

Operator Type	Category	Precedence
Unary	postfix	<i>expr++ expr--</i>
	prefix	<i>++expr --expr +expr -expr ~ !</i>
Arithmetic	multiplicative	<i>* / %</i>
	additive	<i>+ -</i>
Shift	shift	<i><< >> >>></i>
Relational	comparison	<i>< > <= >= instanceof</i>
	equality	<i>== !=</i>
Bitwise	bitwise AND	<i>&</i>
	bitwise exclusive OR	<i>^</i>
	bitwise inclusive OR	<i> </i>
Logical	logical AND	<i>&&</i>
	logical OR	<i> </i>
Ternary	ternary	<i>? :</i>
Assignment	assignment	<i>= += -= *= /= %= &= ^= = <<= >>= >>>=</i>

IN THE NAME OF ALLAH

بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ

THE GRACIOUS, THE MERCIFUL.

LECTURE: 4

JAVA OPERATORS



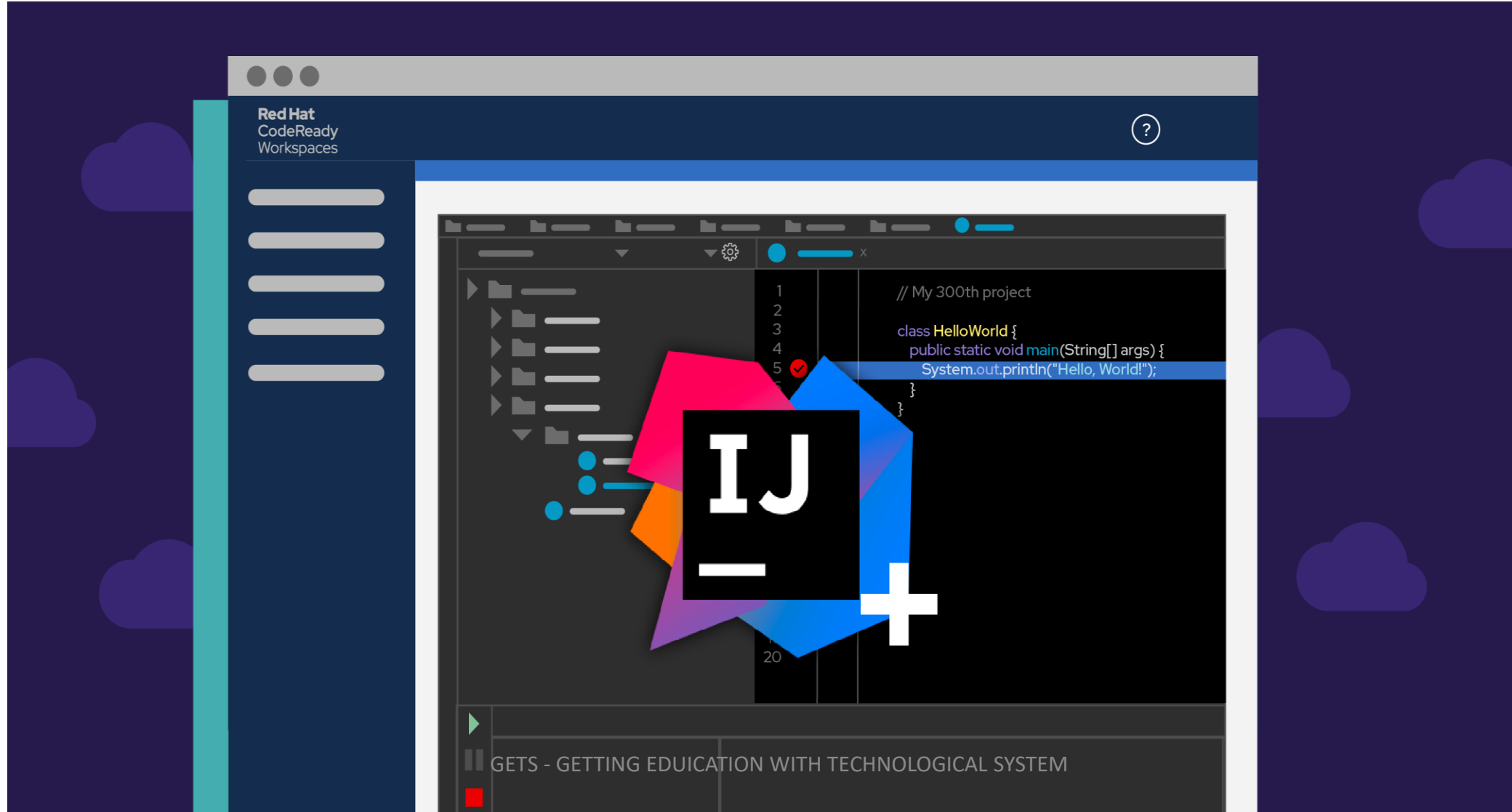


GETS - GETTING EDUCATION WITH TECHNOLOGICAL SYSTEM

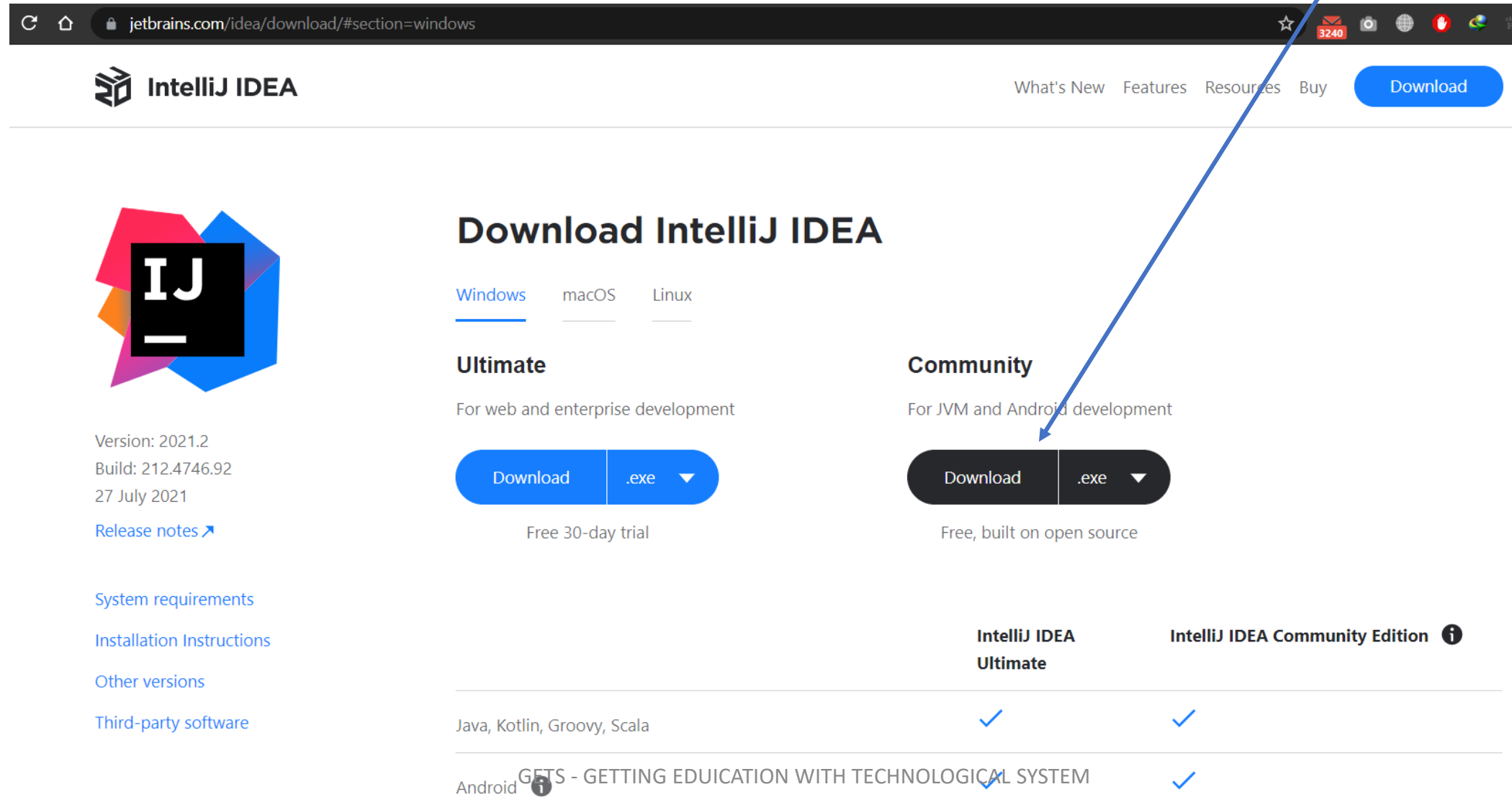
Instructor: Sir A.Rehman Ali Brohi

Operator Type	Category	Precedence
Unary	postfix	<i>expr++ expr--</i>
	prefix	<i>++expr --expr +expr -expr ~ !</i>
Arithmetic	multiplicative	<i>* / %</i>
	additive	<i>+ -</i>
Shift	shift	<i><< >> >>></i>
Relational	comparison	<i>< > <= >= instanceof</i>
	equality	<i>== !=</i>
Bitwise	bitwise AND	<i>&</i>
	bitwise exclusive OR	<i>^</i>
	bitwise inclusive OR	<i> </i>
Logical	logical AND	<i>&&</i>
	logical OR	<i> </i>
Ternary	ternary	<i>? :</i>
Assignment	assignment	<i>= += -= *= /= %= &= ^= = <<= >>= >>>=</i>

We will use IntelliJ Idea IDE for JAVA Programming



Download <https://www.jetbrains.com/idea/download/#section=windows>



