Rover si stia muovendo (0) o che abbia arrestato tutti i movimenti (1).
position(x,y)	Imposta la posizione di Rover sulla griglia virtuale sulla coordinata x,y specificata.
position(x,y,heading,"unit")	Imposta la posizione di Rover sulla griglia virtuale sulla coordinata x,y specificata, e l'orientamento virtuale relativo all'asse x virtuale viene impostato se viene fornito un orientamento (nelle unità per gli angoli specificati). Gli angoli positivi da 0 a 360 vengono considerati in senso antiorario rispetto all'asse x positivo. Gli angoli negativi da 0 a -360 vengono considerati in senso orario rispetto all'asse x positivo.
grid_origin()	Imposta RV come punto di origine corrente della griglia di (0,0).
grid_m_unit(scale_value)	Imposta la spaziatura della griglia virtuale in metri per unità (m/unità) sul valore specificato. 0,1 è il valore m/unità predefinito e viene convertito come 1 unità = 100 mm o 10 cm o 1 dm o 0,1 m. L'intervallo di valid scale_value è compreso tra 0,01 a 10,0.
path_clear()	Cancella le informazioni di percorso o waypoint preesistenti.
zero_gyro()	Ripristina il giroscopio di Rover su 0,0 e cancella i conteggi dell'encoder delle ruote sinistra e destra.

Menu Matematica complessa

Questo sottomenu si trova sotto **Altri moduli**.

Voce	Descrizione
from cmath import *	Importa tutti i metodi dal modulo cmath.
complex(real,imag)	Restituisce un numero complesso.
rect(modulus,argument)	Converte le coordinate polari nella forma rettangolare di un numero complesso.
.real	Restituisce la parte reale del numero complesso.
.imag	Restituisce la parte non reale di un numero complesso.
polar()	Converte la forma rettangolare in coordinate polari di un numero complesso.

Voce	Descrizione
phase()	Restituisce la fase di un numero complesso.
exp() (e alla potenza)	Restituisce e**x.
cos() (Coseno)	Restituisce il coseno di un numero complesso.
sin() (Seno)	Restituisce il seno di un numero complesso.
log() (Logaritmo)	Restituisce il logaritmo naturale di un numero complesso.
log10()	Restituisce il logaritmo in base 10 di un numero complesso.
sqrt() (Radice quadrata)	Restituisce la radice quadrata di un numero complesso.

Menu Tempo

Questo sottomenu si trova sotto **Altri moduli**.

Voce	Descrizione
from time import *	Importa tutti i metodi dal modulo temporale (time).
sleep(seconds)	Sospende l'esecuzione del programma per un numero di secondi specificato.
clock()	Restituisce l'ora corrente del processore come numero mobile espresso in secondi.
localtime()	Converte un'ora espressa in secondi dal 1° gennaio 2000 in una tupla di nove elementi contenente flag di anno, mese, giorno del mese, ora, minuti, secondi, giorno della settimana, giorno dell'anno e ora legale (Daylight Savings Time, DST). Se l'argomento opzionale (secondi) non viene fornito, viene utilizzato l'orologio in tempo reale.
ticks_cpu()	Restituisce un contatore in millisecondi progressivo specifico del processore con punto di riferimento arbitrario. Per misurare il tempo in modo coerente tra diversi sistemi, utilizzare ticks_ms().
ticks_diff()	Misura il periodo tra le chiamate consecutive a ticks_cpu() o ticks_ms(). Questa funzione non deve essere utilizzata per misurare arbitrariamente i periodi di tempo lunghi.

Menu TI System

Questo sottomenu si trova sotto **Altri moduli**.

Nota: Quando si crea un nuovo programma che utilizza questo modulo, si consiglia di utilizzare il tipo di programma **Condivisione dati**. In questo modo verranno importati tutti i moduli pertinenti.

