# HW 2

Joshua Ortiga

Xin Li

Jonathan Le

October 17, 2022

**Problem 2.1.** Define the following sets $S_n \subset \mathbb{R}$

$$S_n = \{x | x \in \mathbb{R}, 0 < x < \tfrac{1}{n}\}.$$

Define the following set:

$$S = \bigcap_{n=1}^{\infty} S_n = S_1 \cap S_2 \cap S_3 \cap ....$$

Prove that $S = \emptyset$.

Proof by contradiction: Assume that there is a set with no intersecting values

$S_n = \{x | x \in \mathbb{R}, 0 < x < \frac{1}{n}\}$ where $\frac{1}{n}$ is the upper bound.

Since $n$ consists of all $\mathbb{R}$ from 0 to $\infty$, $\frac{1}{n}$ will approach 0 but never $= 0$.

Consider a value $j$ where $j = n + 1$ and $S_j = \{x | x \in \mathbb{R}, 0 < x < \frac{1}{j}\}$.
Since $\frac{1}{n} > \frac{1}{j}$, any value in $S_j$ would exist within $S_n$ $\forall n$ since $n$ approaches $\infty$.
$\therefore S \neq \emptyset$.

**Problem 2.2.** Let $F_n$ be the $n^{\text{th}}$ Fibonacci number. Prove that $\forall n \in \mathbb{N}$, $gcd(F_n, F_n + 1) = 1$.

In the Fibonacci sequence: $F_n + 1 = F_n + F_n - 1$.

$\therefore gcd(F_n + F_n - 1, F_n) = gcd(F_n + 1, F_n)$.

Base case: $gcd(F_1, F_2) = 1, \quad (F_1 = 1, F_2 = 1)$

Assume that: $(F_n - 1, F_n) = 1, \quad$ where $n \geq 2$.

Since: $F_n + 1 = F_n + F_n - 1$, and $gcd(F_n + F_n - 1, F_n) = gcd(F_n, F_n + 1)$

$\therefore gcd(F_n, F_n + 1) = 1 \quad \blacksquare$

**Problem 2.3.** $\forall n \in \mathbb{N}$, prove that

$$\sum_{k=1}^{n} k^3 = \left(\sum_{k=1}^{n} k\right)^2.$$

Assume: $\displaystyle\sum_{k=1}^{n} k = \frac{n(n+1)}{2}$.

Then: $\displaystyle\left(\sum_{k=1}^{n} k\right)^2 = \left(\tfrac{n(n+1)}{2}\right)^2 = \tfrac{n^2(n+1)^2}{4}$.

Now assume: $\displaystyle\sum_{k=1}^{n} k^3 = \frac{n^2(n+1)^2}{4}$.

Inductive step: Assume $\left(\frac{n(n+1)}{2}\right)^2$ to be true $\forall n$ to $(n+1)$.

Also consider that $P(n) \Rightarrow P(n+1)$.

$$P(n+1) = \left(\tfrac{n(n+1)}{2} + (n+1)\right)^2$$

$$\downarrow$$

$$\left(\tfrac{n^2+n}{2} + (n+1)\right)^2 = \left(\tfrac{n^2+3n+2}{2}\right)^2$$

$$= \tfrac{(n^2+3n+2)(n^2+3n+2)}{4} = \tfrac{n^4+3n^3+2n^2+3n^3+9n^2+6n+2n^2+6n+4}{4}$$

$$= \tfrac{n^4+6n^3+13n^2+12n+4}{4} = \tfrac{(n+1)(n+1)(n+2)(n+2)}{4} = \tfrac{(n+1)^2(n+2)^2}{4}$$

$P(n+1)$ for $\displaystyle\sum_{k=1}^{n} k^3 = \frac{n^2(n+1)^2}{4} + (n+1)^3$

$$= \tfrac{n^2(n^2+2n+1)}{4} + \tfrac{4(n+1)^3}{4} = \tfrac{n^4+2n^3+n^2}{4} + \tfrac{4(n^3+3n^2+3n+1)}{4}$$

$$= \tfrac{n^4+2n^3+n^2+4n^3+12n^2+12n+4}{4} = \tfrac{n^4+6n^3+13n^2+12n+4}{4}$$

$$= \tfrac{(n+1)(n+1)(n+2)(n+2)}{4} = \tfrac{(n+1)^2(n+2)^2}{4}$$

$\therefore$ Since $\left(\frac{n(n+1)}{2} + (n+1)\right)^2 = \frac{n^2(n+1)^2}{4} + (n+1)^3$

$$\sum_{k=1}^{n} k^3 = \left(\sum_{k=1}^{n} k\right)^2 \blacksquare$$

**Problem 2.4.** Prove that for a list of any size, the function below will return the largest item from the list.

```
if len(lst) == 0:
    return None
elif len(lst) == 1:
    return lst[0]
# lst[~0] represents the last item in the array
# equivilent to lst[-1]
maximum = maxItem(lst[:~0])
if lst[~0] < maximum:
    return maximum
else:
    return lst[~0]
```

Let the list $foo$ = [4, 3, 2, 5, 1, 6, 2].

Following the given code, $foo$ is broken up into pairs via comparing each value with the next element, starting from index 0, and returning the largest value in each pair through every recursive iteration.

Function calls
1. $maximum = maxItem([4, 3, 2, 5, 1, 6])$
2. $maximum = maxItem([4, 3, 2, 5, 1])$
3. $maximum = maxItem([4, 3, 2, 5])$
4. $maximum = maxItem([4, 3, 2])$
5. $maximum = maxItem([4, 3])$
6. $maximum = maxItem([4])$ ← Base case reached, len($foo$) = 1

Returning
1. $maximum = 4$
2. $maximum = 4$ $(3 < 4)$ ← pair
3. $maximum = 4$ $(2 < 4)$ ← pair
4. $maximum = 5$ $(5 \not< 4)$ ← pair

5. $maximum = 5$ $(1 < 5) \leftarrow$ pair
6. $maximum = 6$ $(6 \not< 5) \leftarrow$ pair

We end up with 6, the largest item in $foo$.

$\therefore$ Since $foo[\tilde{}0] < maximum$ will always check for the larger value in each pair, and return the largest of the pair per each recursive iteration. Since we are guaranteed to always reach our base case, and that the only edge case where $foo$ contains no elements is also considered, $maxItem()$ must always return the largest element in the list.
$\blacksquare$