The screenshot shows the JetBrains IntelliJ IDEA download page. At the top, the browser address bar displays the URL `jetbrains.com/idea/download/#section=windows`. The page header includes the IntelliJ IDEA logo, navigation links for 'What's New', 'Features', 'Resources', and 'Buy', and a prominent blue 'Download' button. The main content area is titled 'Download IntelliJ IDEA' and features tabs for 'Windows' (selected), 'macOS', and 'Linux'. Below these tabs, two editions are presented: 'Ultimate' and 'Community'. The 'Ultimate' section describes it as 'For web and enterprise development' and offers a 'Free 30-day trial' with a blue 'Download .exe' button. The 'Community' section describes it as 'For JVM and Android development' and offers a 'Free, built on open source' version with a dark 'Download .exe' button. A blue arrow originates from the URL in the text above and points directly to the 'Download' button in the Community section. On the left side of the page, there is a large IntelliJ IDEA logo, version information (2021.2, Build: 212.4746.92, 27 July 2021), and links to 'Release notes', 'System requirements', 'Installation Instructions', 'Other versions', and 'Third-party software'. At the bottom, a comparison table lists features for both editions, and a footer contains the text 'GETS - GETTING EDUCATION WITH TECHNOLOGICAL SYSTEM'.

IntelliJ IDEA

What's New Features Resources Buy [Download](#)

Download IntelliJ IDEA

[Windows](#) [macOS](#) [Linux](#)

