

Self-Learning Material #2

Decisions and Conditions

Overview

- `if` statement & `if-else` statement
- Nested `if` Block
- `switch` statement
- compare `if-else` vs conditional operation `? :`

Decision and Conditions

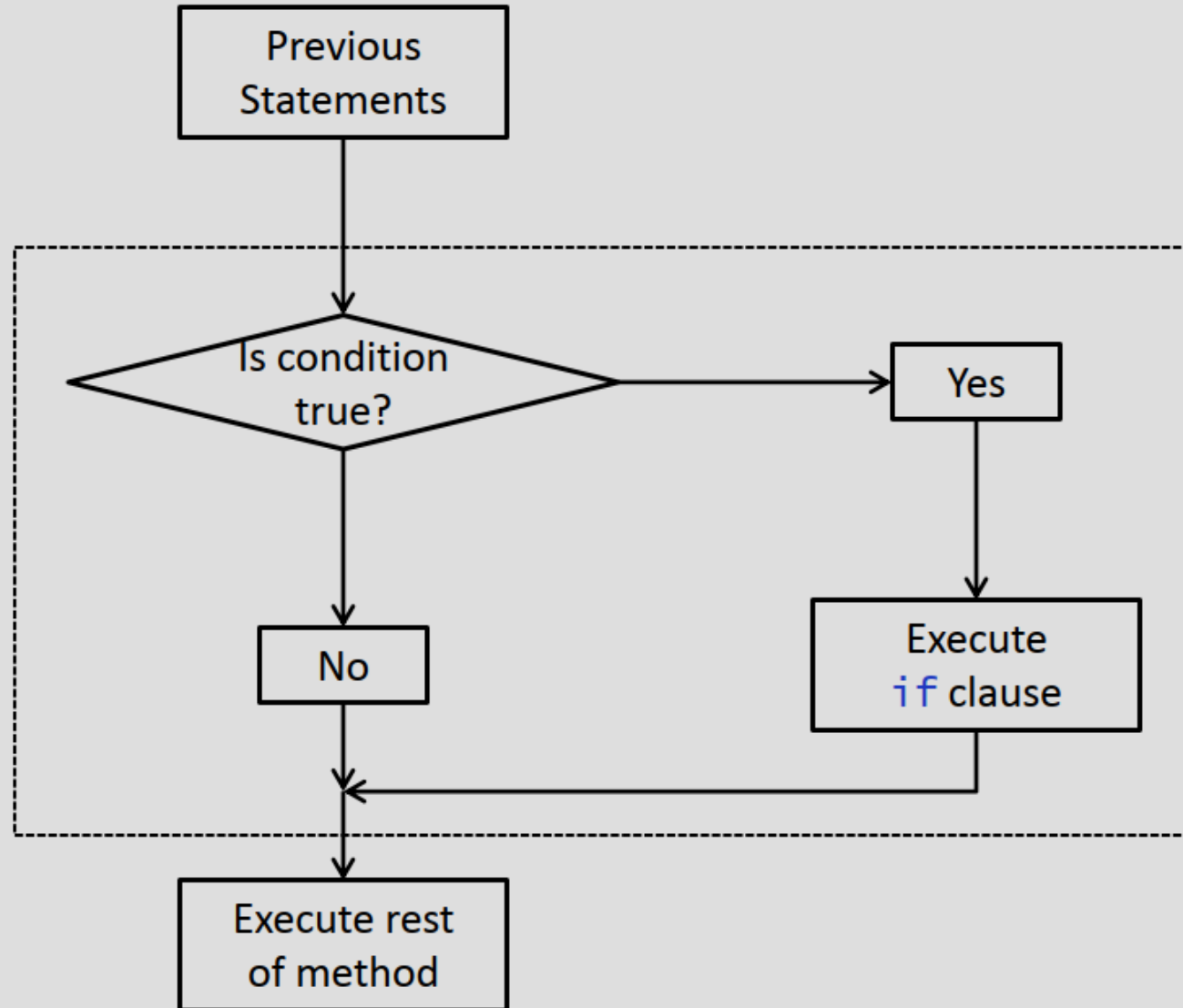
Program takes action based on different scenarios:

- If **temperature higher than 50**, fan speed increases
- If **exam comes**, study; else play
- If **hungry and thirsty**, eat congee; else if **hungry**, eat jelly.

Conditions can be evaluated as true or false. This *controls* how the program works.



