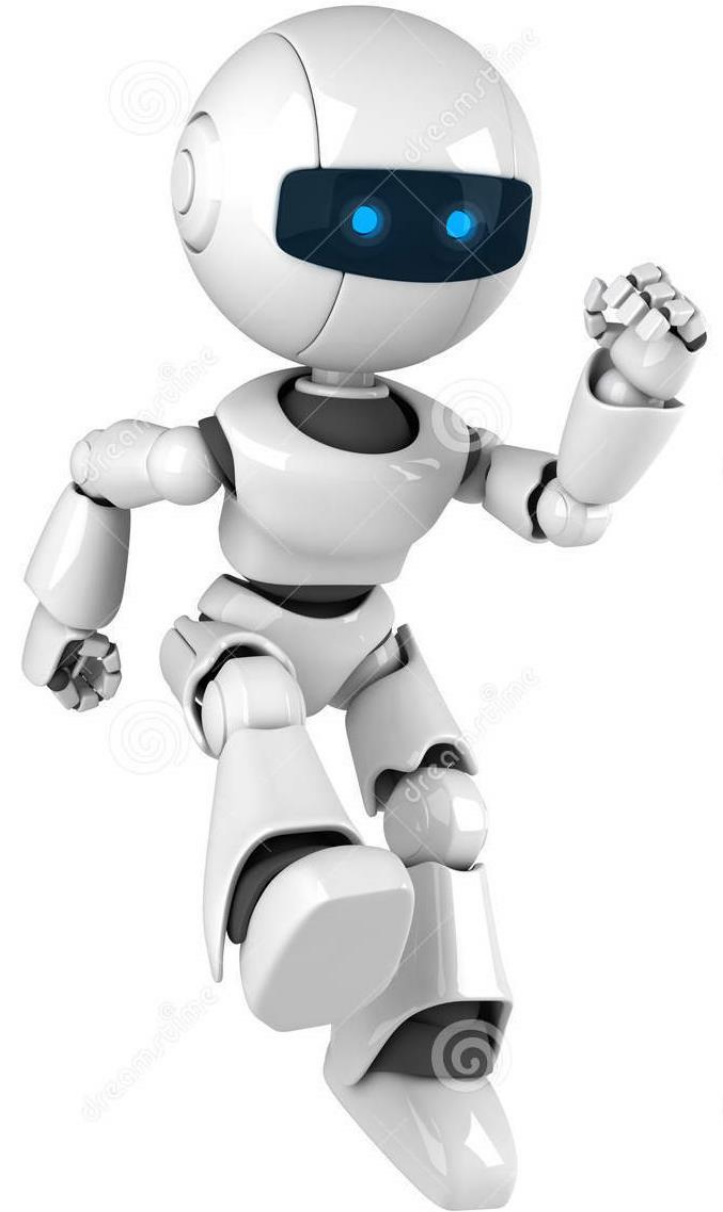


# JUMP STATEMENT



**IN THE NAME OF ALLAH**

**بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ**

**THE GRACIOUS, THE MERCIFUL.**



# **GETS - GETTING EDUCATION WITH TECHNOLOGICAL SYSTEM**

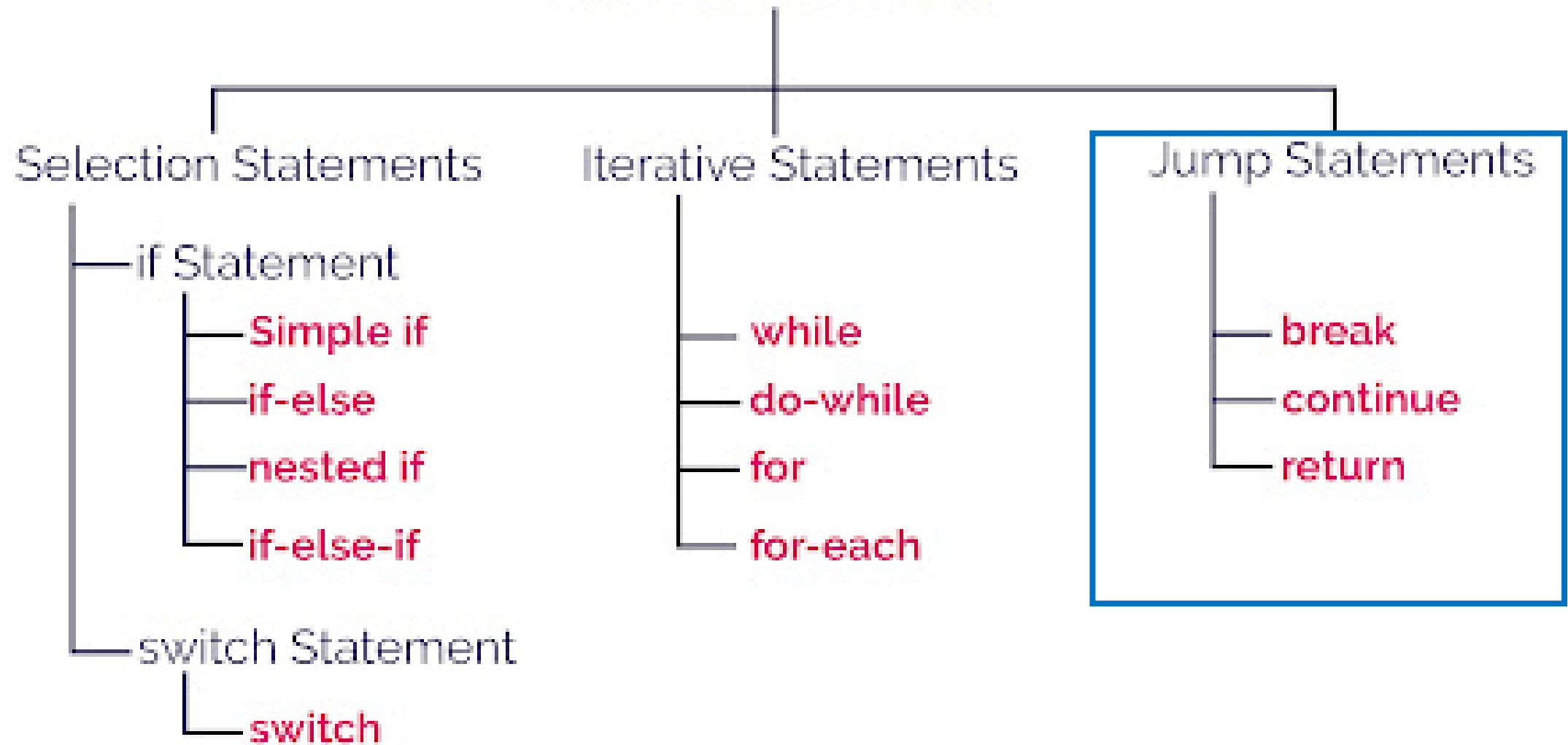
