Unit 4: forループとリスト

Skill Builder 3: ランダムに投げる

このレッスンでは，パターンを調べるため乱数リストを作成します。

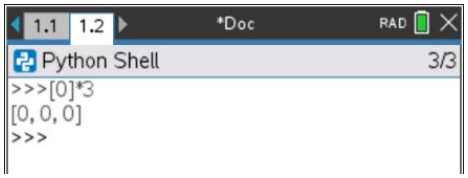
目標

- **randint(a,b)**を使用
- ランダムな整数のリストを作成
- リストのプロパティを分析

Teacher Tip: このレッスンは難しいところがあります。構文エラーがポップアップする場合があり，ヘルプメッセージはあまり役に立ちません。
レッスン最後に選択演習がありますが，これは難しいところがあるので省略してもかまいません。

はじめに：

リスト(または配列)は，1つの変数に多くの異なる値を格納する便利な方法です。Pythonはリストの内容に非常に柔軟性があり(さまざまなタイプの値を保持できますが，これはまれです)，リストを作成する方法がいくつかあります。



注目に値する式の1つは，要素を3回複製する**[0] * 3** です。

3 * [1,2,3]は何を生成しますか。電卓で同じ式を実行すると，右図に示されているようにまったく異なる結果が得られます。

Pythonでリストを操作するときは注意が必要です。これは，結果が慣れていない可能性があるためです。

このレッスンでは基本に固執します。

Teacher Tip: Pythonリストを作成，使用するさまざまな方法については，オンラインドキュメントを確認してください。**print()**ステートメントを頻繁に使って，変数で何が起こっているかを確認します。リストを印刷するのは簡単です：**print(***listname***)**

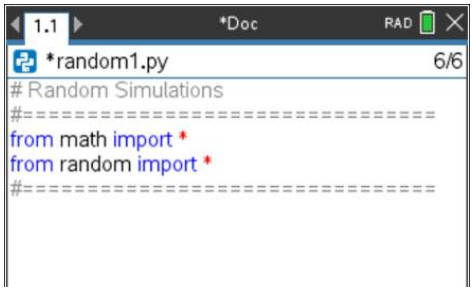
1. このプロジェクトでは，新規のPythonプログラムを作成するときに，**Type:** のドロップダウンリストからRandom Simulations(ランダムシミュレーション)を選択します。

2. ランダムシミュレーションテンプレートは，**math**(数学)と**random**(乱数)の2つのモジュールを提供します。

   Mathモジュールには，sqrt()，trig関数，exp()，log()などの関数があることを確認できます。

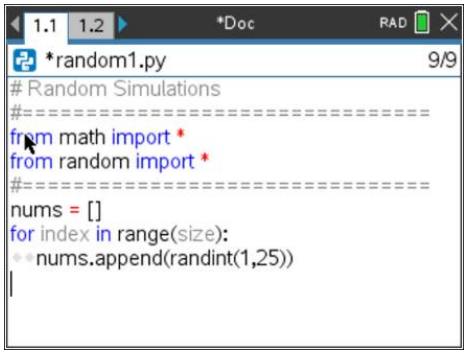
   randomモジュールには，乱数を処理する関数が含まれています。

   便利な関数の1つは，aとbの値の間のランダムな整数を返すrandint(a, b)です。引数は，数値，変数，または数値を生成する式にすることができます。



Teacher Tip: コンピュータは高度なアルゴリズムを使って疑似乱数を生成します。メソッドは予測可能です。randomモジュールに含まれるseed()関数に同じ値を指定すると，プログラムが実行されるたびに同じ乱数列が生成されます。これはテストには役立ちますが，真のランダムイベントをシミュレーションしようとする場合はお勧めできません。乱数に乱数をシード(seed)して，よりランダムな感覚を与えるための手法(多くは内部時計を使用)があります。

3. まず，空のリストを作成します。



nums=[ ] （括弧の間に何もありません）

forループを使って，それぞれが0から25の範囲から選択された100個の乱数のリストを作成します(インデックス変数と範囲サイズを指定します)。

loopブロックは次のとおりです。

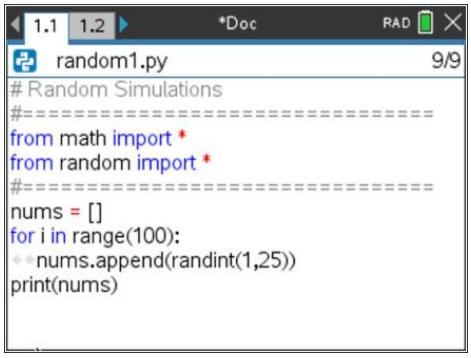
nums.append(randint(1,25))

