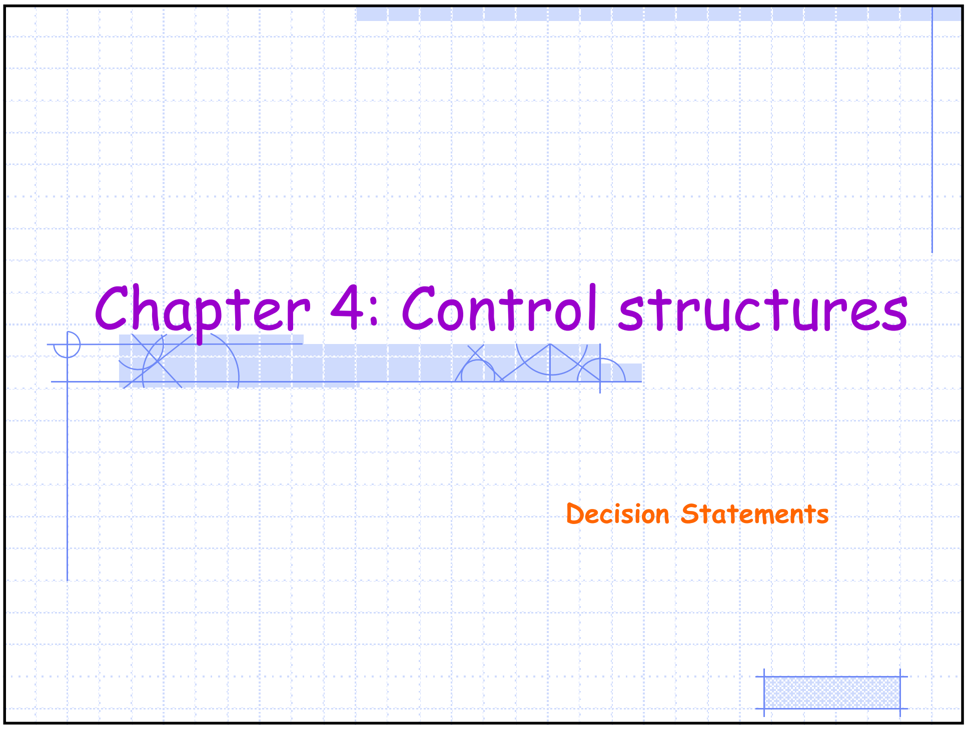
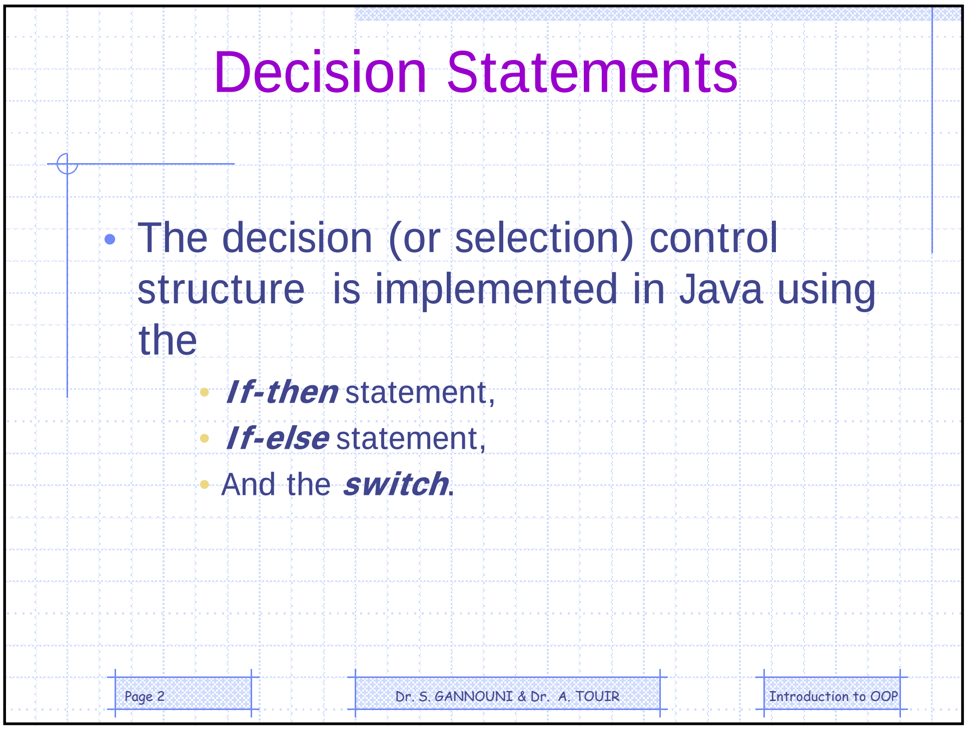


