Week 4 Overview

• Week 3 review
  • Functional Decomposition
    • Top-down design
    • Bottom-up implementation
  • Functions
    • Global vs. Local variables (scope)
    • Arguments/Parameters
    • Return values
Week 4 Overview

• Week 3 review
  • Event handlers
    • `<input>` tag for user input
    • `onclick`, `onfocus`, etc. events
    • Attach code to events

Week 4 Overview

• Outcomes
  • Sketch the solution to a problem requiring conditional execution.
  • Write correct conditional statements to solve a given problem.
ITEC 136
Business Programming Concepts

Week 04, Part 02
Homework Solutions

Homework 2 Solution

• Change maker

```html
<html>
<head><title>Change maker</title></head>
<body>
  <h1>Change maker</h1>
  <p>Author: Todd Whittaker</p>
  <script type="text/javascript" src="change.js"></script>
  <p>Reload or click <a href="">here</a> to run again</p>
</body>
</html>
```
• Change maker

```javascript
var cents = parseInt(prompt("Enter a number of cents", 87));
var remaining = cents;

var quarters = Math.floor(remaining / 25);
remaining = remaining % 25;
var dimes = Math.floor(remaining / 10);
remaining = remaining % 10;
var nickles = Math.floor(remaining / 5);
remaining = remaining % 5;
var pennies = remaining;

document.writeln(cents + " cents is comprised " + "of:<br />");
document.writeln(quarters + " quarter(s)<br />");
document.writeln(dimes + " dime(s)<br />");
document.writeln(nickles + " nickle(s)<br />");
document.writeln(pennies + " pennies<br />");
```
Homework 3 Solution

• Wages calculator

```html
<html>
<head>
    <title>Wages calculator</title>
    <script type="text/javascript" src="wages.js"></script>
</head>
<body>
    <h1>Wages calculator</h1>
    <p>Author: Todd Whittaker</p>
</body>
</html>
```
Homework 3 Solution

• Wages calculator

```html
<label for="payRate">Pay rate:</label><br />
<input type="text" id="payRate" value="25" />
<br />
<label for="hoursWorked">Hours:</label><br />
<input type="text" id="hoursWorked" value="40" />
<br />
<input type="button" value="Calculate" onclick="main()" />
<br />
<div id="output"></div>
</html>
```

```javascript
function main() {
    var LOCAL_RATE = 0.02;
    var STATE_RATE = 0.08;
    var FED_RATE = 0.31;

    // input
    var payRate = readFloat("payRate");
    var hoursWorked = readFloat("hoursWorked");
}
```
Homework 3 Solution

• Wages calculator

```javascript
// process
var grossPay = calcGrossPay(payRate, hoursWorked);
var taxes = calcTaxes(grossPay, LOCAL_RATE, STATE_RATE, FED_RATE);
var netPay = grossPay - taxes;
```

```javascript
// output
document.getElementById("output").innerHTML = 
    "Gross pay is: $" + grossPay.toFixed(2) + ".<br />
    + "Taxes are: $" + taxes.toFixed(2) + ".<br />
    + "Net pay is: $" + netPay.toFixed(2) + ".<br />
};
```
• Wages calculator

```javascript
function readFloat(field) {
    return parseFloat(document.getElementById(field).value);
}

function roundToHundredths(number) {
    return Math.round(number * 100) / 100;
}

function calcGrossPay(payRate, hoursWorked) {
    return hoursWorked <= 40 ? payRate * hoursWorked :
        payRate * (1.5 * hoursWorked - 20);
}
```

```javascript
function calcTaxes(grossPay, localRate, stateRate, fedRate) {
    var localTax = roundToHundredths(grossPay * localRate);
    var stateTax = roundToHundredths((grossPay - localTax) * stateRate);
    var fedTax = roundToHundredths((grossPay - localTax - stateTax) * fedRate);
    return fedTax + stateTax + localTax;
}
```
Conditional Execution

- Conditional Execution
  - Want to do something when a particular condition is met. e.g.:
    - Charging an ATM usage fee only for another bank’s customers, not your own.
    - Tagging an e-mail message as spam only if it contains certain words.
    - Alert the user only if they didn’t fill in a required field on a form.
Conditional Execution

- Ex: If a number is outside the range \([\text{min, max}]\), pop up an error message:

  \[ \text{num} > \text{max} \text{ OR } \text{num} < \text{min} \]
Conditional Execution

- Ex: Determine if the number is even or odd, displaying the result:

  Display "even" if \( \text{num} \mod 2 = 0 \) and "odd" otherwise.
Conditional Execution

- Ex: Display "valid" or "invalid" if an entered number is a valid month of the year:

