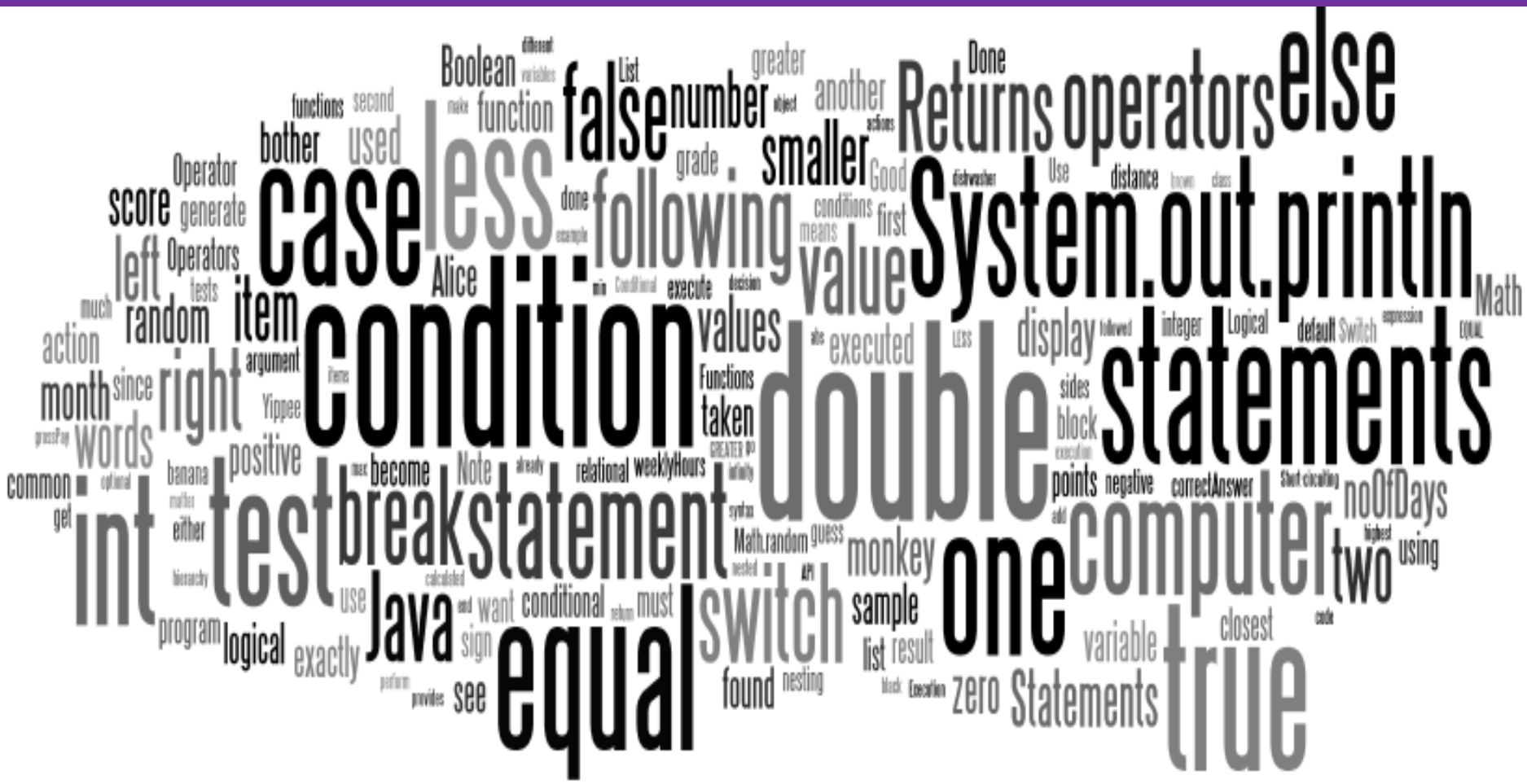
Chapter 4

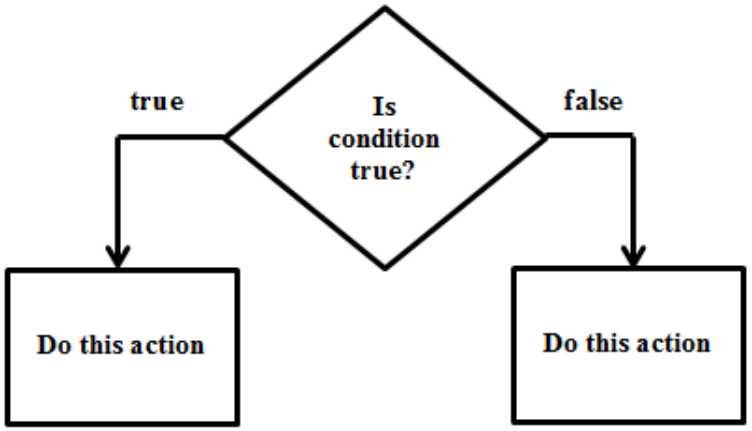


Conditionals

Objectives

- List relational operators.
- List logical operators.
- Use the hierarchy of operators chart to properly construct if/else statements.
- Construct switch statements.
- Use nested if statements.

Decisions



True and false values are also known as Boolean values, named after the 19th century English mathematician George Boole.

Relational Operators

A Boolean test compares primitives, constants, and variables and or objects using the following **relational operators:**

Operator	**Meaning**	**Example**
==	Equal to	x == 3
! =	Not equal to	x != 3
<	Less than	x < 3
>	Greater than	x > 3
<=	Less than or equal to	x <= 3
>=	Greater than or equal to	x >= 3

Note: These relational operators must be typed exactly as above. You can't type a space between the two equal signs and you can't put =< to mean less than or equal.

Alice Example of an if statement :

Math Class

Method Description	Method Call	Result	Argument	Returns
Returns the absolute value	Math.abs(-5.5);	5.5	double	double
Returns the value of the first argument raised to the power of the second argument	Math.pow(5, 2);	25	double, double	double
Returns a positive number that is greater than or equal to 0.0 and less than 1.0.	Math.random( );	Number between 0 and 1	none	double
Returns the closest whole number to the argument	Math.round(6.45);	6	double	double