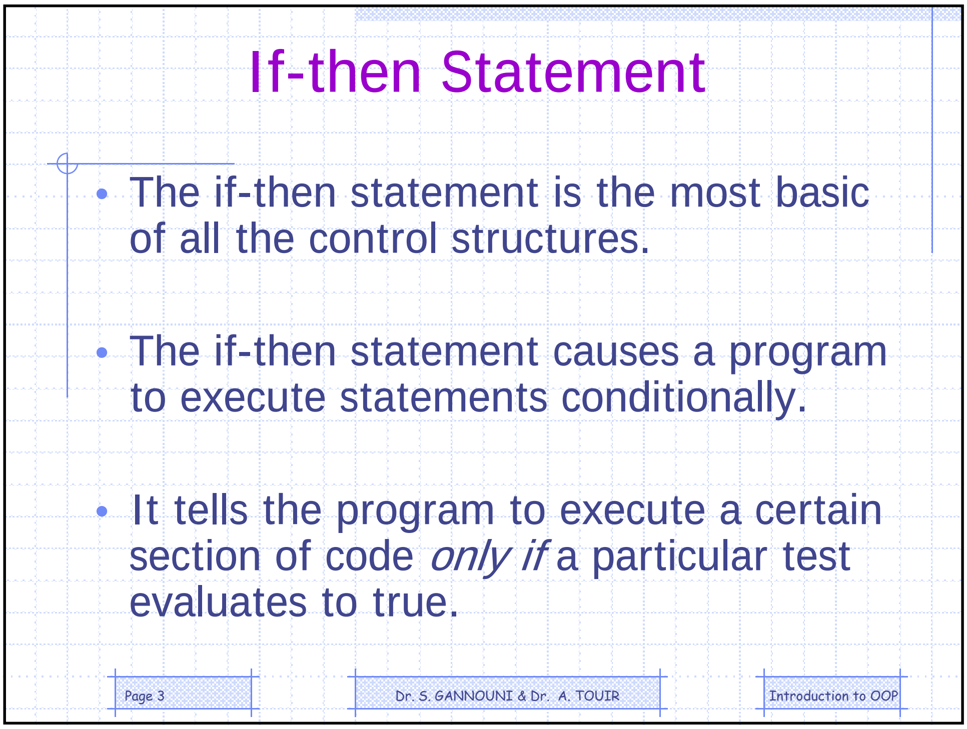
Chapter 4: Control structures

Decision Statements



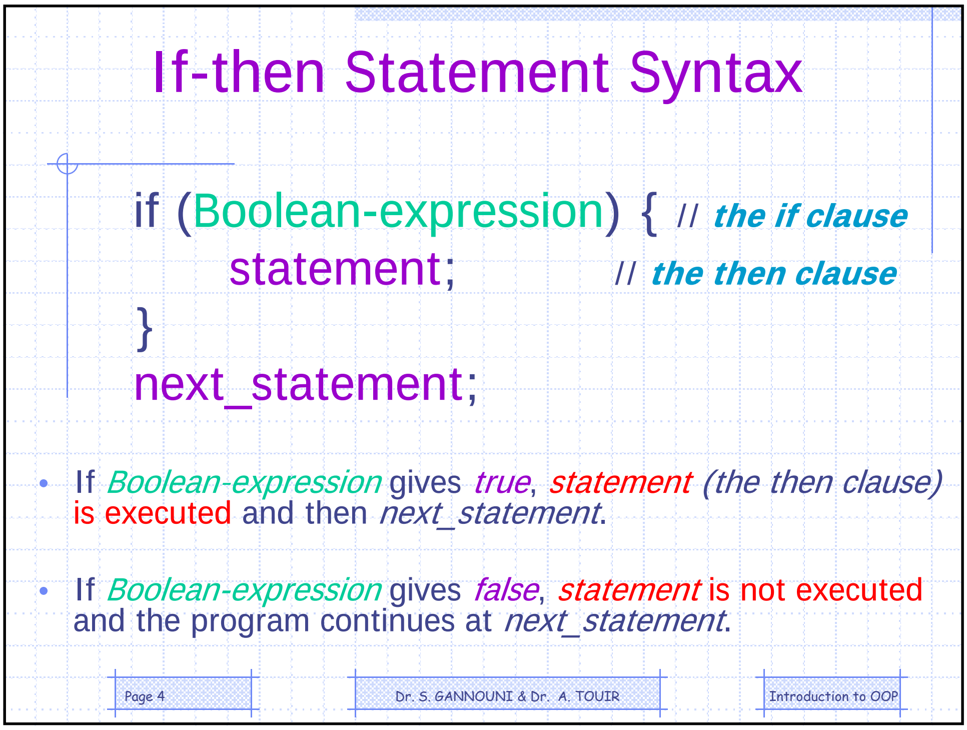
Decision Statements

- The decision (or selection) control structure is implemented in Java using the
  - ***If-then*** statement,
  - ***If-else*** statement,
  - And the ***switch***.



If-then Statement

- The if-then statement is the most basic of all the control structures.
- The if-then statement causes a program to execute statements conditionally.
