VBA と Python の比較
— QuadFnc クラスの実装を例に —

VBA の場合

CommandButton1 をクリックした時のプロシージャ

```
Private Sub CommandButton1_Click()
    Set myFnc = New QuadFnc
    Call myFnc.init(2, 3, -4)
    Call myFnc.solve(sols)
    msg = "solutions␣of␣" & myFnc.str_() & "=0␣:" & vbCrLf
    msg = msg & "(" & Str(sols(0)) & "," & Str(sols(1)) & ")"
    MsgBox msg
    Call myFnc.plot(-3, 1, 0.1)
End Sub
```

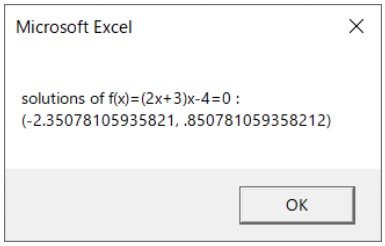
QuadFnc.cls（使用する workbook の VBAProject に追加）

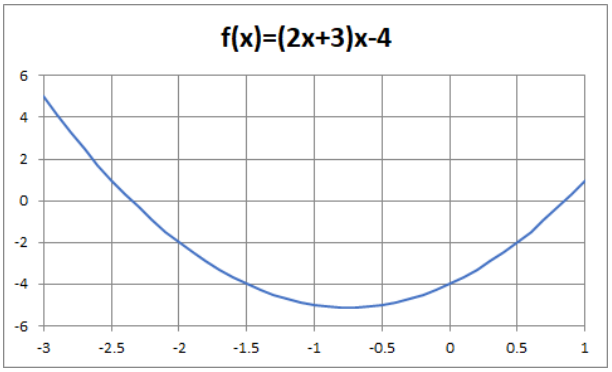
```
Private self_a As Double
Private self_b As Double
Private self_c As Double

Function f(x)
    f = (self_a * x + self_b) * x + self_c
End Function

Sub init(a, b, c)
    self_a = a
    self_b = b
    self_c = c
End Sub

Sub solve(ByRef sols)
    ReDim sols(1)
    If self_a = 0 Then
        er = CVErr(xlErrNA)
        sols(0) = er
        sols(1) = er
        Return
    End If
    dd = self_b ^ 2 - 4 * self_a * self_c
    If dd >= 0 Then
        d = Sqr(dd)
        a2 = self_a * 2
```

```
        sols(0) = (-self_b - d) / a2
        sols(1) = (-self_b + d) / a2
    Else ' give up treating complex number
        MsgBox "There is no solution of real number"
        er = CVErr(xlErrNA)
        sols(0) = er
        sols(1) = er
    End If
End Sub

Function str_()
    If self_b > 0 Then cb = "+" Else cb = ""
    If self_c > 0 Then cc = "+" Else cc = ""
    str_ = "f(x)=(" & Trim(Str(self_a)) & "x"
    str_ = str_ & cb & Trim(Str(self_b)) & ")x"
    str_ = str_ & cc & Trim(Str(self_c))
End Function

Sub plot(xmin, xmax, xdiv)
    n = Int((xmax - xmin) / xdiv)
    xdiv = (xmax - xmin) / n
    XValues = "{" & Str(xmin)
    YValues = "{" & Str(Me.f(xmin))
    For i = 1 To n
        x = xmin + i * xdiv
        XValues = XValues & "," & Str(x)
        YValues = YValues & "," & Str(Me.f(x))
    Next
    XValues = XValues & "}"
    YValues = YValues & "}"
    With ActiveSheet.Shapes.AddChart.Chart
        .ChartType = xlXYScatterLinesNoMarkers
        .HasTitle = True
        .ChartTitle.Text = Me.str_()
        .HasLegend = False
        With .SeriesCollection.NewSeries
            .XValues = XValues
            .Values = YValues
        End With
        With .Axes(xlCategory)
            .MinimumScale = xmin
            .MaximumScale = xmax
            .CrossesAt = xmin
            .HasMajorGridlines = True
        End With
        With .Axes(xlValue)
```

```
            .CrossesAt = .MinimumScale
        End With
    End With
End Sub
```

2,242 bytes





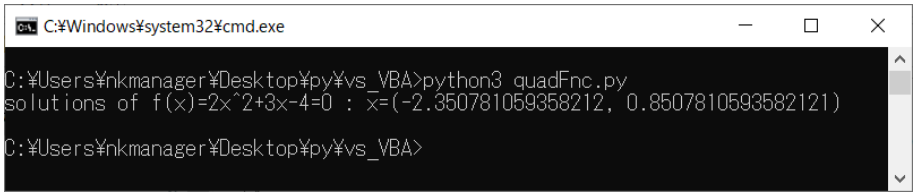
Python の場合

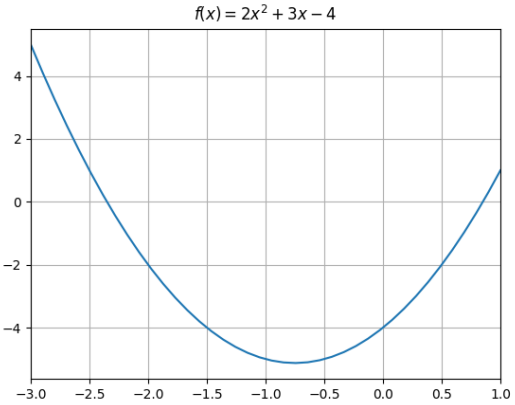
quadFnc.py

```python
import numpy as np
import matplotlib.pyplot as plt

class QuadFnc:

    def __init__(self, a, b, c):
        self.a = a
        self.b = b
        self.c = c

    def f(self, x):
        return (self.a * x + self.b) * x + self.c

    def solve(self):
        if self.a == 0: return (np.nan, np.nan)
        d = pow(self.b * self.b - 4 * self.a * self.c, 0.5)
        a2 = 2 * self.a
        return ((-self.b - d) / a2, (-self.b + d) / a2)

    def plot(self, xmin, xmax, xdiv):
        n = int((xmax - xmin) / xdiv)
        x = np.linspace(xmin, xmax, n)
        y = (self.a * x + self.b) * x + self.c
        plt.plot(x, y)
        plt.title('$' + self.__str__() + '$')
```

```
26          plt.xlim(xmin, xmax)
27          plt.grid()
28          plt.show()
29
30      def __str__(self):
31          return 'f(x)={}x^2{:=+}x{:=+}'.format(self.a, self.b, self.c)
32
33  if __name__ == '__main__':
34
35      myFnc = QuadFnc(2, 3, -4)
36      print('solutions␣omyFnc␣{}=0␣:␣x={}'.format(myFnc, myFnc.solve()))
37      myFnc.plot(-3, 1, 0.1)
```

899 bytes





この QuadFnc クラスは以下のように記述することで他のプログラムで容易に利用することができる*1。

useCls.py

```
1  from quadFnc import QuadFnc
2
3  f = QuadFnc(1, 6, -1)
4  print('solution␣of␣{}=0␣:␣x={}'.format(f, f.solve()))
5  f.plot(-8, 2, 0.2)
```

*1 useCls.py の実行時に quadFnc.py の 34 行目以降は無視される（33 行目が否）。