Ultimate

For web and enterprise development

[Download](#) .exe ▼

Free 30-day trial

Community

For JVM and Android development

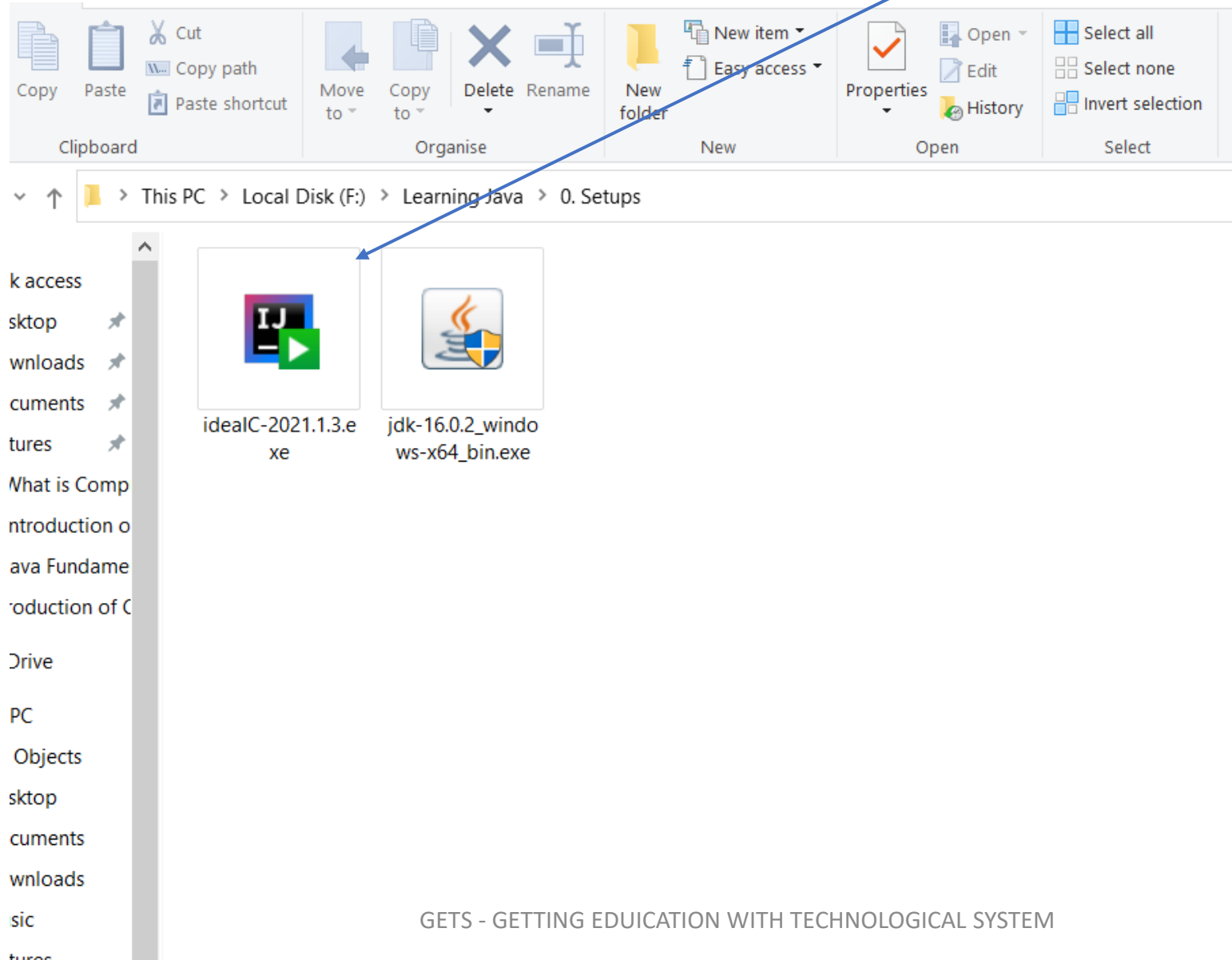
[Download](#) .exe ▼

Free, built on open source

	IntelliJ IDEA Ultimate	IntelliJ IDEA Community Edition ⓘ
Java, Kotlin, Groovy, Scala	✓	✓
Android ⓘ	✓	✓