Instructor: Sir A.Rehman Ali Brohi

# LECTURE: 7

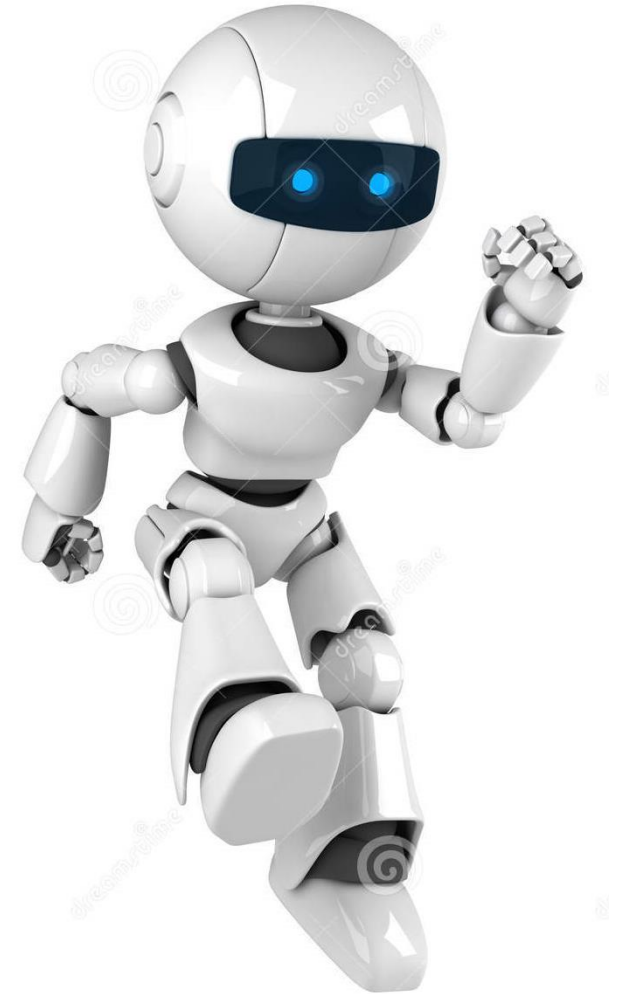
## Jump Statements



# Control Statements



**In Java jump statements are mainly used to transfer control to another part of our program depending on the conditions.**



