

Control Statements (Chapter 5 of Schilit)

Object Oriented Programming BS (AI and
MMG) II

Compiled By:
Abdul Ghafoor

Control Statements

- Change the normal flow of execution
 - Selection Statement:
 - Flow changes based on outcome of an expression
 - Iteration Statement:
 - Repeat one or more statements
 - Jump Statements:
 - Allow you to jump from one section to other



if statement

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

```
int a, b;  
//...  
if(a < b) a = 0;  
else b = 0;
```




Write a java program to find maximum between
two Numbers using only if statement

Write a java program to find wether a number
is positive or negative using if-else

Nested if

```
if(i == 10) {  
    if(j < 20) a = b;  
    if(k > 100) c = d; // this if is  
    else a = c;        // associated with this else  
}  
else a = d;            // this else refers to if(i == 10)
```

Write a java program using if to develop a login program, it should take user name and password as input

Write a java program to check whether the student is pass/fail based on his marks

Write a java program to determine whether a character is vowel or consonant

if-else-if

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

Demo season according to month



switch statement

```
switch (expression) {  
    case value1:  
        // statement sequence  
        break;  
  
    case value2:  
        // statement sequence  
        break;  
  
    .  
    .  
    .  
  
    case valueN :  
        // statement sequence  
        break;  
    default:  
        // default statement sequence  
}
```

```
// A simple example of the switch.
class SampleSwitch {
    public static void main(String args[]) {
        for(int i=0; i<6; i++)
            switch(i) {
                case 0:
                    System.out.println("i is zero.");
                    break;
                case 1:
                    System.out.println("i is one.");
                    break;
                case 2:
                    System.out.println("i is two.");
                    break;
                case 3:
                    System.out.println("i is three.");
                    break;
                default:
                    System.out.println("i is greater than 3.");
            }
    }
}
```

```
// In a switch, break statements are optional.
class MissingBreak {
    public static void main(String args[]) {
        for(int i=0; i<12; i++)
            switch(i) {
                case 0:
                case 1:
                case 2:
                case 3:
                case 4:
                    System.out.println("i is less than 5");
                    break;
                case 5:
                case 6:
                case 7:
                case 8:
                case 9:
                    System.out.println("i is less than 10");
                    break;
                default:
```

Omitting
break from
some cases

Question

- Which types of data a switch can accept?



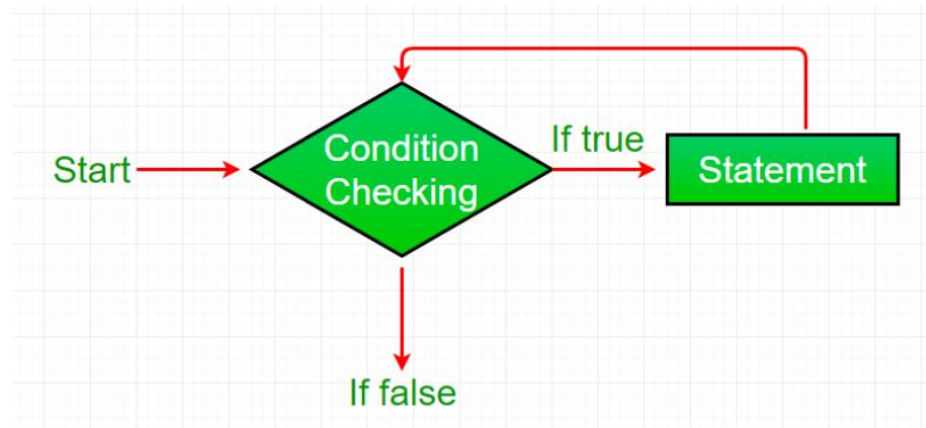
Iteration Statements

while, do-while, for



while

Flowchart For while loop (Control Flow):



- Used when number of repetitions is unknown
- Demo


```
// Program to display numbers from 1 to 5

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

        // declare variables
        int i = 1, n = 5;

        // while loop from 1 to 5
        while(i <= n) {
            System.out.println(i);
            i++;
        }
    }
}
```

