COMP 2804 — Assignment 2

Due: Sunday October 17, 11:59 pm.

Assignment Policy:

- Your assignment must be submitted as a single .pdf file. Typesetting (using Latex, Word, Google docs, etc) is recommended but not required. Marks will be deducted for illegible or messy solutions. This includes but is not limited to excessive scribbling, shadows, blurry photos, messy handwriting, etc.
- **No late assignments will be accepted.**
- You are encouraged to collaborate on assignments, but at the level of discussion only. When writing your solutions, you must do so in your own words.
- Past experience has shown conclusively that those who do not put adequate effort into the assignments do not learn the material and have a probability near 1 of doing poorly on the exams (which is where most of the marks are).
- When writing your solutions, you must follow the guidelines below.
  - You must justify your answers.
  - The answers should be concise, clear and neat.
  - When presenting proofs, every step should be justified.

Question 1:

- Write your name and student number.

Question 2: You have cultivated a unique strain of tomato that is delicious in your backyard, so you decide to grow more. You have 19 plants to start (year 0), and you let them germinate naturally, which means each plant turns into 3 plants the following year. Starting in year 1 you will harvest 10 plants per year of operation. So year 1 is 10 plants, year 2, 20 plants, etc, to sell at the farmer's market. A local squirrel population finds your tomato garden in the first year and eat 21 plants a year thereafter. That means we can express the growth of our tomato garden year by year with the following recursive function.

$$f(0) = 19,$$
$$f(n) = 3 \cdot f(n-1) - 10n - 21, n \geq 1.$$

Prove that the closed form of this recursion is $f(n) = 3^n + 5n + 18, \forall n \geq 0$.
Solution: We prove this by induction.

Base Case: $f(0) = 3^0 + 5 \cdot 0 + 18 = 1 + 18 = 19$. So the base case holds.

Inductive Step: Assume that for all values $k < n$, that $f(k) = 3^k + 5k + 18$. Using the recursive definition, show that this implies that $f(n) = 3^n + 5n + 18$.

$$\begin{aligned} f(n) &= 3 \cdot f(n-1) - 10n - 21 \\ &= 3 \cdot \left(3^{n-1} + 5(n-1) + 18\right) - 10n - 21 \\ &= 3^n + 15(n-1) + 54 - 10n - 21 \\ &= 3^n + 15n - 15 + 54 - 10n - 21 \\ &= 3^n + 5n + 18 \end{aligned}$$

as required.

Question 3: Consider the following recursive algorithm and let n be a power of 3. The subroutine Process($List[i]$) takes a single character from position i in the list $List$ and does some operation to it. Determine the number of times Process($List[i]$) is called as a function of n. Be sure to justify (prove) your answer.

```
Algorithm Thirds(List, n):

    if n = 1:
        return;
    for i in range(n):
        Process(List[i]);
    Thirds(List, n/3)
```

Solution: Let $T(n)$ be the number of calls to Process($List[i]$) in the above algorithm for an input of n. Each call to Thirds($List, n$) calls Process($List[i]$) n times and then recursively calls itself with $n/3$ as a parameter. Let k be the number of times we recursively call Thirds($List, n$). Thus our recursion is:

$$\begin{aligned} T(n) &= T(n/3) + n \\ T(n) &= (T(n/9) + n/3) + n \\ T(n) &= T(n/27) + n/9 + n/3 + n \\ &\dots \\ T(n) &= T(n/3^k) + n/3^{k-1} + n/3^{k-2} + \dots n/3^0 \end{aligned}$$

This recursion ends when $n/3^k = 1$, or when $n = 3^k$, or $k = \log_3 n$. $T(1)$ makes no calls to Process($List[i]$), thus the total number of times Process($List[i]$) is called is

$$n + n/3 + n/3^2 + \dots n/3^{k-1}$$

which we can rewrite as

$$n \cdot \left(\frac{1}{3}\right)^0 + n \cdot \left(\frac{1}{3}\right)^1 + n \cdot \left(\frac{1}{3}\right)^2 + ...n \cdot \left(\frac{1}{3}\right)^{k-1}.$$

We can use the formula

$$\sum_{j=0}^{k-1} ar^j = a\left(\frac{1-r^k}{1-r}\right)$$

where in our case $a = n$ and $r = \frac{1}{3}$. Thus

$$\begin{aligned}\sum_{j=0}^{k-1} n \cdot \left(\frac{1}{3}\right)^j &= n \cdot \frac{1-\left(\frac{1}{3}\right)^k}{1-\frac{1}{3}} \\ &= n \cdot \frac{3}{2}\left(1-\left(\frac{1}{3}\right)^{\log_3 n}\right) \\ &= n \cdot \frac{3}{2}\left(1-\frac{1}{n}\right) \\ &= \frac{3}{2}(n-1)\end{aligned}$$

Therefore PROCESS($List[i]$) is called $\frac{3}{2}(n-1)$ times.

I got lazy and did not include the induction step. If you proved this by induction you got 2 bonus marks. But once you have a closed form, the technique is the same as the question above.

Question 4:

(a) Prove that there cannot be a 00-free bitstring of length n with less than $\lfloor \frac{n}{2} \rfloor$ 1's.