## 3 TYPES OF JUMP STATEMENTS IN JAVA



**BREAK**

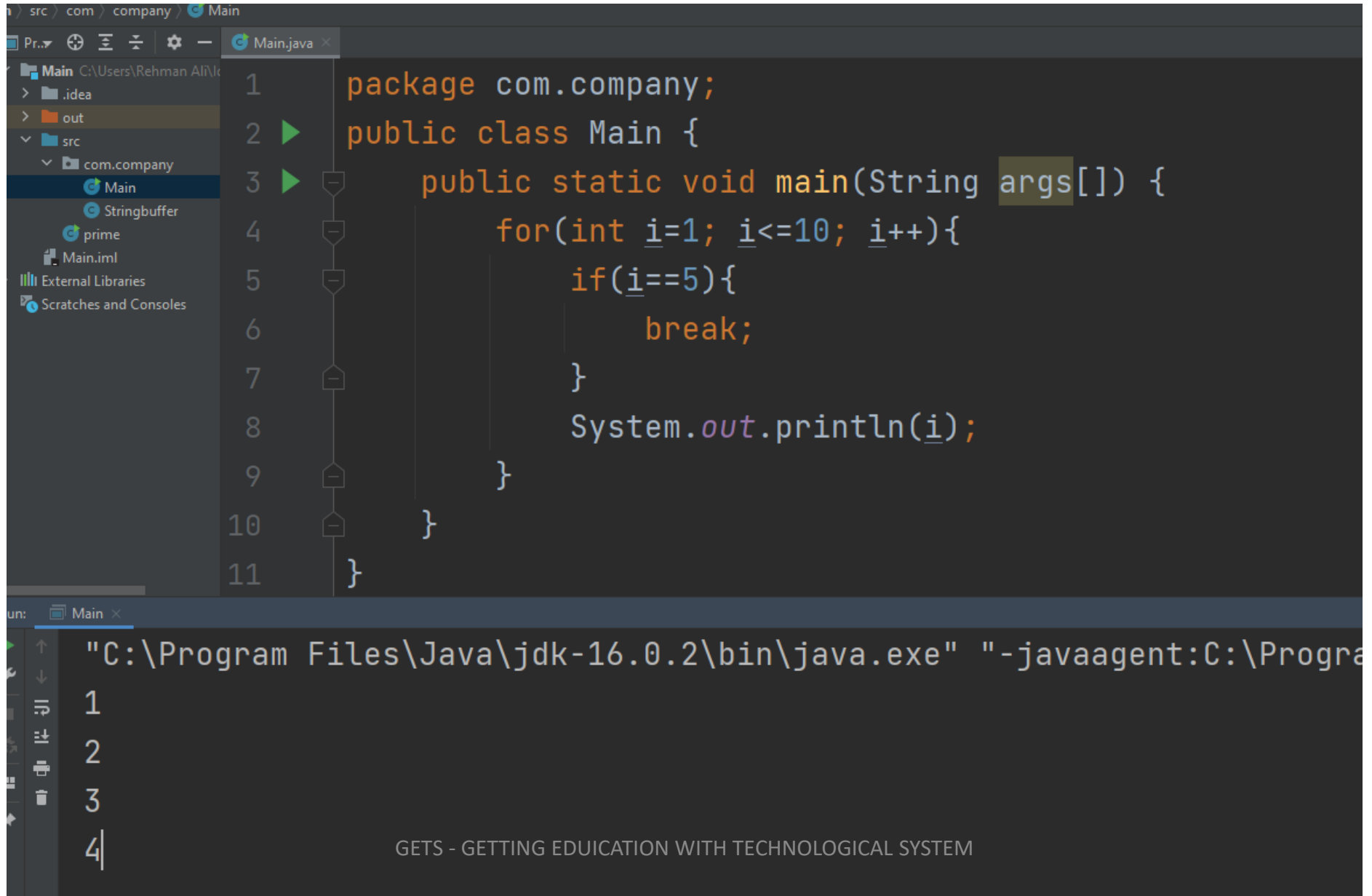
**CONTINUE**

**RETURN**

**When a `break statement` is encountered inside a loop is immediately terminated and the program control resumes at the next statement following the loop.**



# BREAK STATEMENT



```
1 package com.company;
2 public class Main {
3     public static void main(String args[]) {
4         for(int i=1; i<=10; i++){
5             if(i==5){
6                 break;
7             }
8             System.out.println(i);
9         }
10    }
11 }
```

The screenshot shows an IDE with a project named 'Main' in the 'com.company' package. The 'Main.java' file contains a Java program. The program defines a 'Main' class with a 'main' method. Inside the 'main' method, there is a 'for' loop that iterates from 1 to 10. Within the loop, there is an 'if' statement that checks if the current value of 'i' is 5. If it is, the 'break' statement is executed, which immediately exits the loop. After the loop, the program prints the value of 'i' using 'System.out.println(i);'. The IDE's output window at the bottom shows the command to run the program: '"C:\Program Files\Java\jdk-16.0.2\bin\java.exe" "-javaagent:C:\Progra'.

The image shows a screenshot of an IDE with a dark theme. The top part displays a Java file named `Main.java` with the following code:

```
1 package com.company;  
2 public class Main {  
3     public static void main(String args[]) {  
4         for(int i=1; i<=10; i++){  
5             System.out.println(i);  
6             if(i==5){  
7                 break;  
8             }  
9         }  
10    }  
11 }
```

The left sidebar shows the project structure:

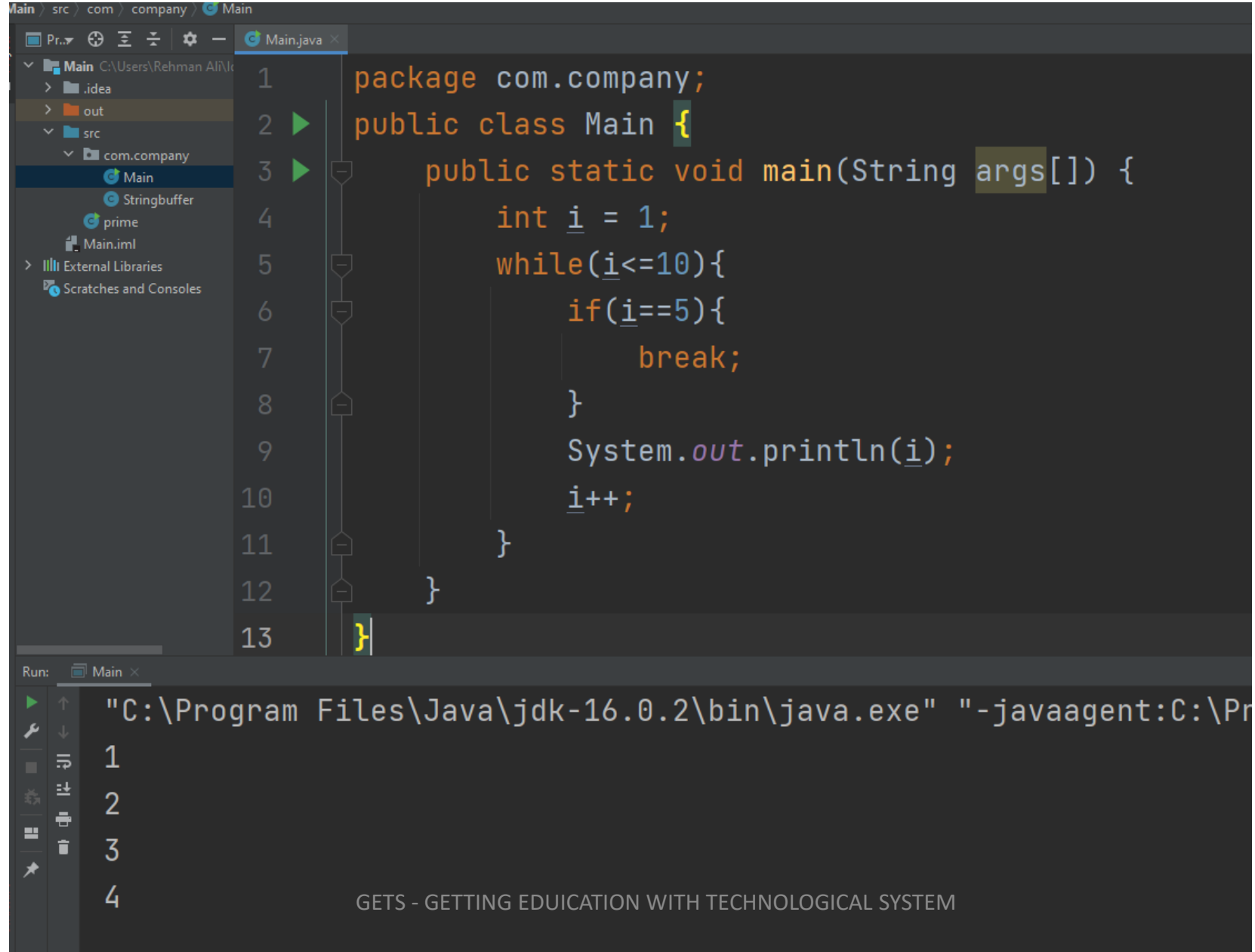
- Main C:\Users\Rehman AI\l...
  - .idea
  - out
  - src
    - com.company
      - Main
      - Stringbuffer
      - prime
      - Main.iml
  - External Libraries
  - Scratches and Consoles

The bottom part of the IDE shows the Run configuration and the output console. The Run configuration is set to `Main`. The output console shows the command used to run the program:

```
"C:\Program Files\Java\jdk-16.0.2\bin\java.exe" "-javaagent:C:\Progr
```

Below the command, there are five empty lines for output, numbered 1 to 5.