- It tells the program to execute a certain section of code *only if* a particular test evaluates to true.



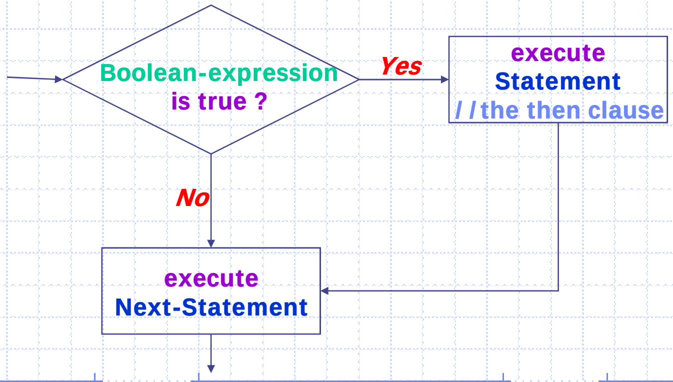
If-then Statement Syntax

```
if (Boolean-expression) { // the if clause
        statement;          // the then clause
}
next_statement;
```

- If *Boolean-expression* gives *true*, *statement* *(the then clause)* is executed and then *next_statement*.
- If *Boolean-expression* gives *false*, *statement* is not executed and the program continues at *next_statement*.