Voce	Descrizione
from ti_system import *	Importa tutti i metodi (funzioni) dal modulo ti_ system.
recall_value("nome")	Richiama una variabile OS predefinita (valore) denominata "name".
store_value("name",value)	Memorizza una variabile Python (valore) in una variabile OS denominata "name".
recall_list("name")	Richiama una lista OS predefinita denominata "name".
store_list("name",list)	Memorizza una lista Python (elenco) in una variabile di lista OS denominata "name".
eval_function("name",value)	Valuta una funzione OS predefinita al valore specificato.
get_platform()	Restituisce "hh" per il palmare e "dt" per il desktop.
get_key()	Restituisce una stringa che rappresenta il tasto premuto. Il tasto '1' restituisce "1", 'esc' restituisce "esc" e così via. Quando viene richiamato senza alcun parametro - get_key() - restituisce immediatamente. Quando viene richiamato con un parametro - get_key (1) - attende finché non viene premuto un tasto.
get_mouse()	Restituisce le coordinate del mouse come tuple di due elementi, la posizione dei pixel dell'area o (-1,-1) se fuori dall'area.
while get_key() != "esc":	Esegue i comandi nel loop "while" finché non viene premuto il tasto "esc".
clear_history()	Cancella la cronologia della Shell.
get_time_ms()	Restituisce il tempo in millisecondi con una precisione al millisecondo. Questa funzionalità può essere utilizzata per calcolare una durata piuttosto che determinare l'ora effettiva.

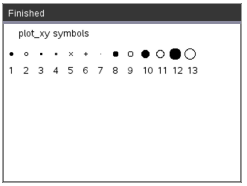
Menu TI Draw

Questo sottomenu si trova sotto **Altri moduli**.

Nota: Quando si crea un nuovo programma che utilizza questo modulo, si consiglia di utilizzare il tipo di programma **Grafica geometrica**. In questo modo verranno importati tutti i moduli pertinenti.

Voce	Descrizione
from ti_draw import *	Importa tutti i metodi dal modulo ti_draw.

Forma

Voce	Descrizione
draw_line()	Disegna una linea a partire dalla coordinata x1,y1 specificata a x2,y2.
draw_rect()	Disegna un rettangolo a partire dalla coordinata x,y specificata con la larghezza e l'altezza specificate.
fill_rect()	Disegna un rettangolo a partire dalla coordinata x,y specificata con la larghezza e l'altezza specificate e lo riempie con il colore specificato (utilizzando set_color o nero se non definito).
draw_circle()	Disegna una circonferenza a partire dalla coordinata del centro x,y specificata con il raggio specificato.
fill_circle()	Disegna una circonferenza a partire dalla coordinata del centro x,y specificata con il raggio specificato e la riempie con il colore specificato (utilizzando set_color o nero se non definito).
draw_text()	Disegna una stringa di testo a partire dalla coordinata x,y specificata.
draw_arc()	Disegna un arco a partire dalla coordinata x,y specificata con la larghezza, l'altezza e gli angoli specificati.
fill_arc()	Disegna un arco a partire dalla coordinata x,y specificata con la larghezza, l'altezza e gli angoli specificati e lo riempie con il colore specificato (utilizzando set_color o nero se non definito).
draw_poly()	Disegna un poligono utilizzando i valori x-list,y-list specificati.
fill_poly()	Disegna un poligono utilizzando i valori x-list,y-list specificati e lo riempie con il colore specificato (utilizzando set_color o nero se non definito).
plot_xy()	Disegna una forma utilizzando la coordinata x,y specificata e il numero specificato da 1 a 13 che rappresenta forme e simboli differenti (vedere di seguito). 

Voce	Descrizione
clear()	Cancella l'intera schermata. Può essere utilizzato con i parametri x,y,width,height per cancellare un rettangolo esistente.
clear_rect()	Cancella il rettangolo nella coordinata x,y specificata con la larghezza e l'altezza specificate.
set_color()	Imposta il colore delle forme che seguono nel programma fino a quando non viene impostato un altro colore.
set_pen()	Imposta lo spessore e lo stile specificati del bordo durante il disegno delle forme (non applicabile quando si utilizzano i comandi di riempimento).
set_window()	Imposta le dimensioni della finestra in cui verranno disegnate le forme. Questa funzione è utile per ridimensionare la finestra in modo che corrisponda ai dati o per cambiare l'origine (0,0) dell'area di disegno.
get_screen_dim()	Restituisce i valori xmax e ymax delle dimensioni dello schermo.
use_buffer()	Attiva un buffer fuori schermo per accelerare il disegno.
paint_buffer()	Visualizza il disegno in buffer. Le funzioni use_buffer() e paint_buffer() sono utili nei casi in cui la visualizzazione di più oggetti sullo schermo potrebbe causare ritardi.

