

CSN-103: Fundamentals of Object Oriented Programming



Control Statements

- Alter the *linear* flow of execution of a program
 - Branch
 - Advance
- Three categories
 - **Selection:** Allow your program to choose different paths of execution
 - **Iteration:** Enable program execution to repeat one or more statements
 - **Jump:** Allow your program to execute in a nonlinear fashion

Java's Selection Statements

- Java supports two selection statements: ***if*** and ***switch***
- **If** : Conditional branch statement

– General form

```
if (condition)
    statement1;
else
    statement2;
```

else block is Optional

```
if (condition)
{
    statement1;
    statement2;
}
else
{
    statement3;
    statement4;
}
```

Nested ifs

- A nested **if** is an **if** statement that is the target of another **if** or **else**


```
if(i < 100)
{
    if(j < 75)
        //Do something;
    if(k > 100)
        //Do something;
    else
        //Do something;
}
else
    //Do something;
```

Use { } to avoid confusion

if-else-if Ladder

- General Form

```
if(condition)  
    statement;  
else if(condition)  
    statement;  
else if(condition)  
    statement;  
.  
.  
else  
    statement;
```

A faint, stylized illustration of a lion lying down, facing left, positioned behind the code text.

- Last else block is optional

@SlackOverflow

A woman sends her programmer husband to the supermarket.

"Bring a carton of milk, and if they have eggs, bring 12".

He returned with 12 cartons of milk.



switch

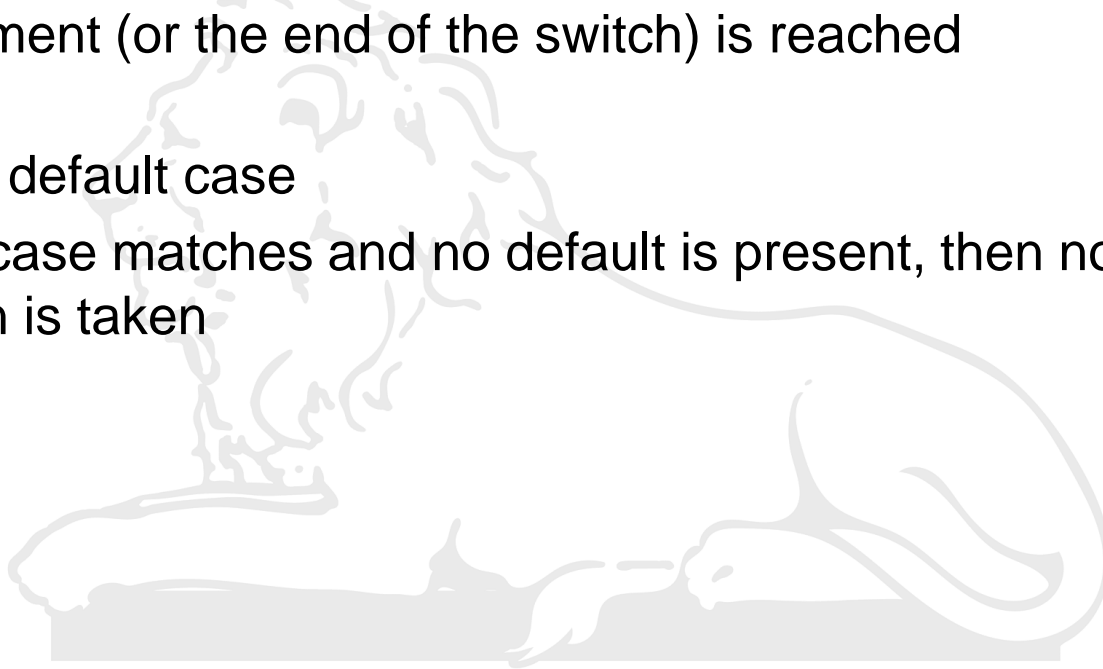
- **switch:** Multiway branch statement
- General form

switch (*expression*)

expression must be of type
byte, short, int, char or String

```
{  
    case value1:  
        // statement sequence  
        break;  
    case value2:  
        // statement sequence  
        break;  
    ...  
    case valueN :  
        // statement sequence  
        break;  
    default:  
        // default statement sequence  
}  
}
```

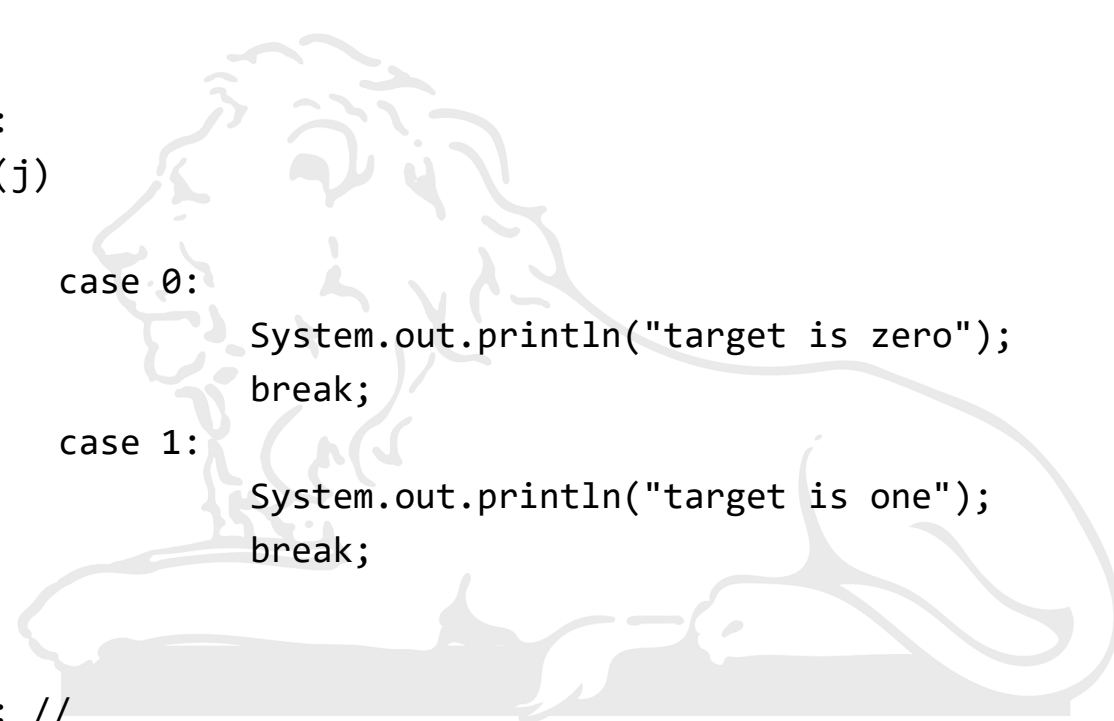
- Optional
 - Adding break statement after each case
 - Execution will continue on into the next **case** until a break statement (or the end of the switch) is reached
 - Adding a default case
 - If no case matches and no default is present, then no further action is taken



Nested switch Statements

- You can use a switch as part of the statement sequence of an outer switch

```
int i,j;
switch(i)
{
    case 1:
        switch(j)
        {
            case 0:
                System.out.println("target is zero");
                break;
            case 1:
                System.out.println("target is one");
                break;
        }
        break;
    case 2: //
}
}
```

A large, faint watermark of a sphinx is visible in the background of the slide, behind the code.