# Break statements using while loop



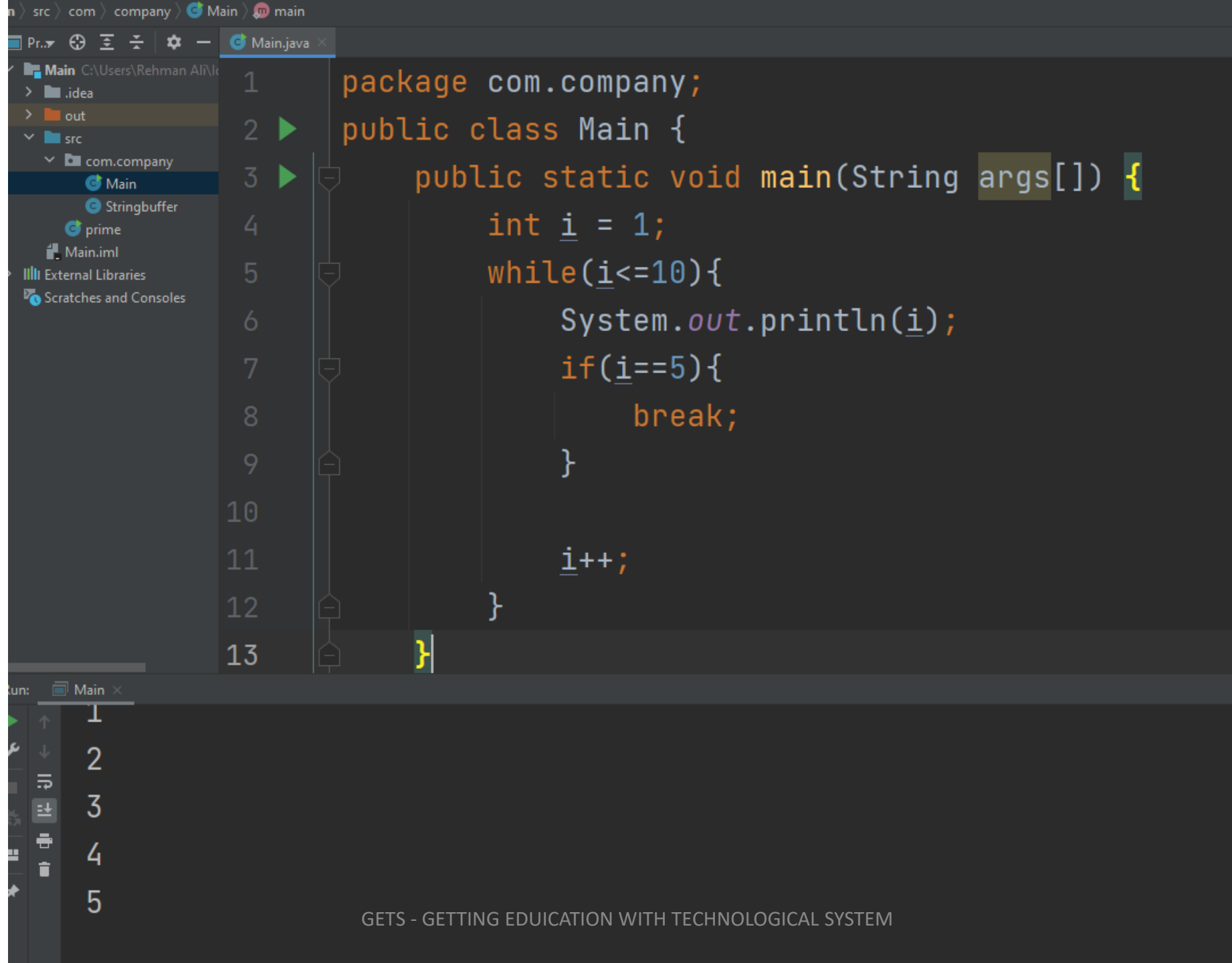
The screenshot displays an IDE with a project named 'Main' in the 'src' directory. The file 'Main.java' is open, showing the following code:

```
1 package com.company;
2 public class Main {
3     public static void main(String args[]) {
4         int i = 1;
5         while(i<=10){
6             if(i==5){
7                 break;
8             }
9             System.out.println(i);
10            i++;
11        }
12    }
13 }
```

