Middlesex University London

Lecture 4

Selections

REVISION TASK

- Anders Ericsson, a professor at Florida State University is a founder of the 10 000 hours theory stating that "10 000 hours practice can make one an expert in a field"
- Knowing the number of practice hours to become an expert, calculate the number of days and a weeks that you would need to spend to practice for that long

THIS WEEK:

- Selections
  - **if** statement
  - **if-else** statements
  - nested **if** and multi-way **if** statements
  - **Switch**
  - Conditional operator

CONTROL FLOW STATEMENTS

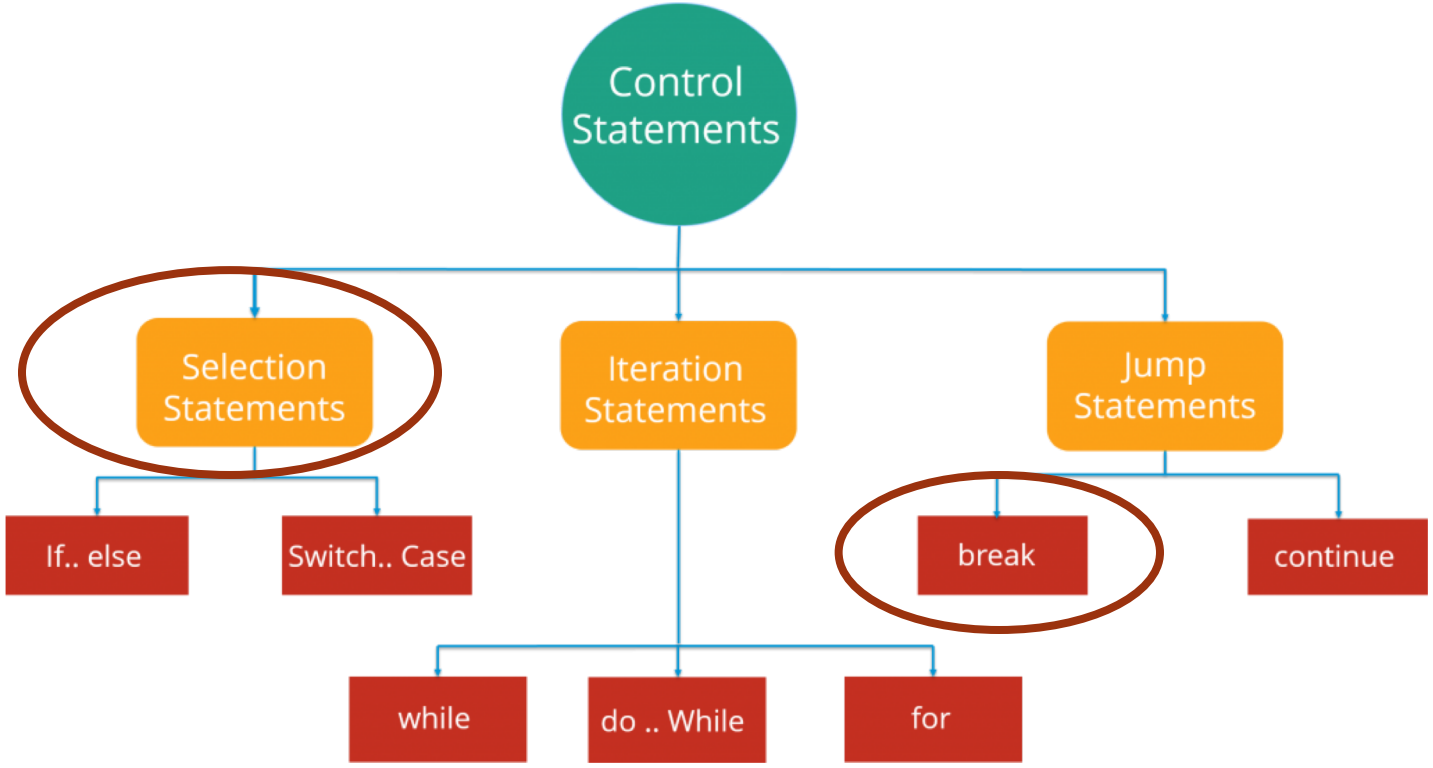


Image taken from:https://www.virginiamayhew.com/flow-chart-basics-programming/java-tutorial-java-programming-made-easy-learn-java-edureka-3/

THE BOOLEAN TYPE AND OPERATORS

- Often in a program you need to compare two values, such as whether *i* is greater than *j*.
  - Java provides six comparison operators (also known as relational operators) that can be used to compare two values.
  - The result of the comparison is a Boolean value: true or false.