Decision Flow chart



If statement

- Java use the follow syntax to make if-decision

```
if (<condition>) {  
    //code to be executed if the condition is true  
}  
//some people do this  
if (<condition>)  
{  
    //code to be executed if the condition is true  
}
```

or



```
if (<condition>)  
    //single line of statement to be executed if the expression is true
```

Example - If

```
if (temperature > 50) {  
    fanSpeed = 50;  
    temperature -= 2;  
}
```

- If `temperature` is not greater than 50, fanSpeed and temperature will not be adjusted.

```
if (examIsComing)  
    study = true;
```

  The indentation and newline are not required by compiler. Yet it is very important for clarity!

Mistake

```
if (examIsOver);  
    play = true;
```

or similarly

```
if (examIsOver); {  
    play = true;  
}
```

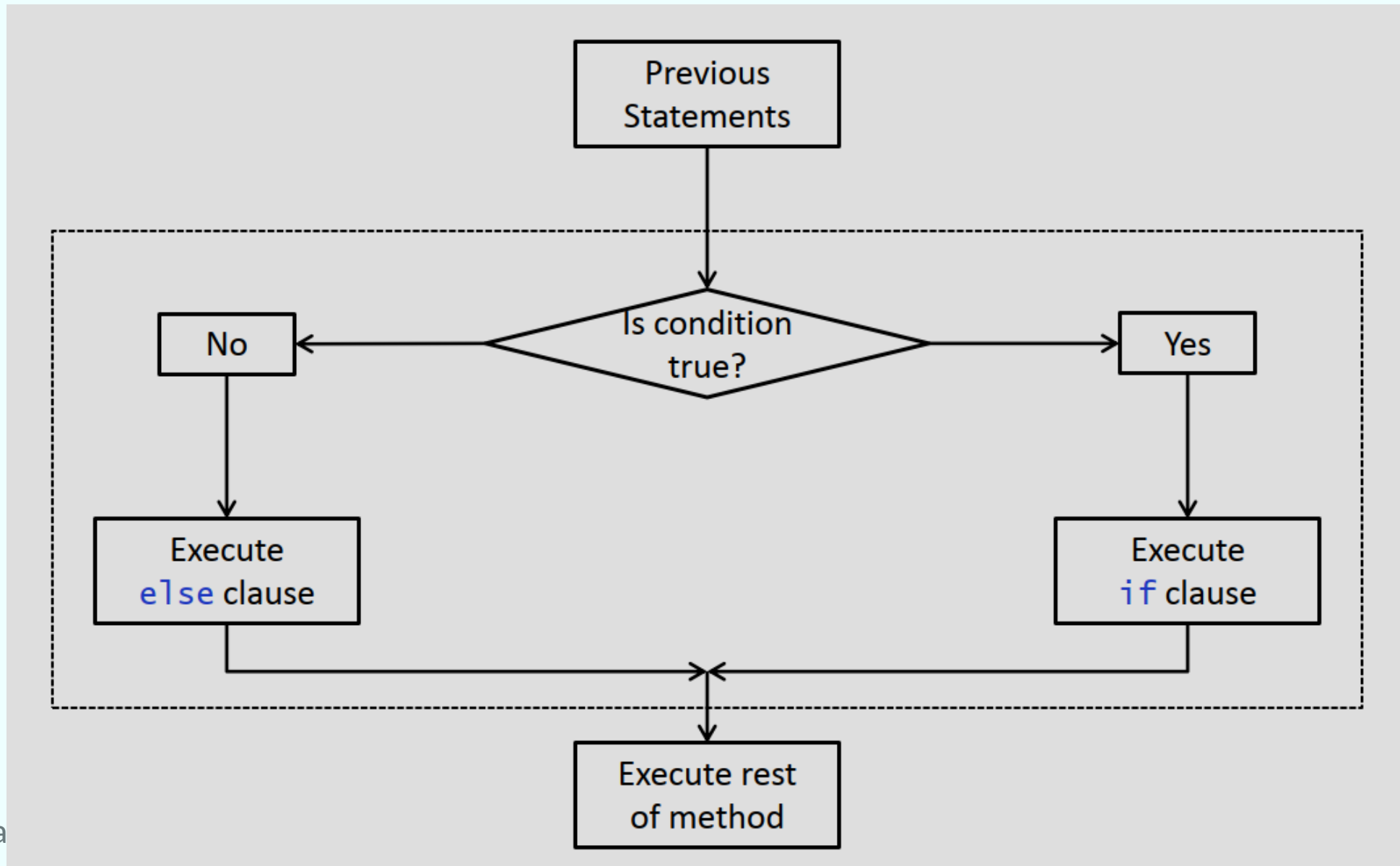
- The first semi-colon creates an empty statement and set play always be true.

