



# EECS 1710

## Programming for Digital Media

Lecture 8 :: Decision Making 2

# This Week

## *Lecture 7*

- *Error Types*
- *Strings & string methods (revisited)*
- *text(), textWidth(), textSize()*
- *random()*
- *relational and conditional operators*

## Lecture 8

- conditional logic practice
- IF, ELSE
- SWITCH, CASE

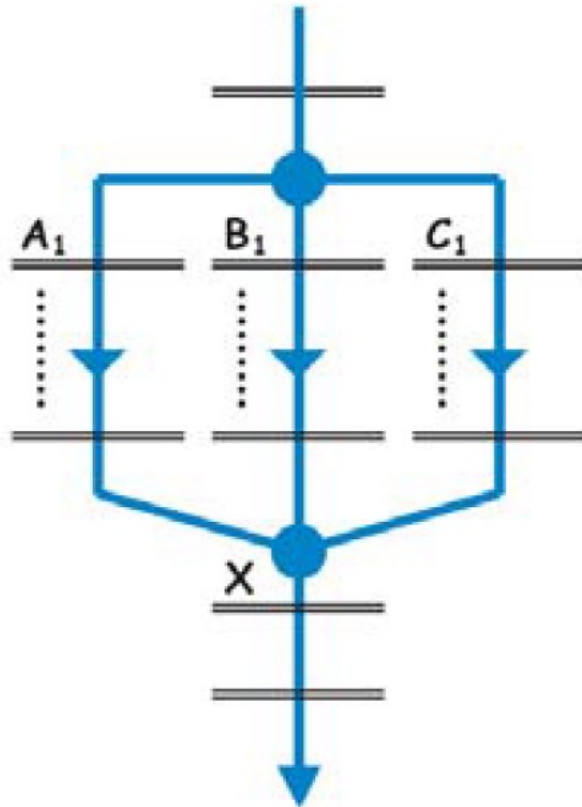
# Making Decisions [2]

Branching

# Flow of execution / Flow of control

- **Branching**

(b)



```
// ... some statements ...
```

```
// perform some test
```

```
// execute if test gives outcome A1
```

```
{ ... }
```

```
// execute if test gives outcome B1
```

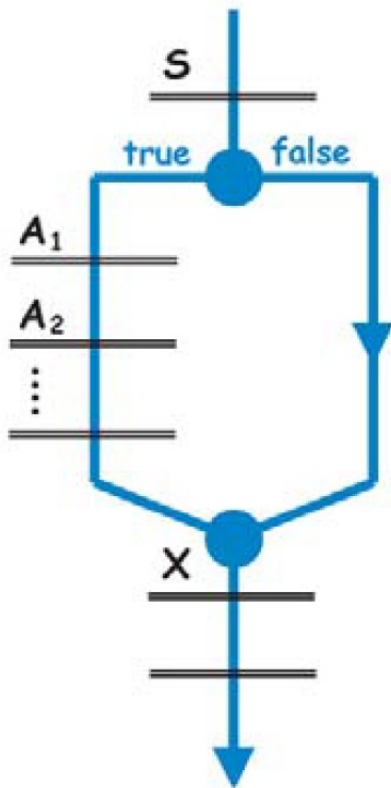
```
{ ... }
```

```
// execute if test gives outcome C1
```

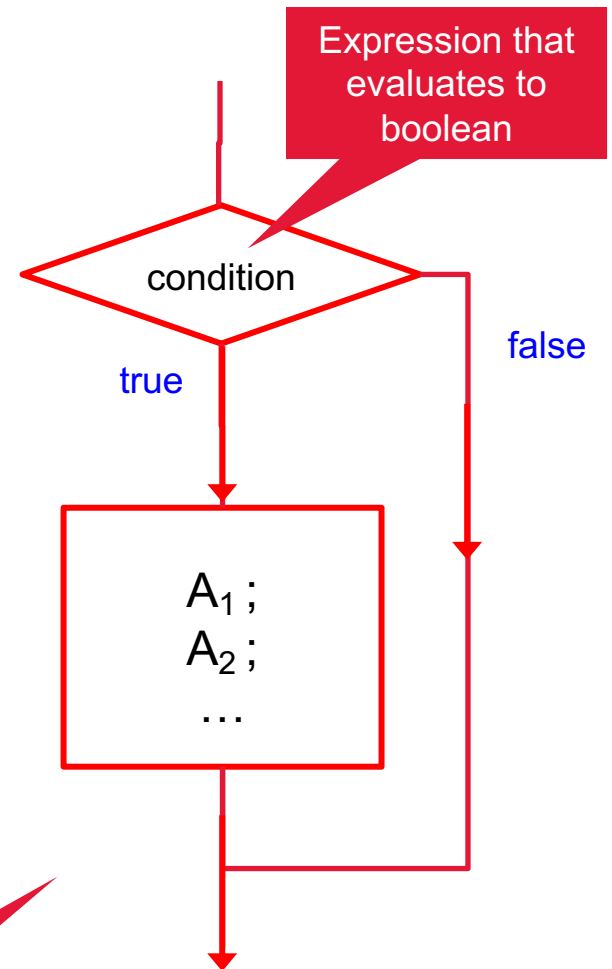
```
{ ... }
```

```
// ... more statements ...
```