COMPARISON OPERATORS

Java operators	Names	Example: score =40	Result
<	Less than	score < 0	false
<=	Less then or equal to	score <= 0	false
>	Greater than	score > 0	true
>=	Greater than or equal to	score >==	true
==	Equal to	score == 0	false
!=	Not equal to	score != 0	true

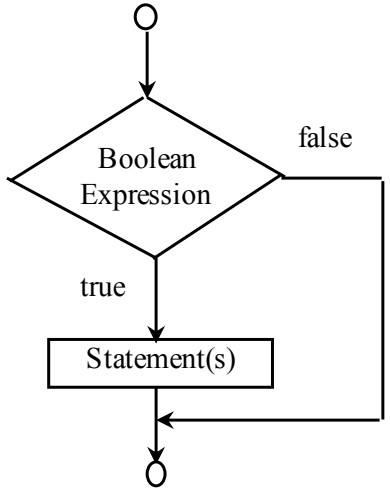
SELECTIONS

- Selections are used for purpose controlling flow of your program
- they tell your program to execute a certain section of code only if a particular test matches the requirements.
- Examples are; if, switch
- The if-then statement is the most basic of all the control flow statements.
- It tells your program to execute a certain section of code only if a particular test evaluates to true.
  - If test score is larger than 40 then mark is pass, otherwise the mark is fail
  - If car speed is larger than 50 then apply breaks, otherwise don't apply breaks
  - If radius is greater or equal to 0 then calculate the area, otherwise don't do anything

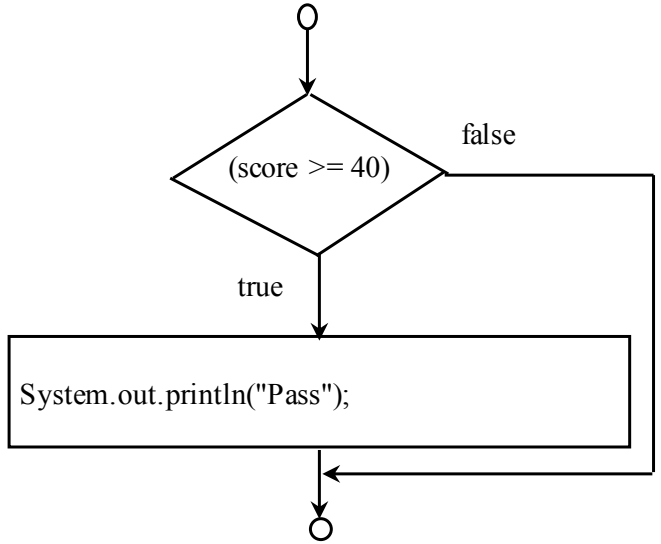
ONE-WAY IF STATEMENTS

```
if (boolean-expression) {
  statement(s)
}
```

```
int score =41;

if(score>40){
    System.out.println("Pass");
}

}
```



(A)

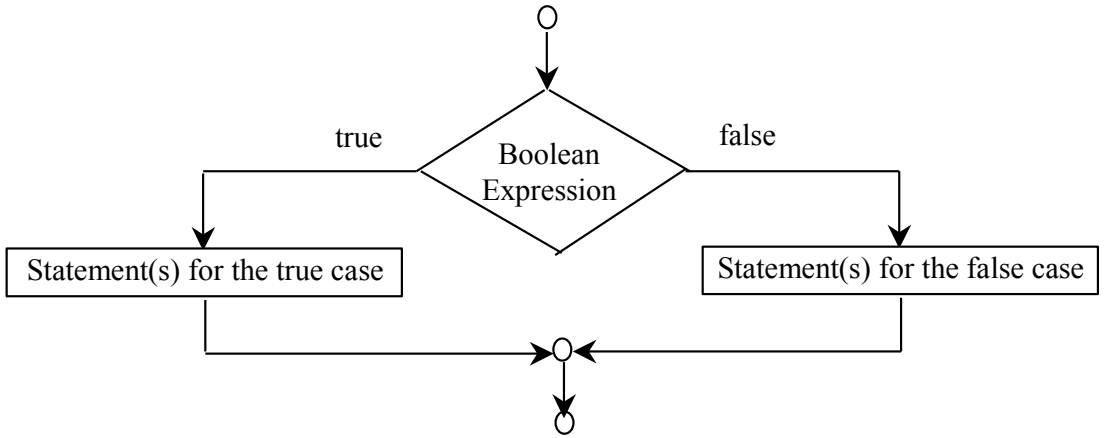


(B)

