Student ID: 795622
Name: DONG GAO
Username: dongg1

1.

(a)

$$\sum_{k=1}^{n}\sum_{j=k}^{2n}1=\sum_{k=1}^{n}(2n-k+1)=\sum_{k=1}^{n}[(2n+1)-k]=\sum_{k=1}^{n}(2n+1)-\sum_{k=1}^{n}k=(2n+1)*n-\frac{n*(n+1)}{2}=\frac{3n^2+n}{2}$$

(b)

We can easily make a conclusion that:

$$\frac{3n^2+n}{2}\in\Theta\left(n^2\right)$$

2.

According to the statement, we could easily get that:

$$T(1)=0$$

$$T(n)=3T\left(\frac{n}{3}\right)+2cn(n>0\ \&\ n\neq 1)$$

$$n=3^m\Rightarrow m=\log_3 n$$

We can transform the recursive formula as follows:

$$T(n)=3T\left(\frac{n}{3}\right)+2cn(n>0\ \&\ n\neq 1)\Rightarrow\frac{T(n)}{n}=\frac{T\left(n/3\right)}{n/3}+2c$$

Let $T'(n)=\frac{T(n)}{n}$, thus $T'(n)=T'\left(\frac{n}{3}\right)+2c$, and $T'(1)=\frac{T(1)}{1}=0$

Use 3^m to replace n in the $T'(n)$, we can get that $T'\left(3^m\right)=T'\left(3^{m-1}\right)+2c$

We can use the "telescoping method" to get that:

$$T'\left(3^m\right)=T'\left(3^{m-1}\right)+2c=T'\left(3^{m-2}\right)+2c*2=T'\left(3^{m-m}\right)+2c*m=T'(1)+2cm=2cm$$

So, $T'(n)=T'\left(3^m\right)=2cm=2c*\log_3 n$

Finally, we can obtain the closed form as $T(n)=n*T'(n)=2cn*\log_3 n(\forall n>0)$

3.

(a)

There are three key points stated in the problem.

(1) This array has distinct elements.
(2) It has been sorted from smallest to largest.
(3) Last but not the least, we just determine whether an item exists rather than finding them all.

So, I thought we could solve this problem by making use of the method or structure used in binary search algorithm. Of course, there are some difference between these two algorithms.
The algorithm I have designed is as follows:

```
function INDEXNUMSEARCH (A[], lo, hi)
if lo>hi || A[lo]>hi || A[hi]<lo then return -1
mid←lo+(hi-lo)/2
if A[mid]=mid then return mid
else
   if A[mid]>mid then
      return INDEXNUMSEARCH (A, lo, mid-1)
   else return INDEXNUMSEARCH (A, mid+1, hi)
```

(b)

Because using the algorithm structure of binary search, we can easily get that the running time of my algorithm is: $t(n) = 1 + \log_2 n$.

The running time in the worst case:

$C(0) = 0$

$C(n) = C(n/2) + 1 (\forall n > 0)$

$C(n) = C(n/2) + 1 = [C(n/4) + 1] + 1 = [C(n/8) + 1] + 1 + 1 = [C(0) + 1] + 1 + ... + 1 = \log_2 n + 1$

4.

(a)

It is the queue data structure that produce the correct answer. We could exclude the stack structure by using the example in 4(b).

(b)

	a	*b*	*c*
a	0	1	1
b	1	0	0
c	1	0	0

The graph, whose adjacency matrix is given above, will cause the stack data structure to never terminate.

5.

Because all the nodes in the cyclic part of the graph have incoming edges, we cannot remove any of them. So the algorithm will stop before the graph topologically sorted. And finally we can get the cyclic part of the graph.

6.

(a)

```
function CALCULATEMI(C[][])
// the input is a two-dimension array, where C[i][j]=C(i,j)(X=x_i, Y=y_i)
        Px[0, 1,…,n-1] ←0,  Py[0,1,…,n-1] ←0
        for i←0 to n-1 do
            for j←0 to n-1 do
                Px[i] ← Px[i]+C[i][j]  //calculate the marginal probability of X_i
                Py[i] ← Py[i]+C[j][i]  //calculate the marginal probability of Y_i
        MI←0
```

```
for i←0 to n-1 do
    for j←0 to n-1 do
        MI=MI+C[i][j]*log(C[i][j]/(Px[i]*Py[j]))
Return MI
```

(b)

$$C(n) = \sum_{i=0}^{n-1} \sum_{j=0}^{n-1} 3 = 3n^2 \in \Theta(n^2)$$