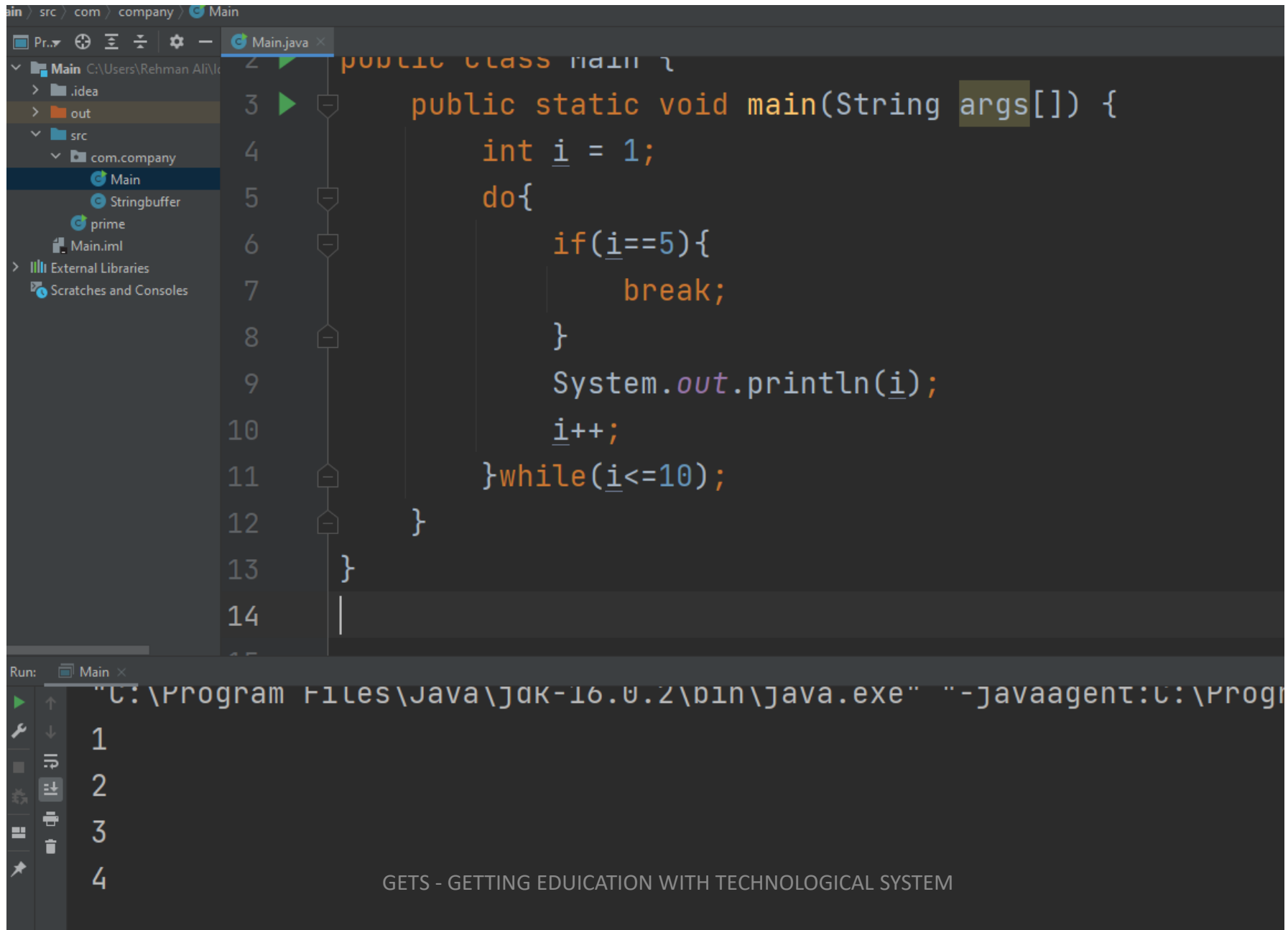
The code is executed, and the output is shown in the 'Run' console:

```
Run: Main x
"C:\Program Files\Java\jdk-16.0.2\bin\java.exe" "-javaagent:C:\Pr
1
2
3
4
```

The output shows the numbers 1 through 4, indicating that the loop was broken out of when *i* reached 5.



# Break statements using do-while loop



The screenshot displays an IDE with a project named 'Main' in the 'src' directory. The 'Main.java' file is open, showing the following code:

```
public class Main {  
    public static void main(String args[]) {  
        int i = 1;  
        do{  
            if(i==5){  
                break;  
            }  
            System.out.println(i);  
            i++;  
        }while(i<=10);  
    }  
}
```

The code is annotated with line numbers 2 through 14. The 'do' block contains an 'if' statement that checks if 'i' is equal to 5. If true, it executes a 'break' statement, which exits the 'do-while' loop. Otherwise, it prints the value of 'i' and increments it. The 'while' condition is 'i <= 10'. Since the loop is 'do-while', it executes at least once. The 'break' statement is triggered when 'i' reaches 5, so the loop terminates after printing 4.

Below the code editor, the 'Run' console shows the command executed: `"C:\Program Files\Java\jdk-16.0.2\bin\java.exe" "-javaagent:C:\Progr`. The console output shows the numbers 1, 2, 3, and 4, each on a new line, indicating the successful execution of the program.

```

Main > src > com > company
Main.java x
Main
Stringbuffer
prime
Main.iml
External Libraries
Scratches and Consoles

public class Main {
    public static void main(String args[]) {
        int i = 1;
        do{
            System.out.println(i);
            if(i==5){
                break;
            }

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

Run: Main x
"C:\Program Files\Java\jdk-16.0.2\bin\java.exe" "-javaagent:C:\Pro
1
2
3
4
5

GETS - GETTING EDUCATION WITH TECHNOLOGICAL SYSTEM
Process finished with exit code 0

```

# JAVA CONTINUE STATEMENT

- The continue statement is used in loop control structure when you need to jump to the next iteration of the loop immediately. It can be used with for loop or while loop.

# CONTINUE STATEMENT

The screenshot displays an IDE window with a Java file named `Main.java`. The code defines a `public class Main` with a `main` method. Inside the `main` method, a `for` loop iterates from `i=1` to `i=10`. Within the loop, an `if` statement checks if `i` is equal to 5. If true, the `continue` statement is executed, which skips the rest of the loop body for that iteration. A comment explains that the `continue` statement will skip the rest of the statement. After the `if` block, the code prints the value of `i` using `System.out.println(i);`. The IDE's run console at the bottom shows the output of the program, which is a list of numbers from 1 to 10, with the number 5 missing, demonstrating the effect of the `continue` statement.