このステートメントの最後にある2つの右括弧に注意してください。一般的な構文エラーです。

Teacher Tip: 次は，リスト内包表記を使う別の方法です。

num=[randint(1,25) for i in range(100)]

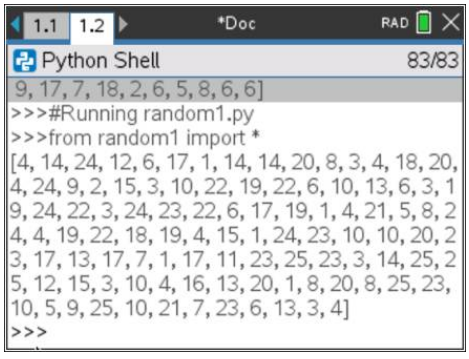
4. プログラムにprintステートメントを追加して(forループの終了後)，数値が作成された後にリストを印刷します。



print(nums)

プログラムを実行して，動作することを確認してください。シェルで**ctrl+R**を押すと，プログラムが再実行され，実行ごとに異なる数値のセットが表示されます。

5. 右図はサンプルの実行例です。

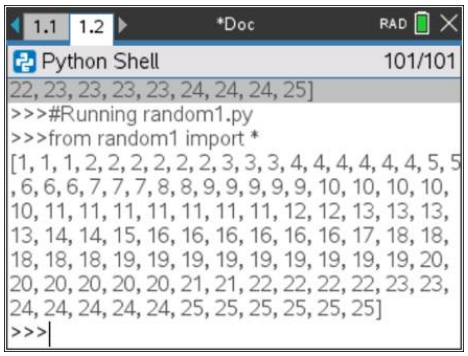


プログラムにコードを追加して，数値の平均を決定します。最初に自分でやってみてください。(前回のレッスンを覚えておきましょう。)

リストに最小値と最大値を表示します(ヒント：**menu > Built-ins > Lists**を参照してください)。

要素を並べ替えることができますか。

6. 並べ替えについて



Pythonには，リストを並べ替える2つのツールがあります。

nums.sort()は，要素を昇順(値が小さい順)で並べ替えます。

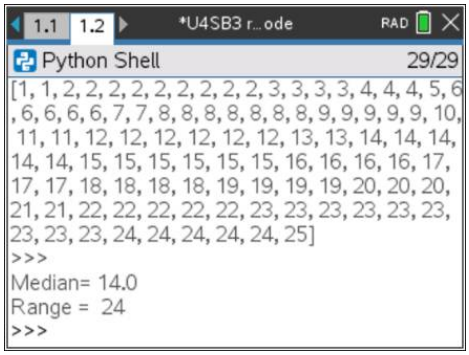
sorted(n)は，ソートされているが元のリストを変更しないリストを返します。

nums2 = sorted(n)を使って，元のリストを保持し，**nums2**という名前の新しいソート済みリストを作成します。

Teacher Tip:
list.**sort()**と**sorted(**list**)**は，リストクラスの2つの異なるメソッドです。
.sort()はリストを変更し，**None**を返します。
sorted()はリストを変更しませんが，ソートされたリストを返します。
2つの異なる構文に注意してください。詳細については，オンラインで確認してください。

選択演習:

7. リストを並べ替えると，データセットの**median**(中央値)と**range**(最小値と最大値の差)を決定できます。



これらの値を表示するよう，プログラムにステートメントを追加してください。

Note: リストの要素の数が偶数の場合，中央値は2つの中央の数値の平均です。

リストの要素の数は，**menu > Built-ins > Lists**にある**len(*<list>*)**関数を使って得られます。**len**はlength(長さ)の省略形で，リストの場合，要素の数を返します。

Q1(第1四分位)とQ3(第3四分位)の値も決定できますか。

Teacher Tip: 選択演習の解答(予約語を使わないように注意してください。予約語とは，システム用に意味が決まっていて，変数名などに使うことのできない語です。)

```
lenth = len(nums2)
if lenth % 2 == 0:
    medi = (nums2[lenth // 2] + nums2[lenth // 2 + 1]) / 2
else
    medi = (nums2[lenth // 2])
print("Median=",medi)
rang = max(nums2) – min(nums2)
print("Range = ",rang)
```

追加課題：**mode**(s)を決定してください。