Iteration	Variable	Condition: $i \leq n$	Action
1st	<div>i = 1</div> <div>n = 5</div>	true	<div>1 is printed.</div> <div>i is increased to 2.</div>
2nd	<div>i = 2</div> <div>n = 5</div>	true	<div>2 is printed.</div> <div>i is increased to 3.</div>
3rd	<div>i = 3</div> <div>n = 5</div>	true	<div>3 is printed.</div> <div>i is increased to 4.</div>
4th	<div>i = 4</div> <div>n = 5</div>	true	<div>4 is printed.</div> <div>i is increased to 5.</div>
5th	<div>i = 5</div> <div>n = 5</div>	true	<div>5 is printed.</div> <div>i is increased to 6.</div>
6th	<div>i = 6</div> <div>n = 5</div>	false	The loop is terminated

while

- while with no body

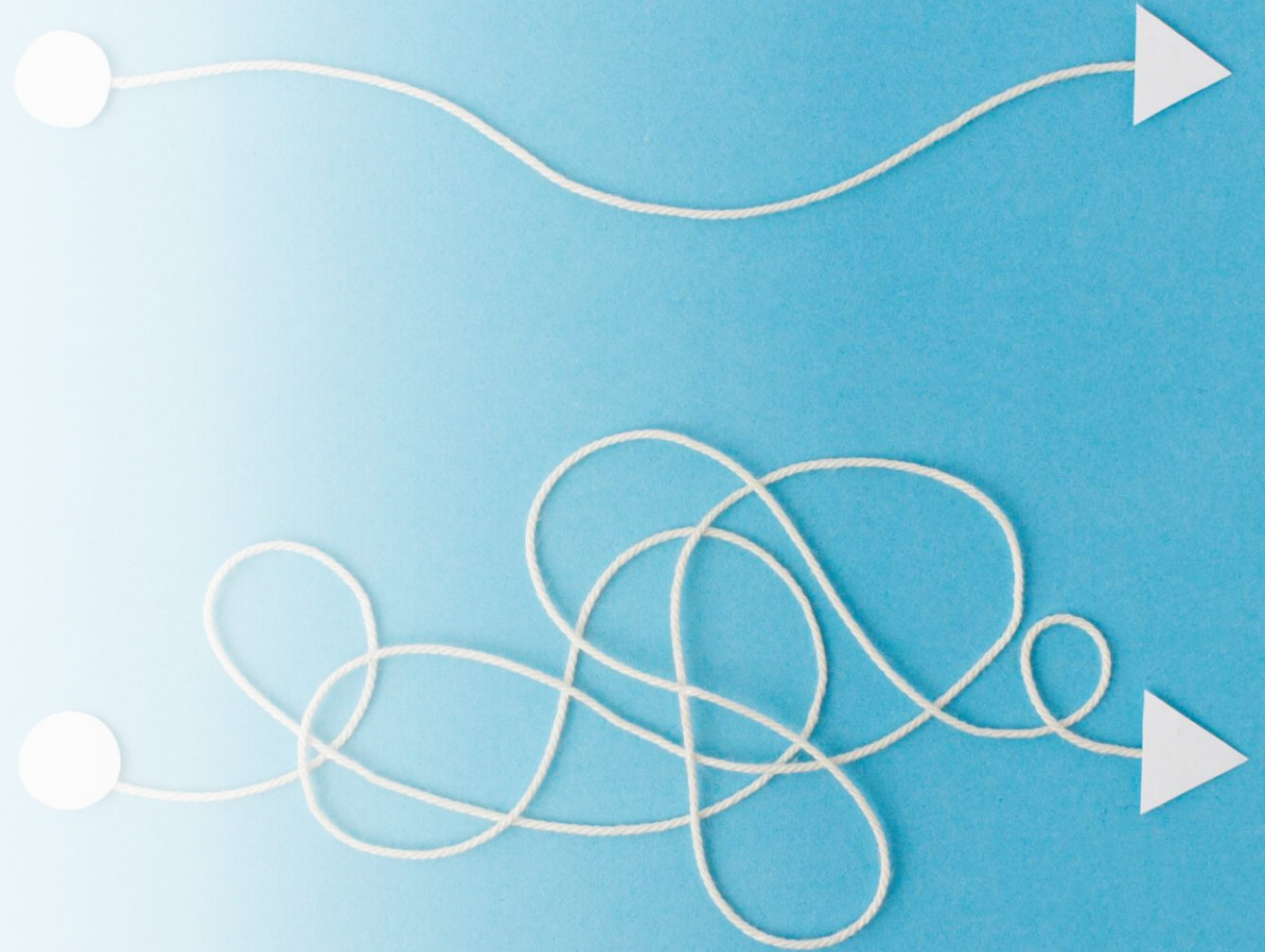
```
// The target of a loop can be empty.
class NoBody {
    public static void main(String args[]) {
        int i, j;

        i = 100;
        j = 200;

        // find midpoint between i and j
        while(++i < --j); // no body in this loop

        System.out.println("Midpoint is " + i);
    }
}
```