Mistake

```
if temperature > 50 {  
    fanSpeed++;  
}
```

- A condition must be surrounded by parenthesis `()`.

Decision Flow chart /w Else



If-Else

- `else` indicates the alternative path when the condition is not true

```
if (<condition>) {  
    //code to be executed if the condition is true  
} else {  
    //code to be executed if the condition is false  
}
```

or if you prefer another indentation

```
if (<condition>)  
{  
    //code to be executed if the condition is true  
}  
else  
{  
    //code to be executed if the condition is false  
}
```

Example

```
if (examIsOver)
    action = "play";
else
    action = "study";
```

```
if (isFree && motivation > 50) {
    action = "go to class";
    knowledge++;
} else {
    action = "go to bed";
    knowledge = knowledge >> 1;
}
```



Under what scenarios would you go to bed?

Mistakes

```
if (examIsOver)
    action = "play";
    mood += 10;
else
    action = "study";
```

✗ Multiple statements must be embraced by `{ }`.

```
if (examIsOver)
    action = "play";
else (!examIsOver)
    action = "study";
```

✗ `else` can't be followed by a condition.



Mistakes

```
if (50 < mark < 70)
    System.out.println("You've got a B");
```

✗ This is an incorrect comparator. Should be `if (50 < mark && mark < 70).`

```
if (a == b == c)
    System.out.println("A and B and C have the same value");
```

✗ Same as above. Should be `if (a == b && b == c).`

  `50 < mark` will be either true or false. Thus `50 < mark < 70` might be evaluated as `true < 70` which makes no sense at all.

Are they the same?

```
if (a > b) {  
    //statement 1  
} else {  
    //statement 2  
}
```

```
if (a > b) {  
    //statement 1  
}  
if (b <= a) {  
    //statement 2  
}
```

Nested if statements

- Nest `if` statements by putting one inside another

```
if (freeTime > 1.5) {  
    if (hasAssignment) {  
        action = "workOnAssignment";  
    } else {  
        action = "watch Kevin's youtube";  
    }  
} else {  
    action = "play";  
}
```

if-else cascade

- Multiple `if-else` can be chained to describe mutually exclusive decisional statements.

```
if (condition1) {  
    //execute if condition1 is true  
} else if (condition2) {  
    //execute if condition2 is true but the above is false  
} else if (condition3) {  
    //execute if condition3 is true but the above are false  
} else if (condition4) {  
    //execute if condition4 is true but the above are false  
} else {  
    //execute if all the above are false  
}
```



The order of the conditions does matter!

Example

```
if (score > 80) {  
    System.out.println("A");  
} else if (score > 65) {  
    System.out.println("B");  
} else if (score > 40) {  
    System.out.println("C");  
} else {  
    System.out.println("F");  
}
```

- If score is 85, output is...
- If score is 70, output is...
- If score is 55, output is...
- If score is 40, output is...

Example 2

```
if (score > 40) {  
    System.out.println("C");  
} else if (score > 65) {  
    System.out.println("B");  
} else if (score > 80 {  
    System.out.println("A");  
} else {  
    System.out.println("F");  
}
```

- If score is 85, output is...
- If score is 70, output is...
- If score is 55, output is...
- If score is 40, output is...

Cascade vs Nested if

- The following codes produce the same result.

Nested If

```
if (score > 40)
    if (score > 65)
        if (score > 80)
            System.out.println("A");
        else
            System.out.println("B");
    else
        System.out.println("C");
else
    System.out.println("D");
```

