CITY
UNIVERSITY OF LONDON
EST 1894

IN1007 Programing in Java

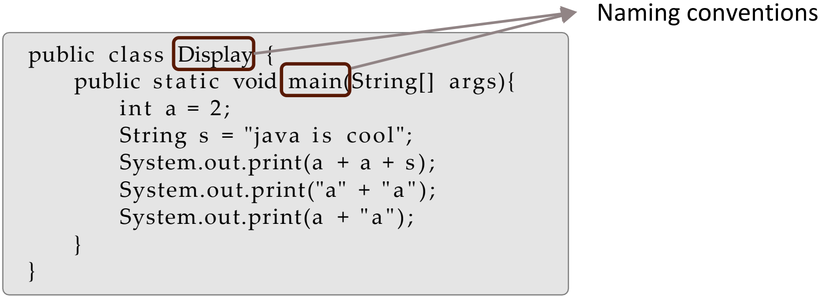
Lecture 2: Control Flow in Java

Announcements

Solutions for last week’s exercises

Video explaining arrays

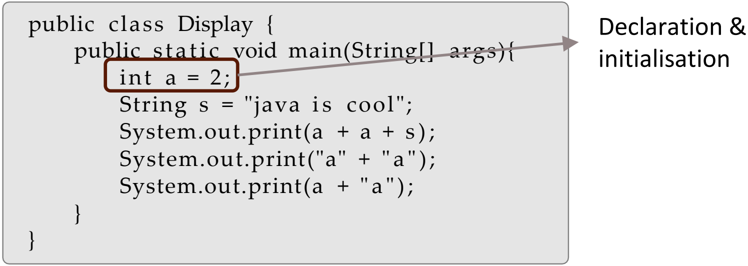
Concepts from Last Week



```
public class Display {
    public static void main(String[] args){
        int a = 2;
        String s = "java is cool";
        System.out.print(a + a + s);
        System.out.print("a" + "a");
        System.out.print(a + "a");
    }
}
```

Naming conventions

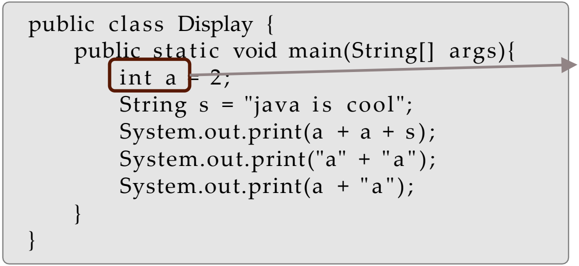
Concepts from Last Week



```
public class Display {
    public static void main(String[] args){
        int a = 2;
        String s = "java is cool";
        System.out.print(a + a + s);
        System.out.print("a" + "a");
        System.out.print(a + "a");
    }
}
```

Declaration & initialisation

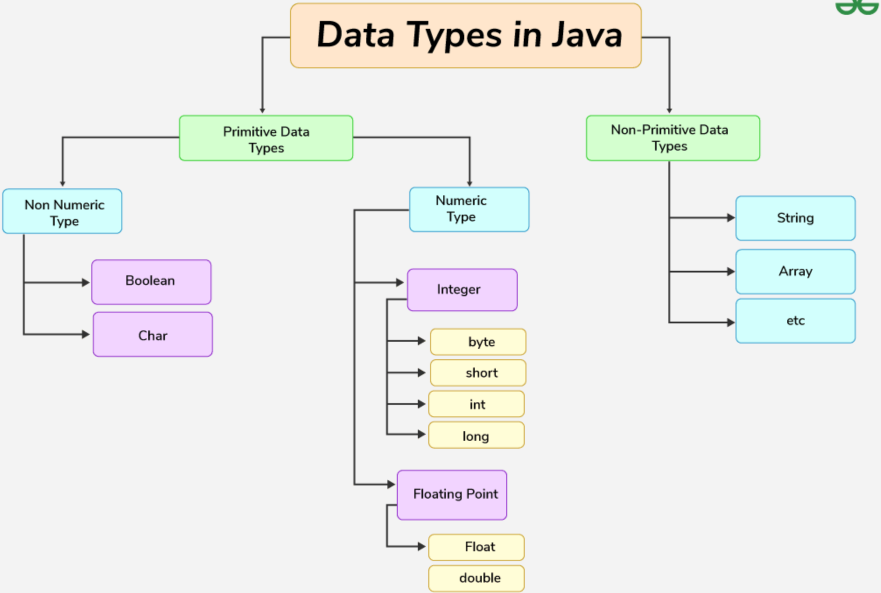
Concepts from Last Week



```
public class Display {
    public static void main(String[] args){
        int a = 2;
        String s = "java is cool";
        System.out.print(a + a + s);
        System.out.print("a" + "a");
        System.out.print(a + "a");
    }
}
```

