

Unit 1: Python入門

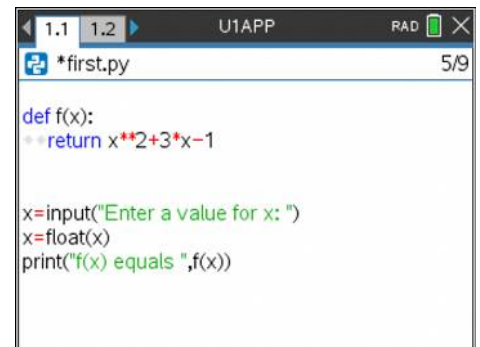
Application: 関数は1つより2つ

この応用では、2番目の関数を作成し、2つの関数を使って数学的性質を調べます。

目標

- Pythonファイルの編集
- Pythonファイルのコピー
- 機能の追加
- シェルでの関数式の評価
- 逆関数の作成

1. 前回のレッスン(ユニット1 スキルビルダー3)では、関数 $f(x)$ を定義しました。この応用(Application)では、そのファイルに別の関数を追加し、それらの関数を使っていくつかの式を評価します。



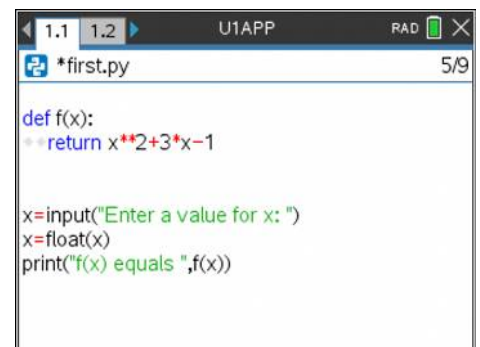
```

1.1 1.2 U1APP RAD 5/9
*first.py
def f(x):
    return x**2+3*x-1

x=input("Enter a value for x: ")
x=float(x)
print("f(x) equals ",f(x))
    
```

1. スキルビルダー3のPythonプログラムを使ってTI-Nspireドキュメントから始めます。右図を参照してください。

**doc > File > Save As...** (ドキュメント>ファイル>名前を付けて保存...)を押して、別の名前でTI-Nspireドキュメントを保存し、別の名前を使います。右図のタイトルバーは、新しい名前U1APPを示しています。



```

1.1 1.2 U1APP RAD 5/9
*first.py
def f(x):
    return x**2+3*x-1

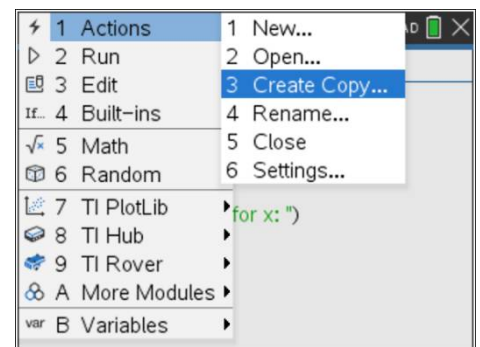
x=input("Enter a value for x: ")
x=float(x)
print("f(x) equals ",f(x))
    
```

2. Pythonエディタで**menu > Actions > Create Copy...** (メニュー>アクション>コピーの作成...)を押して、Pythonプログラムのコピーを作成します。

コピーに別の名前を付けます(既定値では、名前の末尾に1が追加されます)。

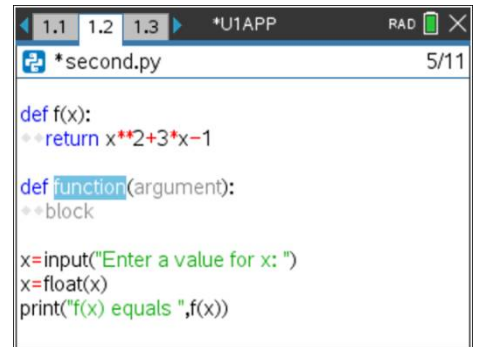
**(Create Copy...(コピーの作成...)が使えない場合、プログラムでctrl+Bを押して保存します。エディタ上部のPythonファイル名の前にアスタリスクを付けしないでください)。**