Using a while
loop take
continuous input
from user until
that input
becomes one



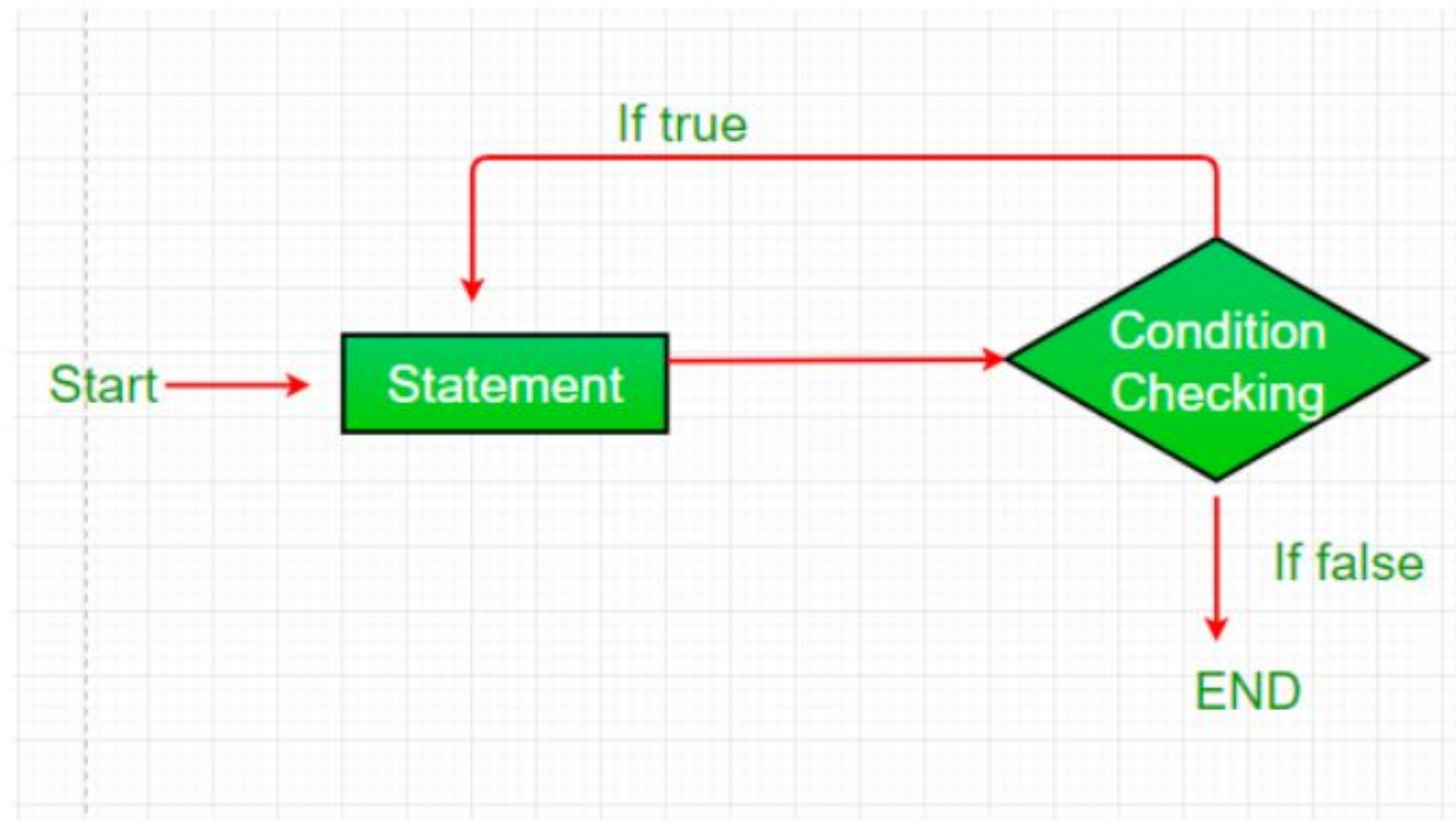
The background is a dark blue gradient with a faint, glowing line graph in red and white. The graph has several peaks and valleys, suggesting data trends. Scattered throughout the background are binary digits (0s and 1s) in a light blue color, some of which are slightly blurred, giving a sense of depth and digital data. The overall aesthetic is tech-oriented and modern.