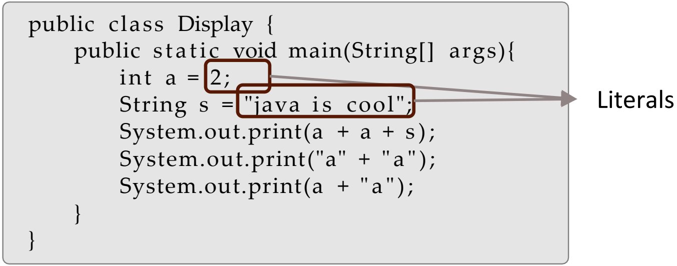
Primitive types

Data Types in Java



- Primitive Data Types
  - Non Numeric Type
    - Boolean
    - Char
  - Numeric Type
    - Integer
      - byte
      - short
      - int
      - long
    - Floating Point
      - Float
      - double
- Non-Primitive Data Types
  - String
  - Array
  - etc

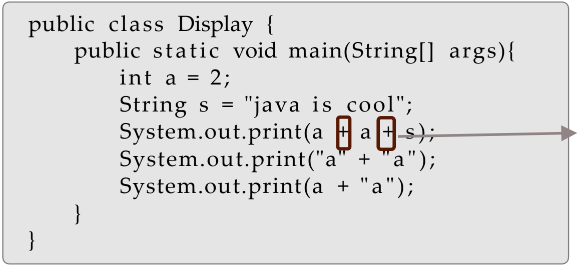
Concepts from Last Week



```
public class Display {
    public static void main(String[] args){
        int a = 2;
        String s = "java is cool";
        System.out.print(a + a + s);
        System.out.print("a" + "a");
        System.out.print(a + "a");
    }
}
```

Literals

Concepts from Last Week



```
public class Display {
    public static void main(String[] args){
        int a = 2;
        String s = "java is cool";
        System.out.print(a + a + s);
        System.out.print("a" + "a");
        System.out.print(a + "a");
    }
}
```

Operator overloading

Today's Lecture

- Control Flow:
  - **If** statement
  - **Switch** statement
  - Loops:
    - **For** loops
    - **While** loops
- **Branching** statements

The if and if-else statements

- if statement -

```
if (Boolean)
   instruction;
```

```
if (Boolean){
   instruction1;
   instruction2;
}
```

- if-else statement -

```
if (Boolean){
   instruction1;
   instruction2;
} else {
   instruction3;
   instruction4;
}
```

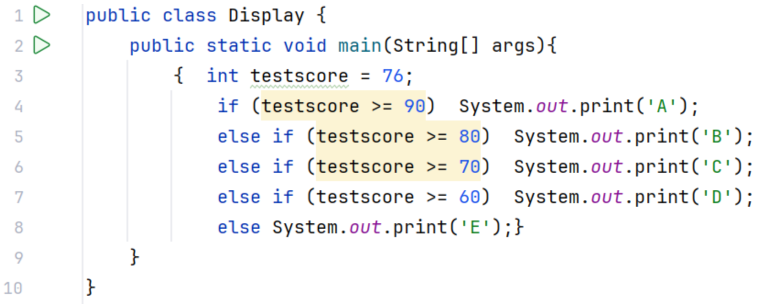
What does the following program display?

```
public class Demo {
    public static void main(String[] args) {
        int testscore = 76;
        if (testscore >= 50)
            System.out.print("Pass ");
            System.out.print("Congratulations!");
        if (testscore < 50)
            System.out.print("Fail ");
            System.out.print("Sorry!");
    }
}
```

