Reppetition

T. M. Smith, Ph.D.
Professor, Computer Studies
Austin Community College
I. Structural Programming
   1. Sequence
      A. Input
      B. Processing Expressions
      C. Output
      D. Functions
   2. Selection
   3. Repetition

II. Object-oriented Programming
   1. Objects
   2. Events

III. Form Validation
DATA TYPES: REVIEW
Data Types

- Definition
  - A specific category of information stored in a variable in the computer’s memory.

- Primitive Data Types
  - Data types to which programmers can assign only one value.

- Complex Data Types
  - Data types to which programmers can assign multiple values.
Primitive Data Types

- Primitive Data Types
  - Numeric Data Type
    - Integer
    - Floating Point
  - String – text characters.
  - Boolean – logical value of true or false.
  - Null – an empty value.
ARRAYS
Complex Data Type: Arrays

- Definition
  - **Array** - A set of data represented by a single variable name.

- Characteristics
  - Visualize the structure of an array as a single column spreadsheet.
    - Each row holds a distinct piece of data.
    - Spreadsheet numbers each row.
    - Row numbers are in numerical sequence.
  - **JS** array uses **zero** as the first element (row).
### Array: Visual Examples

<table>
<thead>
<tr>
<th>Element #</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>123</td>
</tr>
<tr>
<td>1</td>
<td>234</td>
</tr>
<tr>
<td>2</td>
<td>567</td>
</tr>
<tr>
<td>3</td>
<td>890</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Element #</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>&quot;Smith&quot;</td>
</tr>
<tr>
<td>1</td>
<td>&quot;Habib&quot;</td>
</tr>
<tr>
<td>2</td>
<td>&quot;Woerner&quot;</td>
</tr>
<tr>
<td>3</td>
<td>&quot;Mendez&quot;</td>
</tr>
</tbody>
</table>
Array

 ➢ Characteristics

  ❖ An array is a JS object.
  ❖ Arrays belong to the core JS language rather than the Document Object Model.
  ❖ To create an array → assign the new array object to a variable.
  ❖ Element – each piece of data contained in an array.
  ❖ Arrays can store a variety of data types in the array elements.
Create an Array Object

- Rules
  - Keyword – `new` – precedes a call to the *JS* function that generates arrays.
  - `new` creates space in memory for the array.

- Syntax
  
  ```javascript
  variable_name = new Array(# of elements);
  ```

  - Note: *Array()* object receives a single argument representing the number of elements to be contained in the array.
Array Object

- Rules
  - **Index** number is the number of the element.
  - **JS** element numbers start with 0.
  - Declaring the number of array elements in **JS** is optional.
    - ✓ Note: this is not valid in other programming languages.
  - **JS** can create the array without any elements and add new elements to the array as necessary.
  - **JS** can change the size of an array dynamically.
  - If a script assigns a value to an uncreated element in an array, **JS** will automatically create the element, along with any uncreated elements that precede it.
Array Object

Example

```javascript
const car = new Array(3); // creates an array.
```

- Note the parentheses ( ) to declare the array.
- Note the brackets [ ] to specify an element of the array.

Rule

- Array elements which JS has created but has not assigned values have the null value.

Example

```javascript
const car = new Array(3);
```

- At this point, the car array is in memory analogous to a one column, three row table with no data assigned to each cell.
Assign Values to Array Elements

- **Rules**
  - Use same method as assigning values to variables.
  - Include the index for an individual element of the array.

- **Example**
  ```javascript
  car = new Array(3);  //creates array
  car[0] = "Lexus";
  car[1] = "BMW";
  car[2] = "Mercedes Benz";
  ```
  - Script can assign values to an array when JS first creates the array.

- **Example**
  ```javascript
  scores = new Array(78,56,93);
  document.write(scores[1]);  //displays 56
  ```
Rule

- To reference a specific array element – enclose its index number in brackets at the end of the array name.

