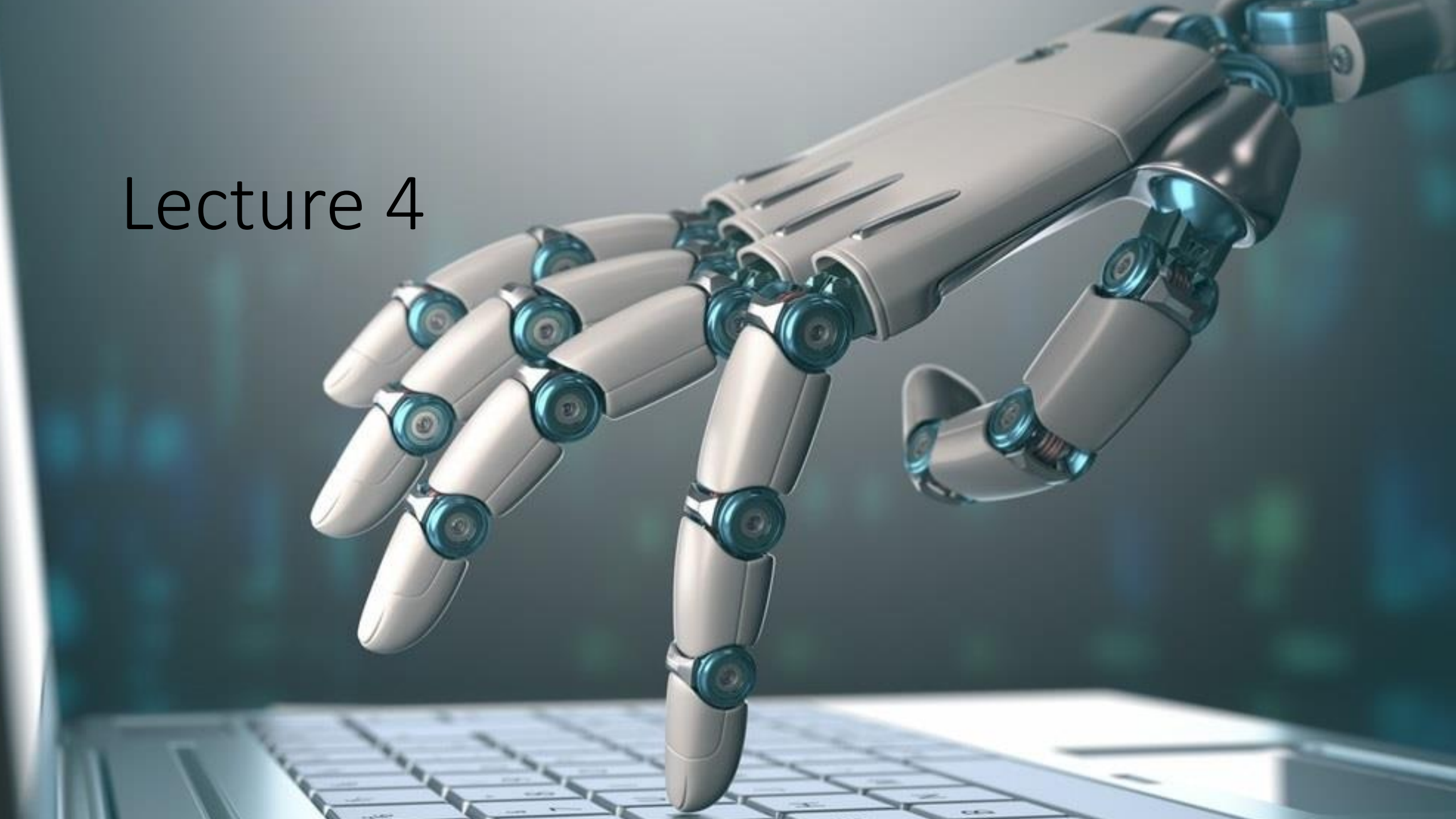
Lecture 4



Agenda

- **While loop**
- For loop
- Do While loop
- Arrays
- Practice

OVERVIEW

- There may be situation when you need to execute a block of code several number of times
- A loop statement allows us to execute a statement or group of statements multiple times
- Looping statements available:
  1. while
  2. for
  3. do...while