What does the following program display?

```
public class IfElseDemo {
    public static void main(String[] args)
     {  int testscore = 76;
       if (testscore >= 90)
           System.out.print(’A’);
       if (testscore >= 80)
           System.out.print(’B’);
       if (testscore >= 70)
           System.out.print(’C’);
       if (testscore >= 60)
           System.out.print(’D’);
       else System.out.print(’E’);}}
```

Correct version



```
public class Display {
    public static void main(String[] args){
        {  int testscore = 76;
            if (testscore >= 90)  System.out.print('A');
            else if (testscore >= 80)  System.out.print('B');
            else if (testscore >= 70)  System.out.print('C');
            else if (testscore >= 60)  System.out.print('D');
            else System.out.print('E');}
    }
}
```

Example: Logic about age

Ask the user to give their age.

- If their age is above 18, you want to display “Adult”.
- Otherwise, you want to display “Underage”.

Additional Example: Logic about Months

Ask the user to give a number between 1 and 12.

Display the corresponding month:

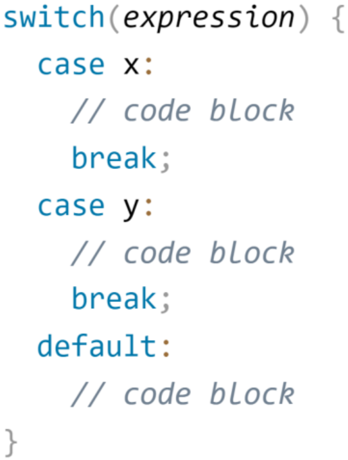
1 = January

2 = February...

Today's Lecture

- Arrays & Java naming conventions (Lecture 1)
- **Control Flow:**
  - **If** statement
  - **Switch** statement
  - Loops:
    - **For** loops
    - While loops
- **Branching** statement

Switch and break



```
switch(expression) {
  case x:
    // code block
    break;
  case y:
    // code block
    break;
  default:
    // code block
}
```

→ works with byte, short, char, int, String and a few others...

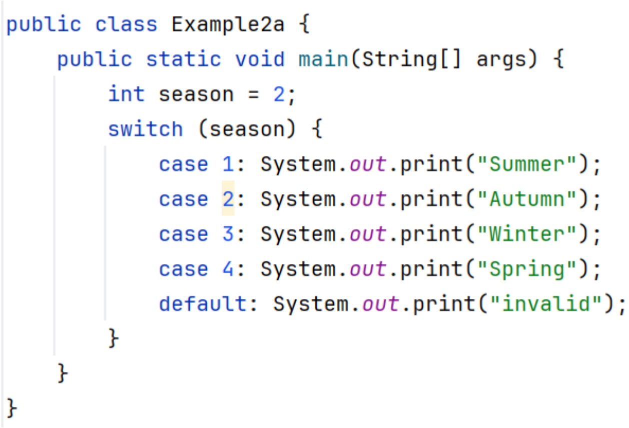
Switch Example:

Try the demo at: W3Schools Tryit Editor

Or use the URL: https://tinyurl.com/2ez9yme6

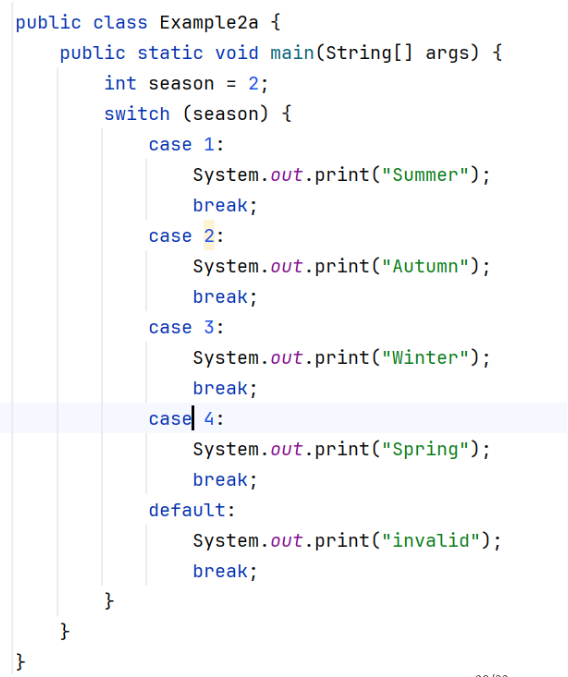
Change the value of the variable called ‘day’ and run to see how the output changes.