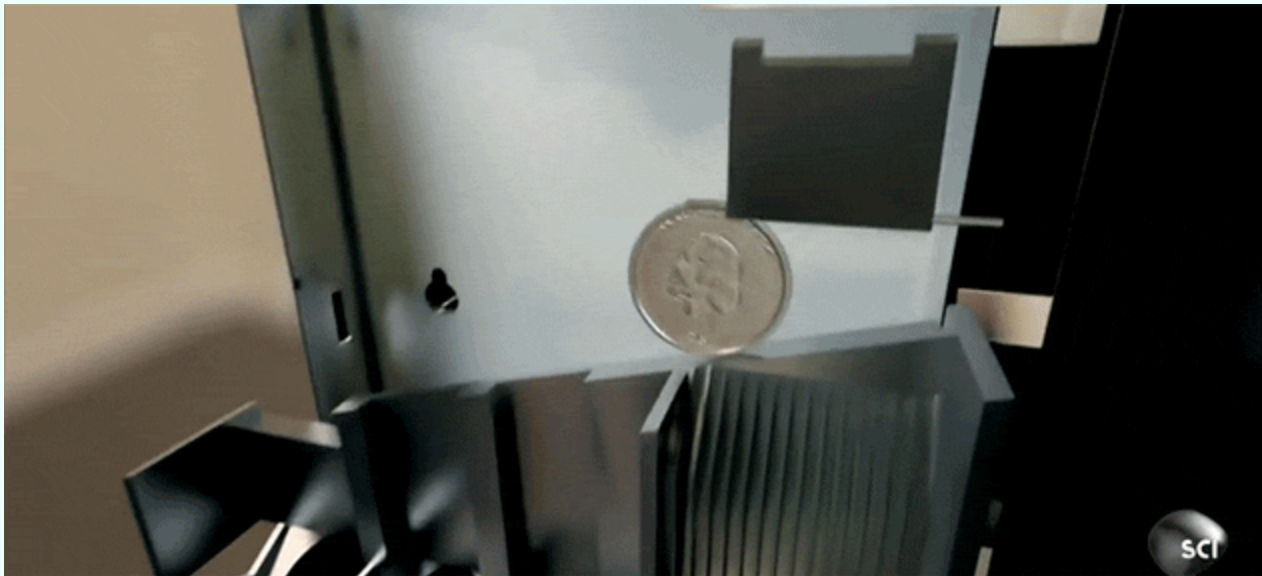
Cascade If

```
if (score > 80)
    System.out.println("A");
else if (score > 65)
    System.out.println("B");
else if (score > 40)
    System.out.println("C");
else
    System.out.println("D");
```