Using a while loop print natural
numbers from 1 to 100

do-while

```
do {  
    // body of loop  
} while (condition);
```


Flowchart do-while loop:



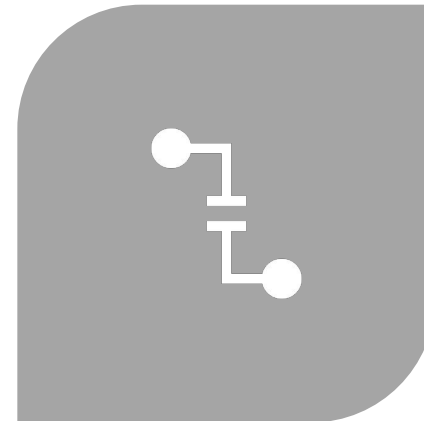


Do sum of first 10 natural numbers
using do while Loop

Task



TAKE AN INTEGER NUMBER AS
INPUT FROM USER, IT SHOULD BE
GREATER THAN 100000



DIVIDE THAT NUMBER WITH 10,
UNTIL IT BECOMES LESSER THAN 100

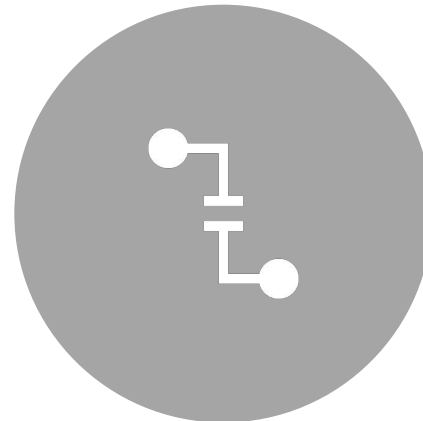
```
// infinite while loop
while(true){
    // body of loop
}
```

```
// infinite do...while loop
int count = 1;
do {
    // body of loop
} while(count == 1)
```

