

1. **Claim** ( $P(n)$ ):

$$3 \sum_{i=1}^{n-1} 4^i = 4^n - 4, \quad n \geq 2.$$

**Proof by induction:**

**Base case** ( $n = 2$ ):

$$3 \sum_{i=1}^1 4^i = 3 \cdot 4 = 12, \quad 4^2 - 4 = 16 - 4 = 12.$$

Thus  $P(2)$  holds.

**Inductive step:** Assume  $P(k)$  holds for some  $k \geq 2$ :

$$3 \sum_{i=1}^{k-1} 4^i = 4^k - 4.$$

We show  $P(k+1)$ :

$$3 \sum_{i=1}^k 4^i = 3 \left( \sum_{i=1}^{k-1} 4^i \right) + 3 \cdot 4^k = (4^k - 4) + 3 \cdot 4^k = 4^{k+1} - 4.$$

Hence  $P(k+1)$  holds. By induction

$$3 \sum_{i=1}^{n-1} 4^i = 4^n - 4 \quad \forall n \geq 2.$$

2. **Claim:** Let  $a_1 = 1$  and  $a_{n+1} = 2a_n + 2^n$ . Then

$$a_n = n 2^{n-1}.$$

**Proof by induction:**

**Base case** ( $n = 1$ ):  $a_1 = 1$  and  $1 \cdot 2^0 = 1$ .

**Inductive step:** Assume  $a_k = k 2^{k-1}$ . Then

$$a_{k+1} = 2a_k + 2^k = 2(k 2^{k-1}) + 2^k = k 2^k + 2^k = (k+1) 2^k.$$

Thus  $a_{k+1} = (k+1) 2^k$ . By induction,

$$a_n = n 2^{n-1}.$$

3. **Claim:** For

$$A(m, n) = \begin{cases} 2n, & m = 0, \\ 0, & m \geq 1, n = 0, \\ 2, & m \geq 1, n = 1, \\ A(m-1, A(m, n-1)), & m \geq 1, n \geq 2, \end{cases}$$

Given  $A(1, n) = 2^n$ .

**Proof by induction on  $n$ :**

**Base case** ( $n = 1$ ):  $A(1, 1) = 2$  and  $2^1 = 2$ .

**Inductive step:** Assume  $A(1, k) = 2^k$ . Then for  $n = k + 1 \geq 2$ ,

$$A(1, k+1) = A(0, A(1, k)) = 2 \cdot A(1, k) = 2 \cdot 2^k = 2^{k+1}.$$

By induction,

$$\boxed{A(1, n) = 2^n}.$$

4. **Recursive definitions:**

(a) Positive integers *not* divisible by 4:

$$S = \{1, 2, 3\}, \quad \text{and if } n \in S, \text{ then } n + 4 \in S.$$

(b) Positive integers that are powers of 2:

$$T = \{1\}, \quad \text{and if } n \in T, \text{ then } 2n \in T.$$

$$\boxed{\text{Answer: } S = \{1, 2, 3\} \cup \{n + 4 : n \in S\}, \quad T = \{1\} \cup \{2n : n \in T\}.$$

5. **Structural induction on**

$$S : 7 \in S, x \in S \implies 2x + 3 \in S, x \in S \implies x^2 + 8 \in S.$$

*Claim:* Every member of  $S$  is a positive integer ending in digit 7.

**Definition:**  $n$  ends in 7 iff  $n \equiv 7 \pmod{10}$ .

**Base:**  $7 \equiv 7 \pmod{10}$ .

**Inductive step:** Assume  $x \equiv 7 \pmod{10}$ . Then

$$2x+3 \equiv 2 \cdot 7 + 3 = 17 \equiv 7 \pmod{10}, \quad x^2+8 \equiv 7^2+8 = 57 \equiv 7 \pmod{10}.$$

Both end in 7. Hence by structural induction the claim holds.

$$\boxed{\text{Answer: All members of } S \text{ are positive integers ending in the digit 7.}}$$

6. **Counterexample:** Find  $n \equiv 7 \pmod{10}$  not in  $S$ .

Take  $n = 27$ . If  $27 \in S$  it must come from

$$27 = 7, \quad 27 = 2x + 3, \quad 27 = x^2 + 8.$$

But  $2x + 3 = 27 \implies x = 12 \not\equiv 7 \pmod{10}$ , and  $x^2 + 8 = 27$  has no integer solution. Hence  $27 \notin S$ .

**Answer:**  $n = 27$ .

7. **Bit-strings  $B$  with**

$$001 \in B, \quad b \in B \implies 11b, 10b, 0b \in B.$$

Let  $a_n = \#\{b \in B : |b| = n\}$  for  $n \geq 2$ .

