

Atividades

Laboratório de Educação Digital



Guião 3 – Descobrimos... o Módulo de sensor ultrassónico

Explorar, Aprender e Inspirar: O Poder dos LED e da TI-Nspire™ CX II-T

- Duração: 50 min;
- Dificuldade: Intermédio;
- Tema(s): STEM; Pensamento Computacional; Relação entre música e matemática; Função afim; Progressão geométrica.

1. Objetivos

- Desenvolver competências inerentes ao trabalho STEM;
- Estudar a função afim, explorando a ordenada na origem e o declive;
- Desenvolver um instrumento musical eletrónico, utilizando conhecimentos de matemática e de programação em Python.

2. Enquadramento

Neste projeto, vamos desenvolver um instrumento musical através de uma abordagem inovadora e cativante para o ensino de matemática e para o desenvolvimento de competências previstas no Perfil dos Alunos à Saída da Escolaridade Obrigatória, como sejam, pensamento crítico, resolução de problemas e capacidade de trabalhar em equipa.

Utilizando programação em Python, a interface TI-Innovator™ Hub e um sensor ultrassónico de distância, vamos criar um sistema onde a frequência do som emitido varia de acordo com a distância detetada pelo sensor, utilizando a função afim para relacionar estas duas variáveis.

No fim do projeto, ainda podemos pôr à prova a tua sensibilidade artística para a criação de melodias utilizando o instrumento eletrónico construído.

3. Material necessário

- Calculadora TI-Nspire™ CX II-T
- TI-Innovator™ Hub¹
- Módulo de sensor ultrassónico

¹ Para instruções detalhadas de configuração e atualização iniciais do equipamento, consultar <https://education.ti.com/pt/produtos/micro-controller/ti-innovator-rover/getting-started>



Este recurso pedagógico é disponibilizado com a respetiva licença Creative Commons

4. Para começar...

Este projeto vai utilizar o TI-Innovator™ Hub, conectado ao sensor de distância do módulo de sensor ultrassônico e à TI-Nspire™ CX-II T

Certifica-te que a calculadora está devidamente conectada ao Hub, utilizando o cabo mini USB-A / mini USB-B, tal como mostrado na figura 1:

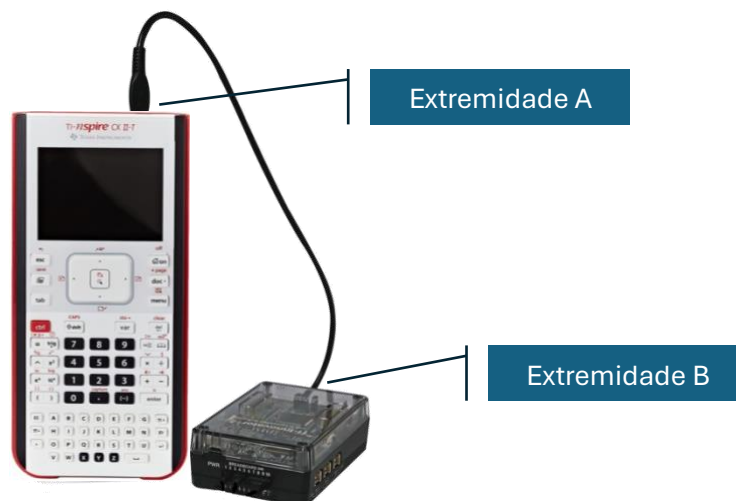


Figura 1 - Conexão do TI Innovator™ Hub à calculadora

Utiliza o cabo fornecido no módulo do sensor ultrassônico para conectar o sensor à entrada IN 1 do Hub, tal como esquematizado na figura 2.

