UNTAR
Universitas Tarumanagara

Terakreditasi BAN PT A (Unggul) | QS STARS RATING SYSTEM 2019 | JABEE | CPA AUSTRALIA | ICAEW CHARTERED ACCOUNTANTS

UNTAR untuk INDONESIA

Object-based Programming

Week 1 – Introduction to Java

1990

- James Gosling, Michael Sheridan, Patrick Naughton
- Oak => Green

1996

- Java 1.0
- Released by Sun Microsystems
- Write Once, Run Anywhere

1998-1999

- J2SE, J2ME, J2EE

2006-2007

- Java SE, Java ME, Java EE
- JVM released under FOSS
- OpenJDK

2010

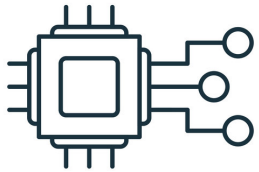
- Acquired by Oracle

2014

- JDK 8

2021

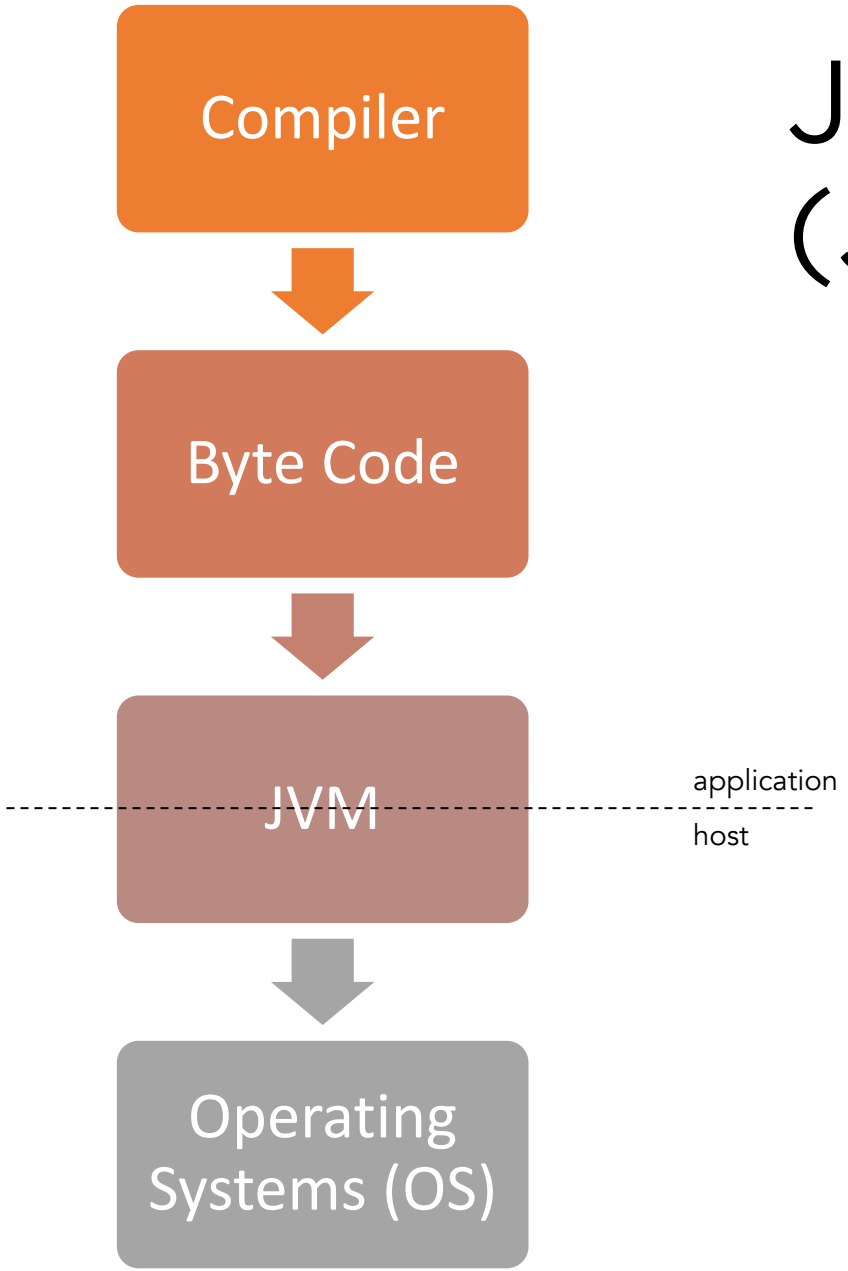
- JDK 16



EMBEDDED SYSTEM

and many more...