for



USED WHEN NUMBER OF
ITERATIONS ARE ALREADY KNOWN



LOOP VARIABLE DECLARED
INSIDE/OUTSIDE LOOP

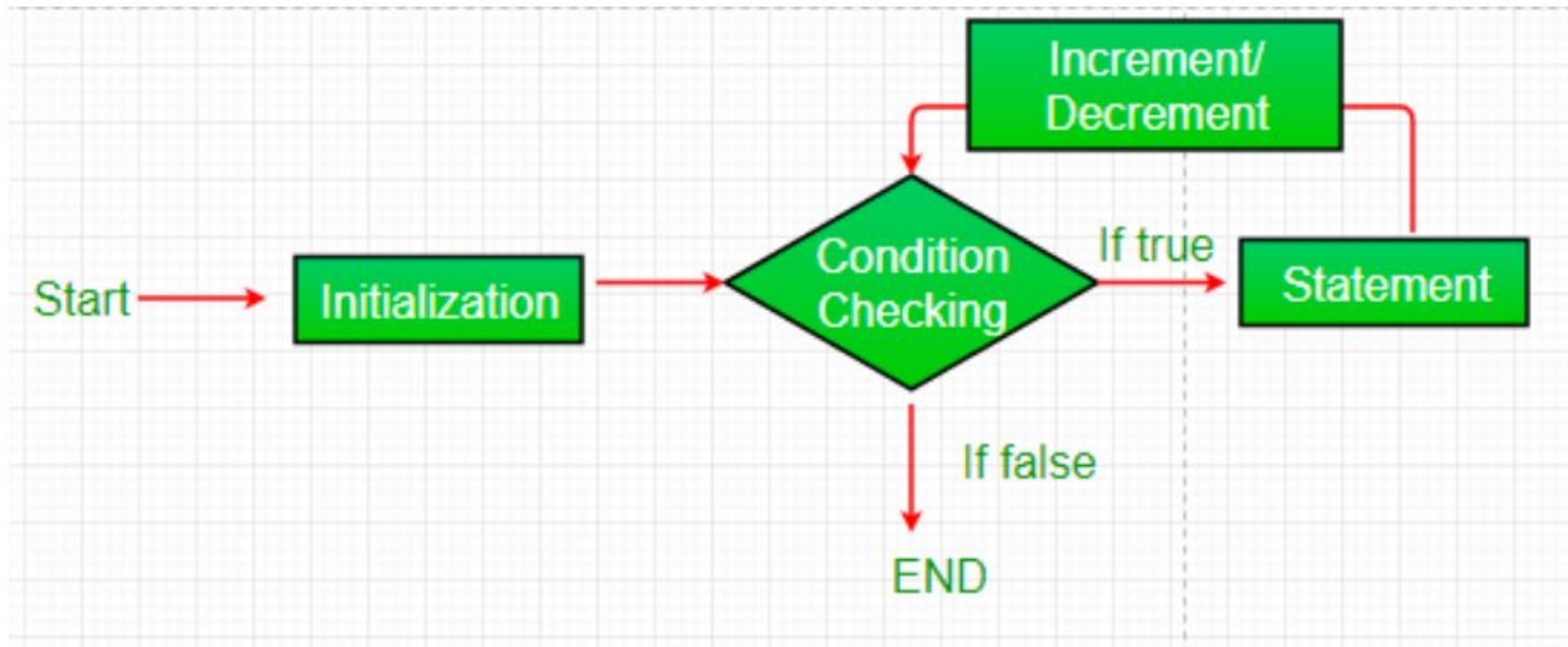
For

```
class Sample {  
    public static void main(String args[]) {  
        int a, b;  
  
        b = 4;  
        for(a=1; a<b; a++) {  
            System.out.println("a = " + a);  
            System.out.println("b = " + b);  
            b--;  
        }  
    }  
}
```

```
// Using the comma.  
class Comma {  
    public static void main(String args[]) {  
        int a, b;  
  
        for(a=1, b=4; a<b; a++, b--) {  
            System.out.println("a = " + a);  
            System.out.println("b = " + b);  
        }  
    }  
}
```


For

Flow chart for loop (For Control Flow):



For

- All three parts of for loop are optional
- An infinite loop

```
for( ; ; ) {  
    // ...  
}
```

For loop Tasks

- Write a for loop to print first n natural numbers in reverse order
- Write a for loop to print output like this:
 - Line 1
 - Line 2
 - Line 3
 - .
 - .
 - .
 - Line 10