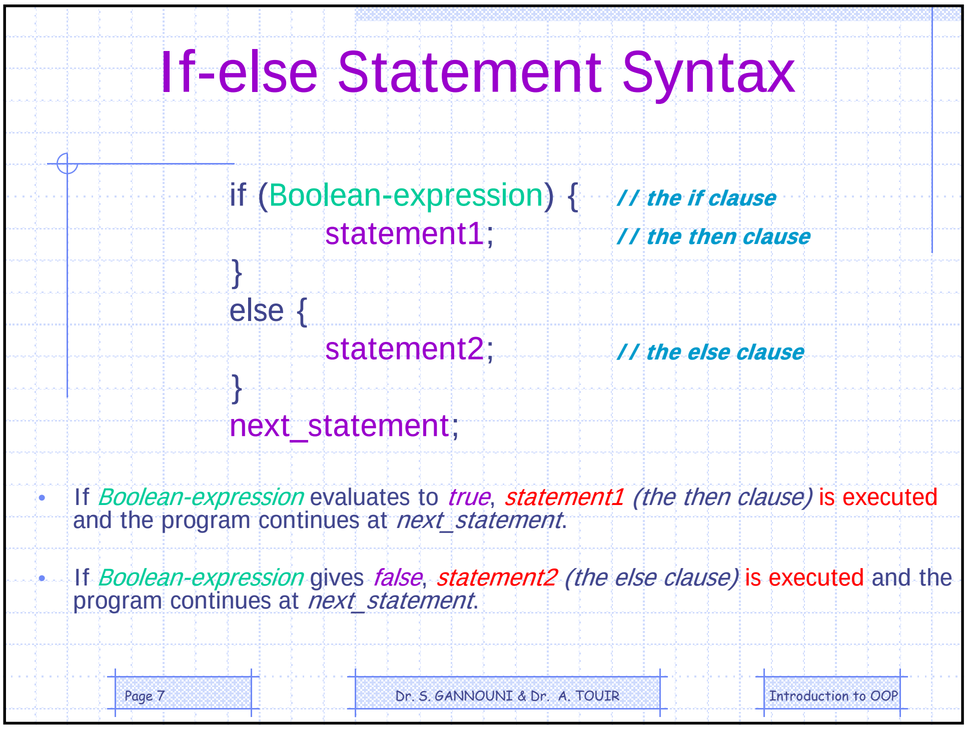
If-then Statement Flow Chart

```
if (Boolean-expression) {       // the if clause
          Statement;            // the then clause
}
Next_Statement;
```



If-else statement

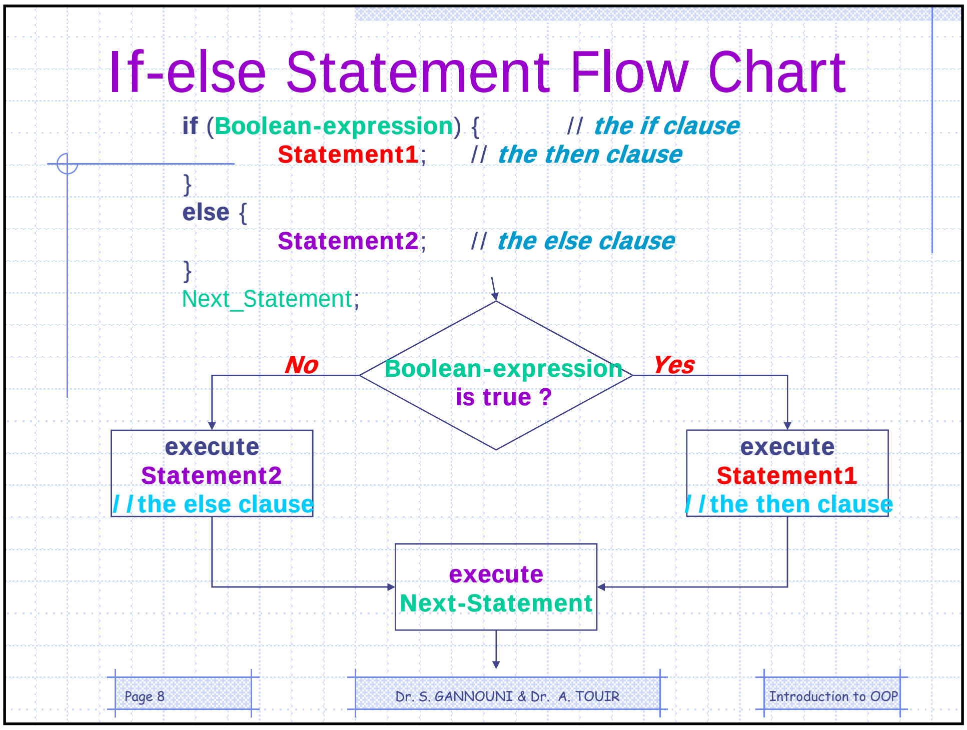
- The **if-else** statement provides a secondary path of execution when the *if clause* evaluates to false .
- It extends the basic if-then statement by adding the *else clause* in order to do something when the if clause is false