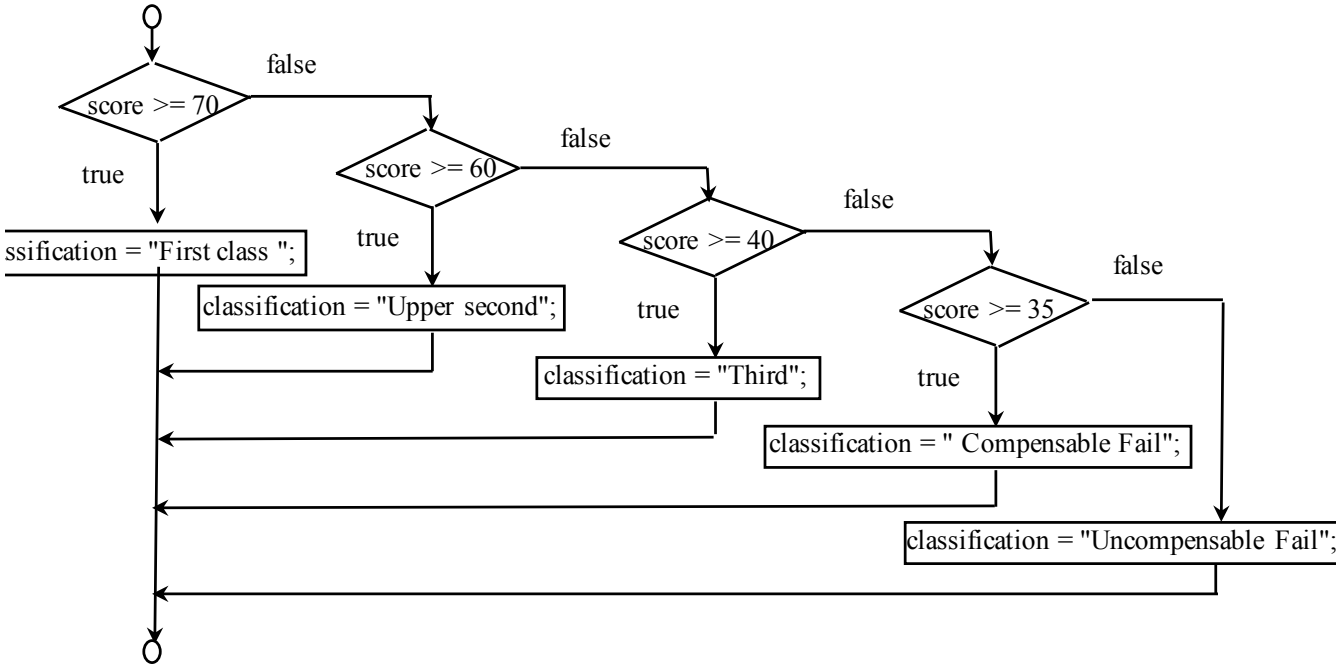
THE TWO-WAY IF STATEMENT

```
if (boolean-expression) {
  statement(s)-for-the-true-case;
}
else {
  statement(s)-for-the-false-case;
}
```

```
int score =41;

if(score>40){
    System.out.println("Pass");
}else{
    System.out.println("Fail");
}
```



NESTED IF




```
    int score =63;

    String classification = "";

    if(score>=70){
        classification = "First class ";
    }else if (score >=60){
        classification = "Upper second class";
    }else if (score >=50){
        classification = "Lower second class";
    }else if (score >=40){
        classification = "Third";
    }else if (score >=35){
        classification = "Compositable Fail";
    }else {
        classification = "Uncompositable Fail";
    }

    System.out.println(classification);

}
```

