Monday January 21

Lecture 5

- Lab Test I (week of Jan. 28)
  - guide
  - tutorial video
  - two example tests

Why Selective Actions

```java
import java.util.Scanner;
public class ComputeArea {
  public static void main(String[] args) {
    Scanner input = new Scanner(System.in);
    final double PI = 3.14;
    System.out.println("Enter the radius of a circle:");
    double radiusFromUser = input.nextDouble();
    double area = radiusFromUser * radiusFromUser * PI;
    System.out.print("Circle with radius " + radiusFromUser);
    System.out.println(" has an area of " + area);
  }
}
```

If the user enters a positive radius value as expected:

```
Enter the radius of a circle:
3
Circle with radius 3.0 has an area of 28.26
```

However, if the user enters a negative radius value:

```
Enter the radius of a circle:
-3
Circle with radius -3.0 has an area of 28.26
```

Not Equal to

!

! ?

```
int x = 3;
int y = 4;
int z = 4;
```

T

3 4
x != y

T

T
3 4
!(x == y)

F

y != z
4 4

!(y == z)
4 4
T

F

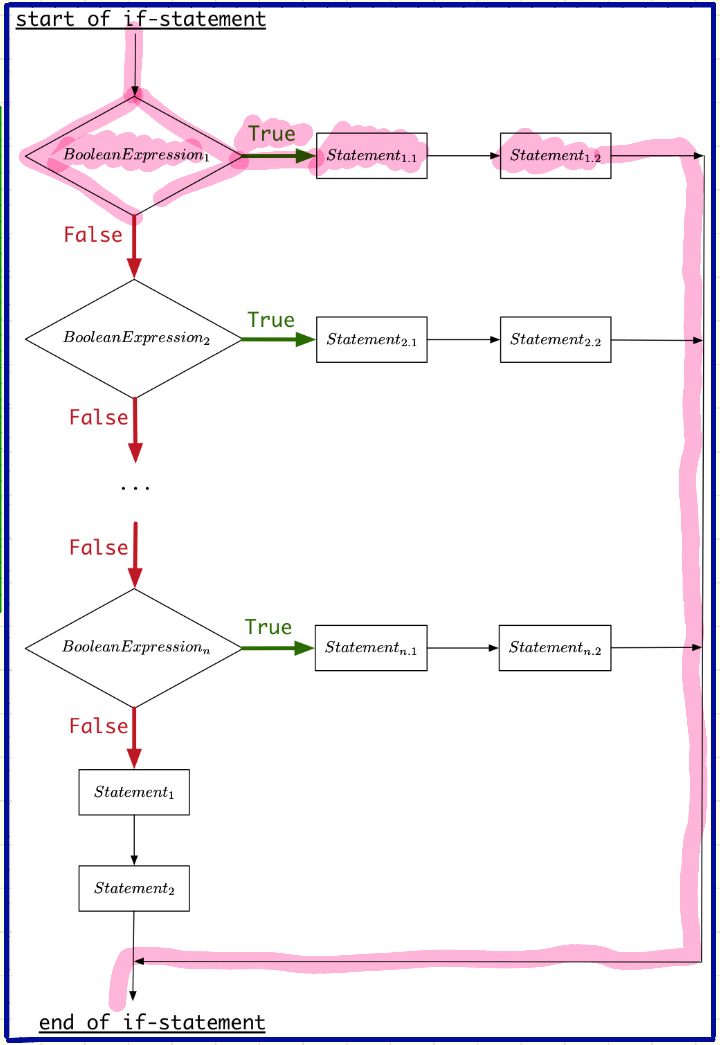
A Single If-Statement

```
if ( BooleanExpression_1 ) {      /* Mandatory */
  Statement_1.1;  Statement_2.1;
}
else if ( BooleanExpression_2 ) {  /* Optional */
  Statement_2.1;  Statement_2.2;
}
... /* as many else-if branches as you like */
else if ( BooleanExpression_n ) {  /* Optional */
  Statement_n.1;  Statement_n.2;
}
else {  /* Optional */
  /* when all previous branching conditions ar
  Statement_1;  Statement_2;
}
```

Syntax

Case I: BooleanExpression$_1$ evaluates to true

Semantics/ Meaning



Only **first** satisfying branch *executed*; later branches *ignored*.

```
int i = -4;
if(i < 0) {
  System.out.println("i is negative");
}
else if(i < 10) {
  System.out.println("i is less than than 10");
}
else if(i == 10) {
  System.out.println("i is equal to 10");
}
else {
  System.out.println("i is greater than 10");
}
```

