Unit 6: micro:bit with python

Skill Builder 3: 光センサ

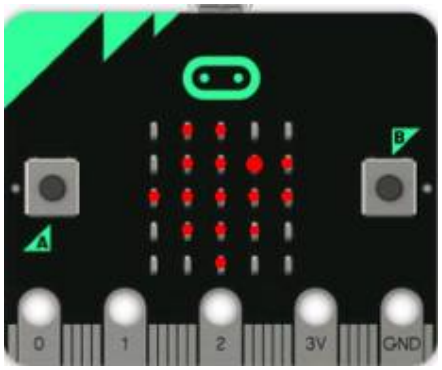
このレッスンでは，micro:bitの光センサをモニターし，さらに分析するためにデータをTI-Nspireリストに保存します。

目標

- Micro:bitの輝度を読み取って表示
- PythonからTI-Nspire CX IIにデータ転送
- micro:bitから収集したデータの調査

1. micro:bitは，ディスプレイLEDを使って周囲の光レベルを読み取ることができます。

   ディスプレイLEDは，出力だけでなく入力デバイスとしても使用でき，micro:bitに当たる光の量を測定するセンサとして利用できます。



2. 新規のドキュメントで，新規のPythonプログラムを開始します。

   [home]を押し，**New > Add Python >New…** (新規>Pythonの追加>新規…)を選択します。
   プログラムに**bright**という名前を付けます。

   光センサが検出できる値を確認するには，BBC micro:bitメニューを使って次の短いプログラムを作成します。

```
from microbit import *

while get_key() != "esc":
    bright = display.read_light_level()
    print("brightness = ",bright)
```

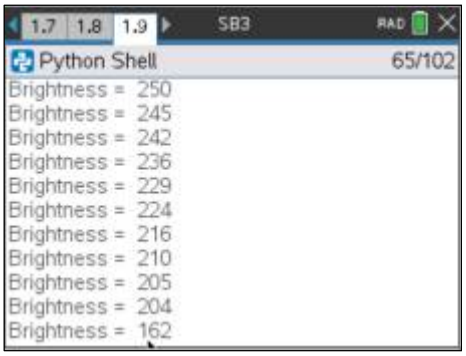
   [menu] > More Modules > BBC micro:bit >Sensors (メニュー>その他のモジュール>BBC micro:bit>センサ)で
   var= .read_light_level()を見つけます。**bright**は入力する変数名になります。

   Note: [menu] > More Modules > BBC micro:bit > Commands> while get_key() != "esc" (メニュー>その他のモジュール> BBC micro:bit>コマンド>while get_key()!= "esc")
   最後の2つのステートメントは，**while**ループ本体になるようインデントすることを忘れないでください。

3. プログラムを実行し，micro:bitのディスプレイを光源に向けます。ディスプレイに何が表示されていても問題ありません。Micro:bitを光源に近づけたり遠ざけたりして，TI-Nspire CXⅡ画面で値の変化を観察します。0から255の間で変化する値が表示されるはずです。

   想像どおり，光源から離れるほど光レベルの値は低くなります。つぎに，プログラムにコードを追加して，光レベルデータを収集し，光と時間の散布図を作成します。

   [esc]を押してプログラムを終了し，エディタに戻ります。



4. **while**ループの前に2つの空のリスト**times**(時間)，**brights**(明るさ)を作成します。

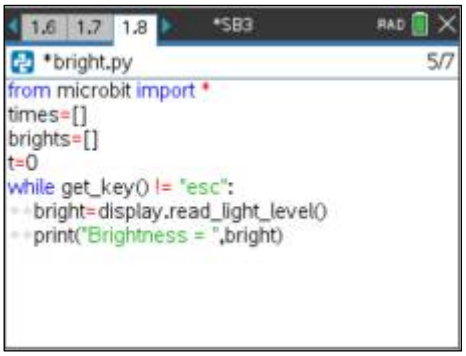
   times = [ ]
   brights = [ ]

   角かっこは，キーパッド**[ctrl]+左括弧**または**[menu] > Built-ins > Lists** (メニュー>組み込み>リスト)により入力します。

   また，**while**ループの前に，0から開始する時間カウンター変数(t)を追加します。

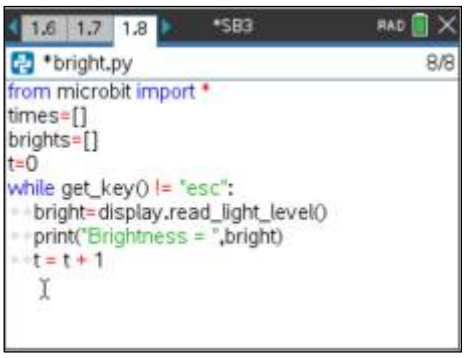
   t = 0

   timeモジュールがあるため，'time'という単語を変数として使うことは避けてください。また，リスト名には多くの値が含まれているため，複数形にすることをお勧めします。



5. **while**ループ本体の**print**ステートメントの後に，タイマー変数tを増やすステートメントを追加します。光の読み取りの間に1秒の時間間隔を使うので，次を使います。

   ♦♦t = t + 1

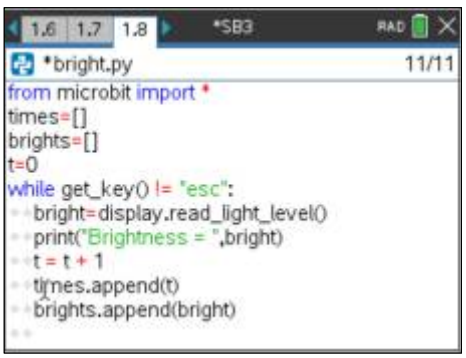


6. 次のステートメントを使って，bright値(明るさ)とt値(時間)をそれぞれのリストに追加します。

   ♦♦times.append(t)
   ♦♦brights.append(bright) ←"s"に注意!