TRACE IF-ELSE STATEMENT

```
int score =63;

String classification = "";

if(score>=70){
    classification = "First class ";
}else if (score >=60){
    classification = "Upper second class";
}else if (score >=50){
    classification = "Lower second class";
}else if (score >=40){
    classification = "Third";
}else if (score >=35){
    classification = "Compositable Fail";
}else {
    classification = "Uncompositable Fail";
}

System.out.println(classification);

}
```

The condition is false

TRACE IF-ELSE STATEMENT

```
    int score =63;

    String classification = "";

    if(score>=70){
        classification = "First class ";
    }else if (score >=60){
        classification = "Upper second class";
    }else if (score >=50){
        classification = "Lower second class";
    }else if (score >=40){
        classification = "Third";
    }else if (score >=35){
        classification = "Compositable Fail";
    }else {
        classification = "Uncompositable Fail";
    }

    System.out.println(classification);

}
```

The condition is true

TRACE IF-ELSE STATEMENT

```
int score =63;

String classification = "";

if(score>=70){
    classification = "First class ";
}else if (score >=60){
    classification = "Upper second class";
}else if (score >=50){
    classification = "Lower second class";
}else if (score >=40){
    classification = "Third";
}else if (score >=35){
    classification = "Compositable Fail";
}else {
    classification = "Uncompositable Fail";
}

System.out.println(classification);

}
```

classification = “Upper second class”

TRACE IF-ELSE STATEMENT

```
    int score =63;

    String classification = "";

    if(score>=70){
        classification = "First class ";
    }else if (score >=60){
        classification = "Upper second class";
    }else if (score >=50){
        classification = "Lower second class";
    }else if (score >=40){
        classification = "Third";
    }else if (score >=35){
        classification = "Compositable Fail";
    }else {
        classification = "Uncompositable Fail";
    }

    System.out.println(classification);

}
```

Executes the println statement

TASK

- Write a program when you ask user for a number (using Scanner) and print even if the number is even or odd if the number is odd.

IF ELSE IN A PRINT OUT STATEMENT

```
(boolean-expression) ? exp1 : exp2
```

```
Scanner sc = new Scanner(System.in);
System.out.println("Enter a number to check if the number is even or odd");
int num = sc.nextInt();

System.out.println((num % 2 == 0)? num + " is even" : num + " is odd");
```

LOGICAL OPERATORS

If then else statement can use logical operator to check for two conditions

Operator	*Name*	**Example (assume age = 24, gender = 'F')**
!	not	!(age > 18) is false, because (age > 18) is true.
&&	and	(age > 18) && (gender == 'F') is true, because (age > 18) and (gender == 'F') are both true.
||	or	(age > 34) || (gender == 'F') is true, because (gender == 'F') is true
^	exclusive or	(age > 34) ^ (gender == 'F') is true, because (age > 34) is false but (gender == 'F') is true.

TRUTH TABLE FOR OPERATORS

! not	
p	!p
true	false
false	true

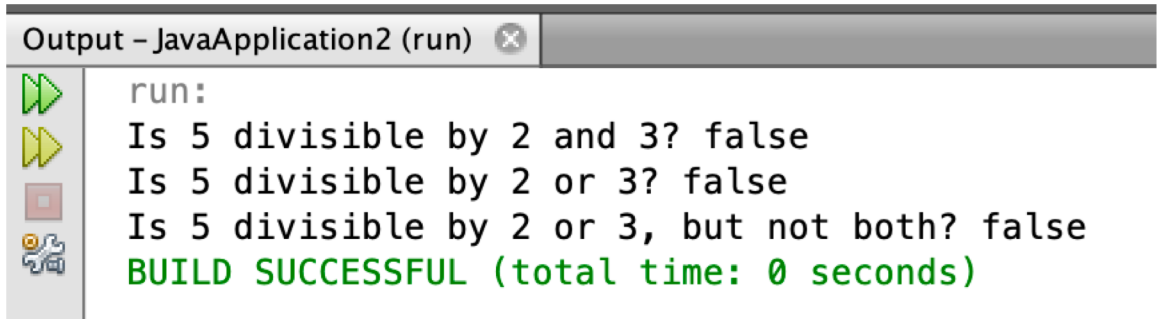
&& and		
p1	p2	p1 && p2
false	false	false
false	true	false
true	false	false
true	true	true