For-each version of for loop

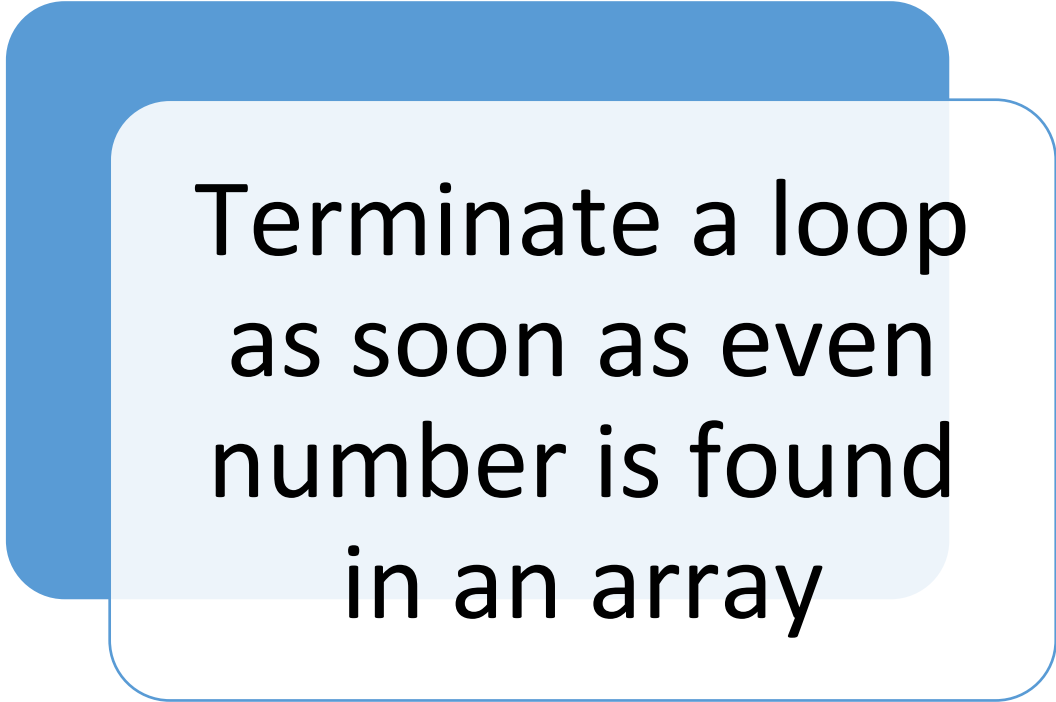
for(type itr-var : collection) statement-block