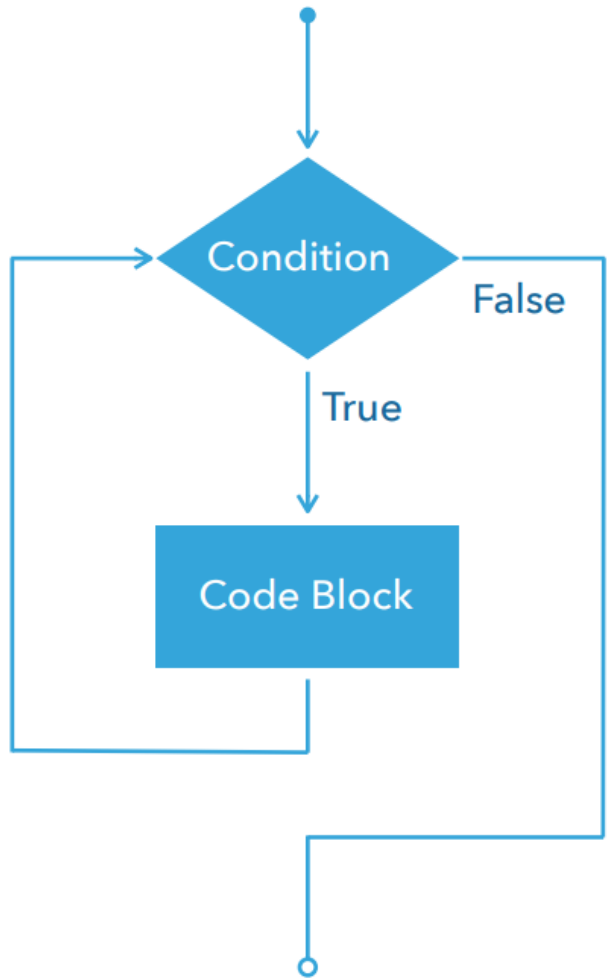
COMMON LOOPS STRUCTURE

- There is a control variable, called the loop counter
- Loop variable must be initialized
- The increment or decrement of the control variable, which is modified each time the iteration of the loop occurs
- The loop condition that determines if the looping should continue or the program should break from it