What does this program display?

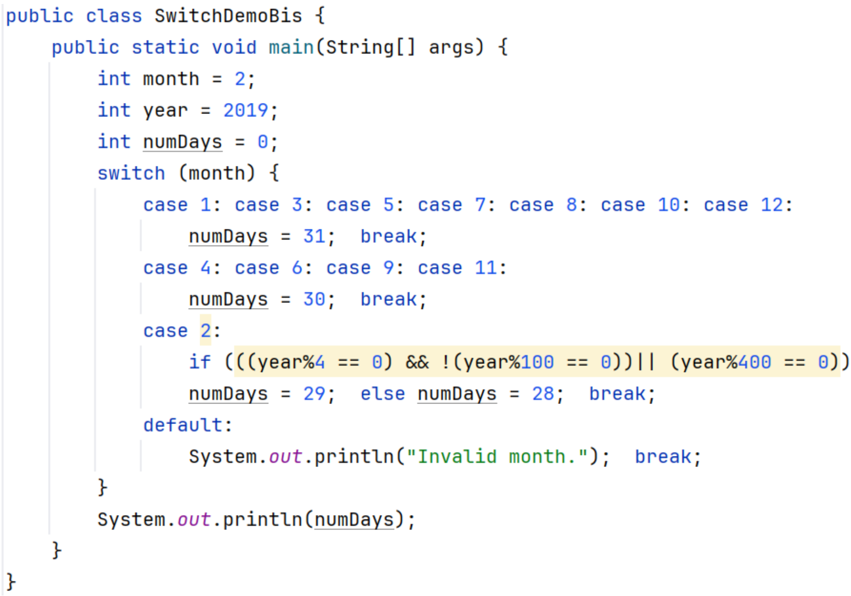


```java
public class Example2a {
    public static void main(String[] args) {
        int season = 2;
        switch (season) {
            case 1: System.out.print("Summer");
            case 2: System.out.print("Autumn");
            case 3: System.out.print("Winter");
            case 4: System.out.print("Spring");
            default: System.out.print("invalid");
        }
    }
}
```

Improved version



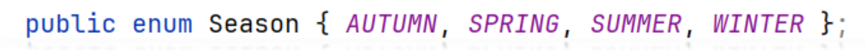
```java
public class Example2a {
    public static void main(String[] args) {
        int season = 2;
        switch (season) {
            case 1:
                System.out.print("Summer");
                break;
            case 2:
                System.out.print("Autumn");
                break;
            case 3:
                System.out.print("Winter");
                break;
            case 4:
                System.out.print("Spring");
                break;
            default:
                System.out.print("invalid");
                break;
        }
    }
}
```



```java
public class SwitchDemoBis {
    public static void main(String[] args) {
        int month = 2;
        int year = 2019;
        int numDays = 0;
        switch (month) {
            case 1: case 3: case 5: case 7: case 8: case 10: case 12:
                numDays = 31;  break;
            case 4: case 6: case 9: case 11:
                numDays = 30;  break;
            case 2:
                if (((year%4 == 0) && !(year%100 == 0))|| (year%400 == 0))
                numDays = 29;  else numDays = 28;  break;
            default:
                System.out.println("Invalid month.");  break;
        }
        System.out.println(numDays);
    }
}
```