-4 T

-4 T

```
i is negative
```

A Single If-Statement

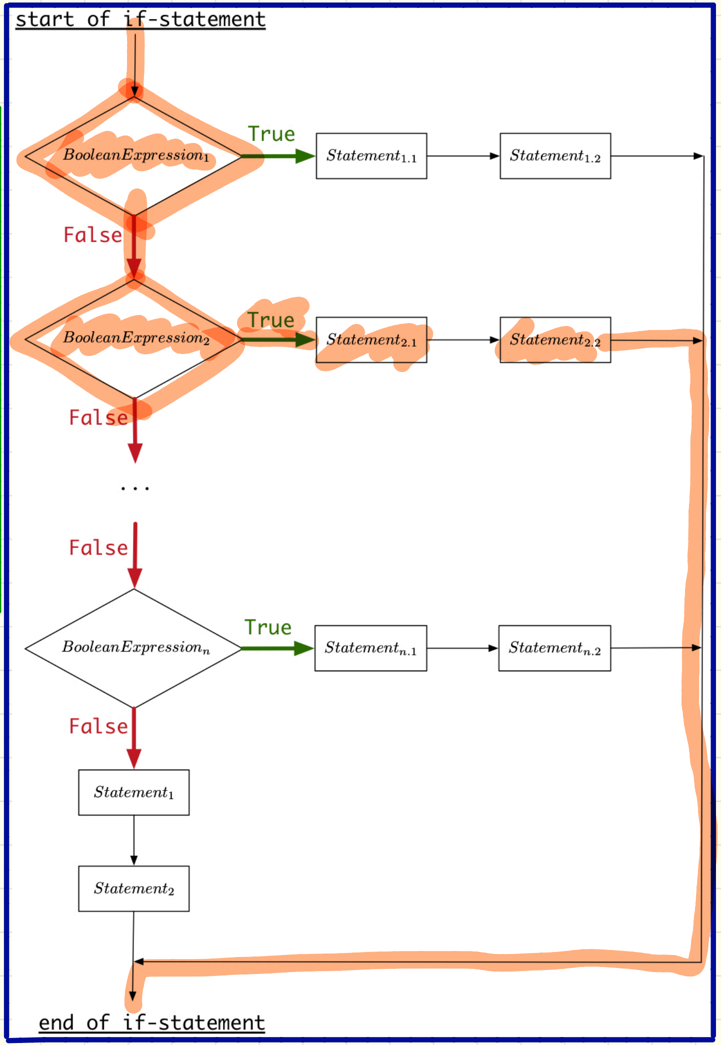
```
if ( BooleanExpression1 ) {     /* Mandatory */
  Statement1.1;  Statement2.1;
}
else if ( BooleanExpression2 ) {     /* Optional */
  Statement2.1;  Statement2.2;
}
... /* as many else-if branches as you like */
else if ( BooleanExpressionn ) {     /* Optional */
  Statementn.1;  Statementn.2;
}
else {  /* Optional */
  /* when all previous branching conditions ar
  Statement1;  Statement2;
}
```

F

Syntax

Case 2: BooleanExpression1 evaluates to false but BooleanExpression2 evaluates to true

Semantics/ Meaning



Only **first** satisfying branch *executed*; later branches *ignored*.

```
int i = 5;          5
if (i < 0) F {
x  System.out.println("i is negative");
}                        5    T
-> else if (i < 10)  {
  System.out.println("i is less than than 10");
}
else if(i == 10)  {
  System.out.println("i is equal to 10");
}
else  {
  System.out.println("i is greater than 10");
}
```

```
i is less than 10
```

A Single If-Statement

```
if ( BooleanExpression₁ ) {   /* Mandatory */
  Statement₁.₁;  Statement₂.₁;
}
else if ( BooleanExpression₂ ) {   /* Optional */
  Statement₂.₁;  Statement₂.₂;
}
... /* as many else-if branches as you like */
else if ( BooleanExpressionₙ ) {   /* Optional */
  Statementₙ.₁;  Statementₙ.₂;
}
else {  /* Optional */
  /* when all previous branching conditions ar
  Statement₁;  Statement₂;
}
```

