Homework 2
(Chosen from solutions from various homeworks, thanks!)

1) If the set of stack operations included a MULTIPUSH operation, which pushes $k$ items onto the stack, would the $O(1)$ bound on the amortized cost of stack operations continue to hold?

No. If a MULTIPUSH performs in $O(k)$ runtime and there are $n$ MULTIPUSH operations, it takes $O(kn)$ time. $O(kn)/n$ is an amortized cost of $O(k)$. 

3) Suppose we perform a sequence of $n$ operations on a data structure in which the $i$th operation costs $i$ if $i$ is an exact power of 2, and 1 otherwise. Use aggregate analysis to determine the amortized cost per operation.

Powers of 2: 1, 2, 4, ... $2^{\log_2 n}$

Sum: $\sum_{i=0}^{\log_2 n} 2i = 2^{\log_2 n} + 1 - 1 = 2^{\log_2 n} - 1 = 2n$

Non-power operations: $O(n)$

Total cost of operations: $n + 2n = 3n = O(n) = O(1)$

Problems 17.1, 17.3
Problem 2.1

2.1 a) \[ \sum_{i=1}^{n} i = \frac{n(n+1)}{2} \]

Base Case: \( n = 1 \) \[ \sum_{i=1}^{1} i = 1 = \frac{1(1+1)}{2} \]

Inductive Step:
Assume \( \sum_{i=1}^{k} i = \frac{k(k+1)}{2} \) Prove \( \sum_{i=1}^{k+1} i = \frac{(k+1)(k+2)}{2} \)

\[ \sum_{i=1}^{k+1} i = \sum_{i=1}^{k} i + (k+1) \]
\[ = \frac{k(k+1)}{2} + k + 1 \quad \text{Induction Hypothesis} \]
\[ = \frac{k^2 + k + 2k + 2}{2} \]
\[ = \frac{k^2 + 3k + 2}{2} \]
\[ = \frac{(k+1)(k+2)}{2} \]

b) \[ \sum_{i=0}^{n} x^i = \frac{x^{n+1} - 1}{x - 1} \]

Base Case: \( n = 0 \) \[ \sum_{i=0}^{0} x^i = 1 = \frac{x^{0+1} - 1}{x - 1} \]

Inductive Step:
Assume \( \sum_{i=0}^{k} x^i = \frac{x^{k+1} - 1}{x - 1} \) Prove \( \sum_{i=0}^{k+1} x^i = \frac{x^{k+2} - 1}{x - 1} \)

\[ \sum_{i=0}^{k+1} x^i = \sum_{i=0}^{k} x^i + x^{k+1} \]
\[ = \frac{x^{k+1} - 1}{x - 1} + x^{k+1} \quad \text{Induction Hypothesis} \]
\[ = \frac{x^{k+2} - 1 - x^{k+1}}{x - 1} \]
\[ = \frac{x^{k+2} - x^{k+1}}{x - 1} \]
\[ = \frac{x^{k+2} - x^{k+1}}{x - 1} \]
Problem 2.2

\[ f(n) = \begin{cases} 
1 & \text{if } n = 0 \\
\sum_{i=0}^{n-1} f(i) & \text{if } n > 0
\end{cases} \]

Prove \( f(n) = 2^{n-1}, \forall n > 0 \)

Base case: \( n = 1 \)
\[ f(1) = 1 = 2^{1-1} \]

\[ f(n) = 1 + \sum_{i=0}^{n-1} f(i) \]
\[ f(n+1) = 1 + \sum_{i=0}^{n} f(i) \]

\[ f(n+1) - f(n) = 1 + \sum_{i=0}^{n} f(i) - 1 = \sum_{i=0}^{n} f(i) = f(n) \]

Which proves
\[ f(n+1) = 2f(n) = 2(2^{n-1}) = 2^n \]

Problems 2.12, 2.15

2.12

Prove: \( n! > 2^n \) for \( n \geq 4 \)

Base step: \( 4! > 2^4 \)
\[ 24 > 16 \quad (\text{True}) \]

Inductive step:
\[ (n+1)! > 2^{n+1} \quad \text{when } n = k+1 \quad (k \geq 4) \]
\[ (n+1)! = n! \cdot (n+1) \] (Definition of factorial)
\[ \geq 2^n \cdot (n+1) \quad \text{(since } n! \geq 2^k) \]
\[ = 2^n \cdot 2 \quad (n = 4) \]

Therefore, we can conclude that \( n! > 2^n \) for all integers \( n \geq 4 \).

2.15

Prove: \[ \sum_{i=1}^{k} \frac{1}{\log i} = O(n^{k+1} \log n) \] for every pos. int. \( k \).

Let \[ S_k = \frac{1}{\log 1} + \frac{1}{\log 2} + \frac{1}{\log 3} + \ldots + \frac{1}{\log n} \]
\[ \leq \log 1 + \log 2 + \log 3 + \ldots + \log n \]
\[ \leq \log n + \log n + \log n + \ldots + \log n \] (Logarithmic maximization)
\[ \leq n \cdot \log n \]
\[ = n^k \log n \]

\[ \therefore \sum_{i=1}^{k} \frac{1}{\log i} = O(n^{k+1} \log n) \]
Prove formally that the following function $f(n)$ returns $n!$ with the precondition that $n \geq 0$.

```java
static int f(int n) {
    if (n < 2) return 1;
    else return n * f(n-1);
}
```

Knowing that $n! = 1 \times 2 \times 3 \times \ldots \times (n-1) \times n$.

**Base case**

- $n = 0 \rightarrow 0! = 1$
- $n = 1 \rightarrow 1! = 1$

**From Function**

- If $n = 0$
  - returns 1
- If $n = 1$
  - returns 1
- If $n = 2$
  - returns $2 \times f(n-1) = 2 \times 1 = 2$
- If $n = 3$ or any value $\geq 3$
  - returns $n \times (n-1) \times (n-2) \times \ldots \times 1$

We can see that the function behaves the same as the factorial $n!$, with the same base case when $n = 0 \rightarrow 0! = 1$ giving $n! = 1$ when $n = 0$ and $n = 1$, otherwise $n! = (n) \times (n-1) \times \ldots \times 1$.