College of Information Studies

University of Maryland Hornbake Library Building College Park, MD 20742-4345

Web Infrastructure

Week 3

INFM 603

The Key Ideas

- Questions
- Structured Programming
- Modular Programming
- Data Structures
- Object-Oriented Programming

Algorithms

- A finite sequence of well-defined instructions designed to accomplish a certain task

- Named for the Persian mathematician Al-Khwarizmi

High level Languages

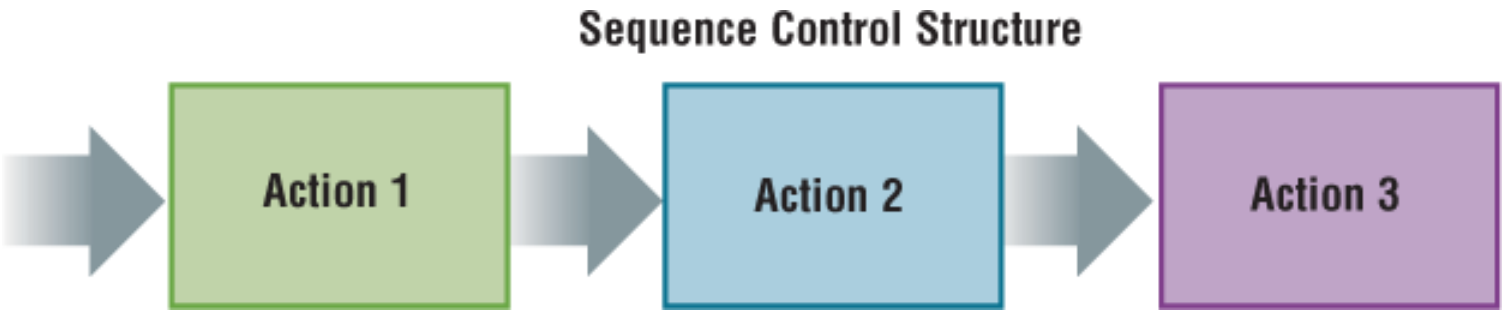
- Procedural (modular) Programming
  - Group instructions into meaningful abstractions
  - C, Pascal, Perl
- Object oriented programming
  - Group “data” and “methods” into “objects”
  - Naturally represents the world around us
  - C++, Java, JavaScript, PHP, Ruby

Basic Control Structures

- Sequential
  - Perform instructions one after another
- Conditional
  - Perform instructions contingent on something
- Repetition
  - Repeat instructions until a condition is met

Not much different from cooking recipes!