GETS - GETTING EDUCATION WITH TECHNOLOGICAL SYSTEM

But we have already have at 0.Setups folder

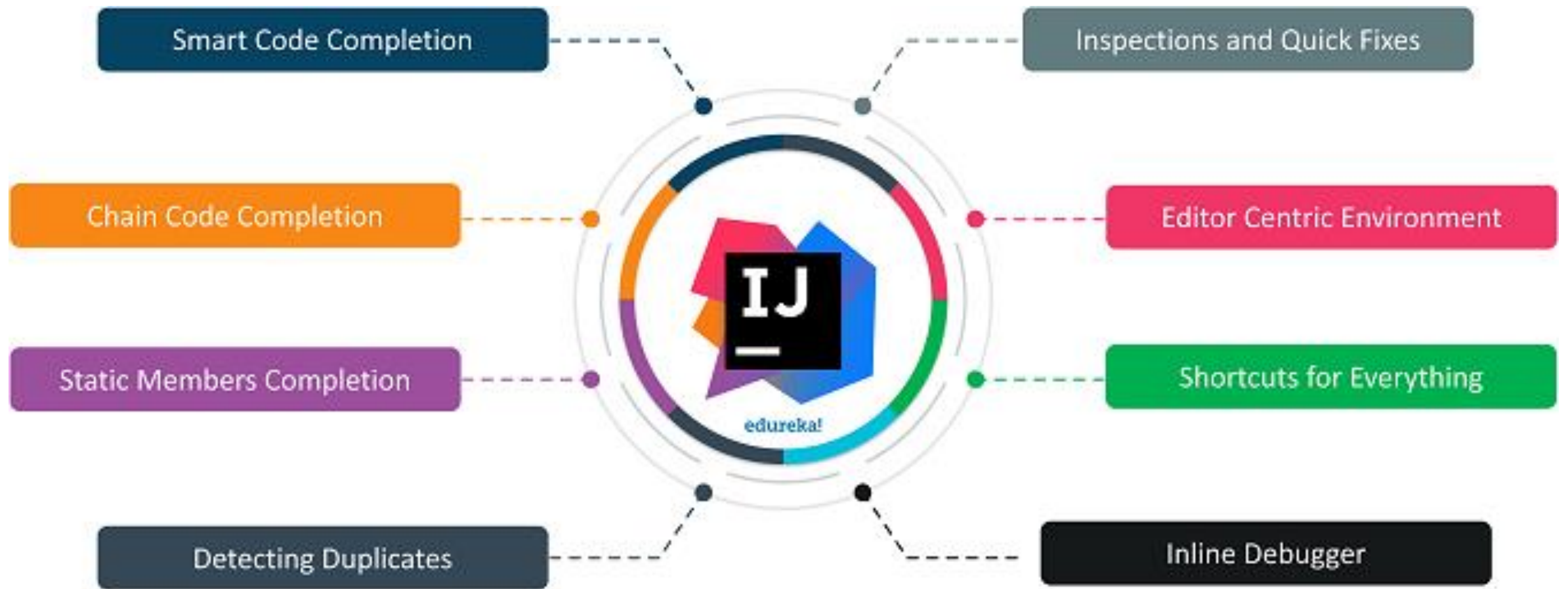


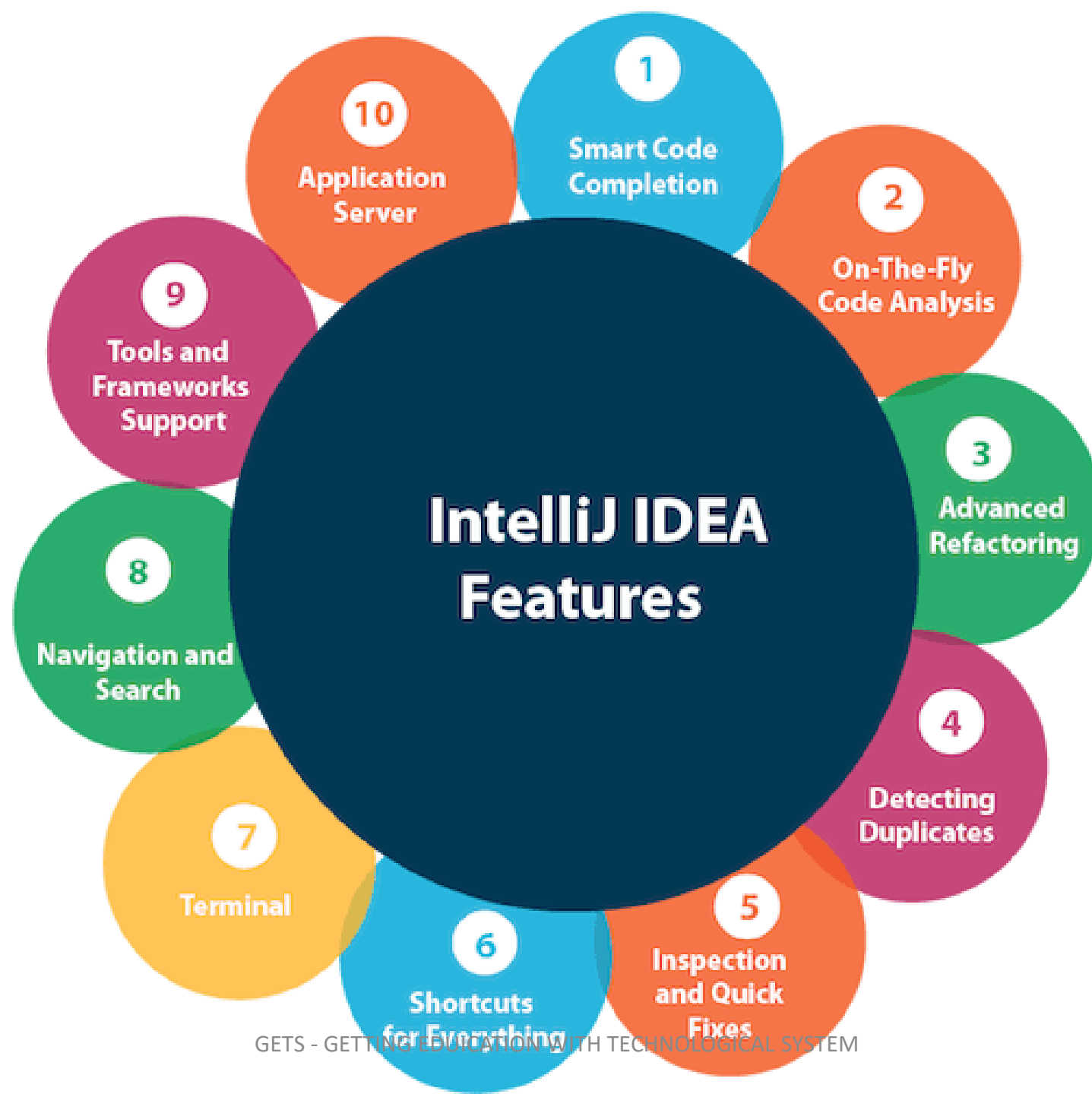
What is IDE?

