Problem Set 0

Name: Akshay Raman

Problem 0-1.

(a) $A \cap B = \{6, 12\}$

(b) $|A \cup B| = |\{1, 3, 6, 9, 12, 13, 15\}| = 7$

(c) $|A - B| = |\{1, 9, 13\}| = 3$

Problem 0-2.

(a) $E[X] = \frac{1*0+3*1+3*2+1*3}{8} = 1.5$

(b) $E[Y] = \frac{21*21}{36} = 12.25$

(c) $E[X + Y] = E[X] + E[Y] = 13.75$

Problem 0-3.

(a) **True.** $100 \equiv 18 \pmod 2 \iff 2|100 - 18 \iff 2|82$

(b) **False.** $100 \equiv 18 \pmod 3 \iff 3|100 - 18 \iff 3|82$

(c) **False.** $100 \equiv 18 \pmod 4 \iff 4|100 - 18 \iff 4|82$

Problem 0-4.

Proof by induction.

For all n $\in \mathbb{N}$,

$$P(n) := \sum_{i=1}^{n} i^3 = \left[\frac{n(n+1)}{2}\right]^2$$

Base Case: P(0) is true, both sides evaluate to 0.

Inductive Step: Assume P(n) is true, where n is a non-negative integer.

Then

$$\begin{aligned}
\sum_{i=1}^{n} i^3 + (n+1)^3 &= \left[\frac{n(n+1)}{2}\right]^2 + (n+1)^3 \\
&= \frac{(n+1)^2(n^2+4n+4)}{4} \\
&= \left[\frac{(n+1)(n+2)}{2}\right]^2
\end{aligned}$$

which proves P(n+1) is true.

So it follows by induction that P(n) is true for all non-negative n. ■

Problem 0-5. *Proof by induction.*

Let P(n) be the proposition that, for all $n \geq 1$, any connected graph $G = (V, E)$ with $|V| = n$ and $|E| = n - 1$ is acyclic.

Base Case: P(1) is true, since that graph only has a single vertex and no edges.

Inductive Step: Assume P(n) is true, where any connected graph G(n) with n vertices and n-1 edges is acyclic. Consider a connected graph G(n+1) with n+1 vertices and n edges. Since G(n+1) is connected, all vertices are connected by an edge. The average degree is $2n/(n + 1) < 2$. So there must be at least one vertex with degree 1. This vertex cannot be part of a cycle since nodes in a cycle require a degree ≥ 2. Removing this (weakest) vertex and the associated edge yields a graph in G(n) which is acyclic by P(n). So, P(n+1) is also true.

So it follows by induction that P(n) is true for all $n \geq 1$. ■

Problem 0-6. Submit your implementation to `alg.mit.edu`.

```python
def count_long_subarray(A):
    '''
    Input:  A     | Python Tuple of positive integers
    Output: count | number of longest increasing subarrays of A
    '''
    count = 1
    length = 1  # max size
    current = 1 #current subarray size
    for i in range(1, len(A)):
        if A[i-1] < A[i]:
            current += 1
        else:
            current = 1

        if current == length:
            count += 1
        elif current > length:
            length = current
            count = 1
    return count
```