Flowcharts

- Flowcharts are used to show the flow of control of an algorithm as it runs step by step in a visual manner.

- Flowcharts consist of the following components:
  - An oval labeled START
  - A sequence of boxes with algorithm operations
  - Arrows that indicate the order that the boxes are evaluated in the algorithm
  - A oval labeled STOP or END
Sequential Flow

- An assignment operation sets a variable to a specific value or changes a variable to a new value.
  - Represented in a flowchart using a rectangle with the assignment operation described inside.
  - Example:
    
    ![Assignment Operation Diagram]

Input

- An input operation sets a variable to a data value given by the user of the algorithm.
  - Represented in a flowchart using a rhombus with a small arrow pointing in from the side.
  - The contents of the rhombus include the variable being initialized.
  - Example:
    
    ![Input Operation Diagram]
Output

- An output operation displays the data value in a variable or a message to the user of the algorithm.
  - Represented in a flowchart using a rhombus with a small arrow pointing out from the side
  - The contents of the rhombus include the variable or message being displayed
  - Examples:

Example

- Algorithm to find the roots of a quadratic equation:
  \[ ax^2 + bx + c = 0 \]

\[
\text{Set root1} = \frac{-b - (b^2 - 4ac)^{1/2}}{2a} \\
\text{Set root2} = \frac{-b + (b^2 - 4ac)^{1/2}}{2a} \\
\text{Output root1, root2} \\
\text{Stop}
\]
Conditional Operations

- A conditional operation determines the direction of flow in an algorithm
  - Represented in a flowchart using a diamond with a test condition inside
  - Example:

```
X > 100
  yes
    no
```

Example

- Revised algorithm to find the roots of a quadratic equation $ax^2 + bx + c = 0$

```
b^2 - 4ac \geq 0
  yes
  no
    Set root1 = (-b - (b^2 - 4ac)^{1/2}) / (2a)
    Set root2 = (-b + (b^2 - 4ac)^{1/2}) / (2a)
    Output “No real roots”
  Output root1, root2
  Stop
```
Loop Operations

- A loop operation indicates the sequence of operations that are to be repeated along with a condition to control the number of repetitions
  - Represented in a flowchart using a diamond with a test condition inside
  - Includes one or more flowchart operations with an arrow that flows back to an earlier point in the flow of the algorithm

Loop Operations (cont’d)

- Example:

```
<table>
<thead>
<tr>
<th>Operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set x = 0</td>
</tr>
<tr>
<td>Set y = 1</td>
</tr>
<tr>
<td>y &gt; 10</td>
</tr>
<tr>
<td>no</td>
</tr>
<tr>
<td>Add y to x</td>
</tr>
<tr>
<td>Add 1 to y</td>
</tr>
<tr>
<td>Output x</td>
</tr>
</tbody>
</table>
```

BODY OF THE LOOP
Loop Operations (cont’d)

- Example:
  
  - Set $x = 0$
  - Set $y = 1$
  - Add $y$ to $x$
  - Add 1 to $y$
  - Output $x$

- Example:
  
  - Set $p = 1$
  - Do the following 10 times:
    - Multiply $p$ by 2
  - Output $p$

  Add an additional unused variable to control the number of times the loop repeats.
Bubble Sort (Original Algorithm)

1. Input n
2. Input a vector of values A[0], A[1], ..., A[n-1]
3. Do the following n times:
   a. Let i = 0
   b. Do the following n-1 times:
      i. If A[i] > A[i+1], exchange A[i] and A[i+1]
      ii. Add 1 to i
4. Output A
Bubble Sort (Original Algorithm)

1. Input n
2. Input a vector of values A[0], A[1], ..., A[n-1]
3. Let x = 0
4. Do the following while x is not equal to n:
   a. Let i = 0
   b. Do the following n-1 times:
      i. If A[i] > A[i+1], exchange A[i] and A[i+1]
      ii. Add 1 to i
   c. Add 1 to x
5. Output A
### Bubble Sort

*(Original Algorithm)*

1. **Input** \( n \)
2. **Input** a vector of values \( A[0], A[1], \ldots, A[n-1] \)
3. Set \( i = 0 \)
4. Set \( \text{max} = A[0] \)
5. Do the following \( n-1 \) times:
   a. Add 1 to \( i \)
   b. If \( A[i] > \text{max} \), set \( \text{max} = A[i] \)
6. Output \( \text{max} \)

### Exercise

Draw a flowchart for the following algorithm:

1. Input \( n \)
2. Input a vector of values \( A[0], A[1], \ldots, A[n-1] \)
3. Set \( i = 0 \)
4. Set \( \text{max} = A[0] \)
5. Do the following \( n-1 \) times:
   a. Add 1 to \( i \)
   b. If \( A[i] > \text{max} \), set \( \text{max} = A[i] \)
6. Output \( \text{max} \)
Subroutines

- An operation that is executed a number of times at different places in your algorithm can be extracted out and made into a subroutine.
- A call to a subroutine is represented by a rectangle with a wide arrow to its right.
- A separate flowchart for the subroutine is written in the same way as the initial “main” flowchart that describes the algorithm.

Bubble Sort
(Original Algorithm)

Start

Input n

Input A[0],...,A[n-1]

Set x = 0

Set i = 0

Set y = 0

y = n-1

A[i] > A[i+1]

Exchange A[i] and A[i+1]

Add 1 to i

Add 1 to y

Add 1 to x

x = n

Output A

Stop

Extract this out as a subchart.
Bubble Sort (Original Algorithm)
Main flowchart

1. Start
2. Input n
3. Input A[0].....A[n-1]
4. Set x = 0
5. Add 1 to x
6. Call BubbleUp
7. x = n
8. yes -> Output A
   no -> Add 1 to x
9. Stop

Bubble Sort (Original Algorithm)
Subchart

1. Start
2. BubbleUp
3. Set i = 0
4. Set y = 0
5. y = n-1
6. yes -> Add 1 to y
   no -> A[i] > A[i+1]
7. no -> A[i] < A[i+1]
8. yes -> Exchange A[i] and A[i+1]
9. no -> Add 1 to i
10. Stop