Induction

$P(n) \equiv$ statement where n is used

- true base case $P(0)$ $P(1)$
- true inductive $P(n) \Longrightarrow P(n+1)$

all positive

$$\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$$

$$P(n) \equiv \sum_{i=1}^{n} i = \frac{n(n+1)}{2}$$

$$\underbrace{1 + \underbrace{2 + \cdots + 99}_{101} + 100}_{101}$$

$$50 \cdot \underbrace{51}_{101}$$

$$50 \cdot 101$$

$$P(n) \equiv \sum_{i=1}^{n} i = \frac{n(n+1)}{2}$$

$$P(1) \equiv \sum_{i=1}^{1} i = \frac{1(1+1)}{2}$$

$$1 = \frac{1 \cdot 2}{2} = 1 \quad \checkmark$$

$P\ k \Rightarrow P(k+1)$

$$\sum_{i=1}^{k} i = \frac{k(k+1)}{2}$$

$$\sum_{i=1}^{k} i + k+1 = \frac{k(k+1)}{2} + k+1$$

$$\sum_{i=1}^{k+1} = \frac{k(k+1)+2k+2}{2}$$

$$= \frac{k^2+k+2k+2}{2}$$

$$= \frac{k^2+3k+2}{2}$$

$$= \frac{(k+1)(k+2)}{2}$$

$\equiv P(k+1)$

$$P(n) \equiv \sum_{i=0}^{n} 2^i = 2^{n+1} - 1$$

$$P(0) \equiv 2^0 = 2^1 - 1$$

$$1 = 2 - 1 \quad \checkmark$$

$$P(k) \Rightarrow P(k+1)$$

$$P(k) \equiv \sum_{i=0}^{k} 2^i = 2^{k+1} - 1$$

$$\sum_{i=0}^{k} 2^i + 2^{k+1} = 2^{k+1} - 1 + 2^{k+1}$$

$$\sum_{i=0}^{k+1} 2^i = 2 \cdot 2^{k+1} - 1$$

$$= 2^{k+2} - 1$$

$$\sum_{i=0}^{k} 2^i + 2^{k+1} =$$

$$2^0 + 2^1 + \cdots + 2^k + 2^{k+1}$$

$$\downarrow$$

$$\Sigma$$

$$2^1 \cdot 2^{k+1} - 1$$

$$P(k+1) \equiv \sum_{i=0}^{k+1} 2^i = 2^{k+2} - 1$$

$\forall a > 1.$ $\quad P(n) = a^1 + a^2 + \cdots a^n = \sum_{i=1}^{n} a^i = \frac{a(a^n - 1)}{a - 1}$

$$P(1) \equiv a^1 = \frac{a(\cancel{a-1})}{\cancel{a-1}} = a^1 \quad \checkmark$$

$P(k) \Rightarrow P(k+1)$

$$P(k) \equiv \sum_{i=1}^{k} a^i = \frac{a(a^k - 1)}{a-1}$$

$$\boxed{\sum_{i=1}^{k} a^i} + a^{k+1}$$

$$\boxed{a^1 + a^2 + a^3 + \cdots + a^k} + a^{k+1}$$

$$\left[\sum_{i=1}^{k} a^i + a^{k+1} = \frac{a(a^k - 1)}{a-1} + a^{k+1} \right] \qquad = \frac{a^{k+2} - a}{a-1}$$

$$\left[\sum_{i=1}^{k+1} a^i = \frac{a(a^k - 1) + a^{k+1}(a-1)}{a-1} \right] \qquad \leq \frac{a(a^{k+1} - 1)}{a-1}$$

$$= \frac{a^{k+1} - a + a^{k+2} - \cancel{a^{k+1}}}{a-1}$$

$\checkmark$ $P(k+1)$

Correctness

- Sequence — happens once $x = x + 1$
- Selection — may happen if (x > 0) { x = x * 2; }
- Repetition — may happen 0 or more times

Function increment(n):

    Return n+1

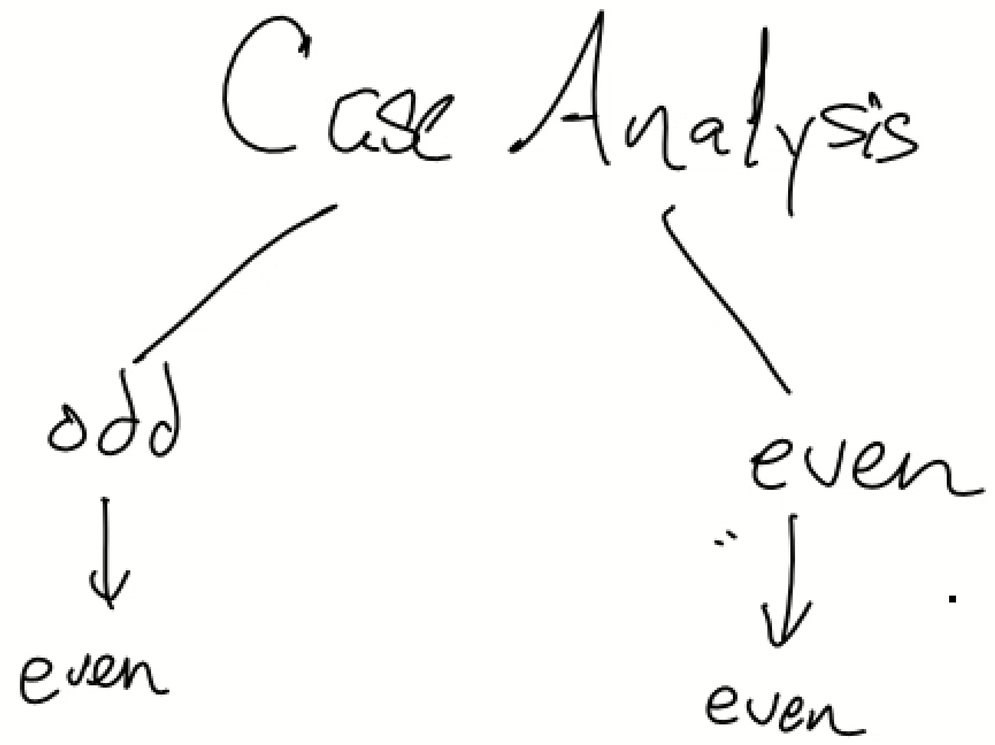
EndFunction

For any positive n, increment(n) is positive

```
Function evenize(n):
    If n is odd:
        Return n+1
    Else
        Return n-2
    EndIf
EndFunction
```

evenize returns an even number for all inputs

Case Analysis



```
Function is_sorted(A, length):
    i ← 1
    While i<length:
        If A[i]< A[i-1]:
            Return False
        i ← i+1
    End While
    Return True
EndFunction
```

Loop invariant

S(i):

The subarray from A[0] up to A[i-1] is sorted at the top of the while loop.

S(1): one el array always sorted.

S(k) ⇒ S(k+1)

First k sorted
Check el at index k
If out of order, I will return false
i increased
Return to top, where S(k+1) true