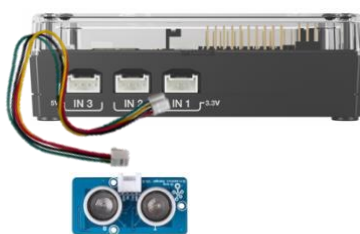


Figura 2 – Conexão do sensor ultrassônico ao Hub

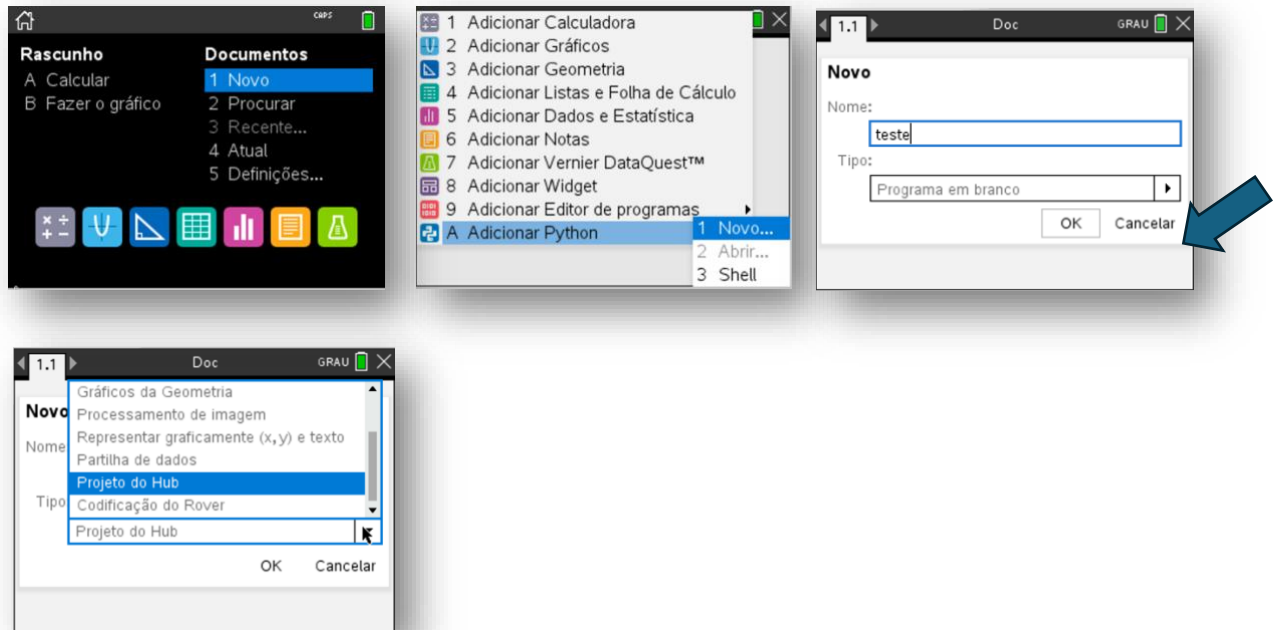
Vamos, para já, criar um programa em Python, muito simples, que apenas vai produzir um som de frequência 440 Hz, durante 0.2 segundos (ainda sem usar a informação do sensor de distância!)



Este recurso pedagógico é disponibilizado com a respetiva licença Creative Commons



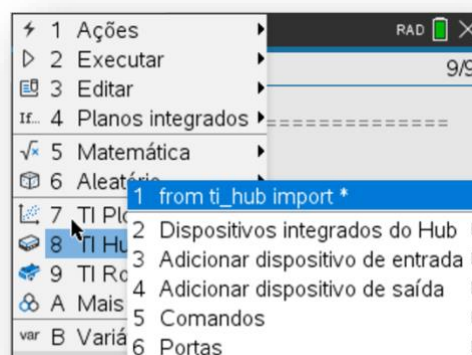
Começando por pressionar o botão  , adiciona uma página de Python, pré-configurada para projetos de Hub, utilizando os passos apresentados de seguida:



Deverás ter obtido um ecrã semelhante ao da figura:



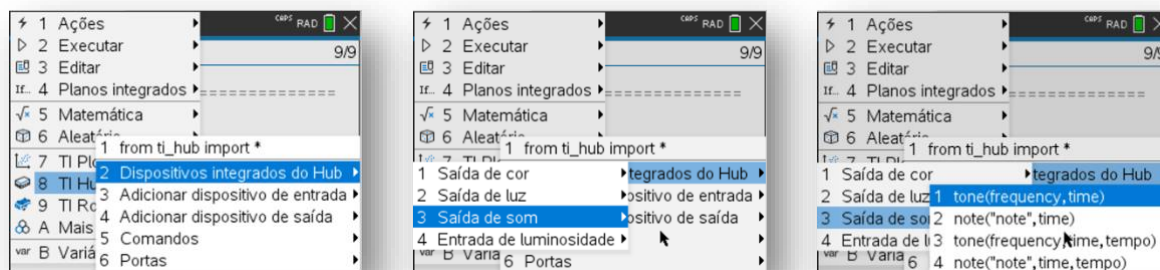
Todos os comandos necessários à construção de programas com o Hub podem ser facilmente acedidos a partir de  8 (TI Hub) :



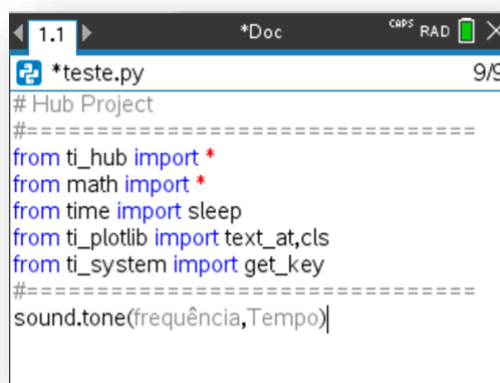
Este recurso pedagógico é disponibilizado com a respetiva licença Creative Commons



Dado que o nosso primeiro programa tem como objetivo produzir um som de uma determinada frequência, e a emissão de som faz parte integrante do Hub (não sendo necessário conectar nenhum dispositivo adicional), podemos encontrar esse comando escolhendo 2 (Dispositivos integrados do Hub) | 3 (Saída de som) | 1 (tone(frequency,time)):



Deverás ter obtido um ecrã semelhante ao da figura:



Resta introduzir um valor numérico para a frequência (neste caso, 440), e para o tempo (neste caso, 0.2), e executar o programa utilizando **ctrl** **R**. O Hub deverá produzir um som de 440 Hz, durante 0.2 segundos.

Acabaste de programar um dispositivo eletrónico para produzir a nota musical (Lá) utilizando o Python!

Todos os outros programas podem ser realizados de forma semelhante, adicionando comandos adicionais que podes encontrar a partir de **menu** (não te esqueças de regressar à página anterior (1.1), clicando em **ctrl** seta de cursor para a esquerda.)

Os próximos desafios irão convidar-te a fazer isso.



Este recurso pedagógico é disponibilizado com a respetiva licença Creative Commons

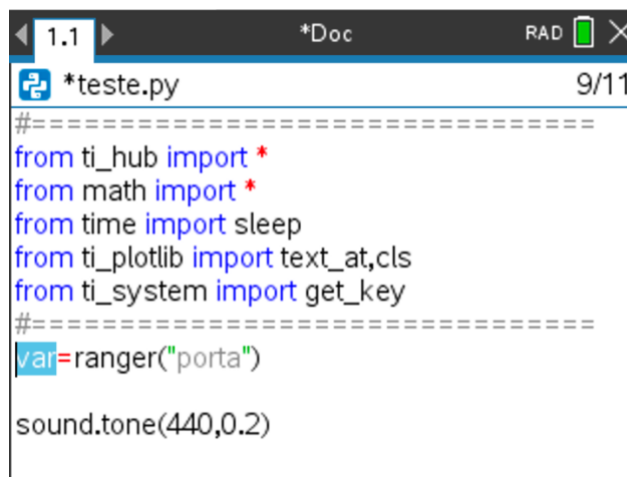
5. Desafios

1. Vamos agora utilizar a informação do sensor de distância para determinar a frequência do som emitido, utilizando uma função afim.

