

# SELECTIONS AND REPETITIONS WITH CODE EXAMPLES

## IF

“IF” block will be executed if the condition is fulfilled or true.

```
int num = 3

if (num > 2) {
    System.out.println(
        "Big Tree");
}

//output
//Big Tree
```

## IF-ELSE

“ELSE” block will be executed if the condition in “IF” block is not fulfilled or false.

```
int num = 1

if (num > 2) {
    System.out.println("Big Tree");
} else {
    System.out.println("not Tree");
}

//output
//not Tree
```

## CHAINING IF-ELSE

The first block that the condition is fulfilled will be executed

```
int num = 1

if (num > 2) {
    System.out.println("Big Tree");
} else if (num > 0) {
    System.out.println("Small Tree");
} else {
    System.out.println("not Tree");
}

//output
//Small Tree
```

## SWITCH CASE

Switch is usually more compact than lots of nested if else and therefore, more readable. Switch only accepts primitive types as key and constants as cases. The disadvantage using switch is you can fall through to the next case if you omit the break.

```
int day = 1;
switch (day) {
    case 1:
        System.out.println("Monday");
        break;
    case 2:
        System.out.println("Tuesday");
        break;
}

//output
//Monday
```

## WHILE LOOP

Use while loop if the number of iteration is not fixed.

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

//output
//01234
```

## DO WHILE LOOP

Same as while loop but “do” block will be executed first before checking the condition.

```
int i = 1;
do {
    System.out.print(i);
    i++;
}
while (i < 0);

//output
//1
```

## FOR LOOP

Use for loop if the number of iteration is fixed.

```
for (int i = 2; i < 5; i++) {
    System.out.print(i);
}

//output
//234
```

## REFERENCES

<https://www.javatpoint.com/java-if-else>  
<https://www.javatpoint.com/java-switch>  
<https://www.javatpoint.com/java-for-loop>