Conditional Execution

- Ex: Given a height and a weight, determine if a person healthy or unhealthy:
  
  Body mass index: uses weight (kilograms) and height (meters) according to the following formula:
  
  \[
  bmi = \frac{w}{h^2}
  \]

  Anything outside the range [19,26] is considered unhealthy.
Conditional Execution

• Nested conditions

• Multiple decisions within one another

Ex: Normal BMI for women is [19.1, 25.8], and for men is [20.7, 26.4]:

[Diagram of conditional execution with nested conditions and multiple decisions.]
if/else Statements

- Keywords if and else implement conditional execution

```java
if (<condition>) {
    <statements>
}
```
if/else Statements

• Ex: If a number is outside the range [min, max], pop up an error message:

```javascript
function validRange(num, min, max) {
    if (num < min || num > max) {
        alert(num + " is outside [" + min + ", " + max + "]");
    }
}
```

if/else Statements

• Keywords if and else implement conditional execution

```javascript
if (<condition>) {
    <statements>
} else {
    <statements>
}
```
if/else Statements

• Ex: Determine if the number is even or odd, displaying the result:

```javascript
var num = parseInt(prompt("Enter a number"));
if (num % 2 == 0) {
    alert(num + " is even");
} else {
    alert(num + " is odd");
}
```

Conditional Execution

• Ex: Given a height and a weight, determine if a person healthy or unhealthy:

Body mass index: uses weight (kilograms) and height (meters) according to the following formula:

\[
\text{bmi} = \frac{w}{h^2}
\]

Anything outside the range [19,26] is considered unhealthy.
if/else Statements

• Highly complex conditions require if/else structures within if/else structures.

```java
if (condition1)
    if (condition2)
        doSomething();
    else
        doAnotherThing();
else
    doSomethingEntirelyDifferent();
```

Which “if” does this “else” match? How can it be made to match the other “if”? Called the “dangling else” problem.

```java
if (condition1)
    if (condition2)
        doSomething();
else
    doAnotherThing();
else
    doSomethingEntirelyDifferent();
```

Indentation is only for people, and doesn’t mean anything to the interpreter. This matches the 2nd (closest) if-statement.
if/else Statements

• Highly complex conditions require if/else structures within if/else structures.

```java
if (condition1) {
    if (condition2) {
        doSomething();
    } else {
        doAnotherThing();
    }
} else {
    doSomethingEntirelyDifferent();
}
```

Tip: always use curly braces around the body of if-statements.

Indentation is only for people, and doesn't mean anything to the interpreter. This matches the 2nd (closest) if-statement.
if/else Statements

• Some simple transformations

```java
if (condition1)
  if (condition2)
    doSomething();

if (condition1 && condition2)
  doSomething();
```

```java
if (condition1)
  doSomething();
else if (condition2)
  doSomething();

if (condition1 || condition2)
  doSomething();
```
Case study: Date Validation

• Date validation
  • Given three numbers (month, day, and year) do the three form a valid date?
    • Month: 1-12
    • Day: 1-28 always valid, 29, 30, 31 sometimes valid depending on year
    • No year 0, account for leap year

Case study: Date Validation

• Date validation – User interface

```html
<!DOCTYPE html PUBLIC
  "-//W3C//DTD XHTML 1.0 Transitional//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <meta http-equiv="Content-Type"
      content="text/html; charset=iso-8859-1" />
    <title>Date Validation</title>
    <script language="JavaScript" type="text/javascript"
      src="DateValidation.js"></script>
  </head>
</html>
```
Case study: Date Validation

• Date validation – User Interface

```html
<table>
  <tr><td>Month:</td></tr>
  <tr><td><input type="text" id="month" name="month" /></td></tr>
  <tr><td>Day:</td></tr>
  <tr><td><input type="text" id="day" name="day" /></td></tr>
  <tr><td>Year:</td></tr>
  <tr><td><input type="text" id="year" name="year" /></td></tr>
  <tr><td colspan="2"><input type="button" value="Validate" onclick="main('month', 'day', 'year')" /></td></tr>
</table>
```

Case study: Date Validation

• Date validation – main program

```javascript
function main(monthId, dayId, yearId)
{
    var month = getInt(monthId);
    var day = getInt(dayId);
    var year = getInt(yearId);
    if (isValidDate(month, day, year))
        alert("Valid date");
    else
        alert("Invalid date");
}
```
Case study: Date Validation

• Date validation – user input

```javascript
function getInt(id) {
  return parseInt(document.getElementById(id).value);
}
```

