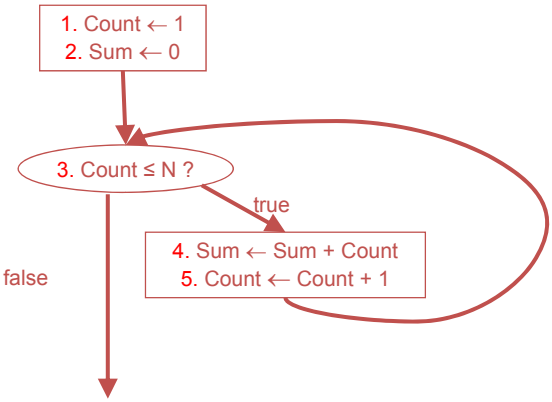
Loop Example 1: Sum from 1 to N

GIVEN:	N	(A positive integer)
INTERMEDIATE:	Count	(Counter going from 1 to N)
RESULT:	Sum	(Sum of integers 1 to N)
HEADER:	Sum ← Sum1ToN(N)	
BODY:		



1

Trace of Sum1toN(3)

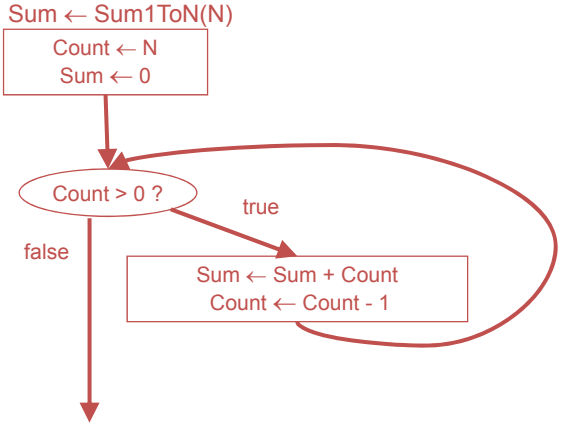
Instructions	N	Count	Sum
Init.	3	?	?
1.		1	
2.			0
3. TRUE			
4.			1
5.		2	
3. TRUE			
4.			3
5.		3	
3. TRUE			
4.			6
5.		4	
3. FALSE			

2

Sum from 1 to N: Variation (I)

GIVENS: N *(A positive integer)*
INTERMEDIATES: Count *(Counter from N to 1)*
RESUTS: Sum *(sum of integers from 1 to N)*
HEADER: Sum ← Sum1ToN(N)
BODY:

From N to 1, with intermediate counter

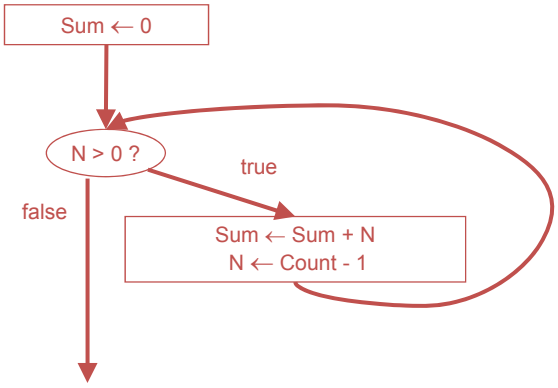


3

Sum from 1 to N: Variation (II)

GIVENS: N *(A positive integer)*
INTERMEDIATES: (none)
RESUTS: Sum *(sum of integers from 1 to N)*
HEADER: Sum ← Sum1ToN(N)
BODY:

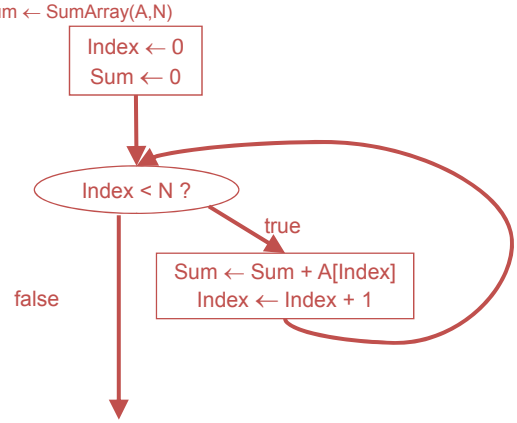
Attention:
N (given) is modified *locally* only. This approach works, but can be confusing.



