This is how to write an algorithm (Algorithm 1) for finding the maximum element in a finite sequence (Slide 14 in Class Slides).

**Algorithm 1: Max finds the maximum number**

**Input:** A finite set $A = \{a_1, a_2, \ldots, a_n\}$ of integers

**Output:** The largest element in the set

1. $\text{max} \leftarrow a_1$
2. for $i \leftarrow 2$ to $n$ do
   3. if $a_i > \text{max}$ then
      4. $\text{max} \leftarrow a_i$
3. return $\text{max}$

Algorithm 2 is a greedy change-making algorithm (Slide 19 in Class Slides).

**Algorithm 2: Change Makes change**

**Input:** A set $C = \{c_1, c_2, \ldots, c_r\}$ of denominations of coins, where $c_1 > c_2 > \ldots > c_r$ and a positive number $n$

**Output:** A list of coins $d_1, d_2, \ldots, d_k$, such that $\sum i = 1^k d_i = n$ and $k$ is minimized

1. $C \leftarrow \emptyset$
2. for $i \leftarrow 1$ to $r$ do
3.   while $n \geq c_i$ do
4.     $C \leftarrow C \cup \{c_i\}$
5.     $n \leftarrow n - c_i$
6. return $C$

Algorithm 3 and Algorithm 4 will find the first duplicate element in a sequence of integers.
Algorithm 3: FindDuplicate

**Input:** A set sequence of integers \(a_1, a_2, \ldots, a_n\)

**Output:** Location of the first value that repeats a previous value in the sequence

1. \(location \leftarrow 0\)
2. \(i \leftarrow 2\)
3. while \(i \leq n \text{ and } location = 0\) do
   4. \(j \leftarrow 1\)
   5. while \(j < i \text{ and } location = 0\) do
      6. if \(a_i = a_j\) then
         7. \(location \leftarrow i\)
      8. else
         9. \(j \leftarrow j + 1\)

Algorithm 4: FindDuplicate2

**Input:** A set sequence of integers \(a_1, a_2, \ldots, a_n\)

**Output:** Location of the first value that repeats a previous value in the sequence

1. \(location \leftarrow 0\)
2. \(i \leftarrow 2\)
3. while \(i \leq n \text{ and } location = 0\) do
   4. \(j \leftarrow 1\)
   5. while \(j < i \text{ and } location = 0\) do
      6. if \(a_i = a_j\) then
         7. \(location \leftarrow i\)
      8. else
         9. \(j \leftarrow j + 1\)