Unit 6: Rover座標

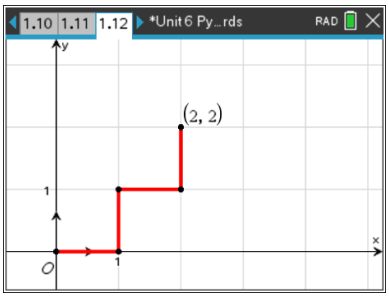
この応用では，Roverが原点(0, 0)から点(2, 2)までグリッド(格子)上を移動する頻度(北と東へのみランダムに回転させる)を確認できます。また，その点に到達する実験的および理論的な確率を調べます。

Application: ランダムウォーク

目標

- 乱数を使って次に進む方向を決定
- ゴール(2,2)に到達するかどうかを判断
- 画面に座標をプロット
- カラーLEDを使って成功または失敗を報告
- 実験を複数回繰り返し，Roverが目標に到達する頻度を決定
- 目標に到達する理論的な確率を調べる

Roverは原点Oからスタートします。各格子点で，Roverは一度に1グリッド単位ずつ，ランダムに東(右)または北(上)に移動できます。Roverがゴール(2, 2)に到達するために取ることができる異なるルートがいくつかあります。1つは，右図で赤/太字で強調表示されています。



Roverはいくつの異なる方法で目標を達成できますか。各動きがランダムである場合，Roverがゴールを逃す可能性はありますか。

Roverが目標を達成する確率はどれくらいですか。

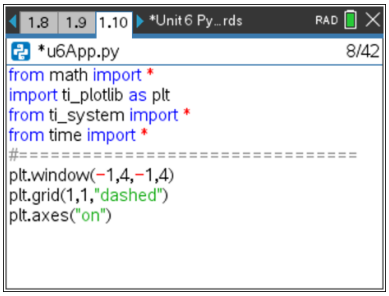
この問題のシミュレーションを作成して，Roverが目標に到達した回数を追跡し，成功した試行の割合を決定します。

失敗について考えてみましょう。Roverが目標を達成できなかったことはどうやって分かりますか。

1. 新規のPython Rover Codingプロジェクトを開始します。

   TI PlotLib > Setup(TI PlotLib>セットアップ)メニューを使って，ルートに沿ったRoverの位置を表示できるように**plt**画面を設定します。

   plt.window(-1, 4, -1, 4)
   plt.grid(1, 1, "dashed")
   plt.axes("on")



2. 試行(trial)回数の変数を設定します：**trials = 10**

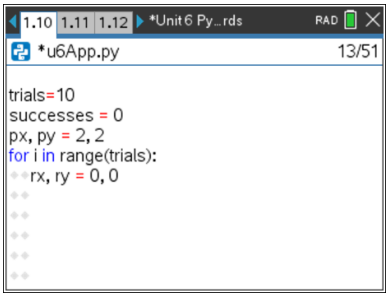
   成功(success)を数える変数を設定します：**successes = 0**

   ゴールの点には2つの変数を設定します：**px, py = 2, 2** (これは有効です!)

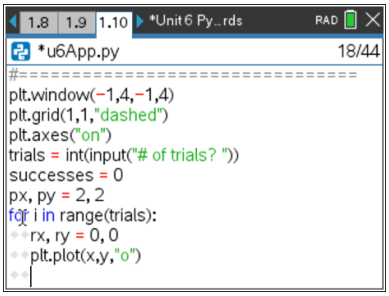
   forループを使って試行を実行します：**for i in range(trials):**

   試行を開始します。

   ループblockで，Roverの位置に2つの変数を使います：**rx, ry = 0, 0**



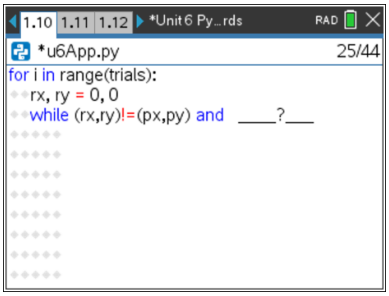
3. グラフ画面にその点をプロットします：**plt.plot(rx,ry,"o")**



4. Roverがゴールの点にいない限り継続する**while**ループを作成します。

   ローバーは失敗しません。どのような状態が失敗になるかを決めることを考えましょう。

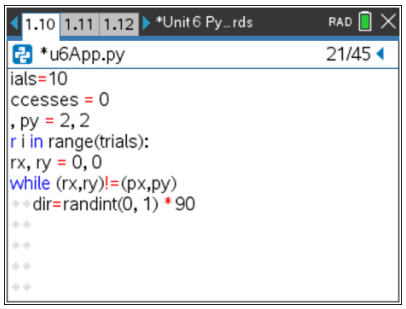
   この条件は，演習として残します。



5. **while**ブロックで，**randint(0, 1)**を使って，東(0°)または北(90°)のどちらに移動するかを決定します。これは，次のステートメントで実現されます。