If-else Statement Syntax

```
if (Boolean-expression) {     // the if clause
        statement1;           // the then clause
}
else {
        statement2;           // the else clause
}
next_statement;
```

- If *Boolean-expression* evaluates to *true*, *statement1 (the then clause)* is executed and the program continues at *next_statement*.
- If *Boolean-expression* gives *false*, *statement2 (the else clause)* is executed and the program continues at *next_statement*.



If-else Statement Flow Chart

```
if (Boolean-expression) {        // the if clause
        Statement1;     // the then clause
}
else {
        Statement2;     // the else clause
}
Next_Statement;
```

Boolean-expression is true ?

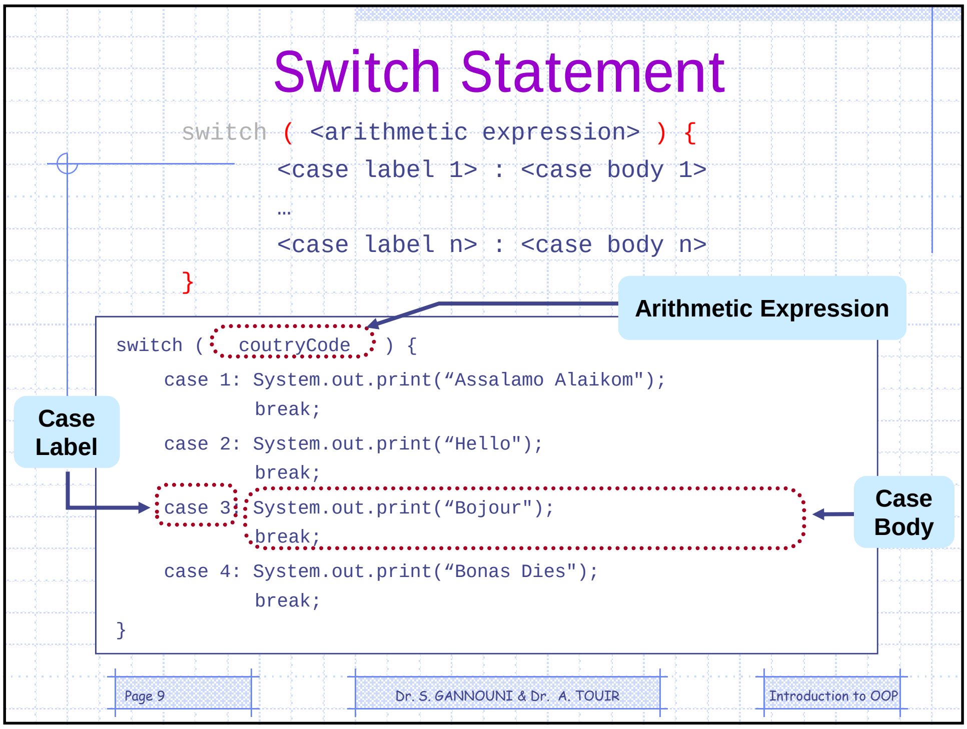
No