Java Virtual Machine (JVM)



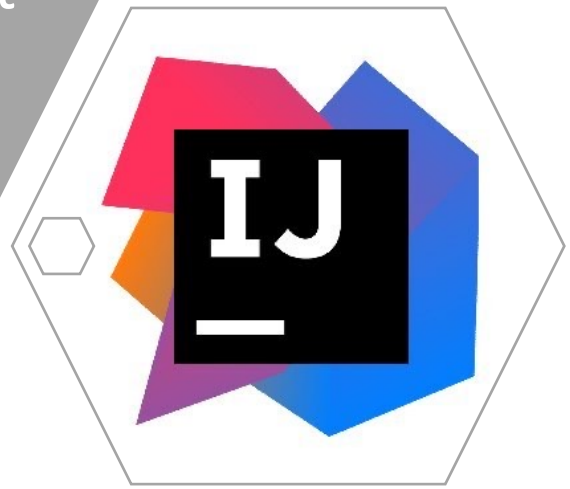
Write Once, Run Anywhere

https://www.jetbrains.com/community/education/#students

Apply using @stu.untar.ac.id email address

Java Development Kit (JDK)

Integrated Development Editor (IDE)



Eclipse, Netbeans, VS Code, ...

https://www.oracle.com/java/technologies/javase-downloads.html

https://jdk.java.net/16/

Question #1

https://bit.ly/3B2zJrL

Java tidak mendukung pembuatan aplikasi di *platform* ...

A. Mobile

B. Embedded system

C. Desktop

D. Web

E. Tidak ada jawaban yang benar

Procedural Programming Basics in Java

Sage, Kingsley. (2019). *Concise Guide to Object-Oriented Programming*. Switzerland: Springer Nature.

HelloWorld.java

```
public class HelloWorld {



}
```

HelloWorld.java

```
public class HelloWorld {
    public static void main(String[] args) {

    }
}
```

HelloWorld.java

```
public class HelloWorld {
    public static void main(String[] args) {
        System.out.println(“Hello World!”);
    }
}
```

HelloWorld.java

```java
/**
 * Write a description of class HelloWorld here
 *
 * @author Janson Hendryli
 * @version 1.0
 */

public class HelloWorld {
    public static void main(String[] args) {
        // This is a one-line comment
        System.out.println(“Hello World!”);
    }
}
```