   .append()は，**[menu] > Built-ins > Lists** (メニュー>組み込み>リスト)にあります

   これらのステートメントは，現在のbright値とt値をリストに追加します。

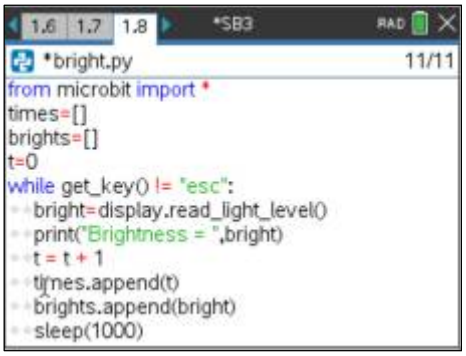


7. サンプリングのタイミング間隔を制御するには，次を追加します。

   ♦♦**sleep(1000)**

   2つの**.append**ステートメントの後に置きます。これにより，サンプル間で1秒間データ収集が一時停止します。

   Sleep()は次の場所にあります：**[menu] > More Modules > BBC micro:bit > Commands** (メニュー>その他のモジュール>BBC micro:bit>コマンド)
   micro:bitを使う場合，**sleep()**は引数にミリ秒を使うことを思い出してください。このサンプリングは1秒に1回行われます。



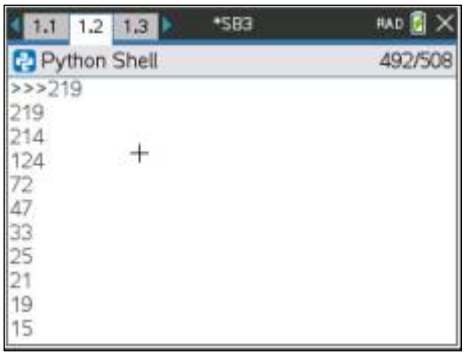
8. **while**ループ本体の後に(インデントは不要です)，Pythonリストを TI-Nspireリストに格納します。両方の環境で同じ名前を使います。

   新しい行の先頭から開始します(インデントはありません)。
   [menu] > More Modules > BBC micro:bit > Commands (メニュー>その他のモジュール>BBC micro:bit>コマンド)で，**store_list()**を見つけます。引用符で囲まれた“TI-Nspire list name”(TI-Nspireリスト名)と，引用符で囲まれていないPythonリスト名の2つの引数を入力します。ツールチップのプロンプトに従います。

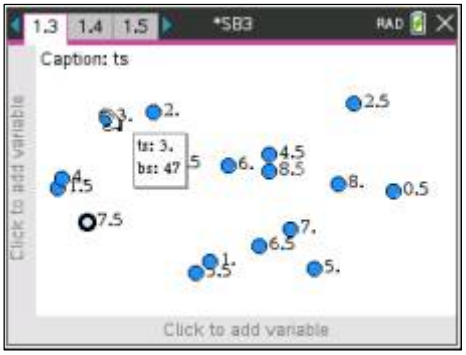
   このプログラムでは，Pythonリストは**[esc]**を押してループを終了したとき，プログラムが終了する直前にのみTI-Nspireリストに保存されます。

9. プログラムを実行(**[ctrl] [R]**)します。光源に近いmicro:bitから始めます。露出した電球やスマートフォンのフラッシュライトがうまく機能します。明るさの読み取り値が10未満になるまで，ゆっくりと，しかし着実にmicro:bitを光から一定の速度で遠ざけます。
   [esc]を押してプログラムを終了します。

   良いデータだと思うまで，このプロセスを繰り返します。

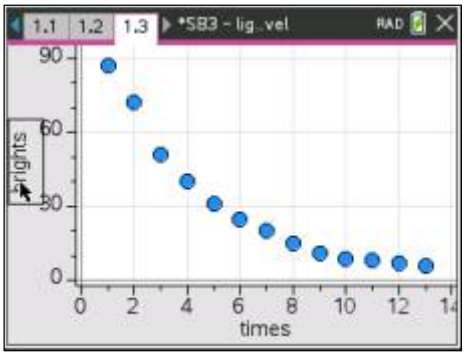


10. プログラムが終了したら，**[ctrl] [doc]**を押してドキュメントにページを追加します。’Add Data & Statistics(データと統計を追加)’を選択します。
    右図のようなページが表示されます。リストの1つが表示され，点が画面に散らばっています。プロットを整理する必要があります。



11. 画面下部にある'Click to add variable(クリックして変数を追加)'をクリックし，リストtimesを選択します。これは独立変数です。

    画面左側にある'Click to add variable(クリックして変数を追加)'をクリックし，リストbrightsを選択します。これは従属変数です。これにより，右図のように散布図は整理されます。



    ここに示すような滑らかなプロットを得るには，プログラムを数回実行する必要があるかもしれません。別データになると，ウィンドウを調整する必要があるときもあります。

    サンプル間の時間間隔を調整することもできますが，必ず **sleep(**1000**)** 引数とカウンター値(t=t+1)の両方を編集してください。

    何か気づきましたか。
    - どのようなパターンが見られますか。この物理現象に適する数学モデルは何ですか。
    - TI-Nspireのデータ分析ツールを使って，データに適した数学モデルを決定します。