Yes

execute Statement2 / /the else clause

execute Statement1 / /the then clause

execute Next-Statement



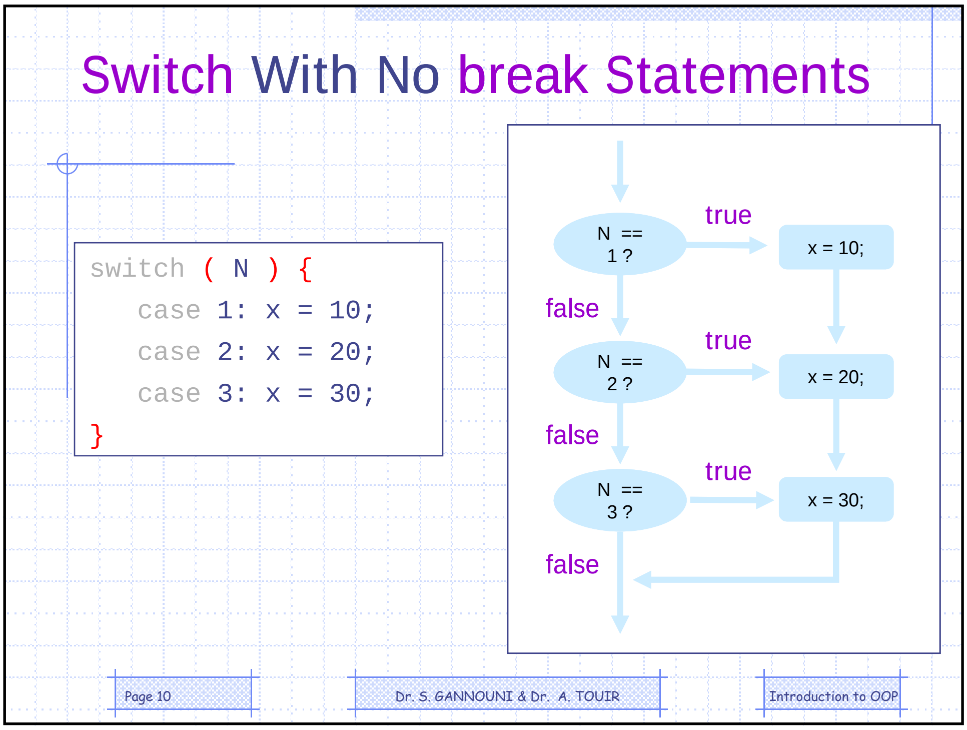
Switch Statement

```
switch ( <arithmetic expression> ) {
        <case label 1> : <case body 1>
        …
        <case label n> : <case body n>
}
```

```
switch ( coutryCode ) {
      case 1: System.out.print(“Assalamo Alaikom");
              break;
      case 2: System.out.print(“Hello");
              break;
      case 3: System.out.print(“Bojour");
              break;
      case 4: System.out.print(“Bonas Dies");
              break;
}
```

Arithmetic Expression
Case Label
Case Body

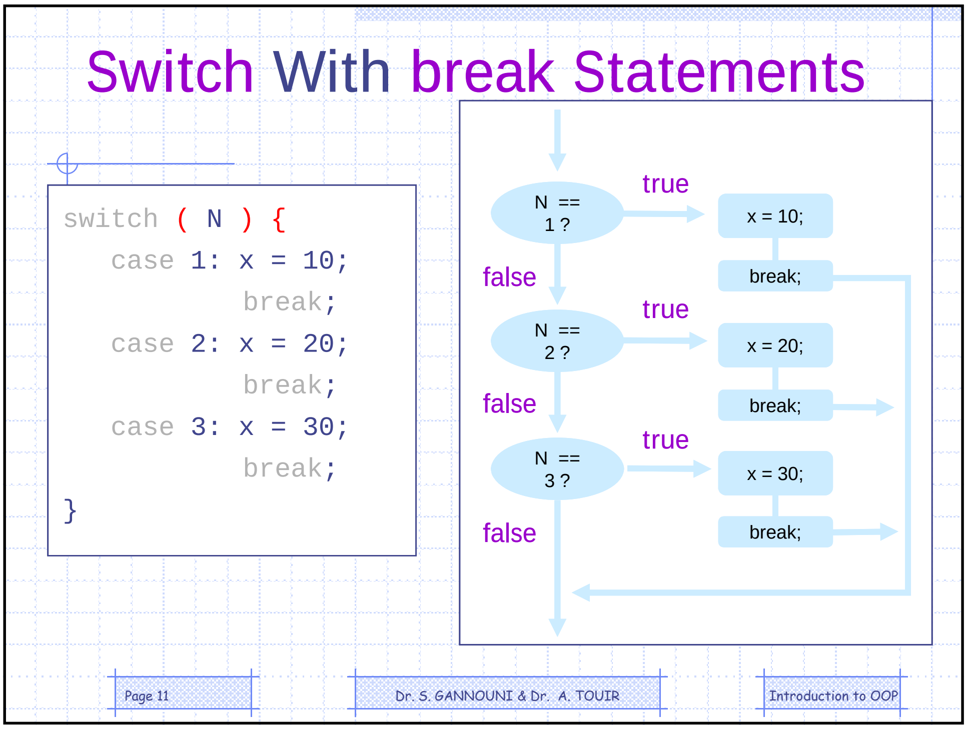
Page 9 | Dr. S. GANNOUNI & Dr. A. TOUIR | Introduction to OOP