Early Termination



Break statement

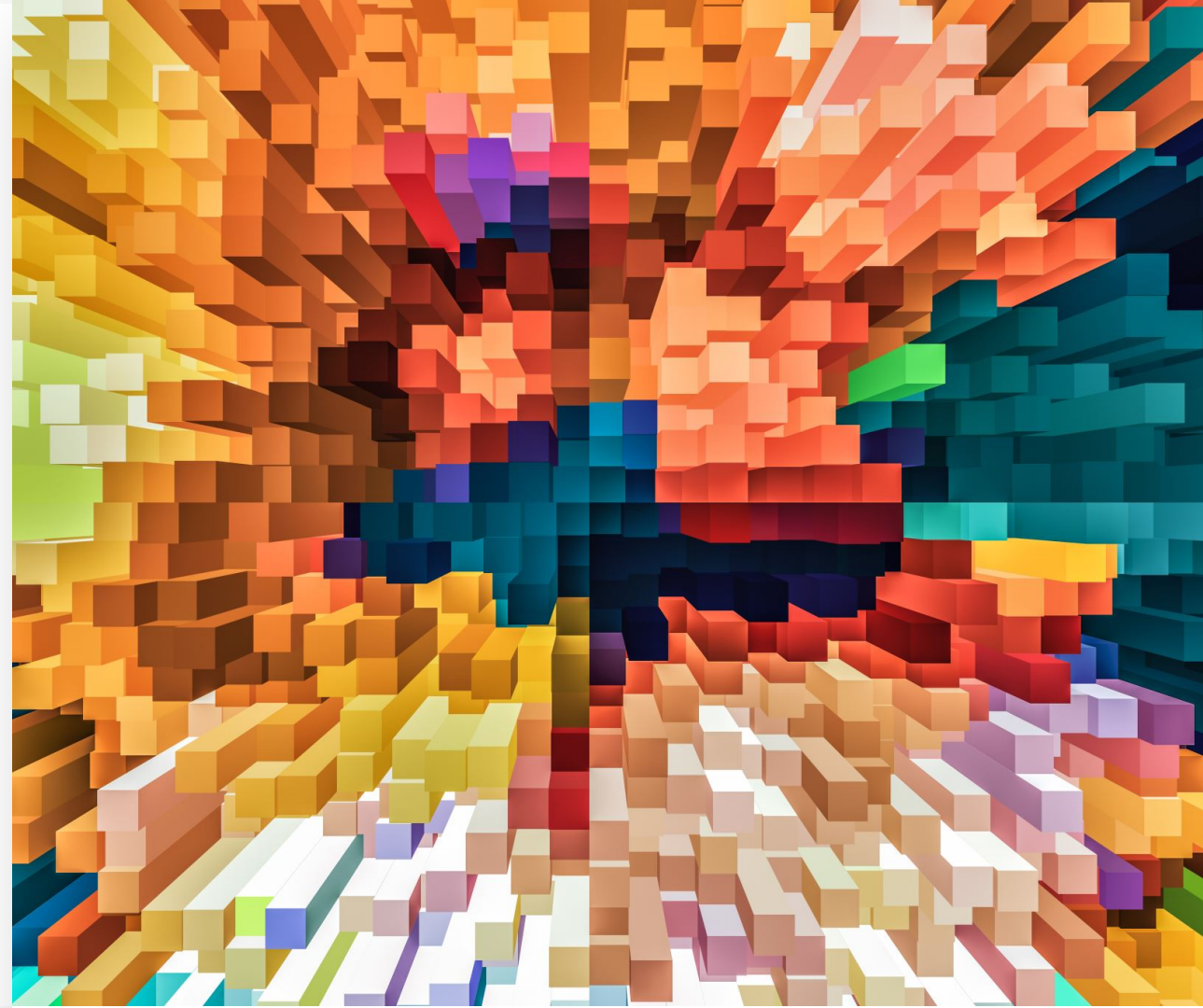
The diagram shows a light blue rounded rectangle representing a loop body, which is partially enclosed by a darker blue rounded rectangle representing the loop structure. The text 'Break statement' is centered within the light blue area.



Terminate a loop
as soon as even
number is found
in an array

The diagram shows a light blue rounded rectangle representing a loop body, which is partially enclosed by a darker blue rounded rectangle representing the loop structure. The text 'Terminate a loop as soon as even number is found in an array' is centered within the light blue area.

Multidimensional Array Iteration through nested loop



Nested Loops

```
public class MultiplicationTable {  
    public static void main(String[] args) {  
        int tableSize = 5;  
  
        // Outer loop: Controls the row (multiplicand)  
        for (int i = 1; i <= tableSize; i++) {  
            // Inner loop: Controls the column (multiplier)  
            for (int j = 1; j <= tableSize; j++) {  
                // Print the product of the current row and column  
                System.out.print(i * j + "\t");  
            }  
            // Move to the next line after each row is printed  
            System.out.println();  
        }  
    }  
}
```

Outer Loop (i)	Inner Loop (j)	Output
1	1	12
1	2	24
2	1	24
2	2	48

Jump Statements

Break:

- Used to terminate immediate loop
- Used to terminate a case statement in switch

Continue:

- Skips a particular iteration of loop

Return:

- Return the control to immediate caller

Return passes control to java run-time system, as Run time system calls the main

```
// Demonstrate return.
class Return {
    public static void main(String args[]) {
        boolean t = true;

        System.out.println("Before the return.");

        if(t) return; // return to caller

        System.out.println("This won't execute.");
    }
}
```



Use of var

Var Keyword

- Var is used to automatically detect data type without specifying the data type.
- Rules:
 - var can be used inside main method any where
 - var can not be used with instance variables
 - var can not be used without initialization
 - var can not be used with arraylists and generics
 - var can not be used as method parameters