# The IF statement



```
Statement-S  
if (condition)  
{  
    Statement-A1  
    Statement-A2  
    .  
    .  
    .  
}  
Statement-X  
.  
.  
.
```



flowchart

# Unit tests (good approach to testing code)

- One possible combination of input value(s) and their expected output value

```
void setup() {  
  
    println("isValidPixel():");  
  
    // each line is an example of a single unit test  
    // a unit test is one possible comb of input and expected output  
    println("(150,58,33) -> exp: true, act: " + isValidPixel(150, 58, 33));  
  
    println("(150,358,33) -> exp: false, act: " + isValidPixel(150, 358, 33));  
    println("(-1,-100,256) -> exp: false, act: " + isValidPixel(-1, -100, 256));  
    println("(255,255,255) -> exp: true, act: " + isValidPixel(255, 255, 255));  
  
}
```

## LEAP YEAR ALGORITHM

=====

**A year is a leap year if either of the following conditions is true:**

- 1. The year is evenly divisible by 4, 100, and 400 (i.e., divisible by all of these)**
- 2. The year is evenly divisible by 4, but not evenly divisible by 100**

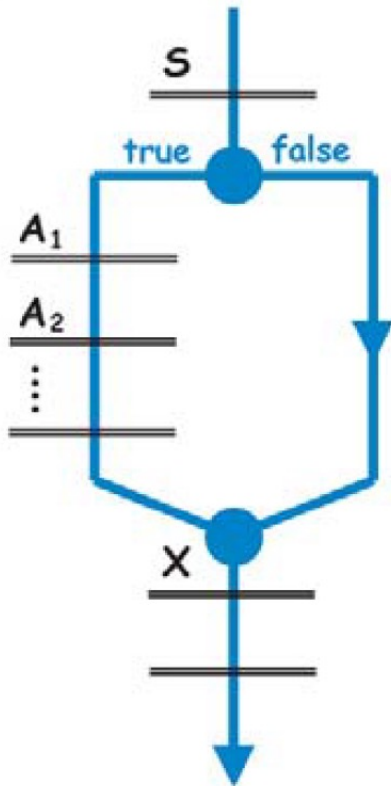
**Otherwise, it is not a leap year!**

=====

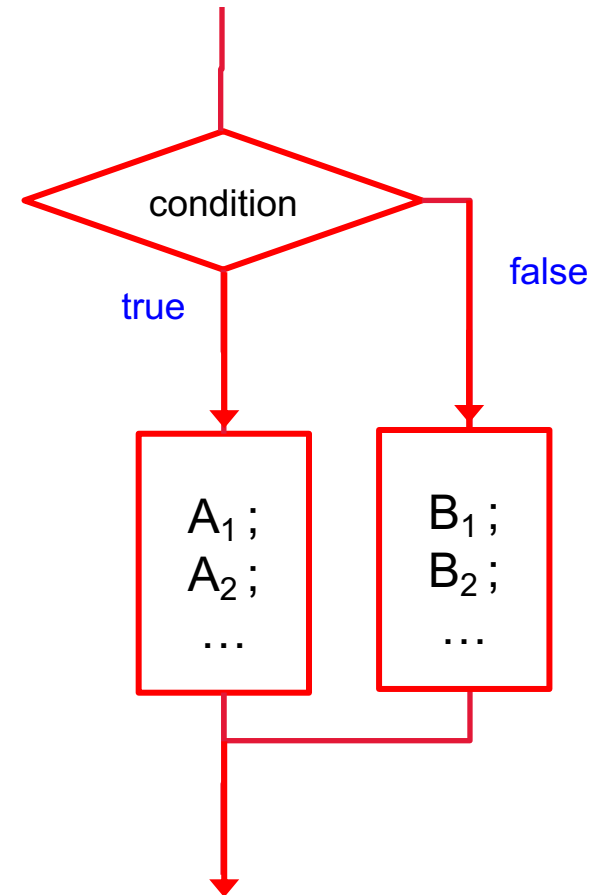
How to implement?