これにより、ドキュメント内に重複コードを持つ、別のPythonエディタアプリが作成されます(ドキュメントU1APPは、ページ1.1にfirst.py、ページ1.2にfirst1.pyを持ちます)。



3. 新規のPythonファイル名は**second.py**です。

つぎに、関数f(x)の下に2番目の関数テンプレートを追加します。空白行で、**menu > Built-ins > Functions** (メニュー>組み込み>関数)を押し、**def function()**を選択します。この場合も、関数の構文には、**def**行の最後にコロンの(:)が含まれています。これは、次のコードが関数の定義であり、ブロックがインデントされていることを示しています。



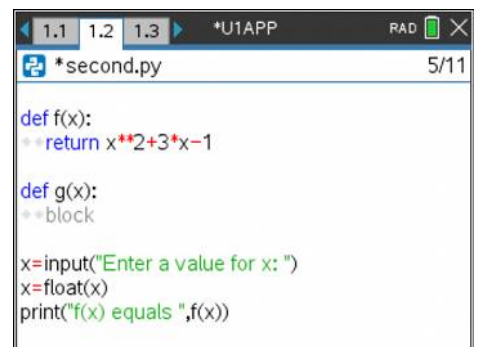
```

def f(x):
    ++return x**2+3*x-1

def function():
    ++block

x=input("Enter a value for x: ")
x=float(x)
print("f(x) equals ",f(x))
    
```

4. 2番目の関数にg(x)という名前を付けます。



```

def f(x):
    ++return x**2+3*x-1

def g(x):
    ++block

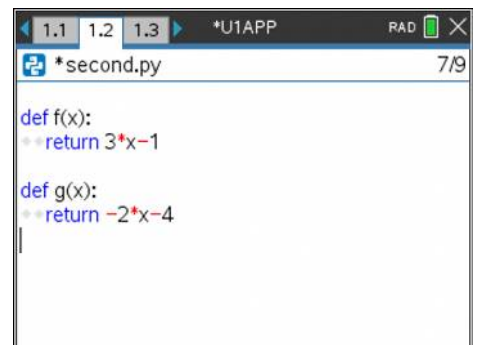
x=input("Enter a value for x: ")
x=float(x)
print("f(x) equals ",f(x))
    
```

5. 関数f(x)からx\*\*2+を削除して、**f(x) = 3 \* x - 1**になるようにf(x)を変更します。  
g(x)を定義します。

```

def g(x):
    ◆◆return -2*x - 4
    
```

プログラム下部にある3行のコードを削除し、2つの関数のみを残します。



```

def f(x):
    ++return 3*x-1

def g(x):
    ++return -2*x-4

x=input("Enter a value for x: ")
x=float(x)
print("f(x) equals ",f(x))
    
```

6. **ctrl+R**を押して、式f(1)+g(1)を入力します。

次のような2つの関数を使って、他の式を試してください。

f(4)+g(1), f(5)+g(2), (f+g)(4), f(g(6))

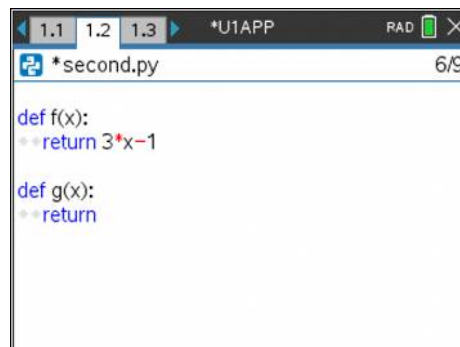


```

>>>#Running second.py
>>>from second import *
>>>f(4)
11
>>>g(1)
-6
>>>f(5)+g(2)
6
>>>f(g(6))
-49
>>>
    
```

7. 合成関数と逆関数

準備はいいですか。エディタに戻り、つねに $f(g(x)) = g(f(x))$ になるように $g(x)$ を変更します( $x$ の値に関係なく)。関数を、注意を払ってテストしてください。



```
1.1 1.2 1.3 *U1APP RAD 6/9
*second.py
def f(x):
    return 3*x-1
def g(x):
    return
```