Example

```javascript
document.writeln(car[2]); // displays "Mercedes Benz"
```

- A script can change the element by assigning a new value to any indexed element in the array.

Application

- If a programmer wants to include a table of information in a document from which a script can look up information without accessing the server, then the programmer includes the data in the document in the form of an array.
Array Object Property

- **length Property**
  - Reflects the number of entries in the array.
  - Note: Because array index values are zero-based, the index of the last cell of an array is one less than the length.
Repition
Repetition

Definition

- A control structure that repeats a set of instructions:
  - a set number of times (count-controlled loop)
    - OR
  - while a condition remains true (event-controlled loop)
- Programmers also call repetition a **looping structure**.
- Programmers call each repetition an **iteration**.

Characteristics

- One instruction set processes multiple data sets.
- Programmers include a counter variable with the looping structure to track the progress of repetition so that the loop ends when the program meets a condition.
- Programmer can nest selection structures, if needed.
Infinite Loop

Definition

- Infinite loop – a program situation in which a loop statement never ends because its conditional never updates or the terminal condition never ends the loop.
- Infinite loops are BIG programming errors.

Rule

- To terminate a loop using the Windows OS press CTRL+ALT+DELETE to access the Task Manager.
- Windows 7: CTRL+SHIFT+ESC to directly access the Task Manager.
Types of Repetition Statements

- Repetition Statements
  - while
  - do...while
  - for
  - continue
  - break
Event-controlled vs. Count-controlled Repetition

- Event-controlled (Indeterminate) Repetition
  - Some event occurs that causes the loop to stop.
  - Event typically is user related, i.e., user input.
  - JavaScript statements: `while`, `do...while`

- Count-controlled (Determinate) Repetition
  - Loop continues until a count variable reaches a specific value.
  - JavaScript statement: `for`

- JavaScript statements that affect a loop execution:
  - `continue`
  - `break`
COUNT-CONTROLLED REPETITION
Controlling Loops

Components of a Count-controlled Loop

- **Initialization**
  - Program assigns an initial value to the loop variable.

- **Condition Evaluation**
  - Program evaluates a condition to determine whether the loop iterates.

- **Alteration**
  - Program changes the loop variable so that the program evaluates the condition differently causing the loop to terminate.
  - If the program does not change the loop variable, then the result could be an infinite loop.
Counter Variable

- **Definition**
  - A *counter* is a variable that increments or decrements with each iteration of a loop statement.

- **Rules**
  - Program can track the progress of a loop with a counter variable.
  - Use variable names such as *count*, or *counter* as the counter variable.
for Statement

Characteristics

- Use for a count-controlled loop.
- Performs essentially the same purpose as the `while` statement.
- `for` statement is a better choice to use for count-controlled loops than a `while` statement.
- ITSE 1411: Always use the `for` statement for count-controlled loops!

Syntax

```javascript
for (initialization_expression; condition; update_statement) {
    JS statement(s);
}
```

- Can omit any of the three parts of `for` statement, but must include semicolons that separate each section.
for Statement

- Operation
  - Start the initialization expression.
    ```java
    count=1 //declare variable count, assign value.
    ```
  - Evaluate condition.
    ```java
    count<=10
    ```
    - If condition is true, then execute statements.
    - If condition is false, then exit for structure.
  - Execute update statement.
    ```java
    ++count
    ```

- Example
  ```java
  for (count=1; count<=10; ++count) 
  document.writeln(count);
  ```
for Statement

Example Web Pages

for Statement 1 (for_1.htm)
for Statement 2 (for_2.htm)
for Statement 3 (for_3.htm)
Purpose

- Use to halt a loop iteration and then to restart the loop with a new iteration.
- Use to stop the loop for the current iteration, but the programmer wants the loop to continue with a new iteration.

Example Web Page

continue Statement 1 (continue_1.htm)
break Statement

- Purpose
  - Use `break` statement within `for` statement to end the loop before its terminal condition occurs.