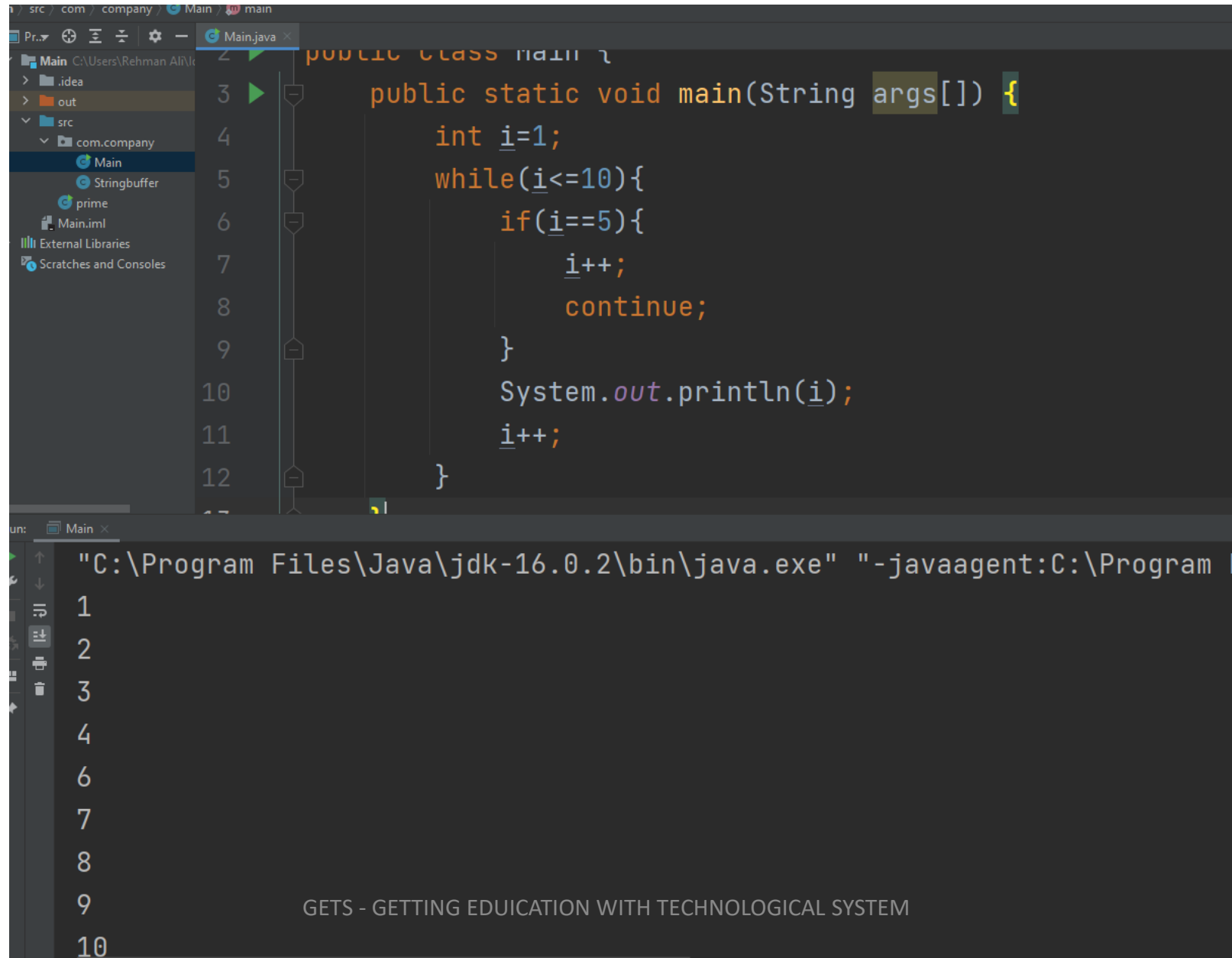
```
public class Main {  
    public static void main(String args[]) {  
        for(int i=1; i<=10; i++){  
            if(i==5){  
                //using continue statement  
                continue; //it will skip the rest of statement  
            }  
            System.out.println(i);  
        }  
    }  
}
```

