Unit 5: TI モジュール

Application: Move it! (それを動かせ!)

この応用では，電卓の矢印キーを使って画面上のオブジェクトを移動することにより，インタラクティブなグラフィカルプログラムの核心を構築します。

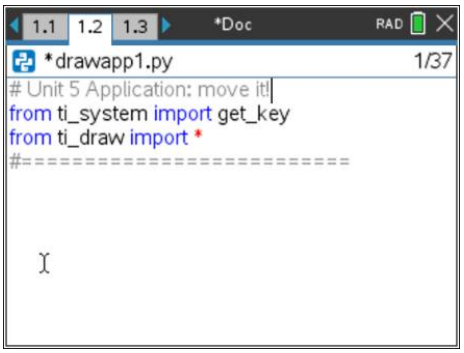
目標

- リッチで魅力的なグラフィカルプログラミングプロジェクトで，このコースのスキルと概念を組合せます。

キーを押すことに反応し，画面上でキーの方向に移動するものを描画するプログラムを作成します。

1. キーの押下を監視するには**get_key()**が必要であり，動き回る何かの画像を作成するには描画関数が必要です。

   空白のPythonファイルで，**ti_system**と**ti_draw**の2つをインポートします。または，テンプレートの1つを使って，不足しているインポートを追加します。



2. 押されたキーを知り，操作できるようにするため，キー値を変数kに格納します。

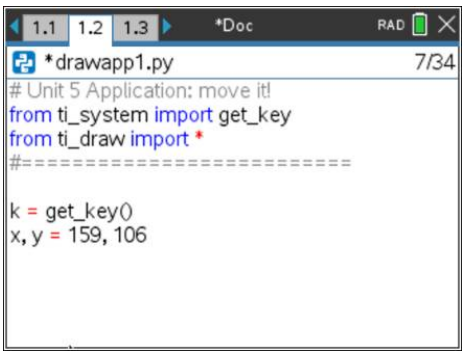
   k = get_key()

   画面中央からオブジェクト(円)をスタートします。

   x=159
   y=106

   これは次のように書くこともできます：**x,y = 159,106**

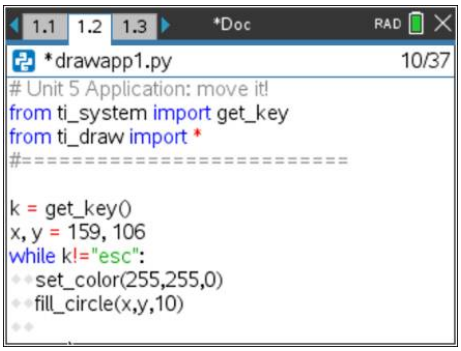
   矢印キーを押すと，これらの値の1つが変更され，オブジェクトが新しい位置に再描画され，移動しているように見えます。



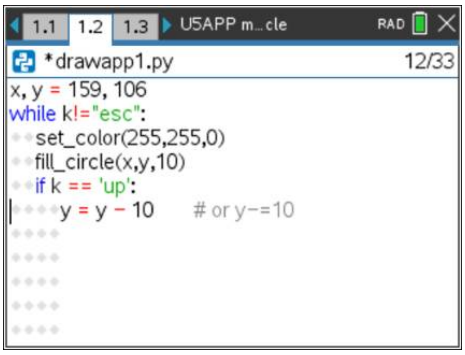
3. メインループを開始します。

   while k != ”esc”:

   カラフルな円を作成します。好きな色と半径を使ってください。



4. 押されたキーを確認します。




```
if k == “up”:
y = y – 10          (または  y -= 10)
```

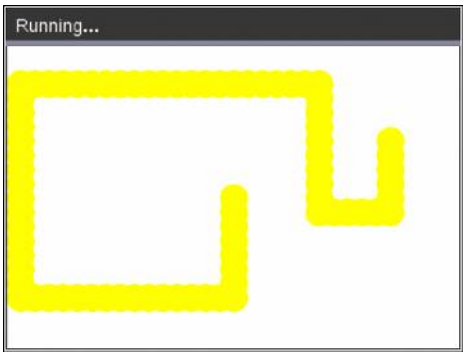
これは，上矢印が押されたとき，y値から10を引くことを意味します。これにより，円が画面上で10ピクセル上に移動します。

他の3つの矢印キー(down(下)，left(左)，right(右))の**if**ステートメントを記述し，xとyを適当に変更します。
すべての**if**ステートメントは，**while**ブロックの一部です。

最初の**k = get_key()**ステートメントはループの一部ではないため，ブロックの下部で**while**ループ内のキー押下(**k = get_key()**)を再度読み取ります。

完了したら，プログラムをテストします。

5. 矢印キーを使って円を移動します。円は動きますか，もしそうなら，おめでとうございます。



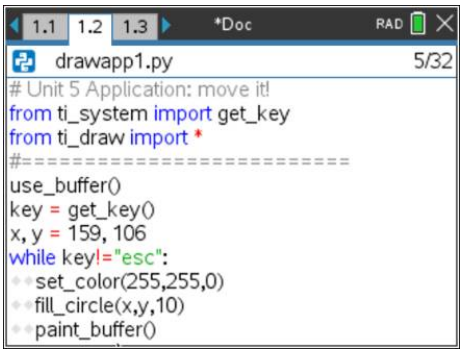
しかし，それは動いた跡を残します。

次の2つを使って，新しい円を描く前に古い円を消去します。

```
set_color(255,255,255)
fill_circle(x,y,10)
```

xとyの値を変更する前です。

6. 円はいつも黄色で描画され，すぐに白で描画されるため，点滅します。黄色の円が描かれたときのみ画面を更新する必要があります。



そのため，プログラムの先頭(whileの前)に，次のステートメントを置きます。

```
use_buffer()
```

これは，**TI Draw > Control**にあります。

そして，黄色い円が描かれた直後に，次のステートメントを配置します。

```
paint_buffer()
```

プログラムを再実行してください。

良くなりましたか。画面は，黄色の円が描かれた後にのみ，再描画されます。

7. 最後に(今のところ)，円が画面から消えるとどうなりますか。それはただ進み続けます。画面の端を検出して動作するように，より多くの**if**ステートメントでプログラムを変更します。ここには2つのオプションがあります。

   a. 端に着いたら，そこにとどまり，画面から離れない。
      (x<0のとき，x=0にする，など)
   b. 画面の一方の端から離れるときは，反対側に折り返す。
      (x<0のとき，x=317にする，など)

   幸運を！

   追加課題：キーが押されていないときでも，円が最後の方向に移動し続けるようにプログラムを変更します。