Notes

- La configurazione predefinita presenta (0,0) nell'angolo in alto a sinistra dello schermo. L'asse x positivo punta verso destra e l'asse y positivo punta verso il basso. Ciò può essere modificato utilizzando la funzione set_window().
- Le funzioni nel modulo ti_draw sono disponibili solo sui palmari e nella vista palmare sul desktop.

Menu immagine TI

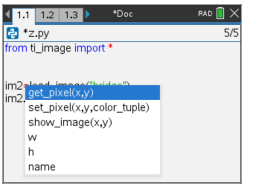
Questo sottomenu si trova sotto **Altri moduli**.

Nota: Quando si crea un nuovo programma che utilizza questo modulo, si consiglia di utilizzare il tipo di programma **Elaborazione immagine**. In questo modo verranno importati tutti i moduli pertinenti.

Voce	Descrizione
from ti_image import *	Importa tutti i metodi dal modulo ti_image.
new_image(width,height,(r,g,b))	Crea una nuova immagine con la larghezza e l'altezza specificate per l'uso nel programma Python. Il colore della nuova immagine è definito dai valori (r,g,b).
load_image("name")	Carica l'immagine specificata dal "nome" (name) per l'uso nel programma Python. L'immagine deve far parte del documento TNS in un'applicazione Note o Grafici. Il prompt "name" visualizzerà i nomi delle immagini (se sono state denominate in precedenza) o un numero che indica il loro ordine di inserimento.
copy_image(image)	Crea una copia dell'immagine specificata dalla variabile "image".

Metodi dell'oggetto immagine

Funzioni aggiuntive relative agli oggetti immagine sono disponibili nell'Editor e nella Shell digitando il nome della variabile seguito da un punto (.).



- **get_pixel(x,y):** Acquisisce il valore (r,g,b) del pixel nella posizione definita dalla coppia di coordinate (x,y).

```
px_val = get_pixel(100,100)
print(px_val)
```

- **set_pixel(x,y,color_tuple):** Imposta il pixel nella posizione (x,y) sul colore specificato in color_tuple.

```
set_pixel(100,100,(0,0,255))
```

  Imposta il pixel in (100,100) sul colore (0,0,255).

- **show_image(x,y):** Visualizza l'immagine con l'angolo in alto a sinistra nella posizione (x,y).
- **w, h, name:** Acquisisce i parametri di larghezza, altezza e nome dell'immagine.

Ad esempio

```
from ti_image import *
```

```
# An image has been previously inserted into the TNS document in a Notes
application and named "bridge"
im1=load_image("bridge")
px_val = im1.get_pixel(100,100)
print(px_val)

# Set the pixel at 100,100 to blue (0,0,255)
im1.set_pixel(100,100,(0,0,255))
new_px = im1.get_pixel(100,100)
print(new_px)

# Print the width, height and name of the image
print(im1.w, im1.h, im1.name)
```

Menu Variabili

Nota: Questi elenchi non includono variabili definite in altre app TI-Nspire™.

Voce	Descrizione
Var: Programma corrente	(Solo editor) Visualizza un elenco di funzioni e variabili globali definite nel programma corrente
Var: Ultimo programma eseguito	(Solo Shell) Visualizza un elenco di funzioni e variabili globali definite nell'ultimo programma eseguito
Var: Tutto	(Solo Shell) Visualizza un elenco di funzioni e variabili globali sia dell'ultimo programma eseguito che da ogni modulo importato

Appendice

Parole chiave Python

Le seguenti parole chiave sono integrate nell'implementazione di TI-Nspire™ Python.