4

Loop Example 2: Sum of Array Values

GIVENS:	A	(An array of numbers)
	N	(Number of array elements)
INTERMEDIATE:	Index	(Array index going from 0 to N-1)
RESULT:	Sum	(Sum of array contents)
HEADER:	Sum ← SumArray(A,N)	
BODY:		



5

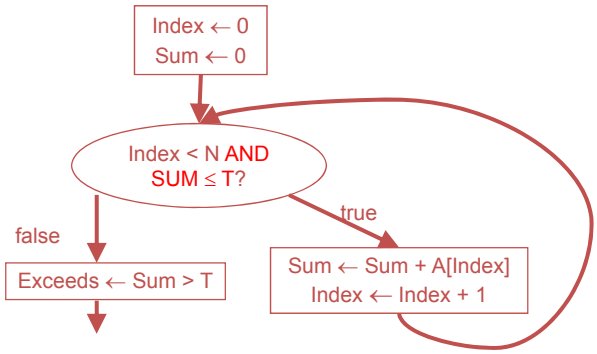
Loop Example 3a): Does sum of array values exceed T?

GIVENS:	A	(An array of numbers)
	N	(Number of array elements)
	T	(A "threshold" value)
INTERMEDIATE:	Sum	(The sum of the array, from example 2)
RESULT:	Exceeds	(Boolean: True if Sum > T and False otherwise)
HEADER:	Exceeds ← SumLargerThanT(A,N,T)	
BODY:		
	Sum ← SumArray(A,N)	
	Exceeds ← Sum > T	

6

Loop Example 3b): Does sum of array values exceed T (efficient)?

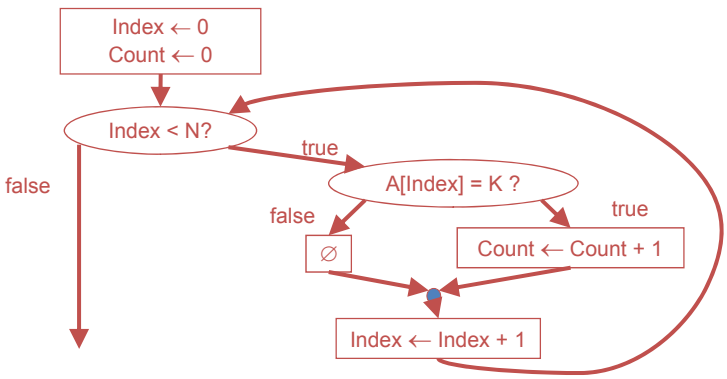
GIVENS:	A	(An array of numbers)
	N	(Number of array elements)
	T	(A "threshold" value)
INTERMEDIATE:	Index	(Array index going from 1 to N)
	Sum	(The sum of the array elements)
RESULT:	Exceeds	(A Boolean: true if T>Sum and false otherwise)
HEADER:	Exceeds ← SumLargerThanT(A,N,T)	
BODY:		



7

Loop Example 4: Count instances of K

GIVENS:	A	(An array of numbers)
	N	(Number of array elements)
	K	(Value for which to count instances)
INTERMEDIATES:	Index	(Array index going from 0 to N-1)
RESULT:	Count	(Number of times value of K is contained in A)
HEADER:	Count ← CountK(A,N,K)	
BODY:		



8

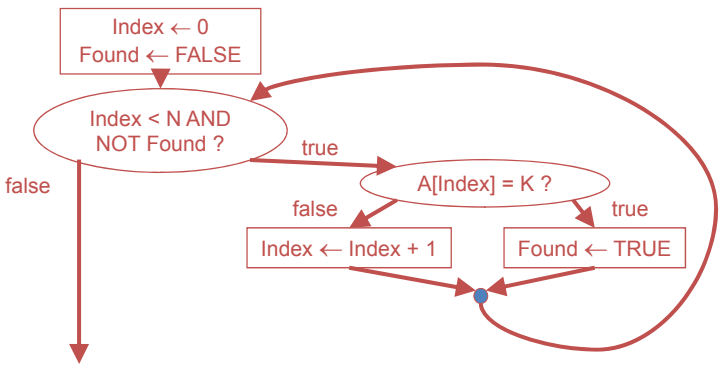
Loop Example 5a)
Is K in an array?