- Example Web Page
  - `break Statement 1` (break_1.htm)
USING while STATEMENT FOR COUNT-CONTROLLED REPETITION
while Statement

- Characteristics
  - Repeats a set of statements as long as a given conditional expression evaluates to \texttt{true}.
  - As long as the conditional expression evaluates to \texttt{true}, the statement or command block that follows will execute repeatedly.
  - Once the conditional expression evaluates to \texttt{false}, the loop ends and the next statement following the \texttt{while} statement executes.
  - To end the \texttt{while} statement once \texttt{JS} executes the desired tasks, include code that tracks the progress of the loop and changes the value produced by the conditional expression.
  - \texttt{while} loop is an example of a \texttt{pretest} loop because the condition is at the top of the loop.
while Loop Flow Chart

Do While Loop

Start

Condition?

True

End

False

Do Task
Program displays “Bettlejuice” 3 times.
while Statement: Syntax

- Syntax
  \[
  \textbf{while} \ (\textit{conditional expression}) \ \{ \\
  \hspace{1em} \textit{JS statements(s)}; \\
  \}
  \]
  - Enclose the conditional expression within parentheses.
  - Indent the statements.
  - Be sure to type the open brace \{ and closing brace \}. 

Repetition
while Statement: Count-controlled Loop

- Example
  ```javascript
  var count = 1;
  while (count <= 5) {
    // JS statement(s);
    ++count;
  }
  ```

- Example Web Page (count-controlled loop)
  [while Statement 1](while_1.htm)

- Note:
  - This example shows that a programmer can use either the `for` or `while` statements to execute a count-controlled loop.
  - **ITSE 1411:** Always use the `for` statement for count-controlled loops because it is easier to write and read the code.
Event-controlled Repetition
Conditions, Counters, & Sentinel Values

- Loop variable **usually** increments by one (1).
  - Loop variable can increment by any value.
- **Decrementing**
  - Subtracting values from a loop variable.
- **Determinate (count-controlled) loop**
  - Loop executes a known number of times.
- **Indeterminate (event-controlled) loop**
  - Number of iterations unknown ahead of time.
  - Use a priming prompt
    - Question to determine whether the loop repeats.
Sentinel Value

- Sentinel value
  - Special value that signals the end of input.
  - Can’t be used as a valid data value.
  - Example Pseudocode:

```plaintext
Declare Numeric invoiceNum
Display "Enter an invoice number or -1 to quit: "
Input invoiceNum
While invoiceNum != -1
    [more processing statements]
    Display "Enter another invoice number or -1 to quit: "
    Input invoiceNum
End While
```
Confirm Dialog Box

- Prompt Dialog Box (Review)
  - Programmers use the **prompt** dialog box to obtain user input.

- Confirm Dialog Box
  - Programmers use the **confirm** dialog box when they want the user to verify or accept something.
  - When a confirm box displays, the user must click either "**OK**" or "**Cancel**" to proceed.
  - When the user clicks "**OK**", the box returns the Boolean value: **true**. When the user clicks "**Cancel**", the box returns the Boolean value: **false**.
  - **confirm()** is a method of the **window** object.

- Syntax
  
  ```javascript
  window.confirm("sometext");
  ```

- Example Web Page
  - [Confirm Dialog Box](confirm_1.htm)
Event-Controlled Loop with **while** Statement

- Example Web Page
  
  **while** Statement – 2 (while_2.htm)
do...while Statement

- Characteristics
  - Executes a statement once, then repeats the execution as long as a given conditional expression evaluates to true.
  - Posttest loop which executes at least once.
do-while Example Pseudocode

Do
    Display "Enter the first number: "
    Input num1
    Display "Enter the second number: "
    Input num2
    total = num1 + num2
    Display "Sum of the numbers: " + total
    Display "More numbers to add (Y/N)? "
    Input more
    While more == "Y"
do...while Statement: Syntax

- Syntax
  ```javascript
  do {
    JS statement(s);
  } while (conditional expression);
  ```
  - The statements execute once before JS evaluates the conditional expression.
  - Different from the while statements in that a do...while statement always executes once.
  - do...while statement is an example of a posttest loop statement.