Returns the rounded positive square root	Math.sqrt(7);	2.6457513	double	double

Example:

Code:

```
double number = 5.49;
System.out.println(Math.round(number));
```

Output:

```
run:
5
BUILD SUCCESSFUL
```

String Methods

```
String s1 = "Now is the winter of our discontent";
String s2 = "Java can be fun and hard ";
```

Method	Result
s1.length( )	35
s2.length( )	25
s1.toUpperCase()	NOW IS THE WINTER OF OUR DISCONTENT
s2.toUpperCase()	JAVA CAN BE FUN AND HARD
s1.toLowerCase()	now is the winter of our discontent
s2.toLowerCase()	java can be fun and hard
s1.startsWith("st")	False
s2.startsWith("Java")	true

String Methods

```
String s1 = "Now is the winter of our discontent";
String s2 = "Java can be fun and hard ";
```

Method	Result
s1.endsWith(“TENT")	false
s2.endsWith("so")	false
s1.replace('e' ,'L')	Now is thL wintLr of our discontLnt
s2.replace('a' , '*')	J*v* c*n be fun *nd h*rd
s1.equals(s2)	false
s1.equalsIgnoreCase(s2)	false
s1.contains("winter")	true
s2.contains("@")	false

Comparing Objects vs Primitives

Comparing Objects

Code:

```
String message1 = "Hello";
String message2 = "hello";
boolean result = message1.equals(message2);
System.out.println(result);
```

Output:

```
run:
false
BUILD SUCCESSFUL
```

Code:

```
DecimalFormat format1=new DecimalFormat(".##");
DecimalFormat format2=new DecimalFormat(".##");
boolean result=format1.equals(format2);
System.out.println(result);
```