Run: Main ×

1  
2  
3  
4  
6  
7  
8  
9  
10



# Continue statement using while loop



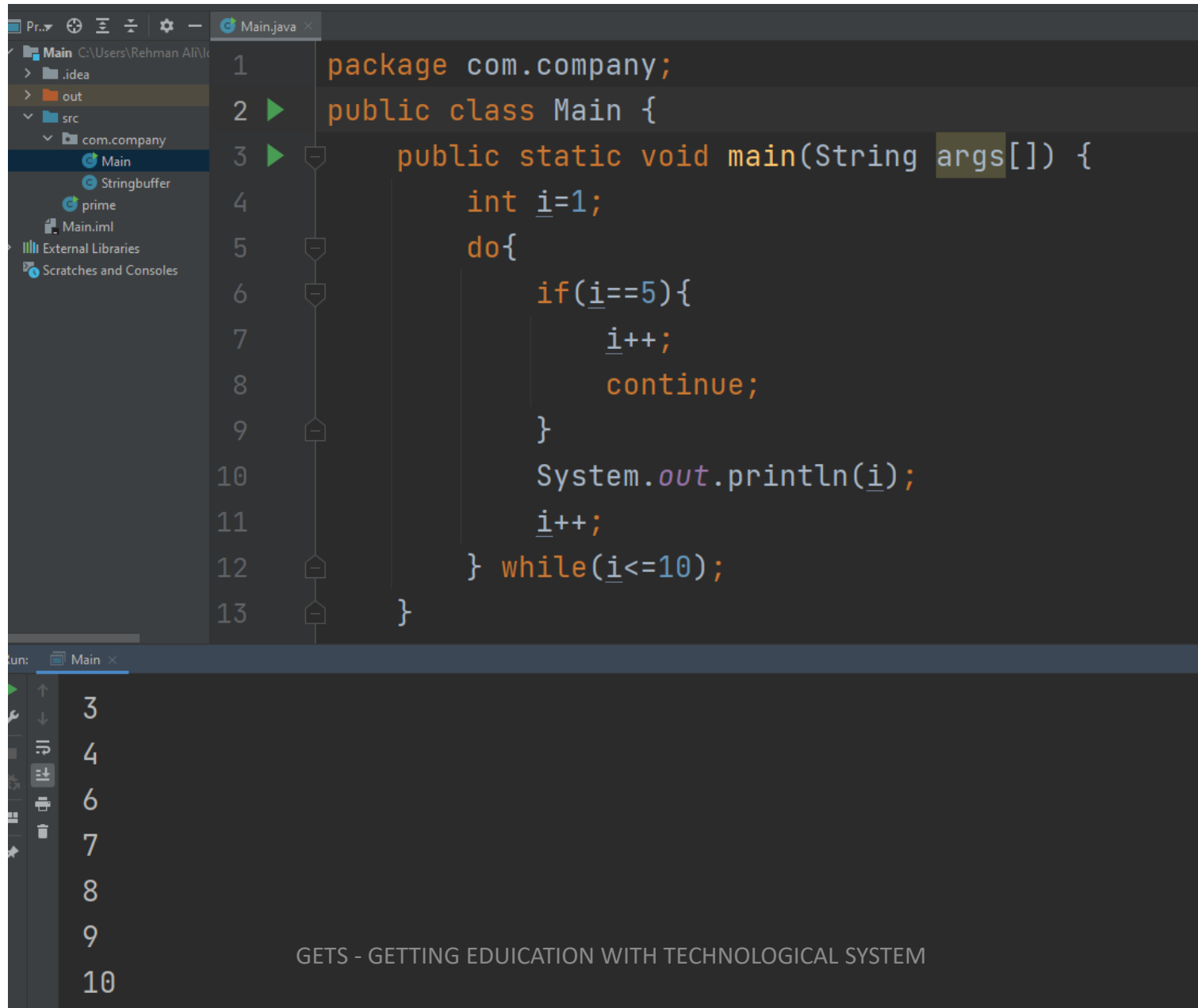
The screenshot displays an IDE with a Java file named `Main.java`. The code implements a `while` loop that prints the value of `i` from 1 to 10, but skips the value 5 using the `continue` statement. The execution output at the bottom shows the sequence of printed numbers: 1, 2, 3, 4, 6, 7, 8, 9, 10.

```
public class Main {  
    public static void main(String args[]) {  
        int i=1;  
        while(i<=10){  
            if(i==5){  
                i++;  
                continue;  
            }  
            System.out.println(i);  
            i++;  
        }  
    }  
}
```

Run: Main ×

```
"C:\Program Files\Java\jdk-16.0.2\bin\java.exe" "-javaagent:C:\Program F  
1  
2  
3  
4  
6  
7  
8  
9  
10
```

# Continue statements using do-while loop



The screenshot displays an IDE with a Java file named `Main.java`. The code defines a package `com.company` and a public class `Main`. Inside the class, there is a static method `main` that takes a `String args[]` parameter. The method initializes an integer `i` to 1 and enters a `do-while` loop. Within the loop, there is an `if` statement that checks if `i` is equal to 5. If true, it increments `i` and then uses the `continue` statement to skip the rest of the loop body and jump back to the start of the loop. If `i` is not 5, it prints the value of `i` and increments it. The loop continues until `i` is greater than 10. The IDE's left sidebar shows a project structure with a `com.company` package containing `Main`, `Stringbuffer`, and `prime`. The bottom panel shows the execution output, which lists the numbers 3, 4, 6, 7, 8, 9, and 10, indicating that the value 5 was skipped.

```
1 package com.company;
2 public class Main {
3     public static void main(String args[]) {
4         int i=1;
5         do{
6             if(i==5){
7                 i++;
8                 continue;
9             }
10            System.out.println(i);
11            i++;
12        } while(i<=10);
13    }
```

Run: Main x

3  
4  
6  
7  
8  
9  
10

# JAVA RETURN STATEMENT

- We will learn in OOP classes