An Integrated Development Environment (IDE) is a software application that provides comprehensive facilities to computer programmers for software development.

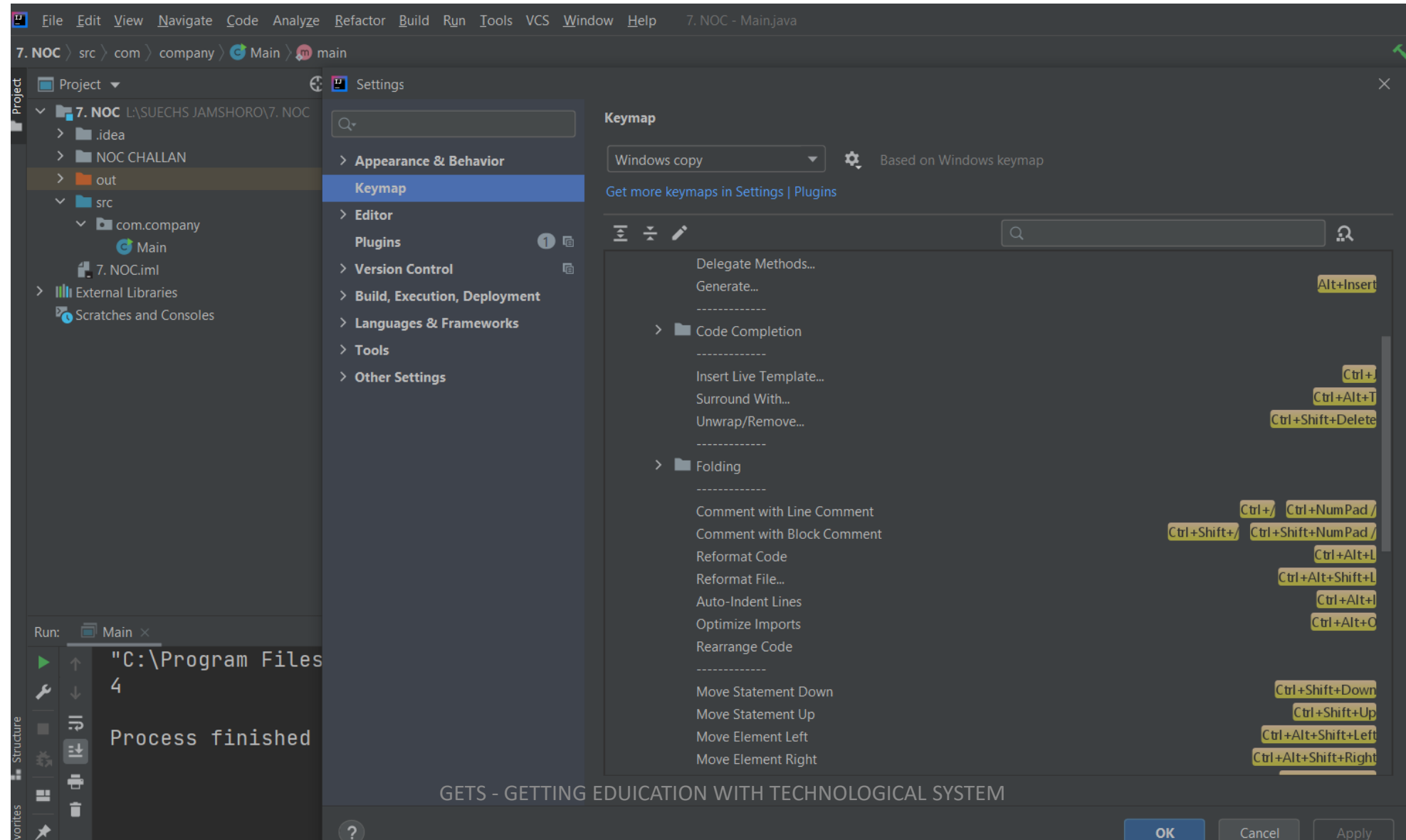
Why IDE?

To reduce the coding time.