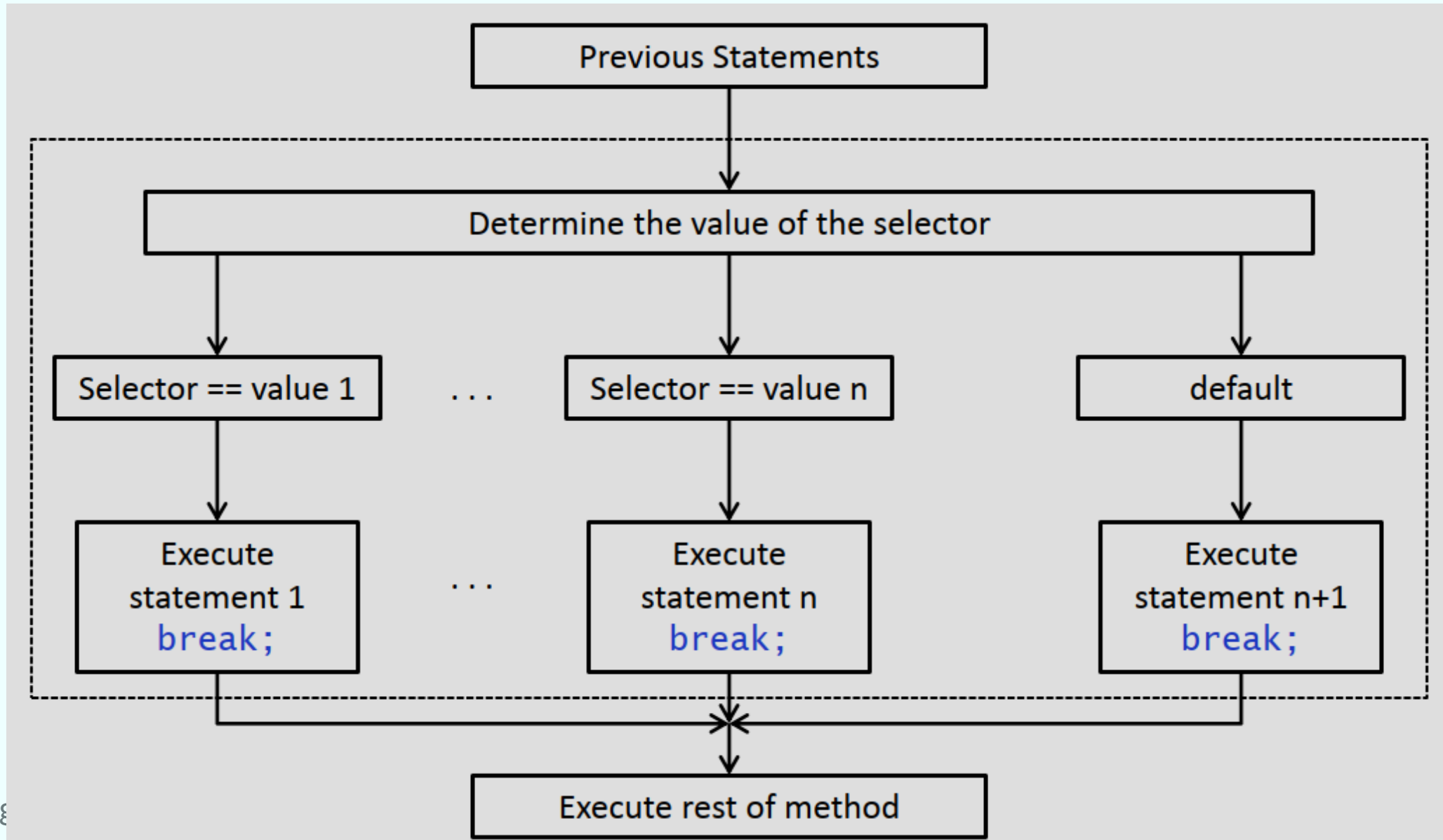
Switch

Switch Statement

- Cascade/nested `if` are good enough to describe our logic.
- Can we do it simpler?
- `switch` is used to choose among more than two mutually exclusive choices based on the values of a variable.
- `switch` works like a vending machine.



Flow chart of a switch statement



Syntax of switch

```
switch (expression) {  
    case 1: //action to be taken, if variable = 1  
        break;  
    case 4: //action to be taken, if variable = 4  
        //another action to be taken  
        break;  
    case 2: //action to be taken, if variable = 2  
        break;  
    default:  
        //action to be taken otherwise  
}
```

The **expression** must be either:

- integer numbers (`byte`, `short`, `int`)
- `char`
- `String`

Example - switch with int

```
int rank;  
//assign some value to rank  
switch (rank) {  
    case 1: System.out.println("Gold medal!");  
        break;  
    case 2: System.out.println("Silver medal!");  
        break;  
    case 3: System.out.println("Bronze medal!");  
        break;  
    default:  
        System.out.println("Thank you for participation!");  
}
```


Example - switch with String

```
String language;  
//assign some value to language  
switch (language) {  
    case "Cantonese": System.out.println("Jou sun ching");  
                        break;  
    case "English"    : System.out.println("Morning!");  
                        break;  
    case "Japanese"   : System.out.println("Ohayo");  
                        break;  
    default           : System.out.println("Hi");  
}
```



It is not necessary to align the colon symbols.

break in a switch

- `break` statement can only appear in a switch statement or a loop
- `break` will jump to the end of the switch statement and execute the rest of the code

```
switch (grade) {  
    case 'A': System.out.println(  
        "Well done! Do you want to be a helper next year?"  
    );  
    break; // goto exit-point  
    case 'B': System.out.println("Good Job!");  
    break; // goto exit-point  
    case 'C': System.out.println("You can do better!");  
    break; // goto exit-point  
    default: System.out.println("Contact your TA immediately!!");  
}  
// Exit-point
```

switch without a break

```
switch (grade) {  
    case 'A': System.out.println(  
        "Well done! Do you want to be a helper next year?"  
    );  
    case 'B': System.out.println("Good Job!");  
    case 'C': System.out.println("You can do better!");  
    default: System.out.println("Contact your TA immediately!!");  
}  
// Exit-point
```

If `grade == 'A'`, it will print

```
Well done! Do you want to be a helper next year?  
Good Job!  
You can do better!  
Contact your TA immediately!!
```

Mistakes

```
switch (4.5 + 3.5) {  
    case 8: System.out.println("8");  
        break;  
    default: System.out.println("not 8");  
}
```

✗ expression cannot be a double.

Mistakes

```
switch (grade) {  
    case 'A', 'B', 'C': System.out.println("Pass");  
                        break;  
    case 'F': System.out.println("Fail");  
}
```

⚠ cases cannot be separated by commas (until Java 13). We don't accept that too!

```
//write this instead  
switch (grade) {  
    case 'A':  
    case 'B':  
    case 'C': System.out.println("Pass");  
                break;  
    case 'F': System.out.println("Fail");  
}
```

Summary

- Switch
- Break