- Example
  ```javascript
  var count = 0;
  do {
    document.writeln("count = ", count);
    ++count;
  } while (count < 2);
  ```
do...while Statement: Example

Example

```javascript
//load array
var class = new Array(4);
class[0] = "Freshman";
class[1] = "Sophomore";
class[2] = "Junior";
class[3] = "Senior";

var count = 0;
do {
    document.write(class[count] + "<br>");
    ++count;
} while (count < 4);
```

Example Web Page

do...while Statement 1 (do-while_1.htm)
Repetition Application: Accumulators

- **Purpose**
  - Accumulators work within a loop to accumulate a value.

- **Example Web Page**
  - [Accumulator 1](accumulator_1.htm)
Repetition Application: Parallel Arrays

- Characteristics
  - Data in two arrays sharing the same index number.
  - Use a for loop to cross reference data in the arrays.

- Example
  - One array contains state abbreviations.
  - One array contains capital cities.
  - Create both arrays, input a state value, use for statement to find match, get index number, then print the corresponding capital city with that index number.

- Example Web Page
  
  Parallel Arrays 1 (parallel_array_1.htm)
  
  Parallel Arrays 2 (state_capitals.htm)
Arrays

- An array contains a set of data represented by a single variable name.
- Programmers call each piece of data contained in an array an element.
- Create an array with the `Array()` constructor object.
- The `Array()` constructor object receives a single argument representing the number of elements to be contained in the array.
- Specifying the number of array elements is optional.
- The numbering of elements within an array starts with an index number of zero (0).
- Programmers can create an array without any elements and then add new elements to the array as needed.
- The size of an array can change dynamically.
- Programmers can assign values to an array’s elements when they first create the array.
Summary Principles

Repetition

- A loop statement repeatedly executes a statement or a series of statements as long as a specific condition is true or until a specific condition becomes true.
- Use the `while` statement for repeating a statement or series of statements as long as a given conditional expression evaluates to true.
- Programmers call each repetition of a looping statement an iteration.
- Programmers must include code that tracks the progress of the `while` statement and change the value produced by the conditional expression once the program performs the desired tasks.
- A counter is a variable that increments with each iteration of a loop statement.
Summary Principles

➢ Repetition
  ❖ If a counter variable is beyond the range of a while statement’s conditional expression, then the program will completely bypass the while statement.
  ❖ In an infinite loop, a loop statement never ends because the program never updates its conditional expression.
  ❖ The do...while statement executes a statement or statements once, then repeats the execution as long as a given conditional expression evaluates to true.
  ❖ Use the for statement for repeating a statement or series of statements as long as a given conditional expression evaluates to true.
  ❖ Programmers can omit any of the three parts of the for statement, but they must include the semicolons that separate each section.
Summary Principles

➢ Repetition

❖ If programmers omit a section of the statement, then they should include code within the body that will end the for statement, or your program may get caught in an infinite loop.

❖ The continue statement halts a looping statement and restarts the loop with a new iteration.
Example Web Pages

- **for** Statement 1 (for_1.htm)
- **for** Statement 2 (for_2.htm)
- **for** Statement 3 (for_3.htm)
- **continue** Statement 1 (continue_1.htm)
- **break** Statement 1 (break_1.htm)
- **while** Statement 1 (while_1.htm)
- **Confirm Dialog Box** (confirm_1.htm)
- **while** Statement – 2 (while_2.htm)
- **do...while** Statement 1 (do-while_1.htm)
- **Accumulator 1** (accumulator_1.htm)
- **Parallel Arrays 1** (parallel_array_1.htm)
- **Parallel Arrays 2** (state_capitals.htm)
The End