Switch With No break Statements

```
switch ( N ) {
    case 1: x = 10;
    case 2: x = 20;
    case 3: x = 30;
}
```

Page 10 | Dr. S. GANNOUNI & Dr. A. TOUIR | Introduction to OOP



Switch With break Statements

```
switch ( N ) {
    case 1: x = 10;
            break;
    case 2: x = 20;
            break;
    case 3: x = 30;
            break;
}
```

N == 1 ?

true

x = 10;

break;

false

N == 2 ?

true

x = 20;

break;

false

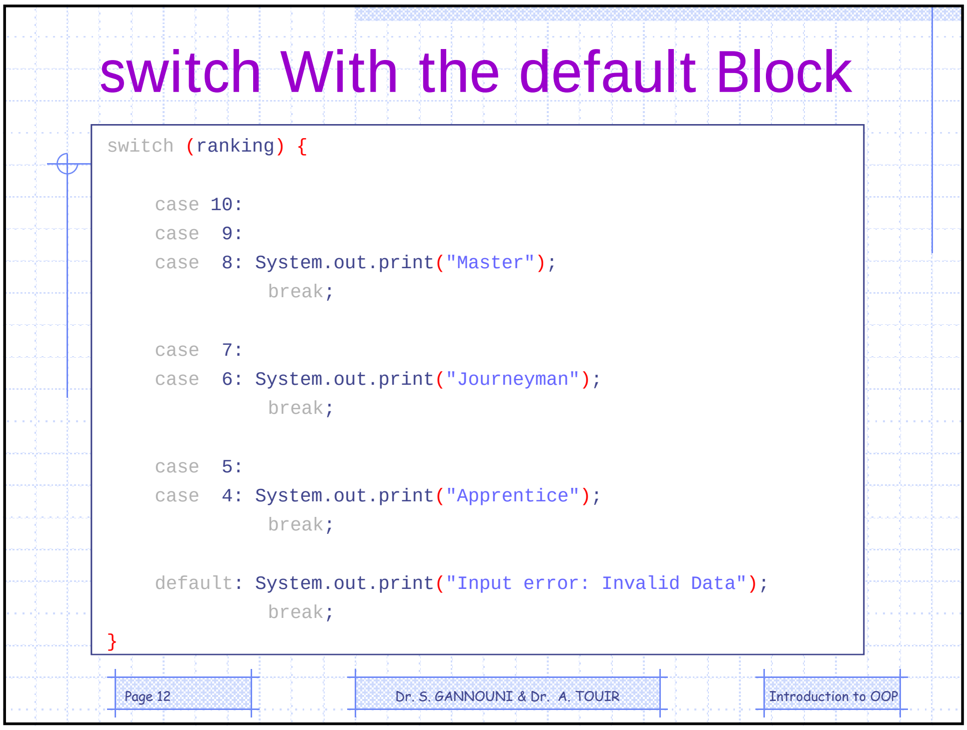
N == 3 ?

true

x = 30;

break;

false



switch With the default Block

```
switch (ranking) {

    case 10:
    case  9:
    case  8: System.out.print("Master");
             break;

    case  7:
    case  6: System.out.print("Journeyman");
             break;

    case  5:
    case  4: System.out.print("Apprentice");
             break;

    default: System.out.print("Input error: Invalid Data");
             break;
}
```