Primitive Data Type

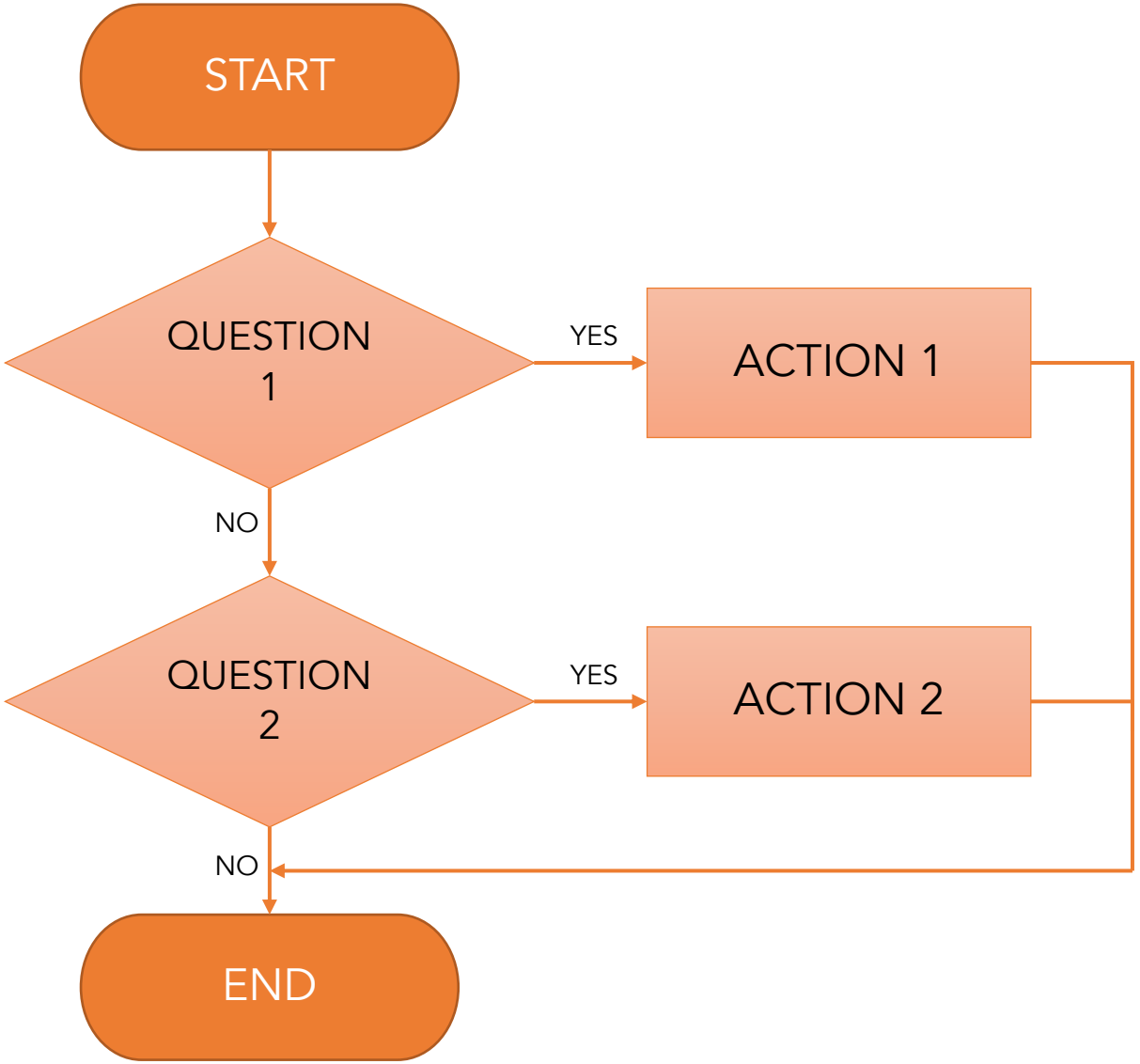
Type	Description	Size	Example Value
`boolean`	True or false	1 bit	`true, false`
`byte`	Integer	1 byte (8 bits)	`-3, -2, -5`
`char`	Unicode character	2 bytes	`'a', '\u0030'`
`short`	Integer	2 bytes	`-3, -2, -5`
`int`	Integer	4 bytes	`-3, -2, -5`
`long`	Integer	8 bytes	`-3L, 0L, 4L`
`float`	Floating point	4 bytes	`1.2f, -1.2e03f`
`double`	Floating point	8 bytes	`1.2, -1.2e03`

```
int x = 3;
long k = 4L;

boolean y = false;

double x1 = 3.14;

float x2 = 1.44f;

char c = ‘m’;
```

HelloWorld.java

```
public class HelloWorld {
    public static void main(String[] args) {
        int a;
        int b = 2;
        double c;
        double d = 4.5;
        double e;

        a = 2;
        a++;
        c = -4.5;
        e = c * d;
        System.out.println("a multiplied by b is " + (a * b));
        System.out.println("The value of e is " + e);
    }
}
```