Solution: If there are $\lfloor \frac{n}{2} \rfloor - 1$ 1's then there are $\lfloor \frac{n}{2} \rfloor$ buckets and $\lfloor \frac{n}{2} \rfloor + 1$ 0's. By the pigeonhole principle, at least 2 0's must be together.

(b) Explain why $\sum_{i=\lfloor \frac{n}{2} \rfloor}^{n} \binom{n}{i}$ is an upper bound on the number of 00-free bitstrings of length n.

Solution: Let i be the number of 1's in the bitstring. For a bitstring of length n, i will give sufficient 1's to have a 00-free bitstrings. However $\binom{n}{i}$ gives every possible ordering of i 1's and $n-i$ 0's, even the ones with 00 in them.

(c) Assume n is even. How many 00-free bitstring of length n have exactly $\frac{n}{2}$ 1's?.

Solution: The procedure is to first write down $\frac{n}{2}$ 1's. There is 1 way to do this. Notice there are $\frac{n}{2} + 1$ places ("buckets") to put a 0. There are $\frac{n}{2}$ 0's. So we choose 1 bucket to leave empty (no 0's) and there are $\frac{n}{2} + 1$ ways to do this.

(d) Given that n is even, how many 00-free bitstring of length n where n is even have exactly $(\frac{n}{2}+1)$ 1's? $(\frac{n}{2}+2)$ 1's?

Solution: We use the same procedure as a above. If we write down$(\frac{n}{2}+1)$ 1's then there are $(\frac{n}{2}+2)$ places to write a 0. We have $(\frac{n}{2}-1)$ 0's to write. We can choose either 3 locations to not write a 0, or , $(\frac{n}{2}-1)$ places to write a 0, so there are $\binom{\frac{n}{2}+2}{3} = \binom{\frac{n}{2}+2}{\frac{n}{2}-1}$ ways to do this. If we write down $(\frac{n}{2}+2)$ 1's there are $(\frac{n}{2}+3)$ places to write one of $(\frac{n}{2}-2)$ 0's, or 5 places without a 0, so there are $\binom{\frac{n}{2}+3}{5} = \binom{\frac{n}{2}+3}{\frac{n}{2}-2}$ ways to do this.

(e) Generalizing on your answer from (d), explain why the following is true for an even value n:

$$\sum_{i=0}^{\frac{n}{2}} \binom{i+\frac{n}{2}+1}{2i+1} = f_{n+2}$$

where f_{n+2} is the $(n+2)th$ Fibonnacci number.

Solution: We've seen in class that the number of 00-free bitstrings of length n is f_{n+2}. Generalizing the answers from (d) we see that for an integer i, $0 \leq i \leq \frac{n}{2}$, that the number of 00-free bitstrings with exactly $\frac{n}{2}+i$ 1's is $\binom{i+\frac{n}{2}+1}{2i+1}$. Summing over all values of i from 0 to $\frac{n}{2}$ thus gives us the total number of 00-free bitstrings of length n.

Note: this would also work without the constraint of n being even, in which case we replace $\frac{n}{2}$ with $\lfloor\frac{n}{2}\rfloor$, and the number of 0's becomes $n-\lfloor\frac{n}{2}\rfloor-i$. At which point the sheer amount of notation starts to become (even more) distracting.

Question 5: Consider a string of characters of length n where each character is chosen from the set $\{0,1,2,3,4,5,6,7,8,9\}$. We will call these *decimal strings*.

(a) We call a decimal string 00-*free* if it does not contain two consecutive 0's. Let D_n be the number of 00-free decimal strings of length n. Determine D_1 and D_2, then express D_n in terms of D_{n-1} and D_{n-2} for $n \geq 3$.

Solution: D_1 can take on any symbol in the set. Since there are 10 symbols, $D_1 = 10$. D_2 can be any string of lenght 2 except for 00. Thus $D_2 = 99$. To determine D_n we look at all the strings of length n that do not start with 0. Since there are 9 other symbols it can start with that are not 0, we get $9 \cdot D_{n-1}$. Next we look at all bitstrings of length n that do start with 0. The next symbol cannot be a 0, thus it must take on one of the other 9 symbols. The number of these strings is thus $9 \cdot D_{n-2}$. Therefore:

$$D_n = 9 \cdot D_{n-1} + 9 \cdot D_{n-2}$$

(b) Find an expression for the number of 00-free decimal strings of length n with precisely i 0's. As in Question 4 there can be at most $n - \lfloor \frac{n}{2} \rfloor$ 0's in the string, so you may assume $i \leq n - \lfloor \frac{n}{2} \rfloor$. Hint: Look at your answer for Question 4.

Solution: If there are i 0's then there are $n - i$ digits that are not 0. That implies there are $n - i + 1$ possible positions to place a 0. The number of ways to choose locations to write 0 is $\binom{n-i+1}{i}$. The number of ways to select a digit from 1 to 9 for the other $n - i$ digits is 9^{n-i}. Thus the number of 00-free decimal strings with i 0's is

$$\binom{n-i+1}{i} \cdot 9^{n-i}$$