False	elif	lambda
None	else	nonlocal
True	except	not
and	finally	or
as	for	pass
assert	from	raise
break	global	return
class	if	try
continue	import	while
def	in	with
del	is	yield

Mappatura dei tasti di Python

Quando si inserisce il codice nell'Editor o nella Shell, la tastiera è progettata per incollare le operazioni Python appropriate o aprire i menu per una facile immissione di funzioni, parole chiave, metodi, operatori, ecc.

Tasto	Mappatura
var	Apre il menu Variabili
sto→	Incolla il segno =
del ←	Elimina il carattere a sinistra del cursore
clear	Nessuna azione
=	Incolla il segno =
|≠≥▸	Incolla i simboli selezionati: • > • < • !=

Tasto	Mappatura
	• >= • <= • == • e • —Oppure— • not • | • & • ~
trig	Incolla la funzione selezionata: • sin • cos • tan • atan2 • asin • acos • atan
?	Visualizza i suggerimenti
:=	Incolla :=
^	Incolla **
$\sqrt[n]{x}$	Nessuna azione
x^2	Incolla **2
$\sqrt{x}$	Incolla sqrt()
×	Incolla il segno di moltiplicazione (*)
""	Incolla le virgolette (")
÷	Incolla il segno di divisione (/)
$\frac{a}{b}$	Nessuna azione
e^x	Incolla exp()
ln	Incolla log()
10^x	Incolla 10**
$\log_a$	Incolla log(value,base)
(	Incolla (
)	Incolla)

Tasto	Mappatura
[ ]	Incolla [ ]
{ }	Incolla { }
(-)	Incolla il segno di sottrazione (-)
≈	Aggiunge una nuova riga dopo la riga corrente
EE	Incolla E
?!	Incolla i simboli selezionati: • ? • ! • $ • ° • ' • % • " • : • ; • _ • \ • #
π	Incolla "pi"
	Comportamento flag esistente
↵	Aggiunge una nuova riga dopo la riga corrente

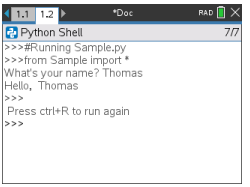
Esempi di programmi Python

Utilizzare i seguenti programmi di esempio per acquisire familiarità con i metodi Python. Essi sono disponibili anche nel file **Getting Started Python.tns** presente nella cartella **Esempi**.

Nota: Se si copia e incolla un codice di esempio contenente gli indicatori di rientro di tabulazione (• •) nel software TI-Nspire™, è necessario sostituire tali istanze con i rientri di tabulazione effettivi.

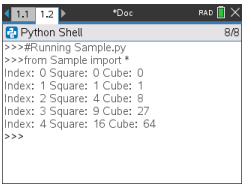
Salve

```
# This program asks for your name and uses
# it in an output message.
# Run the program here by typing "Ctrl R"

name=input("What's your name? ")
print("Hello, ", name)
print("\n Press ctrl+R to run again")
```


1.1 1.2 *Doc RAD
Python Shell 7/7
>>>#Running Sample.py
>>>from Sample import *
What's your name? Thomas
Hello, Thomas
>>>
Press ctrl+R to run again
>>>

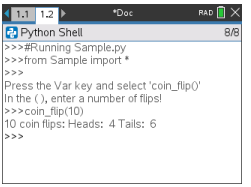
Esempio di loop

```
# This program uses a "for" loop to calculate
# the squares and cubes of the first 5 numbers
# 0,1,2,3,4
# Note: Python starts counting at 0

for index in range(5):
••square = index**2
••cube = index**3
••print("Index: ", index, "Square: ", square,
••••"Cube: ", cube)
```



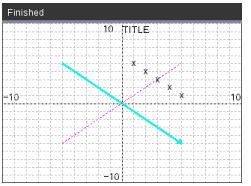
Testa o croce

```
# Use random numbers to simulate a coin flip
# We will count the number of heads and tails
# Run the program here by typing "Ctrl R"

# Import all the functions of the "random" module
from random import *

# n is the number of times the die is rolled
def coin_flip(n):
    heads = tails = 0
  for i in range(n):
# Generate a random integer - 0 or 1
# "0" means head, "1" means tails
    side=randint(0,1)
    if (side == 0):
      heads = heads + 1
    else:
      tails = tails + 1
# Print the total number of heads and tails
  print(n, "coin flips: Heads: ", heads, "Tails: ", tails)

print("\nPress the Var key and select 'coin_flip()'")
print("In the ( ), enter a number of flips!")
```