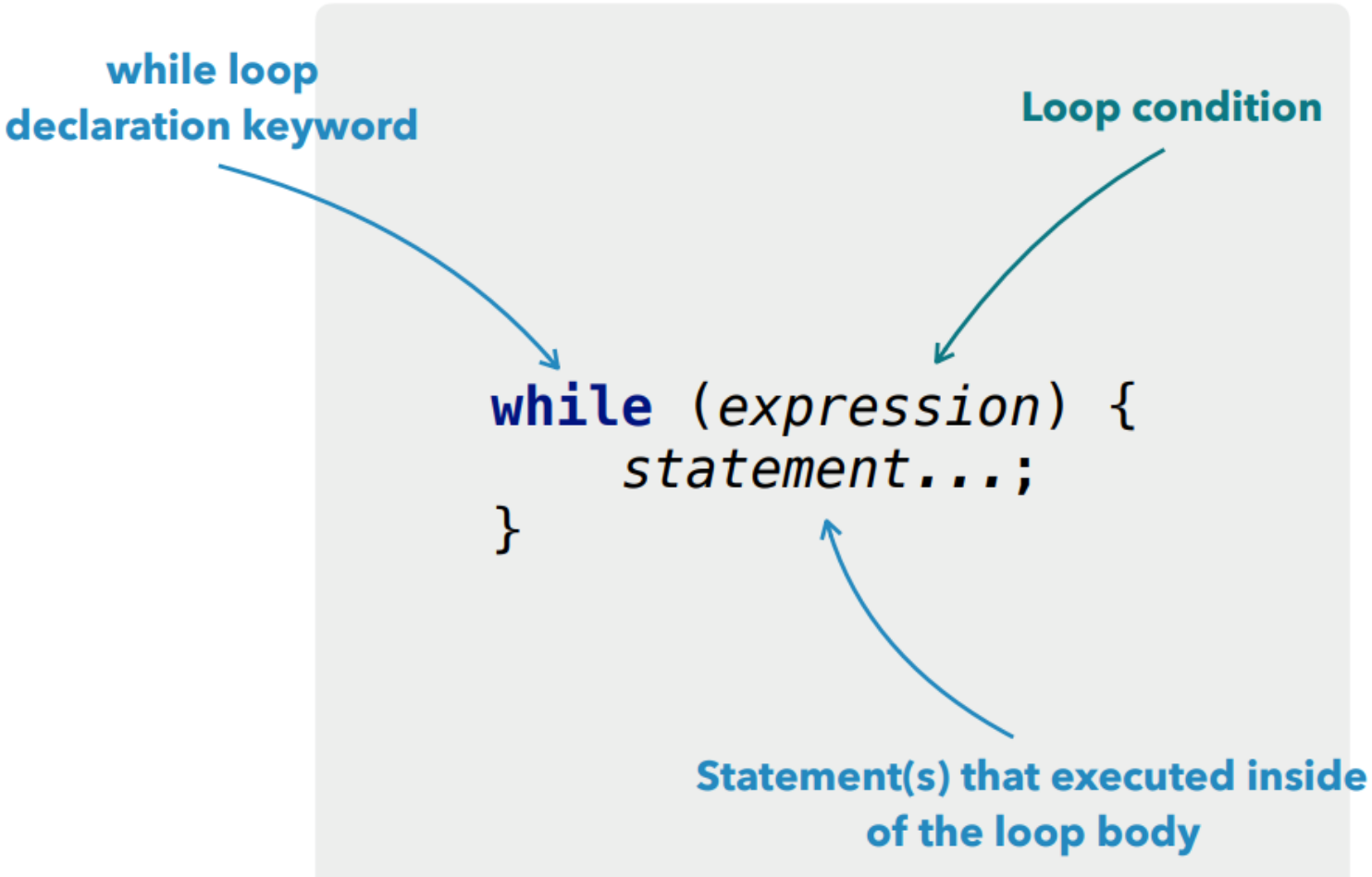
WHILE LOOP: SUMMARY

- Repeats a statement or block of statements while its controlling boolean expression is true
- Boolean expression is evaluated before the first iteration of the loop, hence executed zero or many times
- Usually used when number of iterations depends

WHILE LOOP: FLOWCHART



WHILE LOOP: SYNTAX



WHILE LOOP: CODE EXAMPLE

Code

```
int i = 0;
while (i < 5) {
    System.out.print("i = " + i + "; ");
    i++;
}
```

Console output

```
i = 0; i = 1; i = 2; i = 3; i = 4;

Process finished with exit code 0
```