^ exclusive or		
p1	p2	p1 ^ p2
false	false	false
false	true	true
true	false	true
true	true	false

II or		
p1	p2	p1 && p2
false	false	false
false	true	true
true	false	true
true	true	true

- Here is a program that checks
  - whether a number is exactly divisible by 2 and 3,
  - whether a number is exactly divisible by 2 or 3, and
  - whether a number is exactly divisible by 2 or 3 but not both:

```
   int number = 5;
System.out.println("Is " + number + " divisible by 2 and 3? " +((number % 2 == 0) && (number % 3 == 0)));

System.out.println("Is " + number + " divisible by 2 or 3? " +((number % 2 == 0) || (number % 3 == 0)));

System.out.println("Is " + number +  " divisible by 2 or 3, but not both? " +((number % 2 == 0) ^ (number % 3 == 0)));
```



```java
int score  =60;

if (score >= 40 && score<=49) {
    System.out.println("Pass");
} else if (score >= 50 && score <= 69) {
    System.out.println("Merit");
} else if (score >= 70 && score <= 100){
    System.out.println("Distinction");
} else if (score < 39) {
    System.out.println("Fail");
} else {
    System.out.println("Incorrect input");
}
```

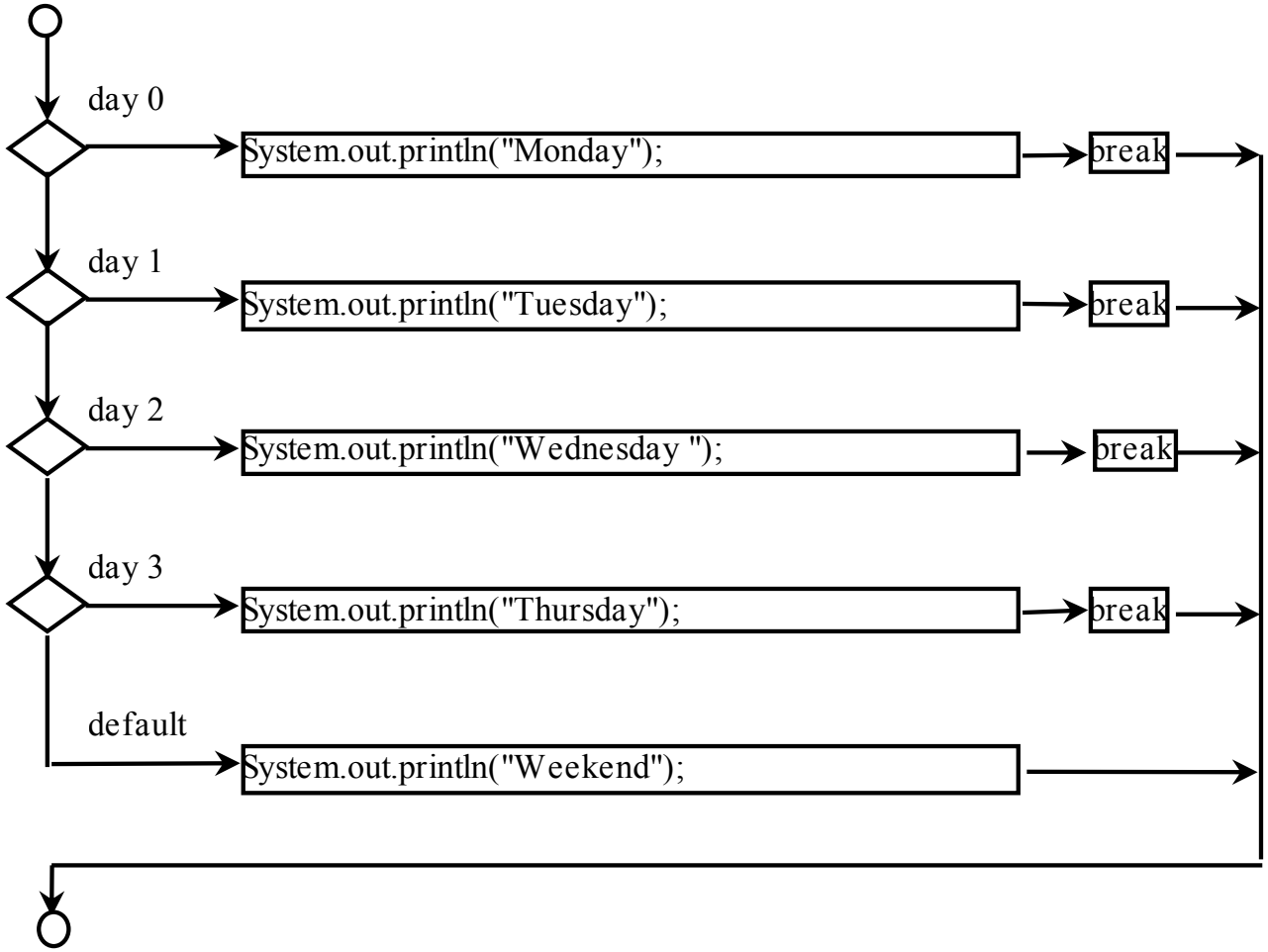
SWITCH STATEMENT

- Switch statement is used for purpose of testing equality against a list of items.
- Each value is called a case, and the variable being switched on is checked for each case.
- Switch statement can operate on byte, short, char, and int, String variables

BREAK STATEMENT

- break is a key word is Java and cannot be used for any other operation
- break is part of branching statements family
- It is used for purpose of escaping the switch statement by preventing further case execution

SWITCH STATEMENT



ANATOMY OF SWITCH STATEMENT