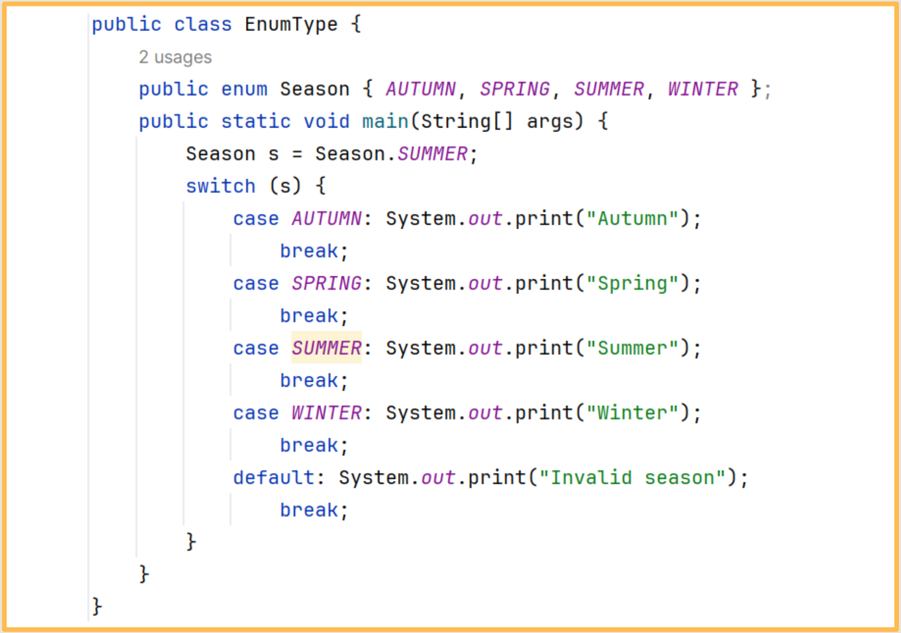
Grouping options together: The **enum** datatype

Enables a variable to be set to a number of predefined constants



```
public enum Season { AUTUMN, SPRING, SUMMER, WINTER };
```

We use capital letters to define constants.



```java
public class EnumType {
    2 usages
    public enum Season { AUTUMN, SPRING, SUMMER, WINTER };
    public static void main(String[] args) {
        Season s = Season.SUMMER;
        switch (s) {
            case AUTUMN: System.out.print("Autumn");
                break;
            case SPRING: System.out.print("Spring");
                break;
            case SUMMER: System.out.print("Summer");
                break;
            case WINTER: System.out.print("Winter");
                break;
            default: System.out.print("Invalid season");
                break;
        }
    }
}
```

Today’s Lecture

- **Control Flow:**
  - **If** statement
  - **Switch** statement
  - **Loops:**
    - **For** loops
    - **While** loops
- **Branching** statement

For loops

How can we display the text “Hello” ten times?

https://cscircles.cemc.uwaterloo.ca/java_visualize/

The structure of a **for** statement

```
for( initialisation; termination; increment)
{
  instruction1;
  instruction2;
}
```

Repeat something a fixed number of times

Example: Display numbers from 1 to 10

```
public class ForDemo {
    public static void main(String[] args){
        for(int i=1; i<=10; i++){
            System.out.println(i);
        }
    }
}
```

Java Visualizer

```
public class ForDemo {
    public static void main(String[] args){
        for(int i=1; i<=10; i++){
            System.out.println(i);
        }
    }
}
```

Example: Display numbers from 1 to 10

```
public class ForDemoV2 {
    public static void main(String[] args){
        int[] array = {1,2,3,4,5,6,7,8,9,10};
        for(int item:array){
            System.out.println(item);
        }
    }}
```

Today’s Lecture

- **Control Flow:**
  - **If** statement
  - **Switch** statement
  - **Loops:**
    - **For** loops
    - **While** loops
- **Branching** statement

The while statement

- while statement -

```
while(condition){
   instruction;
   instruction;
}
```

→ Repeat something as long as some condition is true

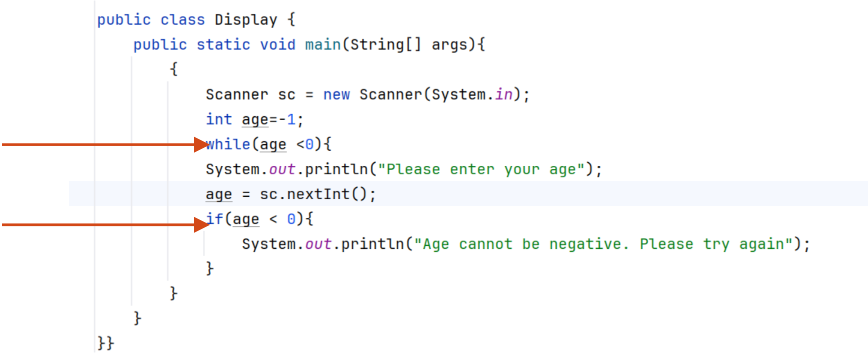
Why do we need another type of loop?

