1.

	A[3]	A[2]	A[1]	A[0]
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5				
6				
7				
8				

```
Increment(A)
1  i ← 0
2  while i < length(A) and A[i] = 1
3        do A[i] ← 0
4           i = i + 1
5. if i < length(A)
6.    then A[i] ← 1
```

$$\sum_{i=0}^{\text{w.i}} \frac{n}{2^i}$$

$$< \sum_{i=0}^{\infty} \frac{n}{2^i}$$

$$< n \sum_{i=0}^{\infty} \frac{1}{2^i}$$

$$< n \frac{1}{1-\frac{1}{2}}$$

$$< 2n$$

$n = 2^k$ increment operations

$T(n) = 2n$

$T(n)/n = 2 = O(1)$

~~n/2~~

$\frac{size(T)}{2}$

$2 \cdot size(T) = size(T)$

$$c_i = \begin{cases} k+1 & \text{if } k = 2^j,\ j = 0,1,\ldots \\ 1 & \text{otherwise} \end{cases}$$

$$\sum_{i=1}^{n} 1 + \sum_{k=0}^{\lfloor \lg n \rfloor} 2^k$$

$$= n + \frac{2^{\lfloor \lg n \rfloor + 1} - 1}{2 - 1}$$

$$= n + 2 \cdot 2^{\lfloor \lg n \rfloor} - 1$$

$$< (n + 2 \cdot 2^{\lg n} - 1) + 1$$

$$< 3n = T(n)$$

```
enqueue(Q, x):
    push(S, x)
```

```
dequeue(q):
    if S' empty then
        while S not empty do
            push(S', pop(S))
    return pop(S')
```

enqueue: 1

dequeue: { non-empty S'_{i-1} : 1

empty S'_{i-1} : $|S_{i-1}|$

Use $\Phi_i = |S_i|$.

enqueue(Q, x): $a_i = c_i + \Phi_i - \Phi_{i-1} =$
$1 + (\Phi_{i-1} + 1) - \Phi_{i-1} = 2$

dequeue(Q) { S_{i-1} empty: $a_i = c_i + \Phi_i - \Phi_{i-1} = |S_{i-1}| + 1 + 0 - |S_{i-1}| = 1$

not-empty: $\Phi_i = \Phi_{i-1}$ thus
$a_i = c_i = 1$

Title: Jun 23 - 16:42 (5 of 5)