Variable that switch statement will operate on

Case which checks if variable is equivalent to 1

Declaration of the switch statement and variable will it operate on

Code that will be executed if the case is matched

Break statement is used to prevent from further cases being executed

Default statement is called when none of the cases are matched

```
int day = 2;

switch(day)
{
    case 1:
        System.out.println("Monday");
        break;
    case 2:
        System.out.println("Tuesday");
        break;
    case 3:
        System.out.println("Wednesday");
        break;
    case 4:
        System.out.println("Thursday");
        break;
    case 5:
        System.out.println("Friday");
        break;
    default:
        System.out.println("Weekend");
        break;
}
```

TRACE SWITCH STATEMENT

```
int day = 2;

switch(day)
{
    case 1:
        System.out.println("Monday");
        break;
    case 2:
        System.out.println("Tuesday");
        break;
    case 3:
        System.out.println("Wednesday");
        break;
    case 4:
        System.out.println("Thursday");
        break;
    case 5:
        System.out.println("Friday");
        break;
    default:
        System.out.println("Weekend");
        break;
}
```

Match case 2

TRACE SWITCH STATEMENT

```
int day = 2;

switch(day)
{
    case 1:
        System.out.println("Monday");
        break;
    case 2:
        System.out.println("Tuesday");
        break;
    case 3:
        System.out.println("Wednesday");
        break;
    case 4:
        System.out.println("Thursday");
        break;
    case 5:
        System.out.println("Friday");
        break;
    default:
        System.out.println("Weekend");
        break;
}
```