Output:

```
run:
true
BUILD SUCCESSFUL
```

Comparing Primitives

Code:

```
int a = 1;
int b = 1;
boolean result = a == b;
System.out.println(result);
```

Output:

```
run:
true
BUILD SUCCESSFUL
```

Code:

```
char a = 'N';
char b = 'Y';
boolean result = a == b;
System.out.println(result);
```

Output:

```
run:
false
BUILD SUCCESSFUL
```

String Comparisons

Code:

```
String message1 = "Hello";
String message2 = "hello";
boolean result = message1.equalsIgnoreCase(message2);
System.out.println(result);
```

Output:

```
run:
true
BUILD SUCCESSFUL
```

String Comparisons

Code:

```
String message1 = "Hello ";
String message2 = "hello";
message1.trim();
message2.trim();
boolean result = message1.equalsIgnoreCase(message2);
System.out.println(result);
```

Output:

```
run:
false
BUILD SUCCESSFUL
```

Code:

```
String message1 = "Hello ";
String message2 = "hello";
message1 = message1.trim();
message2 = message2.trim();
boolean result = message1.equalsIgnoreCase(message2);
System.out.println(result);
```

Output:

```
run:
true
BUILD SUCCESSFUL
```

If Statements

The syntax contains the keyword **if**, followed by a **condition (boolean test) in parenthesis**, followed by either a **single statement or a block statement to execute if the test is true.** An **optional else** keyword provides the alternative statement to execute if the test is false.

if (condition)
    { statements to do when conditional is true }
else
    { statements to do when conditional is false }

if/else examples:

```
int x = 1;
if (x < 0) {
    System.out.println("x is negative.");
}
```

```
int x = 1;
if (x < 0) {
    System.out.println("x is negative.");
} else {
    System.out.println("x is positive.");
}
```

```
int x = 1;
if (x < 0) {
    System.out.println("x is negative.");
} else if (x == 0) {
    System.out.println("x is 0.");
} else {
    System.out.println("x is positive.");
}
```

```
double r = Math.random(); // generate a random number
if (r >= .5) {   //fair chance of heads and tails
    System.out.println("Heads");
} else {
    System.out.println("Tails");
}
```

Logical Operators

Logical Operator	Description
&	**AND**
|	**OR**
!	**NOT**
^	**Exclusive OR**

Short-Circuiting Logical Operators

Logical Operator	Description
&&	**Short-curcuiting AND**
||	**Short-circuiting OR**

Using Logical Operators

- The following only prints if x is less than y and x is less than z. The && means if x is NOT less than y it won't even bother to test to see if x is less than z.

```
if ( x < y  &&   x < z)
    { System.out.println ("x is less than both y and z" ); }
```

- The following calculates grossPay if either condition is true. The || means that if weeklyHours is less 40, it won't bother to test to see if employeeType is 'P'.

```
if (weeklyHours < 40   ||  employeeType == 'P'  )
    { grossPay =  weeklyHours * hourlyRate; }
```

Switch Statements

Done with if statements:	Done with switch statements:

if (grade == 'A') {
  System.out.println("Great!");
} **else if** (grade == 'B') {
  System.out.println("Good!");
} **else if** (grade == 'C') {
  System.out.println("Nice.");
} **else** {
  System.out.println("Not Good.");
}
	switch (grade) {
	  case **'A':**
	  System.out.println("Great!");
	  break**;**
	  case **'B':**
	  System.out.println("Good!");
	  break**;**
	  case **'C':**
	  System.out.println("Nice.");
	  break**;**
	  default:
	  System.out.println("Not Good.");
	}

Complex If Statements vs. Switch

Complex If Statements:	Switch Statements:

if (month == 2) {
  days =28;
} else if ((month == 4) || (month == 6) ||
  (month == 9) || (month == 11)) {
  days=30;
} else {
  days=31;
}
	switch (month) {
	  case 2:
	  days = 28;
	  break;
	  case 4: case 6: case 9: case 11:
	  days = 30;
	  break;
	  default:
	  days = 31;
	}