Sequential Control Structure

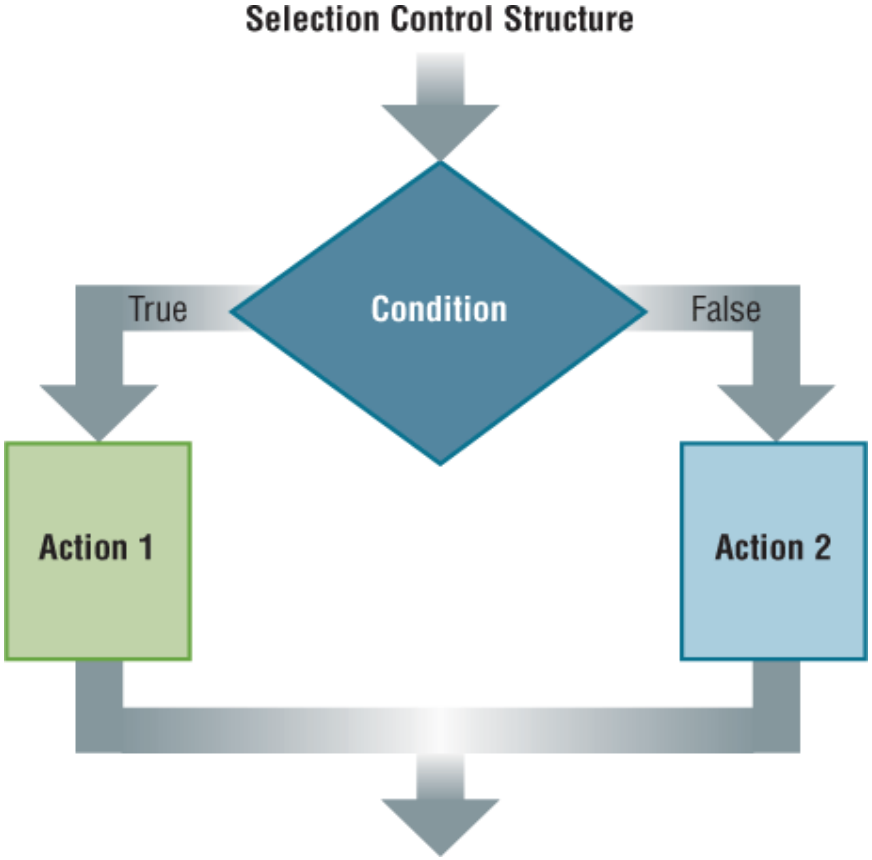



```
a = 2;

b = 3;

c = a * b;
```

Conditional Selection Control Structure



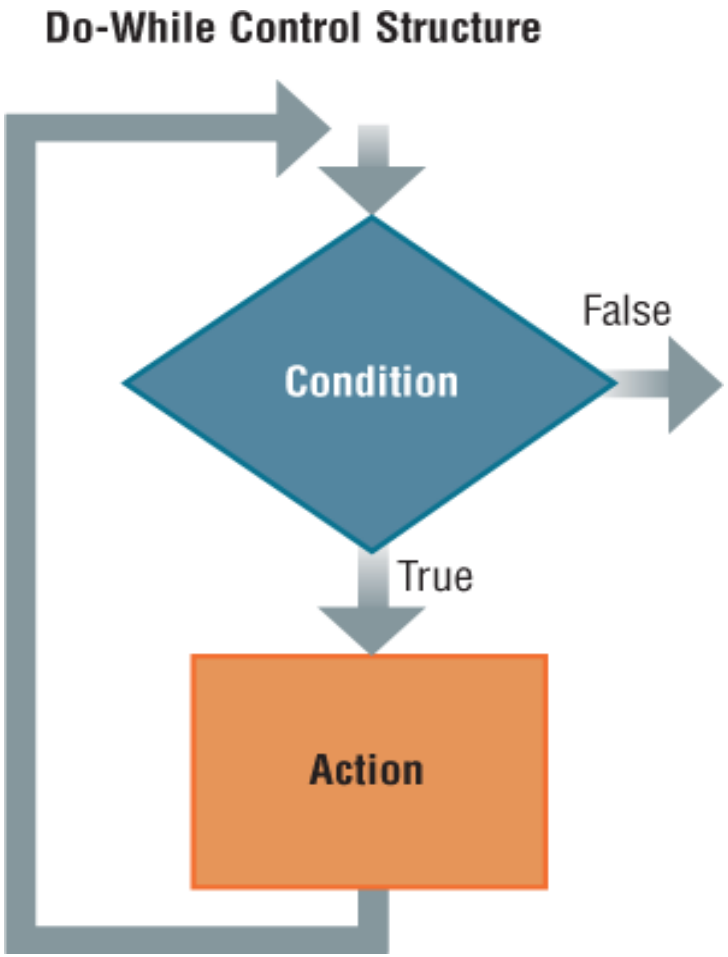

```
if (gender == “male”) {
    greeting = “Hello, Sir”;
} else {
    greeting = “Hello, Madam”;
}
```

```
switch (gender) {
    case “male”:
        greeting = “Hello, Sir”;
        break;
    default:
        greeting = “Hello, Madam”
}
```

Boolean Operators

- x == y — true if x and y are equal [use == not =]
- x != y — true if x and y are not equal
- x > y — true if x is greater than y
- x <= y — true if x is smaller than or equal to y
- x && y — true if both x and y are true
- x || y — true if either x or y is true
- !x — true if x is false

Repetition Control Structure




```
n = 0;
while (n<10) {
    document.writeln(n);
    n++;
}
```

```
for (n=0; n<10; n++) {
    document.writeln(n);
}
```

Key Ideas

- Flowcharts
- Pseudocode
- Stacking and Nesting

Group Exercise

- Calculate the value of a $10,000 investment at the end of each year each year from a list of annual percentage gains or losses, and make a note in each year for which a constant 5% interest rate would outperform the variable rate investment.

2001	−11.9%
2002	−22.1%
2003	28.7%
2004	10.9%
2005	4.9%
2006	15.8%
2007	5.5%
2008	−37.0%
2009	26.5%
2010	15.1%

Pair Exercises

- Print every even number below 873 in the Fibonacci series (see Wikipedia definition).

- Print a 9x9 lower triangular matrix of asterisks.

- Prompt the user to enter a date (number of the month and number of the day), check to see if the date is valid (assume February has 28 days), and reprompt until a valid date is entered.

Design Tips

- Protect against unexpected values
  - Test the value of **all** user input
  - Test the value of critical function parameters
- Verify that every loop will **always** terminate
  - Include a bailout condition, and report it
- Always test for conditions explicitly
  - Trap unexpected conditions with the final else

Programming Tips

- Attention to detail!
  - Careful where you place that comma, semicolon, etc.
- Don’t get cute with the logic or the layout
  - Reflect the structure of your problem clearly
  - Use standard “design patterns”
- Write a little bit of code at a time
  - Add some functionality, make sure it works, move on
- Debug by viewing the “state” of your program
  - Print values of variables using document.writeln();

Documentation Tips

- Reflect your pseudocode in your code
  - Use meaningful variable names
  - Use functions for abstractable concepts
    - And name those functions well
  - Use comments to fill remaining gaps
- Add a comment to identify each revision
  - Give author, date, nature of the change
- Waste space effectively
  - Use indentation and blank lines to guide the eye

