Unit 6: Rover座標

このレッスンでは，座標幾何の距離の公式を使って2点間の距離を計算し，Roverが測定した距離を計算された距離と比較します。**このレッスンでは，メータースティックまたはメートル法の巻尺が必要になります。

Skill Builder 2: 距離の公式

目標

- 2つの異なる点に移動
- マーカーを使って線分を描画
- 関数を使って2点間の距離を計算し，距離を表示
- 点の間の距離を測定
- 測定と計算の誤差を計算

ピタゴラスの定理に基づく距離の公式を思い出しましょう。

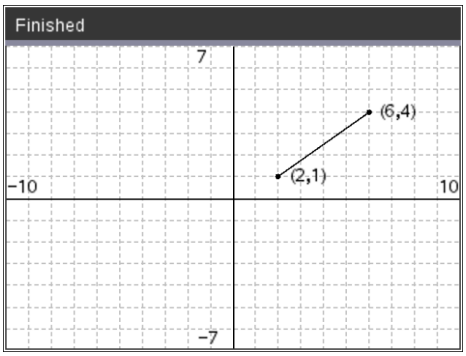
$$d=\sqrt{(x2-x1)^2+(y2-y1)^2}$$

右図に基づいて，次の式はPythonステートメントになります。

d = sqrt((6 - 2)2 + (4 - 1)**2)**

これを計算すると **d = 5**

右図に，3辺が3, 4, 5の直角三角形はありますか。



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

   4つの引数(2組の座標)を取り，2点間の距離を返す**dist**という関数を定義します。

   def function():テンプレートは，**menu > Built-ins > Functions**(メニュー>組み込み>関数)にあります。

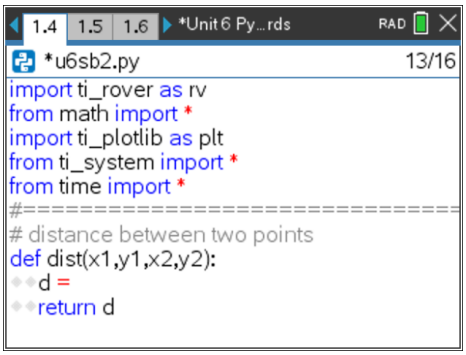
   関数の本体は，次の計算で構成されます。

   $$d=\sqrt{(x2-x1)^2+(y2-y1)^2}$$

   そして，**return**ステートメントです：**return d**

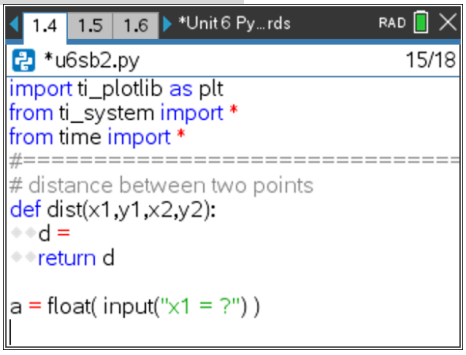
   returnは，**menu > Built-ins > Functions(**メニュー>組み込み>関数)にあります

   これら2つのステートメントは同じ量だけインデント(字下げ)されていることを確認します。



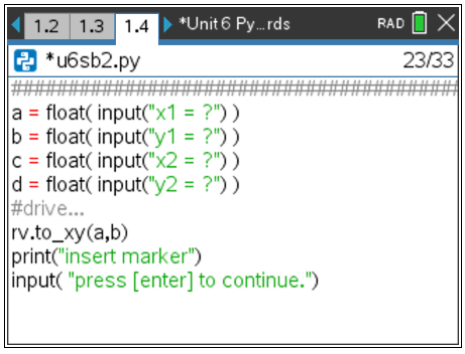
> **Teacher Tip:** グリッド(格子)単位あたり10cmではなく1インチを使うには，Roverの単位を次のように設定します。
>
> **rv.grid_m_unit(0.0254)**

2. 関数の下(**return**ステートメントの後)で，メインプログラムを開始します。コードがインデントされていないことを確認します。2つの点の座標を入力するため(コピー&ペーストを使って)4つの**input()**ステートメントを記述します。入力の簡単なプロンプトを作成し，**float()**関数を使って入力結果を文字列から10進数値に変換します。これら4つのステートメントのうち1つだけが右図に示されています。変数**a**を使って，最初のx座標を格納しています。他の3つの座標には，**b**, **c**, **d**を使います。



Teacher Tip: 入力の具体値としてパラメータ**(a, b, c, d)**を使うことは，これらが関数定義に使われる公式の引数**(x1, x2, y1, y2)**とは異なるということをはっきり意識できます。

3. 4つの**input()**ステートメントの後，Roverを最初の点までドライブさせます。Roverのマーカーホルダーにマーカーを挿入して線分を描画している間，そこで一時停止します。つぎに，2番目の点までドライブを続けます。適切な一時停止ステートメントは，次のとおりです。

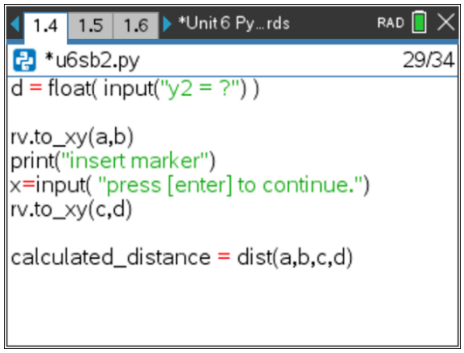


input(“press [enter] to continue.”)

input(「続行するには[enter]を押してください。」）

この入力関数の結果は何も入力されていないため，変数に値を割り当てません。

4. つぎに，プログラムに2点の座標を使って距離関数を評価させます。

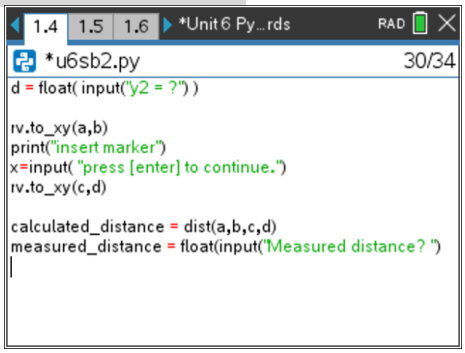


calculated_distance = dist(a, b, c, d)

Teacher Tip: 分かりやすくするため，長い変数名が使われています。学生は，メータースティックまたはメートル法の巻尺を使って線分の長さを測定し，**calculated_distance**と比較します。

文字dは，2つの異なる方法で変数として使われていることに注意します。メインプログラムでは，2番目の点のy座標を表しますが，**dist( )**関数では，計算された距離の値を格納するため使われます。これらの2つの変数は，異なる「スコープ」，つまり変数がlives(存在する) (または：is valid(有効である))プログラムの部分を持っているため，互いに競合しません。

5. 定規または巻尺を使って，Roverが作成した線分の長さを決定します。



input()ステートメントをプログラムに追加して，**measured_distance**を入力できるようにします。

print()ステートメントを追加して，2つの距離変数を表示します。

測定された距離は，計算された距離とどのように比較されますか。

6. 式を使ってパーセント誤差を計算します。

(measured - calculated) / calculated * 100

そして，エラーを出力します。

Teacher Tip: 複数のRoverがある場合は，それぞれでこのプログラムを試して，どのRoverが最も正確であるかを判断します。