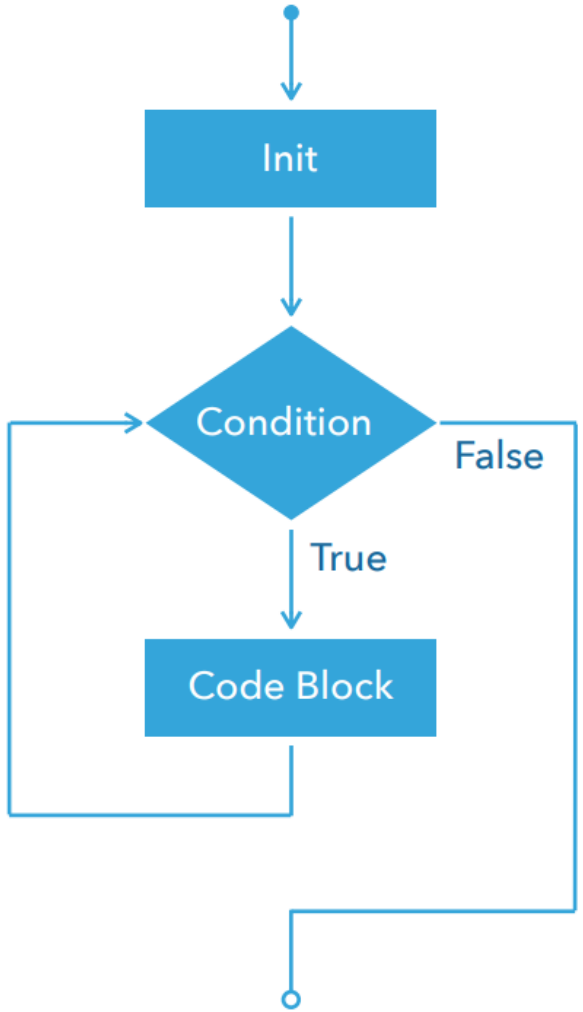
Agenda

- While loop
- **For loop**
- Do While loop
- Arrays
- Practice

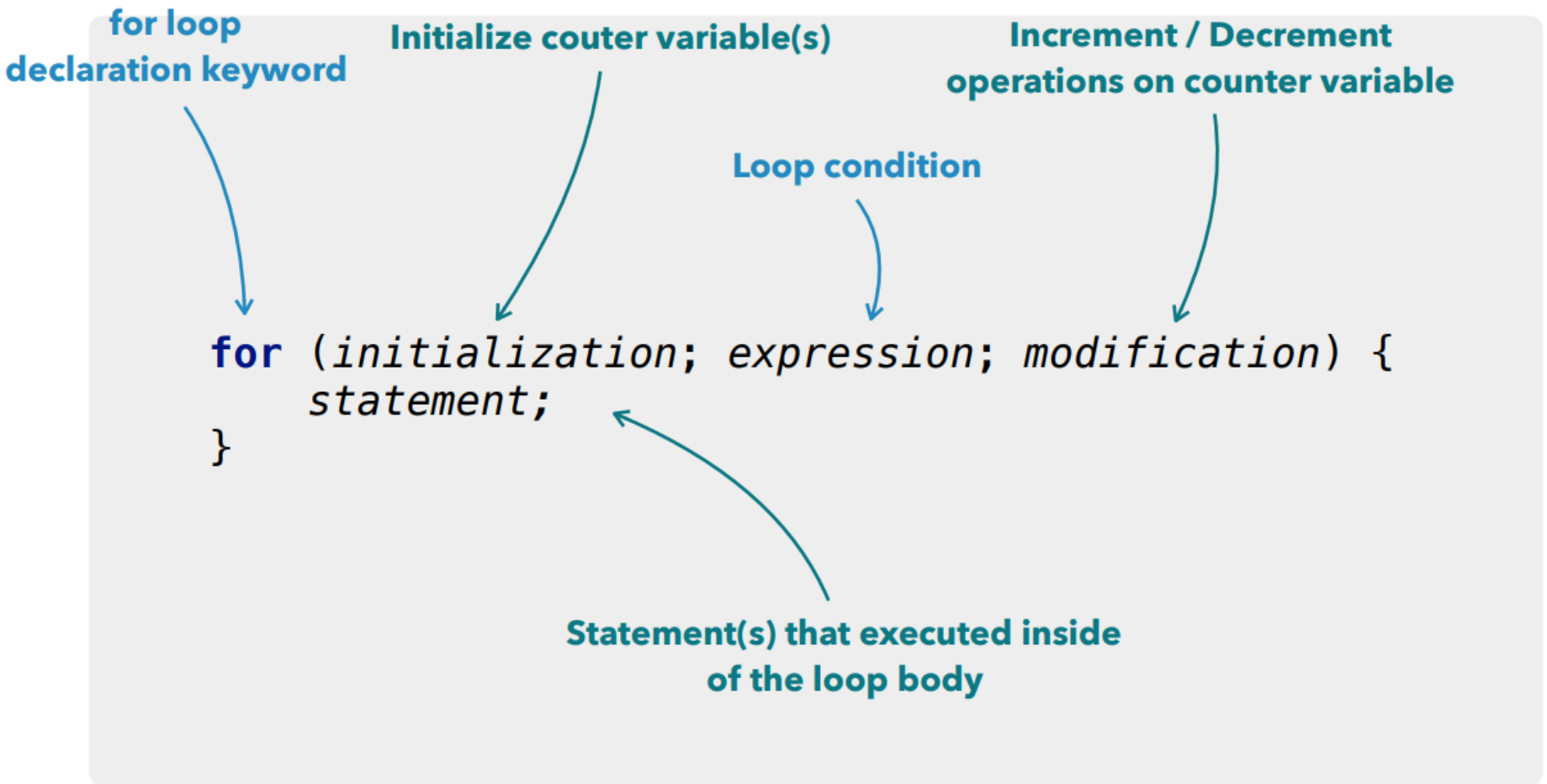
FOR LOOP: SUMMARY

- Control structure that allows us to repeat certain operations by incrementing or decrementing and evaluating a loop counter
- Boolean expression is evaluated before the first iteration of the loop, hence executed zero or many times
- Usually used when number of iterations are known in advance