Conditional Statement



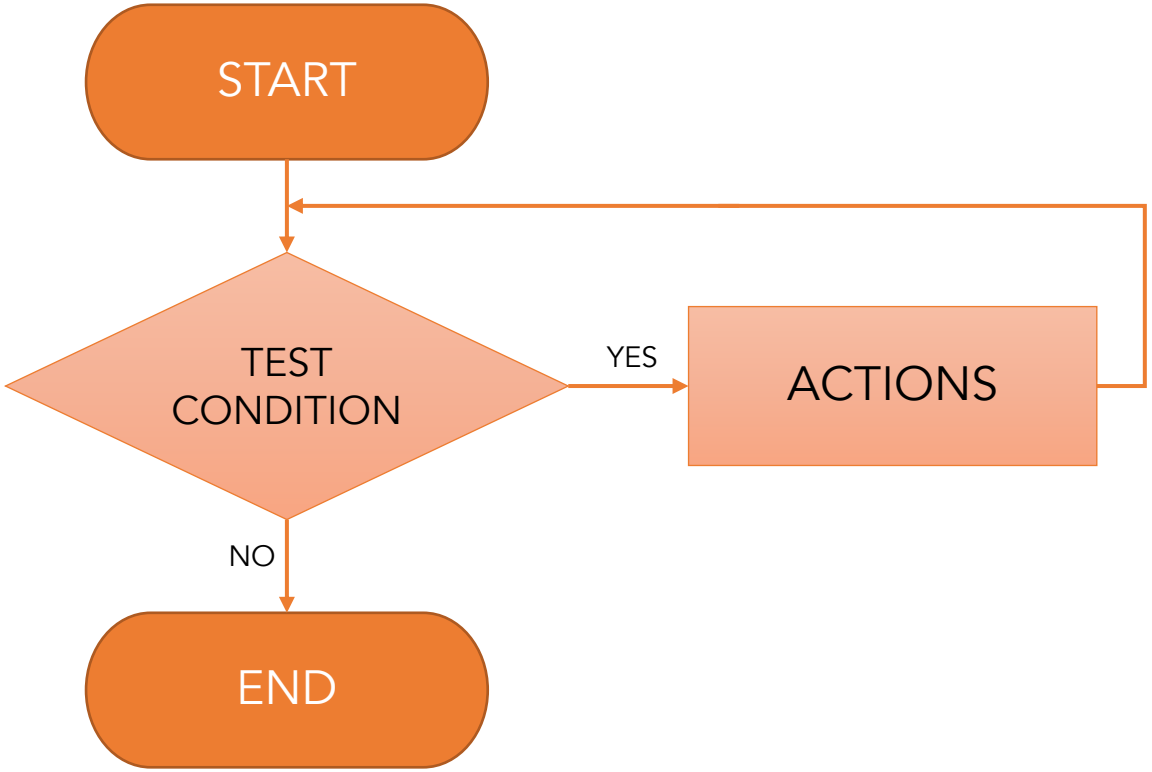
Relational & Logical Operators

Operator	Meaning
==	Is equal to
!=	Is not equal to
>	Is greater than
>=	Is greater than or equal to
<	Is less than
<=	Is less than or equal to
&&	Logical AND
||	Logical OR
!	Logical NOT

```
int a = 1, b = 2, c = 3;
boolean x = true;

if (a == 1) System.out.println(“a has the value of 1”);

if (a != 1)
   System.out.println(“a does not have the value of 1”);

if (a > 2) {
   System.out.print(“a has a value ”);
   System.out.println(“greater than 2”);
}

if ((a == 1) && x)
   System.out.println(”a is 1 AND x is true”);
else
   System.out.println(“a is not 1 AND x is not true”);
```

```
int x = 3;

switch (x) {
    case 1:
        System.out.println(“Option if x has the value 1”);
        break;
    case 2:
        System.out.println(“Option if x has the value 2”);
        break;
    default:
        System.out.println(“x has some other value”);
        break;
}
```

Iteration/ Looping



Anatomy of for loop

```
for (<initial-state>; <test-condition>; <action>) {
    // Body of loop
}
```