Sometimes the number of iterations is not known a priori.

You only know the condition to iterate

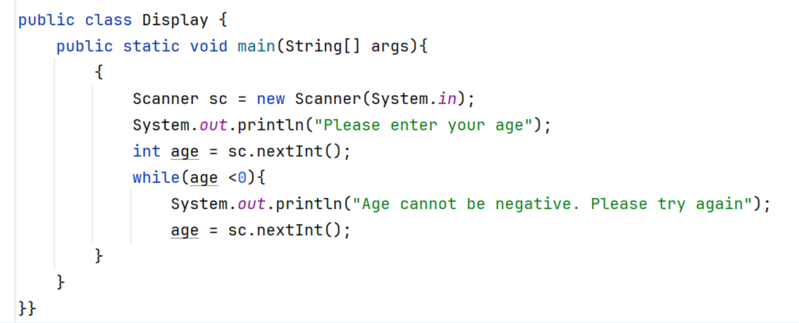
Example: Ask the user to enter a number for their age. As long as they enter a negative number, tell them this cannot be true and ask them again.

Not ideal – but works



```
public class Display {
    public static void main(String[] args){
        {
            Scanner sc = new Scanner(System.in);
            int age=-1;
            while(age <0){
            System.out.println("Please enter your age");
            age = sc.nextInt();
            if(age < 0){
                System.out.println("Age cannot be negative. Please try again");
            }
        }
    }
}}
```

Improved Version



```
public class Display {
    public static void main(String[] args){
        {
            Scanner sc = new Scanner(System.in);
            System.out.println("Please enter your age");
            int age = sc.nextInt();
            while(age <0){
                System.out.println("Age cannot be negative. Please try again");
                age = sc.nextInt();
        }
    }
}}
```

The while statement

```java
public class WhileDemo {
    public static void main(String[] args){
        int i = 1;
        while (i < 11) {
            System.out.println(i);
            i++;
        }
    }
}
```

The do-while statement

- do-while statement -

```
do {  instruction;
} while (condition);
```

→ Repeat something as long as some condition is true

```
public class DoWhileDemo {
    public static void main(String[] args){
        int i = 1;
        do {  System.out.println(i);
            i++;
        } while (i < 11);
    }
}
```

What does the following program display?

```
public class WhileDemoQuiz {
    public static void main(String[] args) {
       int i = 5;
       do {
          System.out.print(i);
          i++;
       } while (i < 5);
       while (i < 6) {
          System.out.print(i);
          i++;
       }
    }}
```

Today’s Lecture

- **Control Flow:**
  - **If** statement
  - **Switch** statement
  - **Loops:**
    - **For** loops
    - **While** loops
- **Branching** statements

Branching statements

→ *Stop the current flow*

Branching statements

→ *Stop the current flow*

- break -

→ terminates a for, while, or do-while loop

Branching statements

→ *Stop the current flow*

- **break -**

→ terminates a for, while, or do-while loop

- **continue -**

→ directly go at the next iteration of a for, while, or do-while loop

Branching statements

→ *Stop the current flow*

- **break -**

→ terminates a for, while, or do-while loop

- **continue -**

→ directly go at the next iteration of a for, while, or do-while loop

- **return -**

→ see Lecture 3 – Week 5

Example:

Check whether a given integer appears in an array of integers.
If this is the case, display the first index where it appears.

Example

Count the number of times a given integer appears in an array of integers.

Today’s Lecture

- **Control Flow:**
  - **If** statement
  - **Switch** statement
  - **Loops:**
    - **For** loops
    - **While** loops
- **Branching** statements

Example

Count the number of times a given integer appears in an array of integers.