• Date validation – months and years

```javascript
function isValidMonth(month) {
  return month >= 1 && month <= 12;
}

function isValidYear(year) {
  return year != 0;
}
```
Case study: Date Validation

- Date validation
  - Leap year
    - Keeps solar year and calendar year synchronized.
    - Rule: any year divisible evenly by 4 is a leap year, unless it is also divisible by 100, unless it is also divisible by 400

Case study: Date Validation

- Date validation
  - Leap year test cases
    - Ex: 1986 is not a leap year (why?)
    - Ex: 1988 is a leap year (why?)
    - Ex: 1900 is not a leap year (why?)
    - Ex: 2000 is a leap year (why?)
Case study: Date Validation

- Date validation

```javascript
function isLeapYear(year) {
    return year % 4 == 0 && year % 100 != 0 || year % 400 == 0
}
```

```javascript
function isValidDay(month, day, year) {
    var result = false;
    if (isValidMonth(month) && isValidYear(year)) {
        // do some calculation...
    }
    return result;
}
```

What is returned if the month or year is invalid?
Case study: Date Validation

• Date validation

```javascript
// here's the calculation...
var maxDay = 31;
if (month == 9 || month == 4 ||
    month == 6 || month == 11)
    maxDay = 30;
else if (month == 2)
    if (isLeapYear(year))
        maxDay = 29;
    else
        maxDay = 28;
result = day >= 1 && day <= maxDay;
```

How are January, March, May, etc handled?

• The working application (properly formatted for course coding conventions):

http://cs.franklin.edu/~whittakt/ITEC136/examples/DateValidation.html
On your own...

- Federal tax calculation schedule X*

<table>
<thead>
<tr>
<th>If taxable income is over--</th>
<th>But not over--</th>
<th>The tax is:</th>
</tr>
</thead>
<tbody>
<tr>
<td>$0</td>
<td>$7,825</td>
<td>10% of the amount over $0</td>
</tr>
<tr>
<td>$7,825</td>
<td>$31,850</td>
<td>$782.50 plus 15% of the amount over 7,825</td>
</tr>
<tr>
<td>$31,850</td>
<td>$77,100</td>
<td>$4,386.25 plus 25% of the amount over 31,850</td>
</tr>
<tr>
<td>$77,100</td>
<td>$160,850</td>
<td>$15,698.75 plus 28% of the amount over 77,100</td>
</tr>
<tr>
<td>$160,850</td>
<td>$349,700</td>
<td>$39,148.75 plus 33% of the amount over 160,850</td>
</tr>
<tr>
<td>$349,700</td>
<td>no limit</td>
<td>$101,469.25 plus 35% of the amount over 349,700</td>
</tr>
</tbody>
</table>


On your own...

- Write a program that inputs the adjusted gross income and outputs the expected tax bill.
Testing programs

• Wouldn’t it be nice if your program told you when you wrote in a bug?
• What if it could do this:

```javascript
if (programHasABug())
    alert("Bug detected!");
```
Testing programs

- We can get close!
- Unit testing
  - Making sure that each function, when provided correct inputs, produces correct outputs.
  - Also, when provided incorrect inputs, it doesn’t do harm

```javascript
function assertEquals(expected, actual, message) {
    if (!(expected == actual)) {
        var str = "Error!" +
            "\n Expected: " + expected +
            "\n Actual: " + actual;
        if (message) {
            str += "\n Message: " + message;
        }
        alert(str);
    }
}
```
Testing programs

- A testing function

```javascript
function assertEquals(expected, actual, message) {
  if (!(expected == actual)) {
    var str = "Error!"
      + "
      Expected: " + expected +
      "\n      Actual: " + actual;
    if (message) {
      str += "\n      Message: " + message;
    }
    alert(str);
  }
}
```

Only if the message is non-null and non-undefined will it appear in the alert.

Testing programs

- Writing a test

```javascript
function testIsValidMonth() {
  // try some valid months...
  assertEquals(true, isValidMonth(3),
    "March should be valid");
  // and some invalid months...
  assertEquals(false, isValidMonth(13),
    "Febtober should be invalid");
}
```
Testing programs

• Writing a test

```javascript
function testIsValidMonth()
{
    // try some valid months...
    assertEquals(true, isValidMonth(3),
        "March should be valid");
    // and some invalid ones...
    assertEquals(false, isValidMonth(13),
        "Febtober should be invalid");
}
```

Testing programs

• Advantages of writing tests
  • Makes you think clearly about the inputs and outputs of functions
  • Makes you write small, testable code
  • Gives you a safety net when you change your code (rerun the tests)
  • You see progress toward a solution (more tests pass)
switch Statements