`<initial-state>`	initial value of some controlling variable that will be set one-time prior to the first evaluation of the test condition to some initial value
`<test-condition>`	a `boolean` test that will determine whether the loop continues to execute
`<action>`	a short section of code that is executed each time the body of the loop has finished executing

initial-state *test-condition* *action*

```
for (int x = 0; x < 10; x++) {
    System.out.println(x);
}
```

`<initial-state>`	initial value of some controlling variable that will be set one-time prior to the first evaluation of the test condition to some initial value
`<test-condition>`	a `boolean` test that will determine whether the loop continues to execute
`<action>`	a short section of code that is executed each time the body of the loop has finished executing

1

initial-state *test-condition* *action*

```
for (int x = 0; x < 10; x++) {
    System.out.println(x);
}
```

In-memory: x = 0 [1]

1 2

initial-state *test-condition* *action*

```
for (int x = 0; x < 10; x++) {
    System.out.println(x);
}
```

```
In-memory:    x = 0                    [1]
              condition = true         [2]
```

1 2

initial-state *test-condition* *action*

```
for (int x = 0; x < 10; x++) {
    System.out.println(x);
}
```

3

```
In-memory:    x = 0                    [1]
              condition = true         [2]
```

1 2 4

initial-state *test-condition* *action*

```
for (int x = 0; x < 10; x++) {
    System.out.println(x);
}
```

3

```
In-memory:    x = 0 + 1 = 1          [4]
              condition = true       [2]
```

5

4

initial-state *test-condition* *action*

```
for (int x = 0; x < 10; x++) {
    System.out.println(x);
}
```

```
In-memory:    x = 0 + 1 = 1          [4]
              condition = true       [5]
```

5

4

initial-state *test-condition* *action*

```
for (int x = 0; x < 10; x++) {
    System.out.println(x);
}
```

6

```
In-memory:    x = 0 + 1 = 1          [4]
              condition = true       [5]
```

A *initial-state* B *test-condition* D *action*

```
for (int x = 0; x < 10; x++) {
    System.out.println(x);
}
```

C

Sequence: A ⇨ B ⇨ C ⇨ D ⇨ B ⇨ C ⇨ D ⇨ and so on...

Question #2

https://bit.ly/3B2zJrL

A B C

```
for (int x = 0; x < 10; x += 2) {
    System.out.println(x);
}
System.out.println("Done.");
```

D

Perintah apa yang dijalankan sebelum perintah ini?

Question #3

https://bit.ly/3B2zJrL

Berapa nilai variabel x setelah looping selesai dijalankan?

A. 0

B. 9

C. 10

D. 11

E. Semua jawaban salah

```
for (int x = 0; x < 10; x++) {
    System.out.println(x);
}
```

Anatomy of `while` loop

```
while (<test-condition>) {
    // Body of loop
}
```

```
int i = 0;

while (i < 10) {
    System.out.println(i);

    i++;
}
```

```
int i = 0;

while (i < 10) {
    System.out.println(i);

    i++;
}
```

What happen if we omit this line?

Anatomy of do-while loop

```
do {
    // Body of loop
} while (<test-condition>);
```

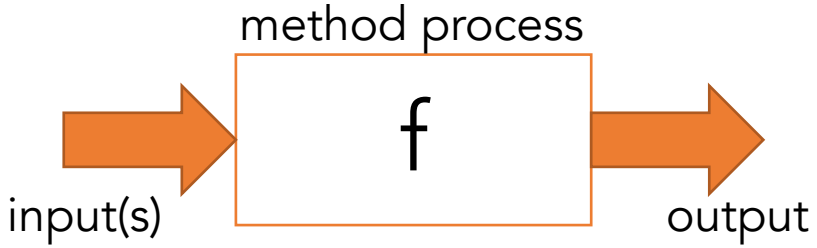
```
int i = 0;

do {
    System.out.println(i);

    i++;
} while (i < 10);
```

while vs do-while

```
int i = 0;

while (i < 10) {
    System.out.println(i);
    i++;
}
```

```
int i = 0;

do {
    System.out.println(i);
    i++;
} while (i < 10);
```

Methods



- A **method** is a function that belongs to a class.
- A method performs some useful behaviour made up using code.
- A method can have any number of inputs, and either one or zero output.
- A method with zero outputs is called a **void method**.