Para que o programa em Python “saiba” que o sensor ultrassónico está conectado à entrada IN 1 do Hub, é necessário adicionar ao programa uma linha de código com essa informação. Vamos adicionar essa linha no início do programa (**após** as importações automáticas e **antes** da produção do som pelo Hub).

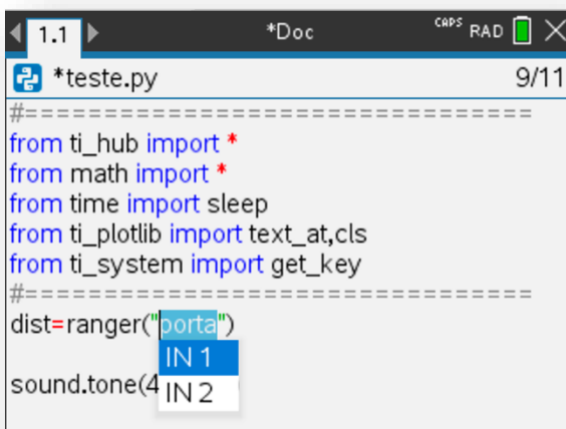
Para isso, clicar em  e seleccionar 8 (TI Hub) | 3 (Adicionar dispositivo de entrada) | 2 (Ranger)

Deverás ter um ecrã semelhante ao da figura seguinte:

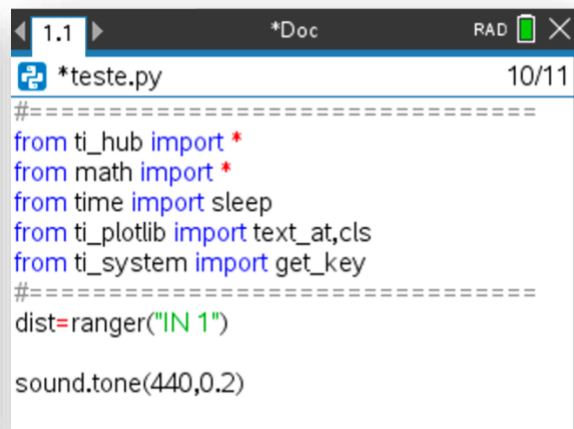


```
#=====
from ti_hub import *
from math import *
from time import sleep
from tiplotlib import text_at,cls
from ti_system import get_key
#=====
var=ranger("porta")
sound.tone(440,0.2)
```

Para que o Python identifique o *objeto* sensor ultrassónico, temos de lhe atribuir um nome e identificar a porta do Hub usada para o conectar. Assim, o texto “var” assinalado a azul deve ser substituído pela designação, por exemplo, dist e, ao deslocar o cursor até onde aparece a designação “porta”, devemos seleccionar IN 1, tal como nas imagens seguintes:



```
#=====
from ti_hub import *
from math import *
from time import sleep
from tiplotlib import text_at,cls
from ti_system import get_key
#=====
dist=ranger("porta")
sound.tone(440,0.2)
```

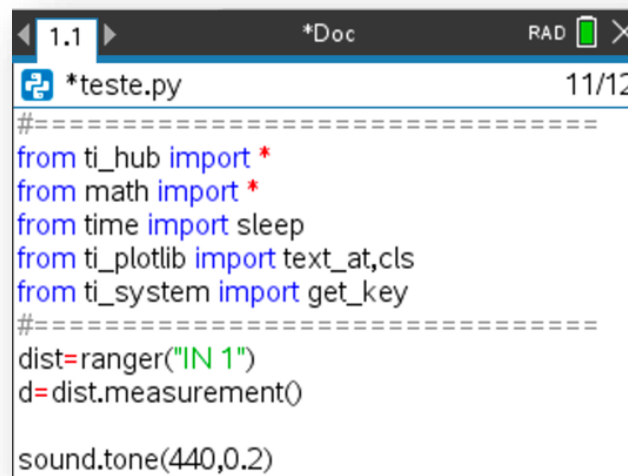


```
#=====
from ti_hub import *
from math import *
from time import sleep
from tiplotlib import text_at,cls
from ti_system import get_key
#=====
dist=ranger("IN 1")
sound.tone(440,0.2)
```