GIVENS:	A	(An array of numbers)
	N	(Number of array elements)
	K	(Value to find)
INTERMEDIATES:	Count	(Number of times K is in array, from algorithm 4)
RESULT:	Found	(Boolean : true if K is in array and false otherwise)
HEADER:	Found ← FindK(A,N,K)	
BODY:		
	Count ← CountK(A,N,K)	
	Found ← Count > 0	

9

Loop Example 5b)
Is K in an array?

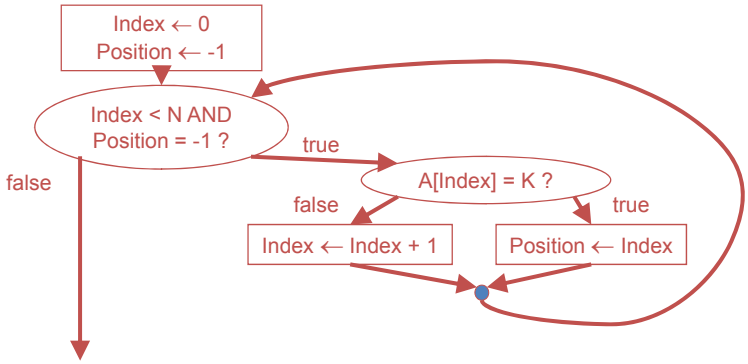
GIVENS:	A	(An array of numbers)
	N	(Number of array elements)
	K	(Value to find)
INTERMEDIATES:	Index	(Array index going from 1 to N)
RESULT:	Found	(Boolean : true if K is in array and false otherwise)
HEADER:	Found ← FindK(A,N,K)	
BODY:		



10

Loop Example 6: Find K's position in an array

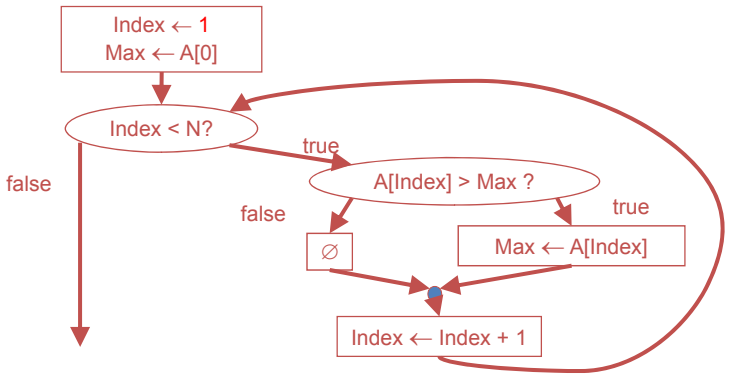
GIVENS:	A	(An array of numbers)
	N	(Number of array elements)
	K	(Value to find)
INTERMEDIATES:	Index	(Array index going from 1 to N)
RESULT:	Position	(Position of K in array, or -1 if K is not contained in array)
HEADER:	Position ← WhereIsK(A,N,K)	
BODY:		



11

Loop Example 7: Find Maximum Value in Array

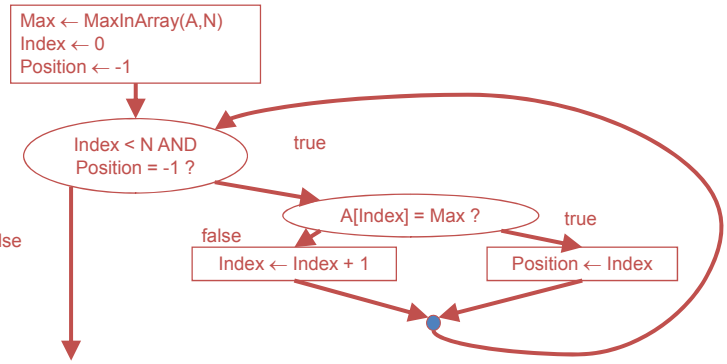
GIVENS:	A	(An array of numbers)
	N	(Number of array elements)
INTERMEDIATES:	Index	(Array index going from 1 to N)
RESULT:	Max	(Maximum value contained in A)
HEADER:	Max ← MaxInArray(A,N)	
BODY:		



