C200 Programming Assignment № 6

Dr. M.M. Dalkilic
Computer Science
School of Informatics, Computing, and Engineering

Indiana University, Bloomington, IN, USA

March 31, 2022

The HW is due on **Thursday, April, 07 at 10:59 PM EST**. Please commit, push and submit your work before the deadline.

Problem 1: Recursion

The binomial coefficient is used in almost every area that involves computation. You will implement it (the binomial coefficient is denoted by C) and a related recurrence (denoted by B) which will use the binomial coefficient function.

$$\binom{n}{0} = \binom{n}{n} = 1, \text{for all } n \geq 0 \tag{1}$$

$$\binom{n}{k} = \binom{n-1}{k} + \binom{n-1}{k-1} \tag{2}$$

$$B_0 = 1 \tag{3}$$

$$B_n = \frac{-\sum_{k=0}^{n-1} \binom{n+1}{k} B_k}{n+1} \tag{4}$$

$$B(1) = \frac{-\sum_{k=0}^{0} \binom{2}{k} B_0}{1+1} \tag{5}$$

$$= -\frac{\binom{2}{0}1}{2} \tag{6}$$

$$= -.5 \tag{7}$$

$$B(2) = -\frac{[\sum_{k=0}^{1} \binom{3}{k} B_k]}{3} \tag{8}$$

$$= -\frac{[\binom{3}{0}B_0 + \binom{3}{1}B_1]}{3} \tag{9}$$

$$= -\frac{[1(1) + 3(-.5)]}{3} \tag{10}$$

$$= -\frac{[1 - 1.5]}{3} = .5/3 = .1\bar{6}$$

$$B(3) = -\frac{[\sum_{k=0}^{2} \binom{4}{k} B_k]}{4} \tag{11}$$

$$= -\frac{[\binom{4}{0}B_0 + \binom{4}{1}B1 + \binom{4}{2}B2]}{4} \tag{12}$$

$$= -\frac{[1(1) + 4(-.5) + 6(1.\bar{6})]}{4} \tag{13}$$

$$= -\frac{[1 - 2 + 1]}{4} = 0 \tag{14}$$

Here are some outputs:

output

```
C(1,0) = 1
C(2,0) = 1
C(2,1) = 2
C(3,0) = 1
C(3,1) = 3
C(3,2) = 3
```

```
C(4,0) = 1
C(4,1) = 4
C(4,2) = 6
C(4,3) = 4
B(0) = 1
B(1) = -0.5
B(2) = 0.16666666666666666
B(3) = -0.0
B(4) = -0.033333333333333305
B(5) = -7.401486830834377e-17
```

Programming Problem 1: Recursion

- Complete the functions C() and B().
- You won't be penalized for not using recursion but these functions can be done easily by recursion.
- You can use sum().
- *hint:* We can also write the equation-4 as list comprehension, but remember to add the entries:

```
[C(m+1, k)*B(k) for k in range(n)]
```

- Since we cannot write $\binom{x}{y}$ in Python so we will write the function as C(n, k) for binomial coefficient.
- For B( ) (equation-4), you'll need to use your binomial coefficient function.

Problem 2: Modeling Data and Judging Goodness of a Model

Years + 1900	Population $\times 10^6$
0	1650
10	1750
20	1860
30	2070
40	2300
50	2560
60	3040
70	3710
80	4450
90	5280
100	6080
110	6870

Table 1: Human Population from 1900-2010 in millions.

Exponential functions are a class of functions that find use in virtually every field. Here is the general form:

$$f(x) \quad = \quad a(b^x) \tag{15}$$

where a, b are constants. Exponential functions grow very fast–we can see it in real-life with the spread of COVID 19 infections. To get a better understanding of these kinds of functions, we'll look at the population growth of people. Table 1 has approximate values for a little over a century beginning with the earth's population of people in 1900s.