F

F

F

X

X

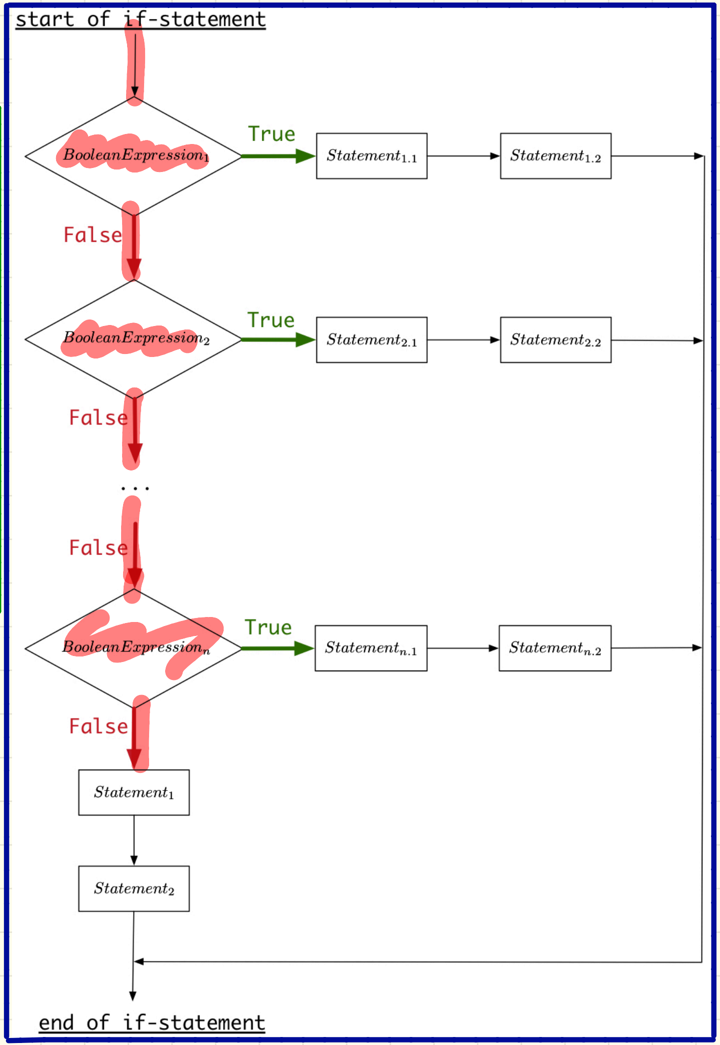
X

Syntax

Case 3 : BooleanExpression1 evaluates to false

but BooleanExpression2 evaluates to false

Semantics/ Meaning



A Single If-Statement

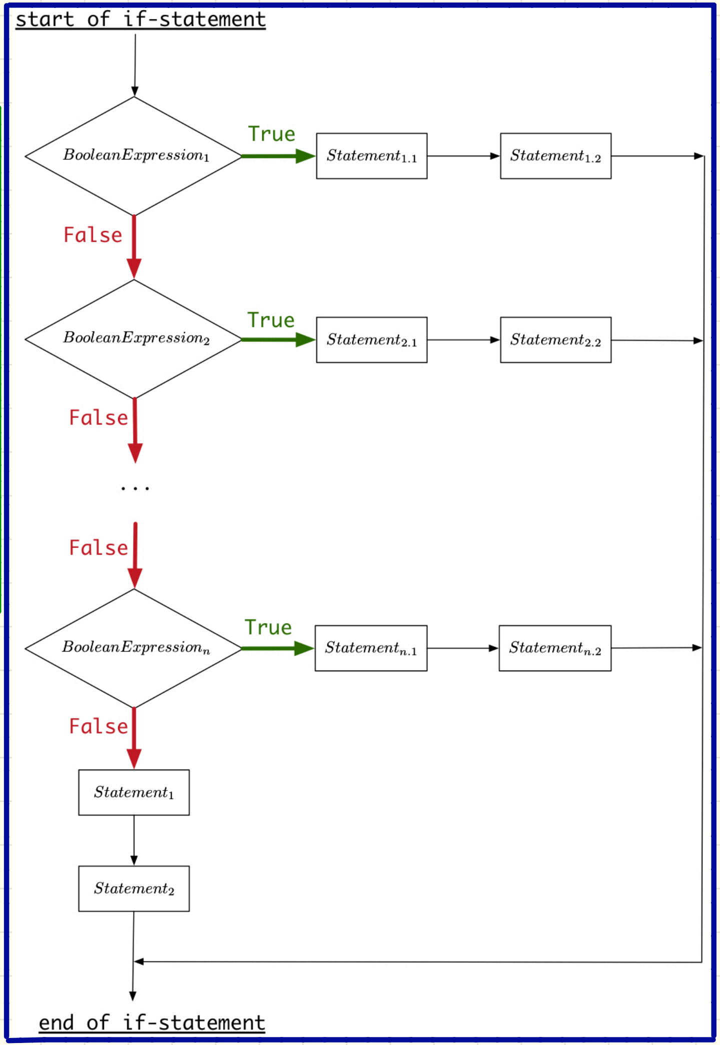
```
if ( BooleanExpression1 ) {   /* Mandatory */
  Statement1.1;  Statement2.1;
}
else if ( BooleanExpression2 ) {   /* Optional */
  Statement2.1;  Statement2.2;
}
... /* as many else-if branches as you like */
else if ( BooleanExpressionn ) {   /* Optional */
  Statementn.1;  Statementn.2;
}
else {  /* Optional */
  /* when all previous branching conditions ar
  Statement1;  Statement2;
}
```