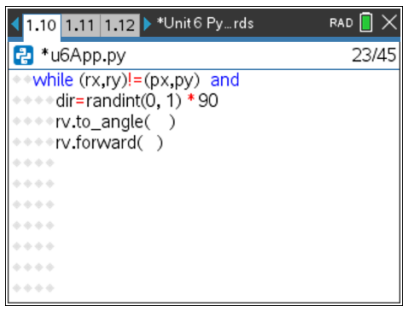
   dir = randint(0, 1) * 90

   randint()ステートメントを使うには，**from random import***をインポートする必要があります。Rover Codingテンプレートには含まれていません。



6. Roverを正確な角度(**dir**)に向けて1単位前進させます。
   (右図の記述は不完全です。)

   プロット速度を制御するため**rv.wait_until_done()**ステートメントを追加します。



7. Roverの位置変数**rx**と**ry**を更新します。

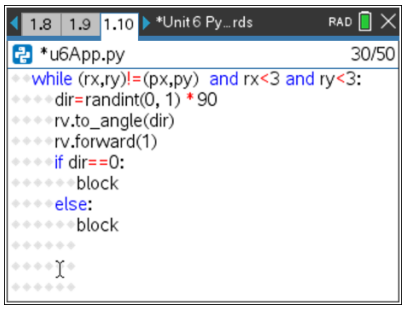
   Roverが東に移動した場合(dir == 0),

   rxに1を足します。

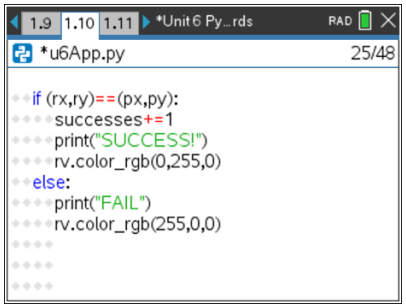
   そうでないとき,

   ryに1を足します。

   plt.plot(rx,ry,"o") (図には示されていません)を使って，画面内のRoverの位置をプロットします。



8. **while**ループが終了したら，Roverが成功したかどうかを判断します。



if (rx, ry) == (px, py):

変数**successs**に1を加えて，成功をカウントします。
“SUCCESS!” または “FAIL”を印刷します。
Roverが成功した場合はカラーLEDを**GREEN**（緑）で点灯します。それ以外はカラーLEDを**RED**（赤）で点灯します。

つぎに，次のように入力します(表示されていません)。

a) Roverが(0, 0)に戻り，東向き(角度0°)になるための2つのステートメント

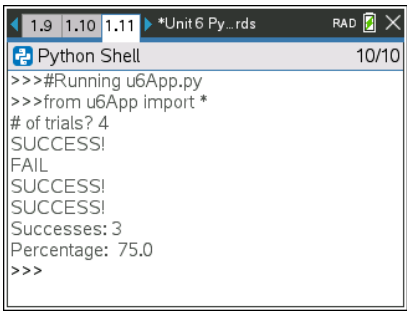
b) LEDをオフにするステートメント

c) **plt.cls()**で始まるプロット画面を再描画し，3つの**Setup**ステートメントを再度使うステートメント

9. **for**ループが終了した後(すべての試行が終了したとき)，実験結果を印刷します。

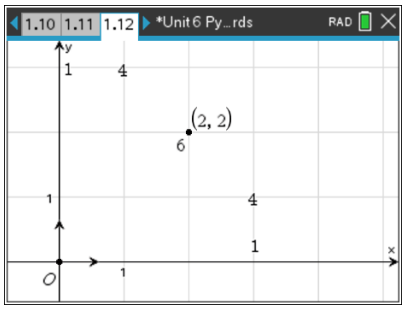
a）**successes**(成功)の総数

b）成功の割合(**successes / trials *100**)



右図は，10回の試行サンプルです。パーセンテージは，成功の実験的確率(**experimental probability**)です。

10. **理論的確率(theoretical probability)**



(0, 0)から(2, 2)へは，6つの成功ルートがあります。

x=3またはy=3のいずれかへは，10個の失敗ルートがあります(図の1と4を参照)。理論上の確率は次のとおりです。

$$\frac{\text{success}}{\text{success + fail}} = \frac{6}{6+10} = \frac{3}{8}$$

成功率は37.5%です。

あなたの実験はどれくらい近づきましたか。より正確な実験的確率(experimental probability)を得るためには何をすべきですか。