Arrays

- A set of elements
  - For example, the number of days in each month
- Each element is assigned an index
  - A number used to refer to that element
    - For example, x[4] is the fifth element (count from zero!)
  - Arrays and repetitions work naturally together

Using JavaScript with Forms

HTML:

```
<form name="input" action="">
   Please enter a number:
   <input size="10" value=" " name="number"/>
</form>
<form name="output" action="">
   The sum of all numbers up to the number above is
   <input size="10" value=" " name="number" readonly="true"/>
</form>
```

Reads in a value from the first form
(*eval* method turns it into a number)

JavaScript:

```
var num = eval(document.input.number.value);
document.output.number.value = 10;
```

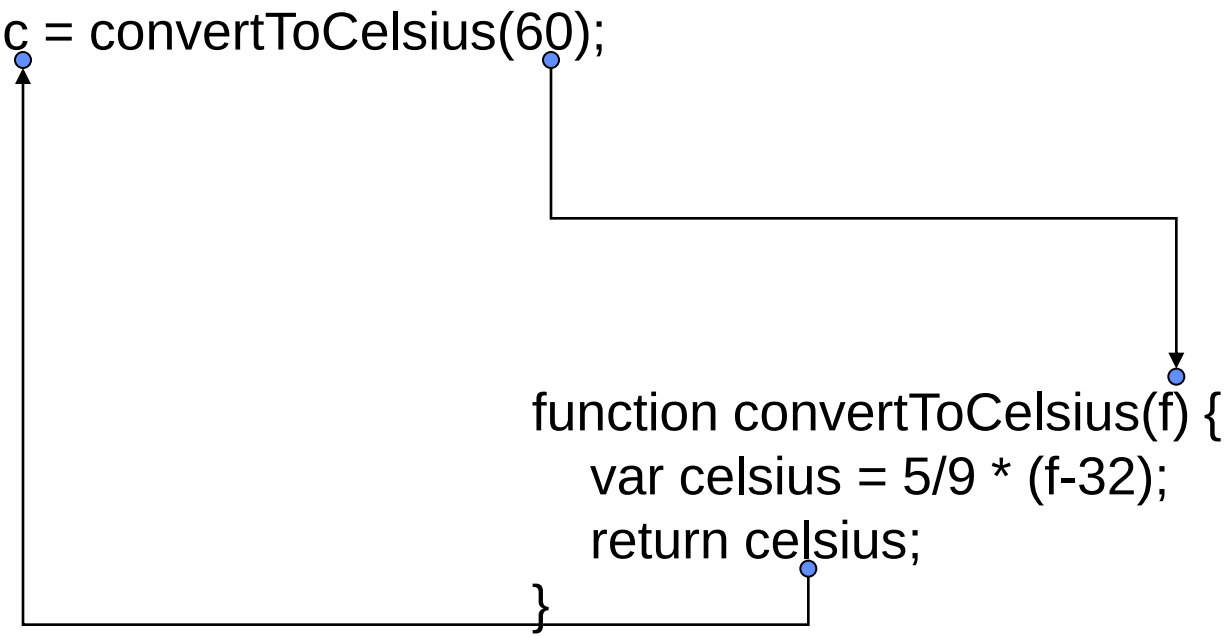
Changes the value in the second form

Functions (non-object "Methods")

- Reusable code for complex "statements"
  - Takes one or more values as "parameters"
  - Returns at most one value as the "result"

```
function convertToCelsius(f) {
    var celsius = 5/9 * (f-32);
    return celsius;
}
```

```
var f = 60;
c = convertToCelsius(f);
```



Writing JavaScript Functions

- Convenient to put it in the <head> section
  - Use <!-- … //--> to prevent display of code

```
…
<head>
<script language="JavaScript" type="text/javascript">
<!--
function calculate() {
  var num = eval(document.input.number.value);
…
  document.output.number.value = total;
}
//-->
</script>
</head>
…
```

Scope of a Variable

- In JavaScript, *var* "declares" a variable

  var mystery; create a variable without defining its type

  var b = true; create a boolean *b* and set it to true

  var n = 1; create an integer *n* and set it to 1

  var s = "hello"; create a string *s* and set it to "hello"

- Variables declared in a function are "local"
  - Same name outside function refers to **different** variable

- All other variables are "global"

Some Useful Predefined “Methods”

- document.writeln(“…”);
  - String gets **rendered** as (X)HTML
  - Include “<br />” to force a line break
- window.alert(“…”);
  - String is **written verbatim** as text
  - Include “\n” to force a line break
- foo = window.prompt(“…”);
  - String is **shown verbatim** as text
  - Result is whatever string the user enters

Handling Events

- Events:
  - Actions that users perform while visiting a page

- Use event handlers to response events
  - Event handlers triggered by events
  - Examples of event handlers in Javascript
    - onMouseover: the mouse moved over an object
    - onMouseout: the mouse moved off an object
    - onClick: the user clicked on an object

Before You Go

On a sheet of paper, answer the following (ungraded) question (no names, please):

What was the muddiest point in today's class?