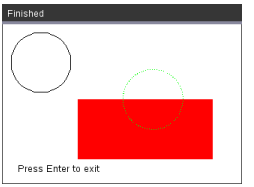
Tracciamento

```
# Plotting example
import ti_plotlib as plt

# Set up the graph window
plt.window(-10,10,-10,10)
plt.axes("on")
plt.grid(1,1,"dashed")
# Add leading spaces to position the title
plt.title("                TITLE")

# Set the pen style and the graph color
plt.pen("medium","solid")
plt.color(28,242,221)
plt.line(-5,5,5,-5,"arrow")

plt.pen("thin","dashed")
plt.color(224,54,243)
plt.line(-5,-5,5,5,"")

# Scatter plot from 2 lists
plt.color(0,0,0)
xlist=[1,2,3,4,5]
ylist=[5,4,3,2,1]
plt.scatter(xlist,ylist, "x")
```



Disegno

```
from ti_draw import *

# (0,0) is in top left corner of screen
# Let's draw some circles and squares
# Circle with center at (50,50) and radius 40
draw_circle(50,50,40)

# Set color to red (255,0,0) and fill a rectangle of
# of width 180, height 80 with top left corner at
# (100,100)
set_color(255,0,0)
fill_rect(100,100,180,80)

# Set color to green and pen style to "thin"
# and "dotted".
# Then, draw a circle with center at (200,100)
# and radius 40
set_color(0,255,0)
set_pen("thin","dotted")
draw_circle(200,100,40)

set_color(0,0,0)
draw_text(20,200,"Press Enter to exit")
```



Immagine

```
# Image Processing
#================================
from ti_image import *
from ti_draw import *
#================================

# Load and show the 'manhole_cover' image
# It's in a Notes app
# Draw a circle on top
im1=load_image("manhole_cover")
im1.show_image(0,0)
set_color(0,255,0)
set_pen("thick","dashed")
draw_circle(140,110,100)
```

Hub

Questo programma utilizza Python per controllare TI-Innovator™ Hub, un microcontrollore programmabile. L'esecuzione del programma senza collegare un TI-Innovator™ Hub visualizzerà un messaggio di errore.

Per ulteriori informazioni su TI-Innovator™ Hub, visitare education.ti.com.

```
#========== Import Section ==========
from ti_hub import *
from math import *
from random import *
from time import sleep
from ti_plotlib import text_at,cls
from ti_system import get_key
#======== End of Import Section =======

print("Connect the TI-Innovator Hub and hit 'enter'")
input()
print("Blinking the RGB LED for 4 seconds")
# Set the RGB LED on the Hub to purple
color.rgb(255,0,255)

# Blink the LED 2 times a second for 4 seconds
color.blink(2,4)

sleep(5)

print("The brightness sensor reading is: ", brightness.measurement())

# Generate 10 random colors for the RGB LED
# Play a tone on the Hub based on the random
# color
print("Generate 10 random colors on the Hub & play a tone")
for i in range(10):
  r=randint(0,255)
  b=randint(0,255)
  g=randint(0,255)
  color.rgb(r,g,b)
  sound.tone((r+g+b)/3,1)
  sleep(1)

color.off()
```

Informazioni Generali

Guida online

education.ti.com/eguide

Selezionare il proprio Paese per maggiori informazioni sul prodotto.

Contattare l'assistenza TI

education.ti.com/ti-cares

Selezionare il proprio Paese per assistenza tecnica e altre risorse.

Informazioni su servizi e garanzia

education.ti.com/warranty

Selezionare il proprio Paese per informazioni sulla durata e sui termini della garanzia o sull'assistenza ai prodotti.

Garanzia limitata. La presente garanzia non pregiudica i diritti spettanti per legge.

Texas Instruments Incorporated

12500 TI Blvd.

Dallas, TX 75243