demo

# The IF/ELSE statement



```
Statement-S
if (condition)
{
    Statement-A1
    Statement-A2
    .
    .
    .
}
else {
    Statement-B1
    Statement-B2
    ...
}
Statement-X
...
```





# IF/ELSE

```
if (condition1) {  
    // if we reach here, condition1 is true  
  
    Statement-A1;  
    Statement-A2;  
    ...  
}  
else {  
    // if we reach here, condition1 is false  
    // i.e. same as (!condition1)  
    // if condition1 is true, we never reach here!  
  
    Statement-B1;  
    Statement-B2;  
    ...  
}
```

# Example: EvenOdd

```
int value = 4;  
// value = 5;  
  
if (value % 2 == 0) {  
    println("Even");  
}  
else {  
    println("Odd");  
}
```

# Ternary Operator "?"

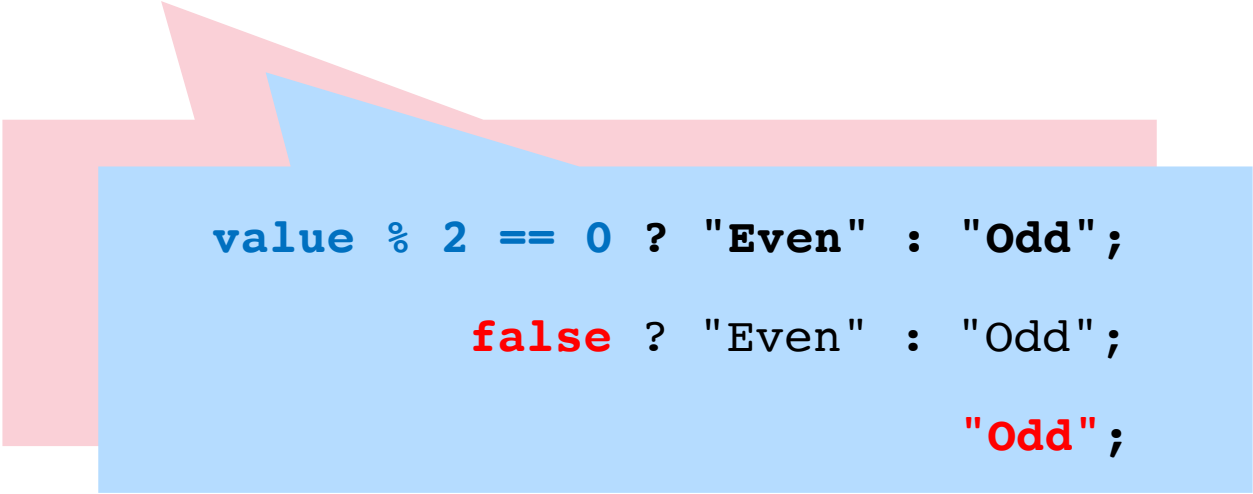
- acts as an IF/ELSE statement would
- useful when only 1 statement to execute if true/false
- Syntax:

*condition ? statement-T : statement-F ;*

# Example: EvenOdd2

```
// assume value exists  
// ternary version of EvenOdd
```

```
String message = value % 2 == 0 ? "Even" : "Odd";  
println(message);
```



```
value % 2 == 0 ? "Even" : "Odd";  
false ? "Even" : "Odd";  
"Odd";
```

# Nested IF/ELSE

- If we “nest” (put) IF statements within IF statements:

```
if (condition1) {  
    // if we reach here, condition1 is true  
    if (condition2) {  
        // if we reach here, this means  
        // condition1 and condition2 must both be true  
    }  
}
```