Agora, sempre que for digitado o nome do *objeto* `dist` seguido de `.` aparece um menu de opções com todas as funcionalidades disponíveis (para o caso deste sensor, `measurement()` ou `measurement_time()`). Sempre que pretendermos medir a distância do sensor a um obstáculo, utilizaremos a opção `measurement()`.

Assim, vamos criar uma variável, por exemplo `d`, que será igual ao valor indicado pelo sensor (será esta variável que permitirá calcular a frequência do som que queremos ouvir).

Assim, devemos digitar `d = dist.`¹ e seleccionar a opção `measurement()`. Deverás obter um ecrã semelhante ao seguinte:



```
#=====
from ti_hub import *
from math import *
from time import sleep
from ti_plotlib import text_at, cls
from ti_system import get_key
#=====
dist=ranger("IN 1")
d=dist.measurement()

sound.tone(440,0.2)
```

Podemos agora utilizar o valor da variável `d` para definir a função *frequência* (`f`) que irá corresponder ao som emitido pelo Hub.

Podemos utilizar, por exemplo, a expressão `f=500 * d`.

Discute com os teus colegas o que deverás alterar no comando `sound.tone()` para que o som passe a ser tocado com a frequência que acabaste de definir.

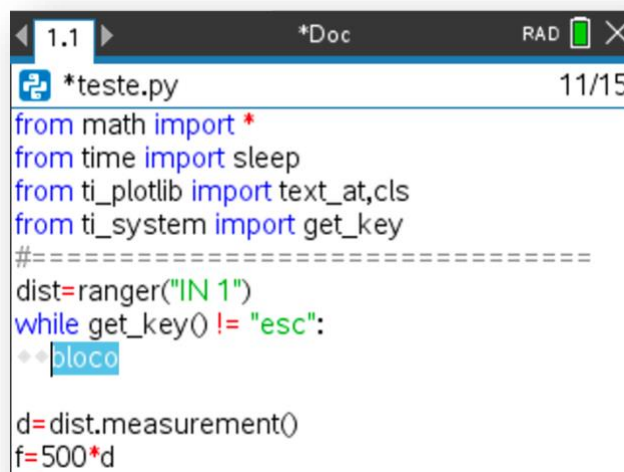
Testa o programa várias vezes (`ctrl` `R`) tendo o cuidado de, **antes** de cada execução, colocar um obstáculo (pode ser a tampa da calculadora, ou a mão) a diferentes distâncias do sensor ultrassónico. Consegues perceber a diferença no som emitido?

2. Apesar de já termos conseguido alguma interatividade com o nosso *instrumento musical*, um instrumento musical digno desse nome não pode tocar apenas uma nota de cada vez! É necessário que o instrumento toque tantas notas quantas as que pretendermos, por exemplo, até que seja premida a tecla `esc`. Para isso, é necessário adicionar uma *estrutura de repetição*, e inserir todas as 3 linhas de código que contribuem para a produção do som (medição de distância; cálculo de frequência, produção de som) nessa estrutura.

¹ Tecla `= trig`



Podes encontrar essa instrução através de  8 (TI Hub) | 5 (Comandos) | 4 (while get_key() != "esc"). Atenção: Este comando deve ser inserido **antes** das 3 linhas de código referentes ao som, e **após** a configuração da ligação do sensor ao IN 1! Deves ter obtido um ecrã semelhante a este:



```
1.1 *Doc RAD 11/15
*teste.py
from math import *
from time import sleep
from tiplotlib import text_at,cls
from ti_system import get_key
#=====
dist=ranger("IN 1")
while get_key() != "esc":
    bloco
d=dist.measurement()
f=500*d
```

Agora, para que o Python “saiba” quais as linhas de código que devem ser repetidas (até que seja premida a tecla ) , deves colocar as 3 linhas em causa com o alinhamento² em que aparece “bloco” (substituindo esta palavra pela primeira linha e adicionando dois espaços em cada uma das outras).