A proposed model of these data is:

$$pop(year) \quad = \quad 1436.53(1.01395)^{year} \tag{16}$$

Can we judge how good this model is? There are many ways, but the simplest is to check the difference between the value of the data and the value from the model for a particular year. This simply tells us how far away are the predictions of the model from the true population in that particular year. If the difference is not high then it means that the model is doing good otherwise if the difference is large then the model's predictions may not be that good as they tend to be far away from the actual population (as given in Table-1).

For example, in 1960, the data says the population was 3040×10^6. The first thing we can do to simplify this even more is to discard $\times 10^6$ and treat the population just as 3040. Similarly, we can subtract 1900 from the year (1960-1900), so the input to the function is 60.

We will write p_i to indicate population for that year. For example, $p_0 = 1650, p_{10} = 1750, p_{20} = 1860$ (you can cross check these values from the table above).

Let's find the absolute value of the difference of the data p_{60} and our model $pop(60)$

$$|3040 - pop(60)| \quad = \quad |3040 - 3298.428492408121| \approx 258.4284924081212 \tag{17}$$

I'm using the value of the function I've written in Python–and kept the decimal places so you can replicate it–but it doesn't contribute significantly to the overall answer. The *closer* the answer is to zero, the better the model! So we can sum all the error:

$$error \quad = \quad \sum_{i=0}^{11} |p_{i\times 10} - pop(i \times 10)| \tag{18}$$

$$= \quad |p_0 - pop(0)| + |p_{10} - pop(10)| + \ldots + |p_{110} - pop(110)| \tag{19}$$

where p_i is the population (from the data) at year $1900 + i$.