default.

Syntax

$\boxed{\text{Case 4}}$: BooleanExpression1 ~ BooleanExpressionn all evaluate to false

Semantics/Meaning



No satisfying branches, and no `else` part, then *nothing* is executed.

```
int i = 12;
if(i < 0) {
  System.out.println("i is negative");
}
else if(i < 10) {
  System.out.println("i is less than than 10");
}
else if(i == 10) {
  System.out.println("i is equal to 10");
}
```

12 F

12 F

12 F

x

x

x

No satisfying branches, then `else` part, if there, is *executed*.

```
int i = 12;
if (i < 0) F {
X System.out.println("i is negative");
}
            F
else if (i < 10)  {
X System.out.println("i is less than than 10");
                         F
else if (i == 10)  {
X System.out.println("i is equal to 10");
}         → default
else {
  System.out.println("i is greater than 10");
}
```

```
i is greater than 10
```

Multi-Way If-Statement with else Part

```
if (score >= 80.0) {
  System.out.println("A");
}
else if (score >= 70.0) {
  System.out.println("B");
}
else if (score >= 60.0) {
  System.out.println("C");
}
else {
  System.out.println("F");
}
```

```
if (           ) {
       A
}
else {
   if (           ) {
        B
   }
   else {
      if (          ) { C }
      else { F };
   }
}
```

always evaluated at runtime? No, only if score >= 80 evaluates to false.

Multi-Way If-Statement without else part

```
String lettGrade = "F";
if (score >= 80.0) {
  letterGrade = "A";
}
else if (score >= 70.0) {
  letterGrade = "B";
}
else if (score >= 60.0) {
  letterGrade = "C";
}
```

50

50

50

score 50

```
String lg = "";
if (s >= 80) {
 x [ lg = "A";
}
else if (s >= 70) {
 x [ lg = "B";
}
else if (s >= 60) {
 x [ lg = "C"
}
→ else { lg = "F"; }
```

radius

invalid : $\text{radius} < 0$

valid : $!(\text{invalid})$

$\hookrightarrow\ !(\text{radius} < 0)$

$\hookrightarrow\ \text{radius} >= 0$

Two Ways to Handling Errors

Test: radius is 9

Test: radius is -5

```
public class ComputeArea {
  public static void main(String[] args) {
    System.out.println("Enter a radius value:");
    Scanner input = new Scanner(System.in);
    double radius = input.nextDouble();
    final double PI = 3.14159;
    if (radius < 0) {  /* condition of invalid inputs */
      System.out.println("Error: Negative radius value!");
    }
    else {  /* implicit:  !(radius < 0), or radius >= 0 */
      double area = radius * radius * PI;
      System.out.println("Area is " + area);
    }
  }
}
```

```
public class ComputeArea2 {
  public static void main(String[] args) {
    System.out.println("Enter a radius value:");
    Scanner input = new Scanner(System.in);
    double radius = input.nextDouble();
    final double PI = 3.14159;
    if (radius >= 0) {  /* condition of valid inputs */
      double area = radius * radius * PI;
      System.out.println("Area is " + area);
    }
    else {  /* implicit:  !(radius >= 0), or radius < 0 */
      System.out.println("Error: Negative radius value!");
    }
  }
}
```

$-5 >= 0$ F