Go to File > setting > keymap > main menu change key setting as you wish



Some shortcuts for IntelliJ Idea for Code

- Type psvm hit enter for public static void main method
- Sout for System.out.println method

1. Java Unary Operators | ++, --, ~, !

```
Main.java x
1  package com.company;
2
3  ▶ public class Main {
4  ▶  ▶ public static void main(String[] args) {
5      //unary operators ++, --
6      int a = 10;
7
8      System.out.println(a++); //10
9      System.out.println(++a); //12
10     System.out.println(a--); //12
11     System.out.println(--a); //10
12 }
13 }
14
15
16
```

1. Unary Operators | ~, !

```
Main.java x
2
3 ▶ public class Main {
4 ▶     public static void main(String[] args) {
5
6         //unary operators ~, !
7         int a = 10;
8         int b = -20;
9         boolean c = true;
10        boolean d = false;
11
12        System.out.println(~a); //-11 --- minus of total positive value which starts from 0
13        System.out.println(~b); //19 --- positive of total minus, positive start from 0
14        System.out.println(!c); //false --- opposite of boolean value
15        System.out.println(!d); //true --- opposite of boolean value
16    }
17 }
18
```

2. Java Arithmetic's | $*$, $/$, $\%$, $-$, $+$

```
Main.java x
1 package com.company;
2
3 public class Main {
4     public static void main(String[] args) {
5
6         //Java Arithmetics *, /, %, -, +
7         int a = 10;
8         int b = 5;
9
10        System.out.println(a+b); //15
11        System.out.println(a-b); //5
12        System.out.println(a/b); //2
13        System.out.println(a%b); //0
14        System.out.println(a*b); //50
15    }
16 }
17
18
```

```
Main.java x
1 package com.company;
2
3 public class Main {
4     public static void main(String[] args) {
5
6         //Java Arithmetics *, /, %, -, +
7
8         System.out.println(10*10/5+3-1*4/2); //21
9     }
10 }
11
12
13
14
```


3. Java Left Shift Operator | <<

```
Main.java x
1  package com.company;
2
3  ▶ public class Main {
4  ▶  public static void main(String[] args) {
5
6      //Java Left shift operators <<
7
8      System.out.println(10<<2); //10*2^2=10*4=40
9      System.out.println(10<<3); //10*2^3=10*8=80
10     System.out.println(20<<2); //20*2^2=20*4=80
11     System.out.println(15<<4); //15*2^4=15*16=240
12 }
13 }
14
15
```

3. Java Right Shift | >>

```
Main.java x
1  package com.company;
2
3  ▶ public class Main {
4  ▶     public static void main(String[] args) {
5
6      // right shift operator
7
8      System.out.println(10>>2); //10/2^2=10/4=2
9      System.out.println(20>>2); //20/2^2=20/4=5
10     System.out.println(20>>3); //20/2^3=20/8=2
11
12     }
13 }
14
15
```

4. Java AND Operator logical && and Bitwise &

```
Main.java x
2
3 ▶ public class Main {
4 ▶     public static void main(String[] args) {
5
6         //Java AND Operator logical && and Bitwise &
7         //logical && if 1st condition is false it will not check the 2nd condition
8         //Bitwise & if 1st condition is false it will also check the 2nd condition
9
10        int a = 10;
11        int b = 5;
12        int c = 20;
13
14        System.out.println(a<b&&a<c); //false && true = false
15        System.out.println(a<b&a<c); //false & true = false
16
17    }
18 }
```

4. Java AND Operator logical && and Bitwise & with increment method

```
Main.java x
4  ▶  public static void main(String[] args) {
5
6      //Java AND Operator logical && and Bitwise &
7      //logical && if 1st condition is false it will not check the 2nd condition
8      //Bitwise & if 1st condition is false it will also check the 2nd condition
9
10     int a = 10;
11     int b = 5;
12     int c = 20;
13
14     System.out.println(a<b&&a++<c); //false && true = false
15     System.out.println(a); //10 because 2nd condition is not checked
16     System.out.println(a<b&a++<c); //false & true = false
17     System.out.println(a); //11 because 2nd condition is also checked
18
19 }
20 }
```

4. Java OR Operator logical || and Bitwise |

```
Main.java x
1  package com.company;
2
3  ▶ public class Main {
4  ▶     public static void main(String[] args) {
5      int a = 10;
6      int b = 5;
7      int c = 15;
8
9      System.out.println(a>b||a>c); //true || false = true
10     System.out.println(a<b|a<c); //false | true = true
11
12
13     }
14 }
15
16
```