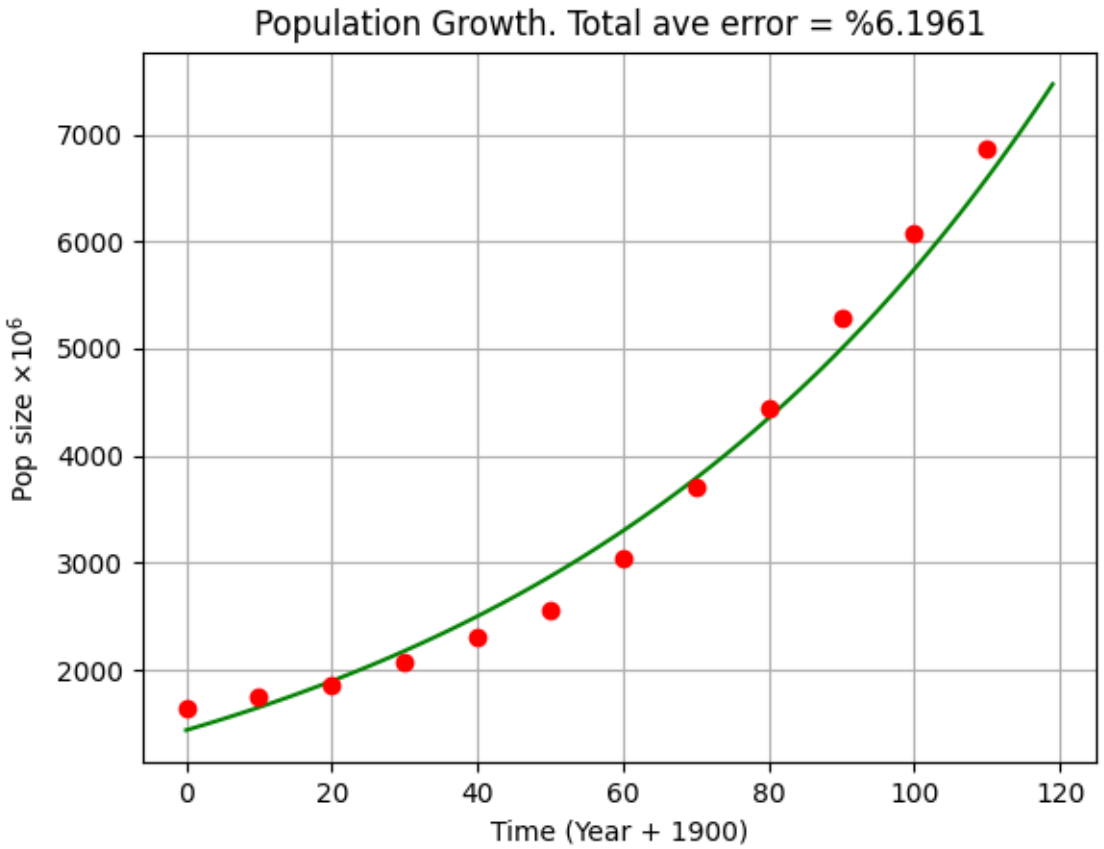

Figure 1: Plot of population growth data (blue dots) and model (green line).

Note: In the starter we have already provided the code for creaitng this plot (line 112-123). After completing all the functions, uncomment line 112-123 and notice that we are storing the output from the functions error (line 112) in a variable called total_error. The variable total_error is used for plotting afterwards. For plotting, we are using the python library 'Matplotlib' - it's already imported in the starter code and the code is already setup. You only need to ensure that the error() function return the correct output. Manually check your functions before attempting to create the plot.

Deliverables for Programming Problem 2

- Create a text file called **pop.txt** that has the same data and format as Table 1 (you can use tab for seperating the columns). You should save this file under the Assignment6 folder and then read it in the get_data() function. Remember your lab for file IO to read the file correctly.
- Complete the functions get_data(), pop() and error() that models the population growth.
- Calculate the total error described above and assign that value to the variable error.
- You'll have in the Assignment6 directory, the file 'pop.txt' and the plot (graph) should appear when we run your code.
- Adapted and extended from, *Biocalculus*, by Steward and Day.

Problem 3: Isograms

Complete the function `isogram` that takes a string and returns True if no letter of the string occurs more than once. You cannot use any built-in Python functions.

Deliverables for Programming Problem 3

- Complete the function isogram()
- An empty string is an isogram

Problem 4: Hexadecimal

Complete the function `Hex` that takes a string in hexadecimal (all capital letters) and returns the decimal equivalent as an integer. A couple of runs are shown.

Output

```
Hex: C1
193

Hex: 7DE
2014
```

Deliverables for Programming Problem 4

- Complete the function Hex().
- You *cannot* use any Python number functions to do the conversion–you must do this from scratch.
- You can use sum() and string reverse [::-1]

Problem 5: 11s

We saw a property of an integer is that if it's divisible by 9, then the sum of the digits is also divisible by 9. Interestingly, this holds true for 11 too.

Deliverables for Programming Problem 4

- Complete the function div_11().
- You *cannot* simply divide by 11 or use integer divide.

Pairs

mabdayem@iu.edu, johnslia@iu.edu
ahnabrah@iu.edu, ashankwi@iu.edu
adamsjf@iu.edu, njindra@iu.edu
dadeyeye@iu.edu, jtkrug@iu.edu
cmaguila@iu.edu, emdelph@iu.edu
ahmedrr@iu.edu, phklein@iu.edu
malshama@iu.edu, mschauss@iu.edu
olalbert@iu.edu, edepke@iu.edu
nalemanm@iu.edu, jjwelp@iu.edu
faysalza@iu.edu, timbogun@iu.edu
rnameen@iu.edu, yudsingh@iu.edu
svamin@iu.edu, mlboukal@iu.edu
jaygul@iu.edu, powelchr@iu.edu
bbacso@iu.edu, zsbanks@iu.edu
rbajaj@iu.edu, cadelaga@iu.edu
cbalbuen@iu.edu, yl181@iu.edu
ikbanist@iu.edu, msronan@iu.edu
mbarrant@iu.edu, ssalama@iu.edu
tymbarre@iu.edu, gcruzcor@iu.edu
dcblakle@iu.edu, gkyoung@iu.edu
gabradle@iu.edu, apoellab@iu.edu