FOR LOOP: FLOWCHART



FOR LOOP: SYNTAX



FOR LOOP: CODE EXAMPLE

Code

```
for (int i = 0; i < 5; i++) {
    System.out.print("i = " + i + "; ");
}
```

Console output

```
i = 0; i = 1; i = 2; i = 3; i = 4;

Process finished with exit code 0
```

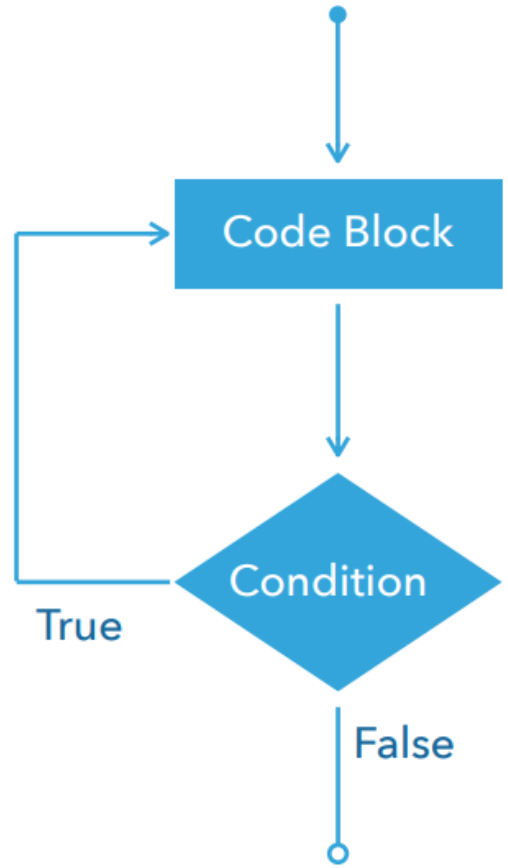
Agenda

- While loop
- For loop
- **Do While loop**
- Arrays
- Practice

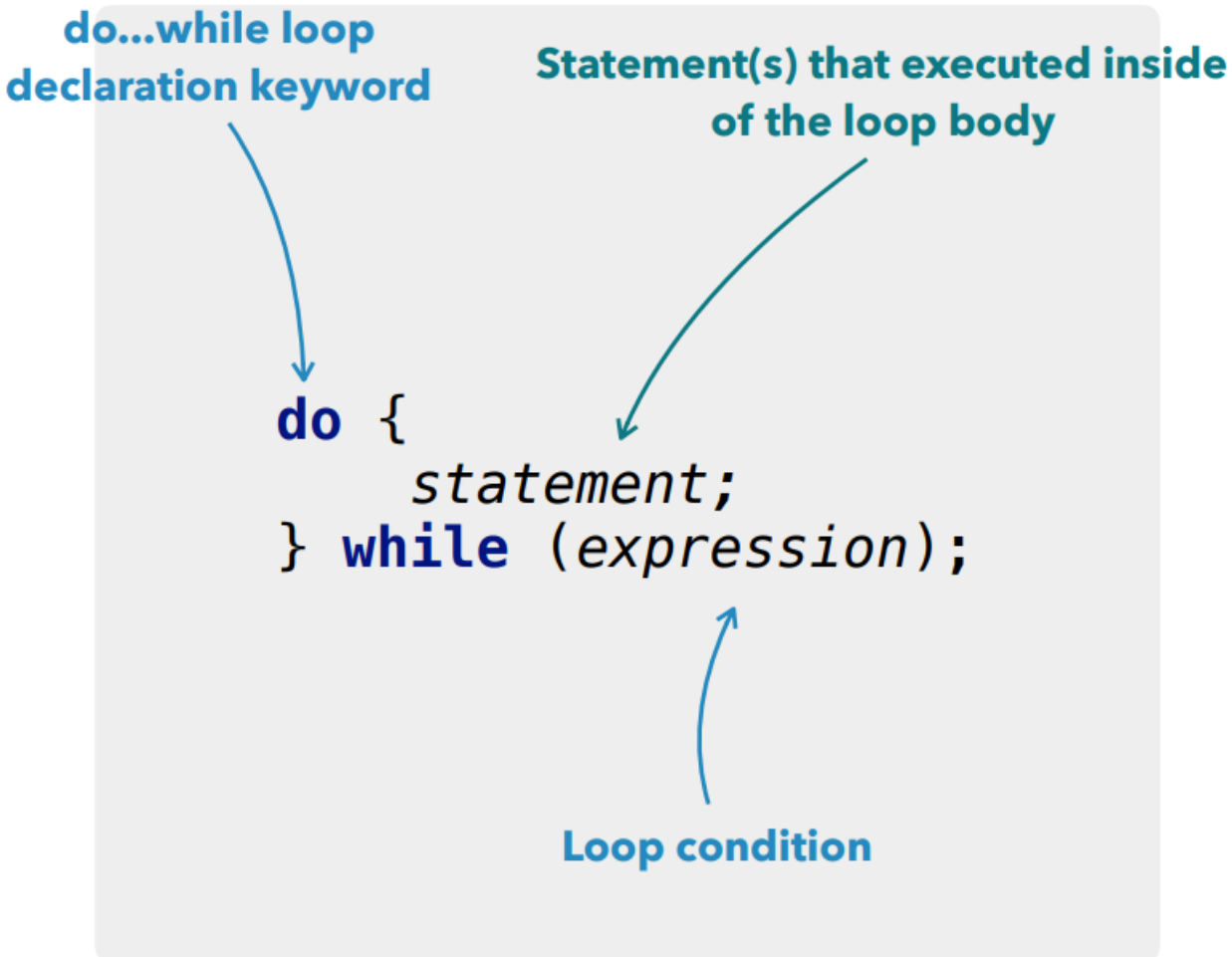
DO WHILE LOOP: SUMMARY

- Repeats a statement or block of statements while its controlling boolean expression is true
- Boolean expression is evaluated after the first iteration of the loop, hence executed one or many times
- Usually used when number of iterations are known in advance

DO WHILE LOOP: FLOWCHART



DO WHILE LOOP: SYNTAX



DO WHILE LOOP: CODE EXAMPLE

Code

```
int i = 0;
do {
    System.out.print("i = " + i + "; ");
    i++;
} while (i < 5);
```

Console output

```
i = 0; i = 1; i = 2; i = 3; i = 4;

Process finished with exit code 0
```