Methods

```
<access-modifier> <return-type> method-name(<format-parameters>) {
    // Body of method
    <return-statement-if-not-void>
}
```

Methods

```
public int calculateArea(int width, int length) {
    int totalArea = width * length;
    return totalArea;
}
```

Methods

```
public void printTotalArea(int width, int length) {
    int totalArea = width * length;
    System.out.println(“Total area is “ + totalArea);
}
```

Methods

```
private boolean isGreaterThan(int a, int b) {
    return (a > b);
}
```

```
import java.util.*;

public class Example {
   private static void printHelloWorld() {
      System.out.println(“Hello World”);
   }

   public static int addTwoNumbers(int a, int b) {
      return a + b;
   }

   public static void main(String[] args) {
      printHelloWorld();

      // read user’s input
      Scanner in = new Scanner(System.in);
      System.out.print(“Input an integer: “);
      int number = in.nextInt();

      System.out.println(number + “ + 10 = “ + addTwoNumbers(number, 10));
}
```

Naming Convention

- Class
  - Nouns
  - Mixed case with the first letter of each internal word capitalized
  - Simple and descriptive
  - Whole words; avoid acronyms and abbreviations (unless widely used, such as URL or HTML)

- Examples
  - `ImageSprite`
  - `HelloWorld`
  - `Raster`

Naming Convention

- Methods
  - Verbs
  - Camel-case
    - Mixed case with the first letter lowercase
    - First letter of each internal word capitalized

- Examples
  - `run()`
  - `runFaster()`
  - `doInBackground()`
  - `getLength()`

Naming Convention

- Variables
  - Camel-case
  - Should not start with underscore _ or dollar sign $ characters (though allowed)
  - One-character variable names should be avoided, except for temporary throwaway variables, e.g. `i`, `j`, `k` for integers

- Examples
  - `int width = 5;`
  - `boolean isAuthenticated = false;`
  - `double earningBeforeInterests = 1000000.0;`

Naming Convention

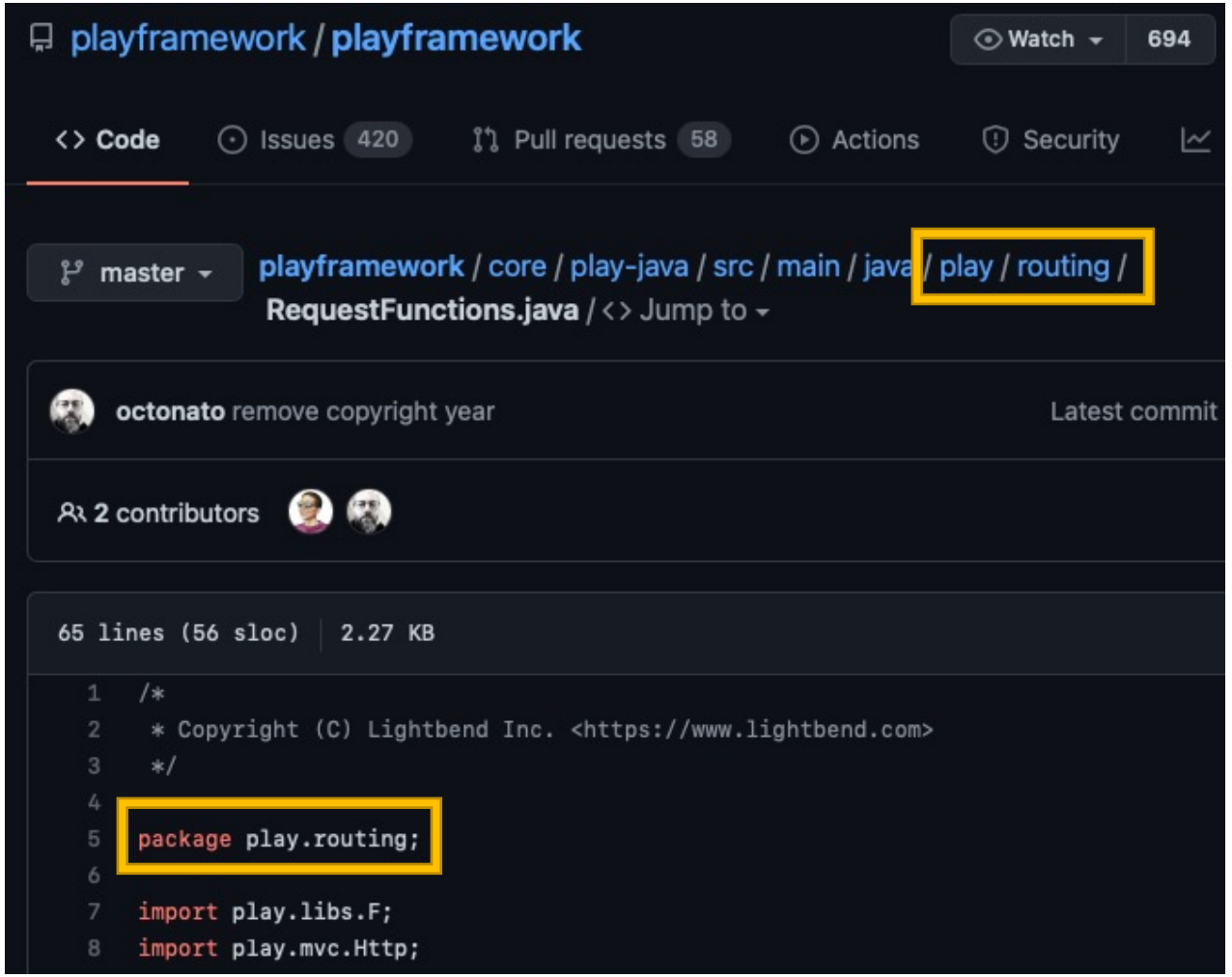
- Constants
  - All uppercase
  - Words separated by underscores _