**Compute:**  $a_2 = 0, a_3 = 1$  (only “001”). For  $n \geq 4$ , each length- $n$  string arises by prefixing

$$0b \text{ (} b \text{ of length } n-1), \quad \text{or} \quad 11b \text{ or } 10b \text{ (} b \text{ of length } n-2).$$

Thus

$$a_n = a_{n-1} + 2a_{n-2}, \quad n \geq 4, \quad a_2 = 0, a_3 = 1.$$

**Answer:**  $a_2 = 0, a_3 = 1, a_n = a_{n-1} + 2a_{n-2}$  for  $n \geq 4$ .

8. **Closed form for**

$$a_n - a_{n-1} - 2a_{n-2} = 0, \quad a_2 = 0, a_3 = 1.$$

Characteristic equation:  $r^2 - r - 2 = 0 \implies r = 2, -1$ ,

$$a_n = \alpha 2^n + \beta (-1)^n,$$

$$a_2 = 0 : \quad 4\alpha + \beta = 0,$$

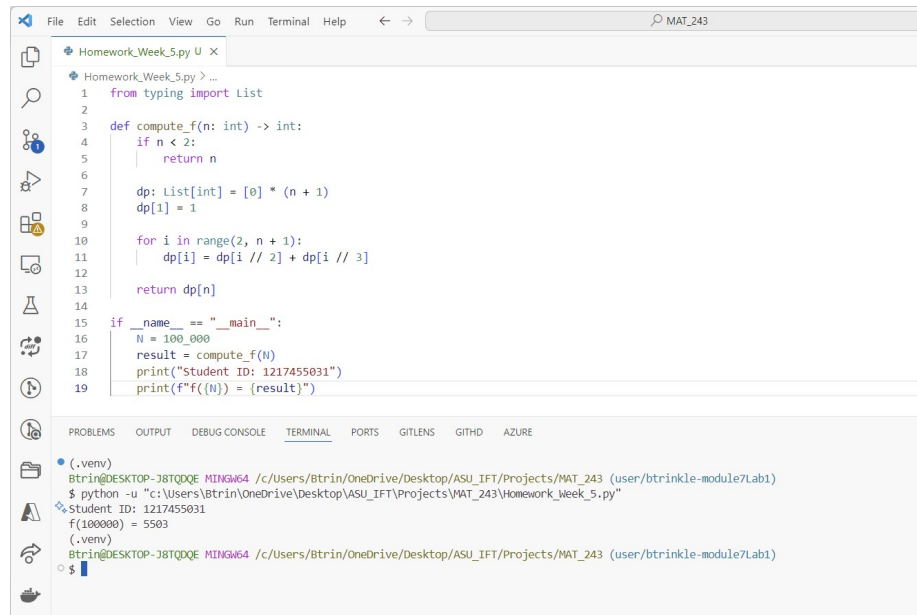
$$a_3 = 1 : \quad 8\alpha - \beta = 1, \implies \alpha = \frac{1}{12}, \beta = -\frac{1}{3}.$$

Hence

$$a_n = \frac{1}{12} 2^n - \frac{1}{3} (-1)^n.$$

**Answer:**  $a_n = \frac{1}{12} 2^n - \frac{1}{3} (-1)^n$ .

## 9. Extra Credit



The screenshot shows a Visual Studio Code editor window with a Python file named `Homework_Week_5.py`. The code implements a dynamic programming function `compute_f` to calculate the number of ways to reach a target `n` using steps of 1 or 2. The function uses a list `dp` for memoization. The main block sets `N = 100000`, calls `compute_f(N)`, and prints the result and the value of `f(N)`.

```
1 from typing import List
2
3 def compute_f(n: int) -> int:
4     if n < 2:
5         return n
6
7     dp: List[int] = [0] * (n + 1)
8     dp[1] = 1
9
10    for i in range(2, n + 1):
11        dp[i] = dp[i // 2] + dp[i // 3]
12
13    return dp[n]
14
15 if __name__ == "__main__":
16     N = 100_000
17     result = compute_f(N)
18     print("Student ID: 1217455031")
19     print(f"f({N}) = {result}")
```

The terminal output shows the execution of the script, displaying the student ID and the result of the function for `N = 100000`.

```
(.venv)
btrin@DESKTOP-38TQDQE MINGW64 /c:/Users/Btrin/OneDrive/Desktop/ASU_IFT/Projects/WAT_243 (user/btrinkle-module7Lab1)
$ python -u "c:/Users/Btrin/OneDrive/Desktop/ASU_IFT/Projects/WAT_243/Homework_Week_5.py"
Student ID: 1217455031
f(100000) = 5503
(.venv)
btrin@DESKTOP-38TQDQE MINGW64 /c:/Users/Btrin/OneDrive/Desktop/ASU_IFT/Projects/WAT_243 (user/btrinkle-module7Lab1)
$
```