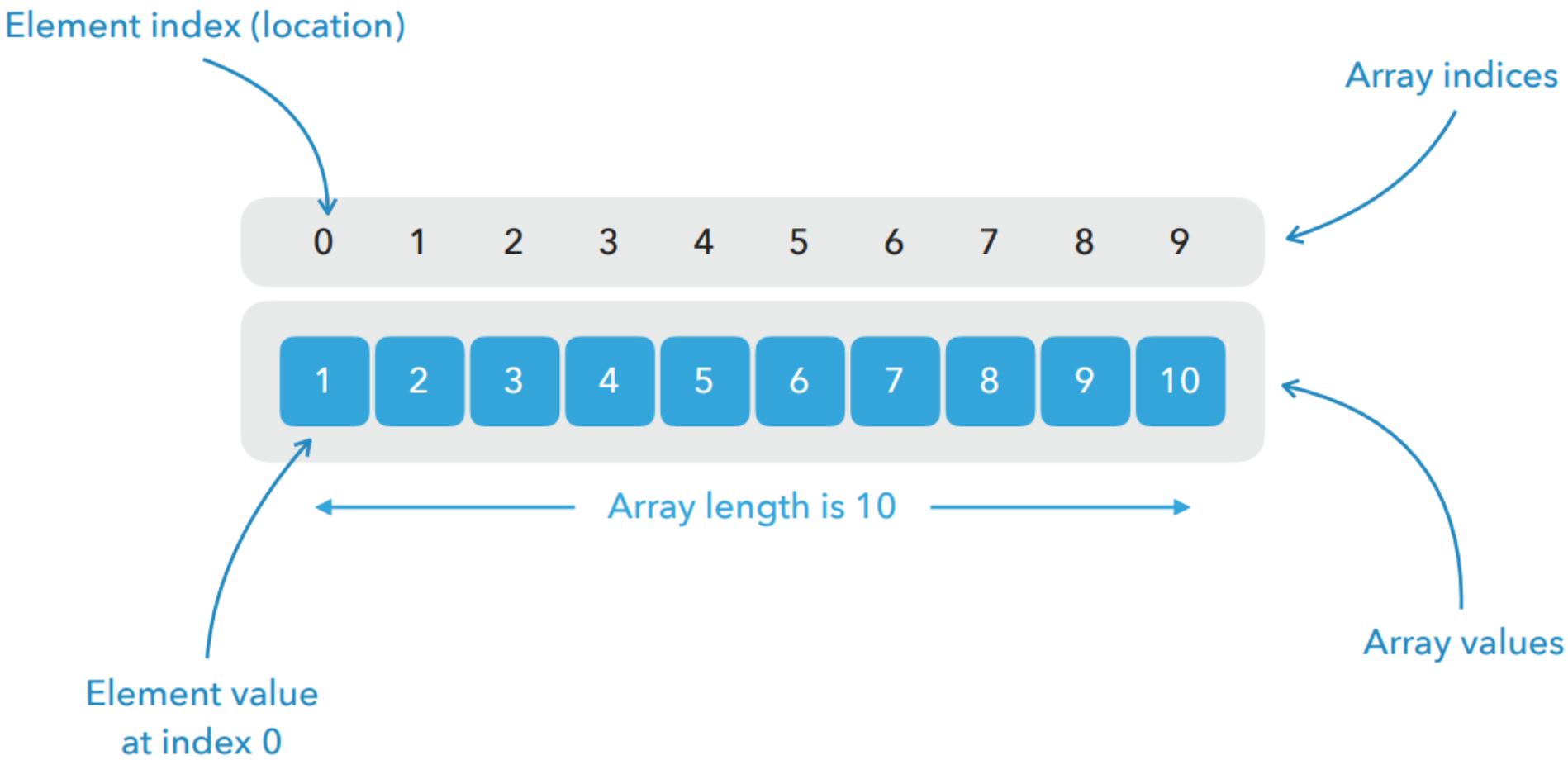
Agenda

- While loop
- For loop
- Do While loop
- **Arrays**
- Practice

DEFINITION

- An array is a container object that holds a fixed number of values of a single type
- The length of an array is established when the array is created
- After creation, its length is fixed

ARRAYS VISUALISATION



ARRAYS DECLARATION: SYNTAX

- Array declaration without instantiation

```
type[] name;
```

- Array declaration with instantiation

```
type[] name = new type[size];
```

- Array declaration with inline initialization

```
type[] name = {var1,.., varN};
```

ARRAY DECLARATION: INSTANTIATION CODE EXAMPLE

Code

```
int[] leapYears = new int[3];

leapYears[0] = 2020; leapYears[1] = 2016; leapYears[2] = 2012;

System.out.println("Leap years = " + Arrays.toString(leapYears));
```

Console output

```
Leap years = [2020, 2016, 2012]

Process finished with exit code 0
```

ARRAY DECLARATION: INLINE INITIALIZATION CODE EXAMPLE

Code

```
int[] leapYears = {2020, 2016, 2012};

System.out.println("Leap years = " + Arrays.toString(leapYears));
```

Console output

```
Leap years = [2020, 2016, 2012]

Process finished with exit code 0
```

WORKING WITH ARRAYS

- When working with arrays, loops are often used because of array iterable nature
- Array contains elements of the single type and size is fixed and known in advance

1. EXAMPLE: PRINTING ARRAY CONTENT

```
public class PrintingArrayDemo {

    public static void main(String[] args) {

        String[] alphabet = new String[5];

        alphabet[0] = "A";
        alphabet[1] = "B";
        alphabet[2] = "C";
        alphabet[3] = "D";
        alphabet[4] = "E";

        for (int i = 0; i < alphabet.length; i++) {
            System.out.println("[" + i + "]: " + alphabet[i]);

        }

    }

}
```