- Examples
  - `static final double MIN_WIDTH = 5.0;`
  - `static final int CPU_PROCESS_ID = 1;`
  - `static final float PI = 3.14F;`

Naming Convention

- Packages
  - All lowercase ASCII letters
  - Commonly is top-level domain names

- Examples
  - com.sun.eng
  - id.ac.untar.fti



In another files, I can do `import play.routing.RequestFunctions`

Exercise – Guessing Game

- The program will generate a random number in the range 1-99
- The user will have up to 10 attempts to guess the number
- If the number guesses is too low, the program will display a “too low” message on console
- If the number guesses is too high, the program will display a “too high” message on console
- The game ends if the user guesses correctly, or has had 10 attempts

Answer!

```
import java.util.*;

public class GuessingGame {
    private static int answer;
    private static Scanner in;

    public static void main(String[] args) { ... }

    private static void startGame() { ... }
    private static int getGuessFromUser() { ... }
    private static void analyseGuess(int guess) { ... }
}
```

```
public static void main(String[] args) {
    in = new Scanner(System.in);
    startGame();
}

...

private static int getGuessFromUser() {
    System.out.print("Enter your guess: ");
    int number = in.nextInt();
    return number;
}
```

```
private static void analyseGuess(int guess) {
    if (guess > answer)
        System.out.println(“Too high”);
    else if (guess < answer)
        System.out.println(“Too low”);
    else
        System.out.println(“Well done!”);
}
```

```
private static void startGame() {
    // initialize the random answer
    Random rnd = new Random();
    answer = rnd.nextInt(99) + 1;

    int numGuesses = 1;
    int userGuess = 0;

    boolean isGameOver = false;

    // the main game loop
    while (!isGameOver) { ... }
}
```

```
while (!isGameOver) {
    userGuess = getGuessFromUser();
    analyseGuess(userGuess);

    numGuesses++;
    isGameOver = (answer == userGuess) || (numGuesses > 10);

    if (isGameOver && numGuesses > 10) {
        System.out.println("Out of guesses!");
        System.out.println("The answer is " + answer);
    }
}
```