Module 01 Problems

For each of problems 1, 2, and 3 below, make 1 of the following determinations:

a) Procedure A is an algorithm
b) Procedure A is NOT an algorithm
c) It is not possible to decide of procedure A is an algorithm

If you determine that situation c) is appropriate, please indicate what additional information you would need to determine if procedure A is an algorithm or not.

If you determine that a) or b) is appropriate, please state your reasoning

1. [12 pts]

```
def A(x):
    i=0
    while (True):
        x[i] = 0
        i = i+1
```

2. [12 pts]

```
def A(x):
    i=0
    while (True):
        x[i] = 0
        i = i+1
        if (i >= 10000000000):
            break
```

3. [12 pts]

```
def A(x):
    i=0
    while (i<10000000):
        x[i] = 0
        i = i+1
        external_procedure(x[i])
```

4. [12 pts] What is the computational complexity of the following program? (Big Oh notation is sufficient).
Assume that X[i,j] references the (i,j) component of matrix X.
Also, assume that N is a parameter describing the size of X as X[N,N].

```
for i in range(N):
    v[i] = 0
    for j in range(N):
        v[i] += A[i,j]*b[j]
```

Be sure to give your reasons for your complexity calculation.

5. [12 pts] What is the computational complexity of the following program? (Big Oh notation is sufficient).
Assume that X[i,j] references the (i,j) component of matrix X
Also, assume that N is a parameter describing the size of X as X[N,N]

```
for i in range(N):
    v[i] = 0
    for j in range(i:N):
        v[i] += A[i,j]*b[j]
```

Be sure to give your reasons for your complexity calculation.

6. [12 pts] What is the computational complexity of the following program? (Big Oh notation is sufficient).
Assume that X[i,j] references the (i,j) component of matrix X
Also, assume that N is a parameter describing the size of X as X[N,M]
M is a symbolic constant completely independent of N.

```
for i in range(N):
    v[i] = 0
    for j in range(M):
        v[i] += A[i,j]*b[j]
```

Be sure to give your reasons for your complexity calculation.

7. [14 pts] Find an efficient algorithm to compute xm, for arbitrary non-negative integer m.
By "efficient", the time complexity of your algorithm should be O(log2(m))
Hint: used divide and conquer.
Please include:

1) a proof sketch that your algorithm is correct,
2) the complexity of your algorithm, and how you determined the complexity.

8. [14 pts] Suppose you were given an array of numbers that were in ascending order. Some evil mastermind, however, has rearranged the list to "cycle around" from some middle point. Find an efficient algorithm to determine the original starting point from the cyclic list. By "efficient", the time complexity of your algorithm should be O(log2(N)), where N is the length of the array.
Example:
Original list: 1, 3, 4, 5, 8, 9
Your rearranged list: 5, 8, 9, 1, 3, 4

Your algorithm should identify the 4th element   as the start of the original list.

Please include:

1) a proof sketch that your algorithm is correct,
2) the complexity of your algorithm, and how you determined the complexity.