Csc2010 Lab 4
Lab experience 3

• Animation of max finding.
• Sort animator in invitation.jar
On the sixth line of the Find Largest algorithm of Figure 2.14 there is an instruction that reads,
While \((i \leq n)\) do

Explain exactly what would happen if we changed that instruction to read as follows:

a. While \((i \geq n)\) do
b. While \((i < n)\) do
c. While \((i = n)\) do
Solution

a) If \( n \leq 2 \), then the test would be true, so the loop would be executed. In fact, the test would never become false. Thus the algorithm would either loop forever, or generate an error when referring to an invalid \( A_i \) value. If \( n > 2 \), then the test would be false the first time through, so the loop would be skipped and \( A_1 \) would be reported as the largest value.

b) The algorithm would find the largest of the first \( n - 1 \) elements and would not look at the last element, as the loop would exit when \( i = n \).

c) For \( n = 2 \) the loop would execute once, comparing the \( A_1 \) and \( A_2 \) values. Then the loop would quit on the next pass, returning the larger of the first two values. For any other value of \( n \), the loop would be skipped, reporting \( A_1 \) as the largest value.
C2.Q18

a. Refer to the pattern-matching algorithm in Figure 2.16. What is the output of the algorithm as it currently stands if our text is

Text: We must band together and handle adversity and we search for the pattern “and”?

b. How could we modify the algorithm so that it finds only the complete word and rather than the occurrence of the character sequence a, n, and d that are contained within another word, such as band?
Solution

(a) The algorithm will find the three occurrences of *and*. First in the word *band*, second in the word *and*, and third in the word *handle*.

(b) We could search for “and”. That is, the word itself surrounded by spaces. Note that the word "and" is special in that it is almost always surrounded by spaces in a sentence. Other words may start or end sentences and be followed by punctuation.
C2. Q20

Design an algorithm that is given a positive integer N and determines whether N is a prime number, that is, not evenly divisible by any value other than 1 and itself. The output of your algorithm is either the message “not prime,” along with a factor of N, or the message “prime.”
Step 1: Get the value for $N$
Step 2: Set the value of $i$ to 2
Step 3: Set the value of $R$ to 1;
Step 4: While ($i < N$ and $R \neq 0$) do Steps 5-6
Step 5: Set $R$ to the remainder upon computing $N/i$
Step 6: Set the value of $i$ to $i + 1$
Step 7: If $R = 0$ then
   Print the message "not prime"
   Else
   Print the message "prime"
   (This algorithm could be improved upon because it is enough to look for divisors of $N$ less than or equal to $\cdot$)
Write an algorithm to read in a sequence of values $V \geq 0$, one at a time, and determine if the list contains at least one adjacent pair of values that are identical. The end of the entire list is marked by the special value $V=-1$. For example, if you were given the following input:

$$14, 3, 7, 7, 9, 1, 804, 22, -1$$

The output of your algorithm should be a ‘Yes’ because there is at least one pair of adjacent numbers that are equal (the 7s). However, given the following input:

$$14, 3, 7, 77, 9, 1, 804, 22, -1$$

The output of your algorithm should be a ‘No’ because there are no adjacent pairs that are equal. You may assume in your solution that there are at least two numbers in your list.
Solution

Set adjacent to NO
Get values for V1 and V2    //We can do this since we know
    //there are at least 2 values
While (V2 ≠ -1) AND (adjacent = NO)
    If V1 = V2
        Set adjacent to YES
    Else
        Set V1 = V2
        Get a new value for V2
End
Print the value of adjacent
Stop