Eric Rasmusen
October 31, 2022

Handout: The Fibonacci Sequence

The Fibonacci Sequence is the sequence of numbers $0, 1, 1, 2, 3, 5, 8, 13, 21, \ldots$.

In math notation, it can be written as $x_1, x_2, x_3, \ldots$, where $x_1 = 0, x_2 = 1$, and $x_i = x_{i-2} + x_{i-1}$.

In Python, you can program them this way:

```
#First I describe what is in this file:

#October 31, 2022
# Python 3
# Code for printing the Fibonacci Sequence

print ("_____________________________________")
#First we import the special packages we need for this program.

import math as  math #This is for euler's number
import decimal as d    #This is for setting the number of decimal places, the precision level.
import matplotlib.pyplot as plt  #This is for drawing plot diagrams

print ("_____________________________________")
#Now we will initialize some values.

d.getcontext().prec = 64 #This sets the number of decimal places obtained using the d.Decimal() c

end = 8   # Set this to however many fibonacci numbers you want.

fibonacci = [( "blank", item) for item in range(0,end+1)] #  Create a list of length n.
#We will change everything inside it later.
#To Pythoners: You can use the Append command.

print("Our dummy initial list is", fibonacci, ". \n")

print ("_____________________________________")
#Here is the essence of the program, which does the real work.

fibonacci[0] =0
fibonacci[1] = 1

print(Our dummy list now is different:'', fibonacci, ". \n")
```

```
for item in range(2,  end+1):
 fibonacci[item] = fibonacci[item-2] + fibonacci[item-1]

print ("The first", end, "Fibonacci numbers are", fibonacci, ". \n")

print ("_____________________________________")
#Now we see how to compute the Golden Ratio
golden_ratio = [( "blank", item) for item in range(0,end+1)] #  Create a dummy list of length n.
#We will change everything inside it later.

for item in range(1, end ):   #This is method 1
  golden_ratio[item]=  fibonacci[item+1]/fibonacci[item]
  golden_ratio[item] = d.Decimal(golden_ratio)# this changes the precision level of the estimate.

print("The golden ratio using method 1 is", golden_ratio, ".\n")
print ("_____________________________________")

#For method 2, see https://twitter.com/pickover/status/1584560114144784384?s=03
#Method 2 is worse for small i . Does it ever get better than method 1? I don't know.
for item in range(1, end ):  #Now method 2
  golden_ratio_2[item]=  fibonacci[item]**(1/item)
  golden_ratio_2[item] = d.Decimal(golden_ratio_2)# this changes the precision

print("The golden ratio using method 2 is", golden_ratio_2, ".\n")

print ("_____________________________________")
 #The Euler number, e, equals about 2.7.
#It is the number e such that the slope of the curve f(x)=e^x  at point x equals  f(x),
#the height of the curve at point x.

fibonacci_estimate=[( "blank", item) for item in range(0,end+1)]
fibonacci_estimate[0] = 0

for item in range(1, end +1):
fibonacci_estimate[item] =  (math.e**(item-2))**.5
print("The square root of  e  to the power", item-2, "is", fibonacci_estimate,
  "and fibonacci number", item,"is", fibonacci[item], ".\n")

print("This is only magically close up to fibonacci number 8.\n" )


print ("_____________________________________")
```

Here is some code to add to plot the last set of estimates and the Fibonacci numbers themselves:

```
integers = [item for item in range(0, end+1)]
print("The integers are", integers)

print("The fibonacci numbers are", fibonacci)
print("The fibonacci estimates are", fibonacci_estimate)

plt.plot(integers,fibonacci, linestyle="None", marker= ".")
plt.plot(integers,fibonacci_estimate, linestyle="None", marker= "x", color="red")

# Set x, y limits for the points covered by the diagram:
plt.xlim(0,end )
plt.ylim(0, fibonacci[end ])

#Label the horizontal and vertical axes
plt.xlabel("Integers")
plt.ylabel("Fibonacci\nNumbers", rotation= "horizontal", horizontalalignment="left" , labelpad=6)
plt.title("The Fibonacci Numbers", color="red", fontsize = 20 )

plt.show()  #Show the plot diagram
```