Executes the println statement

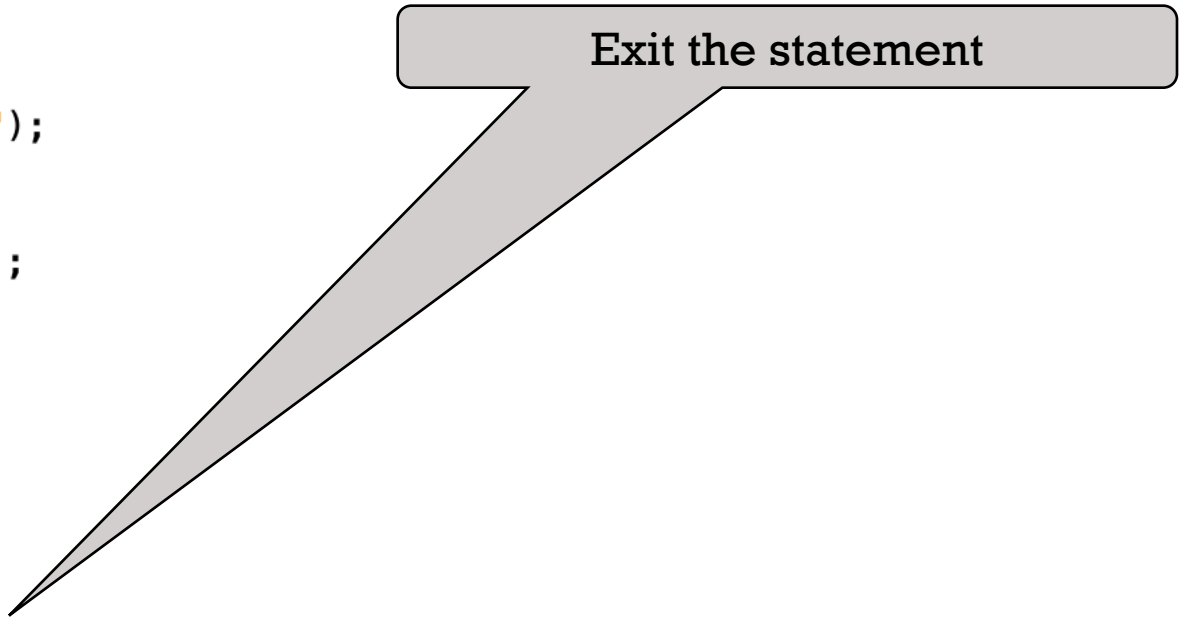
TRACE SWITCH STATEMENT

```
int day = 2;

switch(day)
{
    case 1:
        System.out.println("Monday");
        break;
    case 2:
        System.out.println("Tuesday");
        break;
    case 3:
        System.out.println("Wednesday");
        break;
    case 4:
        System.out.println("Thursday");
        break;
    case 5:
        System.out.println("Friday");
        break;
    default:
        System.out.println("Weekend");
        break;
}
```

Encounter break

TRACE SWITCH STATEMENT

```
int day = 2;

switch(day)
{
    case 1:
        System.out.println("Monday");
        break;
    case 2:
        System.out.println("Tuesday");
        break;
    case 3:
        System.out.println("Wednesday");
        break;
    case 4:
        System.out.println("Thursday");
        break;
    case 5:
        System.out.println("Friday");
        break;
    default:
        System.out.println("Weekend");
        break;
}
```



EXAMPLE OF SWITCH STATEMENT

```
int grade =5;
String classification = "";

switch(grade){
    case 1:
    case 2:
    case 3:
    case 4:
        classification = "First class"; break;
    case 5:
    case 6:
    case 7:
    case 8:
        classification ="Upper second class"; break;
    case 9:
    case 10:
    case 11:
    case 12:
        classification ="Lower second class"; break;
    case 13:
    case 14:
    case 15:
    case 16:
        classification ="Third"; break;
    case 17:
    case 18:
        classification ="Compensatable fail"; break;
    case 19:
        classification ="Uncompensatable fail"; break;
}
System.out.println(classification);
```

KEY THINGS YOU SHOULD UNDERSTAND AFTER WEEK 4

- Basic relational operators and Boolean types
- Basic conditional statements with if-else clauses
- Logical operators and evaluation of conditional expressions
- Switch-case statements

MASTERING YOUR SKILLS

- Read chapter 3 from the Liang book and finish all exercises from the end of the chapter (p. 130-140)
- Extra problems to solve on next slides..

PROBLEM:
A SIMPLE MATH LEARNING TOOL

- This example creates a program to let a first grader practice additions.
- The program randomly generates two single-digit integers number1 and number2 and displays a question such as "What is 7 + 9?" to the student.
- After the student types the answer, the program displays a message to indicate whether the answer is true or false

SIMPLE IF DEMO

- Write a program that prompts the user to enter an integer.
- If the number is a multiple of 5, print HiFive. If the number is divisible by 2, print HiEven.

PROBLEM:
AN IMPROVED MATH LEARNING TOOL

- This example creates a program to teach a first grade child how to learn subtractions.
- The program randomly generates two single-digit integers number1 and number2 with number1 >= number2 and displays a question such as "What is 9 – 2?" to the student.
- After the student types the answer, the program displays whether the answer is correct.

PROBLEM: BODY MASS INDEX

- Body Mass Index (BMI) is a measure of health on weight. It can be calculated by taking your weight in kilograms and dividing by the square of your height in meters. The interpretation of BMI for people 16 years or older is as follows:

BMI	Interpretation
BMI < 18.5	Underweight
18.5 <= BMI < 25.0	Normal
25.0 <= BMI < 30.0	Overweight
30.0 <= BMI	Obese