logbrads@iu.edu, astrouf@iu.edu
kbburnet@iu.edu, samuwagn@iu.edu
lburrola@iu.edu, vimadhav@iu.edu
cbylciw@iu.edu, gaoxinl@iu.edu
aidcarli@iu.edu, jsm13@iu.edu
dcaspers@iu.edu, raia@iu.edu
mathchen@iu.edu, pheile@iu.edu
joecool@iu.edu, sasaluja@iu.edu
ccoriag@iu.edu, jaespin@iu.edu
jacuau@iu.edu, eliserr@iu.edu
ddahodu@iu.edu, yuljiao@iu.edu
rpdeady@iu.edu, linweix@iu.edu
shrdesai@iu.edu, jchobbs@iu.edu
shadoshi@iu.edu, camitong@iu.edu
eeconomo@iu.edu, asaokho@iu.edu
ereilar@iu.edu, mppan@iu.edu
kamdelmo@iu.edu, bgloor@iu.edu
jpenrigh@iu.edu, rosavy@iu.edu
mfanous@iu.edu, lancswar@iu.edu
nfarhat@iu.edu, vvictori@iu.edu
jayfish@iu.edu, owenaj@iu.edu
sydfoste@iu.edu, egmorley@iu.edu
ethfrago@iu.edu, vramkum@iu.edu
fraustom@iu.edu, cjohanns@iu.edu
nfrische@iu.edu, owinston@iu.edu
gillenj@iu.edu, mmansoo@iu.edu
ggivan@iu.edu, mvincen@iu.edu
noahgrah@iu.edu, ryou@iu.edu
halejd@iu.edu, divpatel@iu.edu
ehallor@iu.edu, jthurd@iu.edu
ejharms@iu.edu, chsand@iu.edu
binyhu@iu.edu, daparent@iu.edu
silmudee@iu.edu, wlegear@iu.edu
srimmadi@iu.edu, rlmcdani@iu.edu
aj110@iu.edu, sahmir@iu.edu
yjan@iu.edu, jamoya@iu.edu
gjarrold@iu.edu, kevko@iu.edu
mjerrell@iu.edu, lopezis@iu.edu
sj110@iu.edu, ansiva@iu.edu
kjwalapu@iu.edu, hdwatter@iu.edu
fkanmogn@iu.edu, lzinn@iu.edu

gkarnuta@iu.edu, apapaioa@iu.edu
akaushal@iu.edu, bolabanj@iu.edu
sskauvei@iu.edu, weidzhen@iu.edu
jk130@iu.edu, amystaff@iu.edu
tclady@iu.edu, anniye@iu.edu
jhlazar@iu.edu, jwrohn@iu.edu
jleverty@iu.edu, gtutton@iu.edu
cl101@iu.edu, ibnash@iu.edu
mlumbant@iu.edu, cy30@iu.edu
eluthra@iu.edu, ndvanbur@iu.edu
gmanisca@iu.edu, grtalley@iu.edu
pmanolis@iu.edu, sprabhak@iu.edu
remarche@iu.edu, aramo@iu.edu
tymath@iu.edu, chlzhang@iu.edu
luilmill@iu.edu, shevphil@iu.edu
mooralec@iu.edu, mdtanner@iu.edu
hnasar@iu.edu, sothor@iu.edu
dylomall@iu.edu, evewalsh@iu.edu
perkcaan@iu.edu, cadwilco@iu.edu
srpothir@iu.edu, lzinn@iu.edu
mattroac@iu.edu, dazamora@iu.edu
nps1@iu.edu, yangyuc@iu.edu
evtomak@iu.edu, actoney@iu.edu
sowvemul@iu.edu, rwan@iu.edu