• Switch statements
  • A shortcut to compare many values and conditionally execute code based strictly on equality.
    • Good for a limited number of enumerable options.
    • Bad for testing ranges of values, deciding between two mutually exclusive options.
switch Statements

• Switch statements – suitability

<table>
<thead>
<tr>
<th>Example</th>
<th>If/else or switch?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Determining point values of letter grades.</td>
<td></td>
</tr>
<tr>
<td>Determining letter grades from a percentage.</td>
<td></td>
</tr>
<tr>
<td>Determining insurance rates based on age.</td>
<td></td>
</tr>
<tr>
<td>Determine the name of a month based on a month number.</td>
<td></td>
</tr>
<tr>
<td>Determine form letter salutation based on gender.</td>
<td></td>
</tr>
</tbody>
</table>

switch (expression) {
    case value1:
        statements1;
        break;
    case value2:
        statements2;
        break;
    //...
    case valueN:
        statementsN;
        break;
    default:
        statements;
        break;
}
switch Statements

• Switch statements

```java
switch (<expression>) {
    case <value1>:
        <statements1>;
        break;
    case <value2>:
        <statements2>;
        break;
    //...
    case <valueN>:
        <statements2>;
        break;
    default:
        <statements>;
        break;
}
```

Execution “falls through” to the next case!

• Ex: prerequisites for courses

<table>
<thead>
<tr>
<th>Course</th>
<th>Prerequisite(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ITEC 495</td>
<td>ITEC 400, MIS 484, ITEC 430</td>
</tr>
<tr>
<td>ITEC 400</td>
<td>ITEC 136, ITEC 275</td>
</tr>
<tr>
<td>ITEC 430</td>
<td>MIS 310, MIS 320, MATH 215</td>
</tr>
<tr>
<td>MIS 310</td>
<td>COMM 320</td>
</tr>
<tr>
<td>MIS 320</td>
<td>COMM 320</td>
</tr>
<tr>
<td>ITEC 350</td>
<td>ITEC 136, ITEC 275</td>
</tr>
<tr>
<td>ITEC 450</td>
<td>COMP 281</td>
</tr>
</tbody>
</table>
switch Statements

```javascript
function prerequisitesFor(course) {
    var result = "";
    switch (course) {
        case "ITEC495":
            result += "MIS484, ITEC400, ITEC430"; break;
        case "ITEC350":
            result += "ITEC136, ITEC275"; break;
        case "ITEC136":
            result += "COMP107, MATH160"; break;
        case "MIS310":
            result += "ITEC430":
                result += "MIS310, MIS320, MATH 215"; break;
        case "ITEC450":
            result += "COMP281"; break;
        default:
            alert("Unknown course: " + course)
    }
    return result;
}
```

Note the fall through cases for ITEC 350 and MIS 310.

switch or if / else

- Any switch can be rewritten as a series of if / else statements.
- This is a good exam question...
switch Statements

```javascript
function prerequisitesFor(course) {
    var result = "";
    switch (course) {
        case "ITEC495":
            result += "MIS484, ITEC400, ITEC430"; break;
        case "ITEC350":
        case "ITEC400":
            result += "ITEC136, ITEC275"; break;
        case "ITEC430":
            result += "MIS310, MIS320, MATH 215"; break;
        case "ITEC450":
            result += "COMP281"; break;
        default:
            alert("Unknown course: " + course);
    }
    return result;
}
```

Try it yourself...

• For the suitable cases mentioned previously, try writing a switch statement to determine the answer.
Questions?
Self Quiz

• Question 1: An “if” statement whose condition evaluates to “true” can only execute a single statement after the “if” unless the statements are surrounded by ______________.

• Question 2: When the value returned by a “switch” statement does not match a “case” label, then the statements with the __________ label execute.
Self Quiz

• Question 3: In the following code, which line contains the “if” statement that corresponds to the “else” on line 4?

```
1. if (a < 10)
2.   if (b > 7)
3.     x = 5;
4. else
5.     x = 7;
```

Self Quiz

• Question 4: This program pops up “foo” as the result. What is the problem?

```
1. var x = "5";
2. var result;
3. if (x = 7)
4.   result = "foo";
5. else
6.   result = "bar";
7. alert(result);
```
Self Quiz

• Question 5: Water exists in three states (solid, liquid, and gas) at one atmosphere of pressure based on the current temperature. Given a temperature as input, you are to alert the user about the state of water at that temperature. Should you use an if/else or a switch? Why?
Upcoming Deadlines

• Lab 1 – Due February 2
• Exam 1 – In class February 2
• Reflection paper draft 1 – Due February 2
• Week 6 pre-class exercise – Due February 9