2. EXAMPLE: SUM OF ARRAY ELEMENTS

```
public class SumOfArrayElementsDemo {

    public static void main(String[] args) {

        int[] numbers = {1, 2, 3, 4, 5, 6, 7, 8, 9};
        int sum = 0;

        for (int i = 0; i < numbers.length; i++) {
            sum += numbers[i];
        }

        System.out.println("Sum = " + sum);

    }
}
```

3. EXAMPLE: FIND SMALLEST ELEMENT IN ARRAY

```
public class SmallestArrayElementDemo {

    public static void main(String[] args) {

        int[] numbers = {61, 97, 4, 37, 12};
        int min = numbers[0];

        for (int i = 0; i < numbers.length; i++) {
            if (numbers[i] < min) {
                min = numbers[i];
            }
        }

        System.out.println("min = " + min);

    }

}
```

Agenda

- While loop
- For loop
- Do While loop
- Arrays
- **Practice**

Task 1 – Warm up

- Sum 3 different Random numbers and print them to console

Task 2 – Dices

- Throw 3 random dices (3 random numbers printed to console) and show the message that player wins when all 3 dices show the same number
- Random randomGenerator = **new** Random();
  int rundomNumber1 = randomGenerator.nextInt(6);
  die1 = (**int**) (Math.*random*() * 6) + 1;

Task 3 – Statistics

- Create a Statistics class which will have these methods:
- min – displays min number from array
- max– displays max number from array
- average – displays average number from array

Task 3 – Statistics - Upgrade

- Fill array with randomly generated numbers using for loop

How to work with Console

```
Scanner my_scan = new Scanner(System.in);
String my_str = my_scan.nextLine();
```

```
Scanner scanner = new Scanner(System.in);
int consoleNumber = scanner.nextInt();
```

Task 4 – Calculator - Upgrade

- Upgrade your calculator launch class so user enters a and b number to the console and you print the result

Task 4 – Calculator – Upgrade part 2

- Upgrade your calculator launch class so user enters 1-4 number to the console and calculator does the action that the user selected
- 1 – Sum, 2 – Substract, 3 – Multiply, 4 – Divide
- Any other number – Try again

Task 5

Функциональные требования:

Разработать сервисный класс `ArrayService`, который не имеет состояния и реализует следующие функциональные методы:

- `int[] create(int size)` - метод должен вернуть пустой массив размера `size`;
- `void fillRandomly(int[] array)` - метод должен заполнить переданных массив `array` случайными числами в диапазоне от 0 до 100 включительно;
- `void printArray(int[] array)` - метод должен распечатать переданный массив `array` в консоль (логику необходимо реализовать самостоятельно, не используя класс `Arrays`);
- `int sum(int[] array)` - метод должен вернуть сумму всех элементов массива;
- `double avg(int[] array)` - метод должен рассчитать среднее арифметическое всех элементов массива (в случае, если массив пустой, вернуть 0);

Возможная реализация класса:

```
public class ArrayService {

  public int[] create(int size) {
    // TODO
  }

  public void fillRandomly(int[] array) {
    // TODO
  }

  public void printArray(int[] array) {
    // TODO
  }

  public int sum(int[] array) {
    // TODO
  }

  public double avg(int[] array) {
    // TODO
  }

}
```

Homework

Функциональные требования:

Разработать сервисный класс, который реализует два функциональных метода:

- Рассчет суммы всех чисел в заданном интервале **включительно**. Если начало интервала превышает конец, то необходимо выполнить расчет в обратном порядке. Например, сумма чисел от 3 до 7 равна `3 + 4 + 5 + 6 + 7 = 25` и наоборот: `7 + 6 + 5 + 4 + 3 = 25`.
- Подсчет количества четных чисел в заданном интервале **включительно**. Если начала интервала превышает конец, то необходимо выполнить расчет в обратном порядке. Например, количество четных чисел в диапазоне от 2 до 9 равно `2, 4, 6, 8 => 4` и наоборот: `8, 6, 4, 2 => 4`.

Оба метода должны быть реализованы используя циклы. Логику необходимо реализовать в отдельном классе `NumberService`:

```
public class NumberService {

  public int rangeSum(int start, int finish) {
    //TODO
  }

  public int rangeEvenCount(int start, int finish) {
    //TODO
  }

}
```


THANK YOU FOR YOUR ATTENTION
YouTube