Podes agora correr o programa e variar continuamente a distância de um obstáculo ao sensor ultrassónico, para verificares que o instrumento toca notas de diferentes frequências. Um autêntico instrumento musical eletrónico!

Com base na tabela 1, que contém as frequências de determinadas notas musicais, define uma função afin para que o instrumento musical emita notas cuja frequência varie, aproximadamente, entre o “SOL” mais grave e o “LÁ” mais agudo da tabela. Esta variação deve corresponder a uma gama de distâncias ao sensor compreendida entre 0 e 1,5 metros.

² Este avanço chama-se indentação.

Nota	Frequência (Hz)
SOL	392,00
SOL#	415,30
LÁ	440,00
LA#	466,16
SI	493,88
DÓ	523,25
DÓ#	554,37
RE	587,33
RE#	622,25
MI	659,26
FÁ	698,46
FÁ#	739,99
SOL	783,99
SOL#	830,61
LÁ	880,00

Tabela 1 - Frequência das notas musicais

Testa o programa várias vezes, e faz as modificações que considerares necessárias.

Discute com os teus colegas as alterações de funcionamento no instrumento musical se fossem feitas modificações aos parâmetros seguintes da função afim:

- Ordenada na origem;
- Valor absoluto do declive;
- Sinal algébrico do declive.

Após terem chegado a um consenso sobre o efeito dos parâmetros anteriores, testa o programa com as alterações referidas e verifica se as previsões se confirmaram!

6. Conclusão

Com esta atividade pudeste verificar uma aplicação prática da função afim, e de que forma os seus parâmetros afetam o valor da mesma.

No entanto, esta função não seria adequada para definir as frequências das notas de um piano eletrónico, já que as frequências de notas musicais seguem uma progressão geométrica, de razão $\sqrt[12]{2}$.

Investiga como poderias modificar o teu programa para que a frequência dos sons emitidos correspondesse aos valores da tabela 1, tal como aconteceria se estivéssemos a utilizar um instrumento musical semelhante a um piano.



Este recurso pedagógico é disponibilizado com a respetiva licença Creative Commons

7. Para o professor

Sugestões de resposta aos desafios:

1.

```
1.1 *teste.py 12/13
from math import *
from time import sleep
from ti_plotlib import text_at,cls
from ti_system import get_key
#=====
dist=ranger("IN 1")
d=dist.measurement()
f=500*d
sound.tone(f,0.2)
```

2.

```
1.1 *teste.py 13/14
from math import *
from time import sleep
from ti_plotlib import text_at,cls
from ti_system import get_key
#=====
dist=ranger("IN 1")
while get_key() != "esc":
    d=dist.measurement()
    f=500*d
    sound.tone(f,0.2)
```

3.

```
1.1 *teste.py 13/15
from math import *
from time import sleep
from ti_plotlib import text_at,cls
from ti_system import get_key
#=====
dist=ranger("IN 1")

while get_key() != "esc":
    d=dist.measurement()
    f=392+(880-392)/1.5*d
    sound.tone(f,0.2)
```

Conclusão

```
1.1 *teste.py 14/15
from math import *
from time import sleep
from ti_plotlib import text_at,cls
from ti_system import get_key
#=====
dist=ranger("IN 1")
while get_key() != "esc":
    d=dist.measurement()
    n=int(d*16)
    f=440*2**(n/12)
    sound.tone(f,0.2)
```



Este recurso pedagógico é disponibilizado com a respetiva licença Creative Commons