12

Loop Example 8a): Find Position of Maximum Value in Array

GIVENS:	A	(An array of numbers)
	N	(Number of array elements)
INTERMEDIATES:	Index	(Array index going from 1 to N)
	Max	(Maximum value contained in A)
RESULTS:	Position	(Position of Maximum value contained in A)
HEADER:	Position ← MaxPosInArray(A,N)	
BODY:		



13

Loop Example 8b): Find Position of Maximum Value in Array

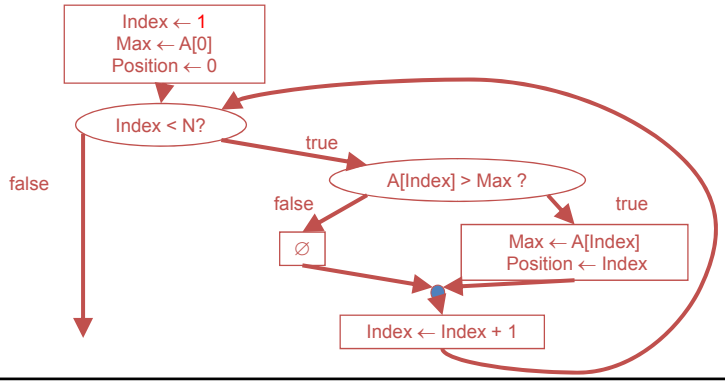
GIVENS:	A	(An array of numbers)
	N	(Number of array elements)
INTERMEDIATES:	Max	(Maximum in array from example 7)
RESULT:	Position	(Position of Maximum value contained in A)
HEADER:	Position ← MaxPosInArray(A,N)	
BODY:		

Max ← MaxInArray(A,N)
Position ← WhereIsK(A,N,Max)

14

Loop Example 8c): Find Position of Maximum Value in Array

GIVENS:	A	(An array of numbers)
	N	(Number of array elements)
INTERMEDIATES:	Index	(Array index going from 1 to N)
	Max	(Maximum value contained in A)
RESULTS:	Position	(Position of Maximum value contained in A)
HEADER:	Position ← MaxPosInArray(A,N)	
BODY:		



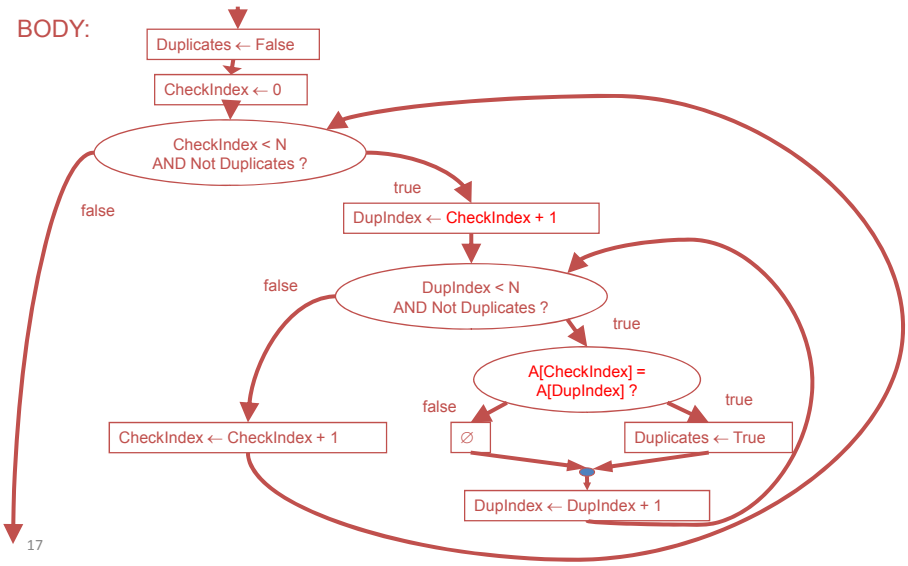
15

Loop Example 9): Determine if array contains duplicate values

GIVENS:	A	(An array of numbers)
	N	(Number of array elements)
INTERMEDIATES:	CheckIndex	(Array index for current element)
	DupIndex	(Array index for duplicate checking of current element)
RESULT:	Duplicates	(True if there are duplicates in A and false otherwise)
HEADER:	Duplicates ← HasDuplicates(A,N)	
BODY:		