- This is equivalent of AND'ing the two conditions

```
if (condition1 && condition2) {  
    // if we reach here, this means  
    // condition1 and condition2 must both be true  
}
```

# Nested IF/ELSE

```
if (condition1) {  
    // if we reach here, condition1 is true  
  
    if (condition2) {  
        // if we reach here, this means  
        // condition1 and condition2 must both be true  
        // i.e. same as (condition1 && condition2)  
    }  
    else {  
        // if we reach here, this means  
        // condition1 is true and condition2 is false  
        // i.e. Same as (condition1 && !condition2)  
    }  
}  
else {  
  
    // if we reach here, condition1 is false only  
    // we haven't considered condition2 at all  
  
}
```

# Nested IF/ELSE

```
if (condition1) {  
    // if we reach here, condition1 is true  
  
    if (condition2) {  
        // if we reach here, this means  
        // condition1 and condition2 must both be true  
        // i.e. same as (condition1 && condition2)  
    }  
    else {  
        // if we reach here, this means  
        // condition1 is true and condition2 is false  
        // i.e. Same as (condition1 && !condition2)  
    }  
}  
else if(condition2) {  
    // if we reach here, condition1 is false and condition2 is true  
    // i.e. same as (!condition1 && condition2)  
}  
else {  
    // if we reach here, condition1 and condition2 both false  
    // i.e. same as (!condition1 && !condition2)  
}  
}
```

## Example: LetterGrade

Score	Grade
≥90	A+
≥80	A
≥75	B+
≥70	B
≥65	C+
≥60	C
≥55	D+
≥50	D
≥40	E
<40	F

```
int score = 32;
String grade = "F";

if (score ≥ 90) grade = "A+";
else if (score ≥ 80) grade = "A";
else if (score ≥ 75) grade = "B+";
else if (score ≥ 70) grade = "B";
else if (score ≥ 65) grade = "C+";
else if (score ≥ 60) grade = "C";
else if (score ≥ 55) grade = "D+";
else if (score ≥ 50) grade = "D";
else if (score ≥ 40) grade = "E";
else grade = "F";

println("score = " + score + ",
        grade = " + grade);
```



# What happens here? (e.g. midterm question)

What value is assigned to fee by the if statement below, when speed is 75?

Does this seem appropriate? Would you modify it (if so how)?

```
double fee;  
  
if (speed > 35) {  
    fee = 20.0;  
}  
else if (speed > 50) {  
    fee = 40.00;  
}  
else if (speed > 75) {  
    fee = 60.00;  
}
```

# Takeaway

- IF/ELSE tests can get pretty messy, pretty quickly..
- This is why sometimes it is better to construct boolean expressions and use them as conditions
- So that we avoid excessively large, nested IF/ELSE statements

# Some practice with conditional logic

- Coding bat examples
  - <https://codingbat.com/java/Logic-1>
  - <https://codingbat.com/java/Logic-2>

Do Logic-1 questions for practice

Try first to return a value (true/false or whatever type question expects)

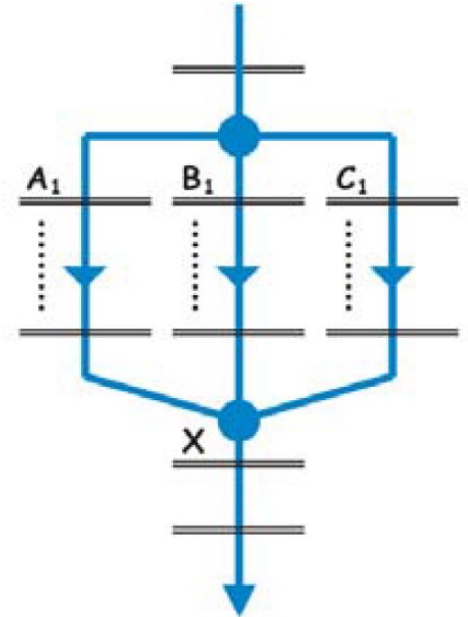
This will show results of many unit tests..

Now modify your code tests PASS (i.e. all in the green!)

When you are confident... try Logic-2 (bit harder)

# More Advanced Branching

- SWITCH STATEMENTS
- (not required, but can make code neater! Especially when there are a large number of branches to consider for a given outcome)
- Usually used when we branch on several alternative VALUES of a given variable/expression



# Example: Menus & Options

- Present a menu of options for a program
- Select a menu option (e.g. key press) and test for it

```
int shapeToDraw = 0;

void showMenuOptions() {
    println("Hit a key to select the shape to draw.");
    println(" - then click and hold mouse while dragging to draw");
    println("=====");
    println("0  - draws a line when dragging (default)");
    println("1  - draws a rect");
    println("2  - draws a circle");
    println("3  - draws an ellipse");
}

String whichShape() {
    if (shapeToDraw == 0) return "LINE";
    if (shapeToDraw == 1) return "RECT";
    if (shapeToDraw == 2) return "CIRCLE";
    if (shapeToDraw == 3) return "ELLIPSE";
}
```

# Menus & Options

```
void setup() {  
    showMenuOptions();  
}  
  
void draw() {  
  
void keyPressed() {  
  
    if (key == '0') shapeToDraw=0;  
    if (key == '1') shapeToDraw=1;  
    if (key == '2') shapeToDraw=2;  
    if (key == '3') shapeToDraw=3;  
  
    println("You chose to draw a "  
            + whichShape()    );  
  
}
```

# Using switch/case?

```
void setup() {  
    showMenuOptions();  
}  
  
void draw() {  
}  
  
void keyPressed() {  
  
    if (key == '0') shapeToDraw=0;  
    if (key == '1') shapeToDraw=1;  
    if (key == '2') shapeToDraw=2;  
    if (key == '3') shapeToDraw=3;  
  
    println("You chose to draw a "  
            + whichShape() );  
}
```

```
void setup() {  
    showMenuOptions();  
}  
  
void draw() {  
}  
  
void keyPressed() {  
  
    switch (key) {  
        case '0': shapeToDraw=0; break;  
        case '1': shapeToDraw=1; break;  
        case '2': shapeToDraw=2; break;  
        case '3': shapeToDraw=3; break;  
    }  
  
    println("You chose to draw a "  
            + whichShape() );  
}
```

# How to process keyPressed() { } ??

- Could use many IF's
  - Recall.. Can get ugly
- lets use SWITCH/CASE



# Drawing while mouse button is down...

- Use a flag to say if drawing is on
  - Set it to off normally
  - When mouse pressed, set to on
  - When mouse released, set back to off
- Draw
  - If draw mode is on, then draw current selected shape
  - If draw mode is off, do nothing
- Keypress?
  - switch shape to draw

```

// add new global vars

// flag
boolean drawNow = false;

// start pos. of object
int startX;
int startY;

// capture start pos and
// update drawNow flag

void mousePressed() {
    drawNow = true;
    startX = mouseX;
    startY = mouseY;
}

void mouseReleased() {
    drawNow = false;
}

```

```

void draw() {

    if (drawNow) {
        background(255, 255, 255);
        switch (shapeToDraw) {
            case 0:
                background(255, 255, 255);
                line(startX, startY, mouseX, mouseY);
                break;
            case 1:
                background(255, 255, 255);
                rectMode(CORNERS);
                rect(startX, startY, mouseX, mouseY);
                break;
            case 2:
                ellipseMode(RADIUS);
                circle(      startX, startY,
                           abs(mouseX-startX));

                break;
            case 3:
                background(255, 255, 255);
                ellipseMode(RADIUS);
                ellipse(      startX, startY,
                           abs(mouseX-startX),
                           abs(mouseY-startY));

                break;
        }
    }
}

```

```

// add new global vars

// flag
boolean drawNow = false;

// start pos. of object
int startX;
int startY;

// capture start pos and
// update drawNow flag

void mousePressed() {
    drawNow = true;
    startX = mouseX;
    startY = mouseY;
}

void mouseReleased() {
    drawNow = false;
}

```

optimized  
version

```

void draw() {

    if (drawNow) {

        background(255, 255, 255);

        switch (shapeToDraw) {
        case 1:
            rectMode(CORNERS);
            rect(startX, startY, mouseX, mouseY);
            break;

        case 2:
        case 3:
            // enters here if either 2, or 3
            background(255, 255, 255);
            ellipseMode(RADIUS);
            ellipse( startX, startY,
                    abs(mouseX-startX),
                    abs(mouseX-startX));
            break;

        default:
            // defaults to drawing a LINE
            line(startX, startY, mouseX, mouseY);
            break;
        }
    }
}

```