16

Loop Example 9): Determine if array contains duplicate values



17

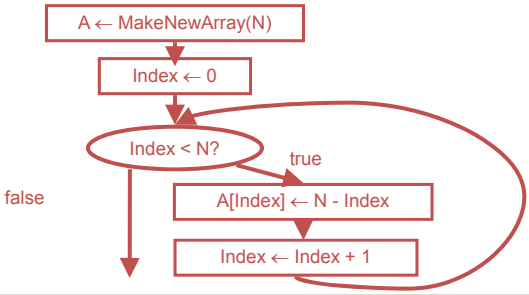
Creating an Array

- Create an array containing the integers 1 to N in reverse order.
- Remember the standard algorithm

  anArrayRefVar ← MakeNewArray(L)

  that creates an array referenced by anArrayRefVar; the array has L positions with unknown values in them.

GIVENS: N (a positive integer)
RESULTS: A (an reference to an array containing N…1)
INTERMEDIATES: Index (an index for A)
HEADER: A ← CreateNDownTo1(N)
BODY:



18

Trace for N=3

statements	N	Index	A
initial values	3	?	?
1. A ← MakeArray(3)			{?, ?, ?}
2. Index ← 0		0	
3. Index < N? true			
4. A[Index] ← N – Index			{3,?,?}
5. Index ← Index + 1		1	
3. Index < N? true			
4. A[Index] ← N – Index			{3,2,?}
5. Index ← Index + 1		2	
3. Index < N? true			
4. A[Index] ← N – Index			{3,2,1}
5. Index ← Index + 1		3	
3. Index < N? false			

19

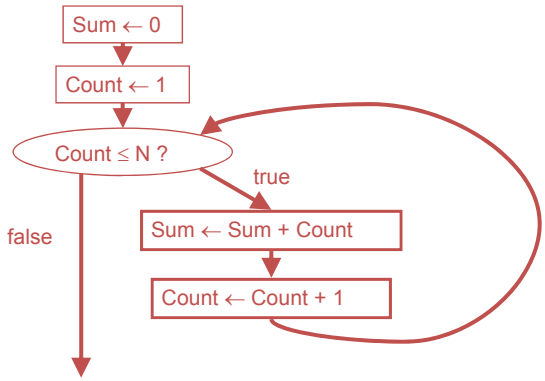
Translate to Program

```
import java.io.* ;

class Sum1ToN
{
   public static int sum1ToN (int n)
   {
      // initialize counter and accumulator sum
      int count = 1;
      int sum = 0;
      // loop
      while (count <= n)
      {
         sum = sum + count;
         count = count + 1;
      }
      // return the result
      return(sum);
   }
}
```

20

FOR loop to add 1 to N



- Translate:

```
sum = 0 ;
for ( count = 1; count <= n; count = count + 1 )
{
  sum = sum + count ;
}
```

21

Translated to a Program

```
import java.io.* ;
class MaxInArray
{
   public static int maxInArray (int [] a)
   {
      // DATA DICTIONARY

      // GIVEN: a          An array of integers
      int max;         // RESULT:         The maximum in the array
      int n;           // INTERMEDIATES:  The length of array a
      int index;       //                 Index into array a

      // Initialise n

      n = a.length;
```

22

Translated to a Program

```
      // ALGORITHM BODY
      max = a[0];
      index = 1;
      while (index < n)
      {
         if (a[index] > max)
         {
            max = a[index];   // update max
         }
         else
         {
            ; // do nothing
         }
         index = index + 1;
      }

      // Return results
      return(max);
   }
}
```

23

Comparing Strings

- What is the value of **`result`** for these examples?
  - Example 1:

```
String str1 = "abcde" ;
String str2 = "abcfg" ;
int result = str1.compareTo(str2);
```

  - Example 2:

```
String str1 = "abcde" ;
String str2 = "ab" ;
int result = str1.compareTo(str2);
```

  - **`result`** is greater than zero

24