4. Java OR Operator logical || and Bitwise | with increment method

```
Main.java x
1  package com.company;
2
3  ▶ public class Main {
4  ▶     public static void main(String[] args) {
5      int a = 10;
6      int b = 5;
7      int c = 15;
8
9      System.out.println(a>b||a++>c); //true || false = true
10     System.out.println(a); //10 because 2nd condition not checked
11     System.out.println(a<b|a++<c); //false | true = true
12     System.out.println(a); //11 because 2nd condition is checked
13
14
15     }
16 }
17
```

4. Java Bitwise exclusive XOR ^

```
Main.java x
1  package com.company;
2
3  ▶ public class Main {
4  ▶     public static void main(String[] args) {
5         int a = 10;
6         int b = 5;
7         int c = 15;
8
9         System.out.println(a < b ^ a > c); //false ^ false = false
10        System.out.println(a > b ^ a > c); //true ^ false = true
11        System.out.println(a < b ^ a < c); //false ^ True = true
12        System.out.println(a > b ^ a < c); //true ^ True = false
13
14
15
16    }
17 }
```

Logical AND Operator (& and &&)

Operand1	Operand2	Returned Value
False	False	False
False	True	False
True	False	False
True	True	True

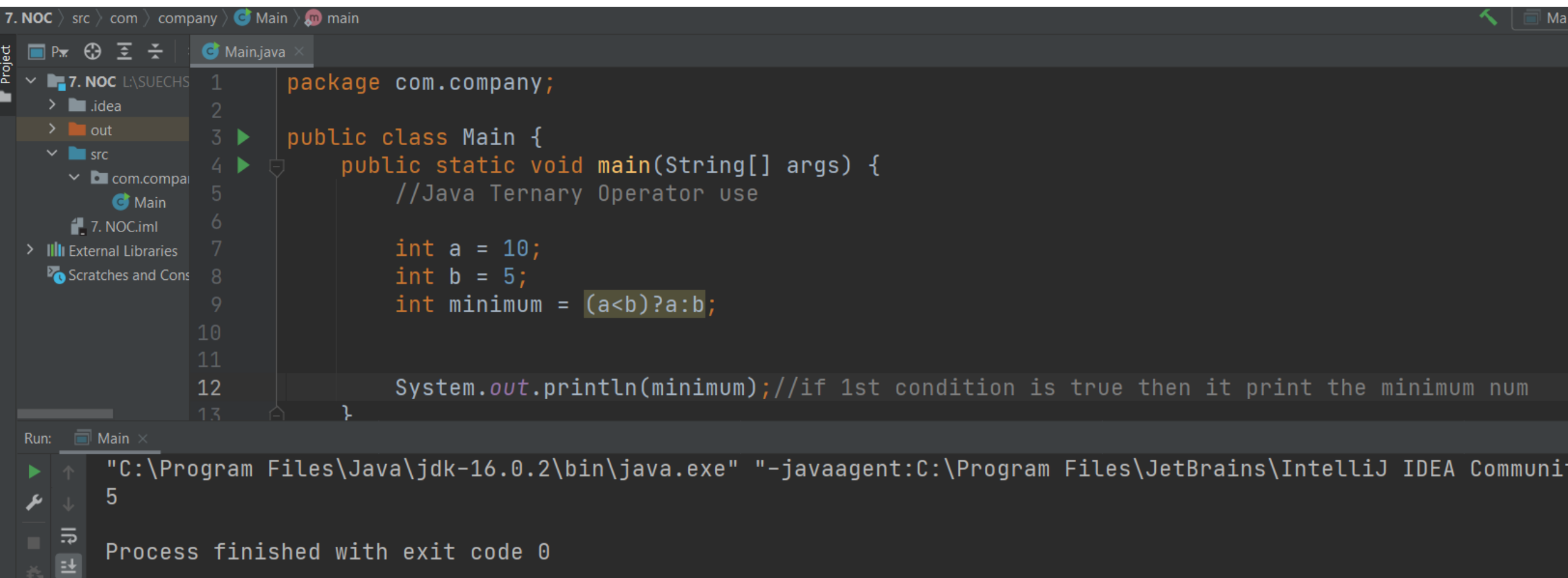
Logical OR Operator (| and ||)

Operand1	Operand2	Returned Value
False	False	False
False	True	True
True	False	True
True	True	True

Logical XOR Operator (^)

Operand1	Operand2	Returned Value
False	False	False
False	True	True
True	False	True
True	True	False

4. Java Ternary operators | ? , :



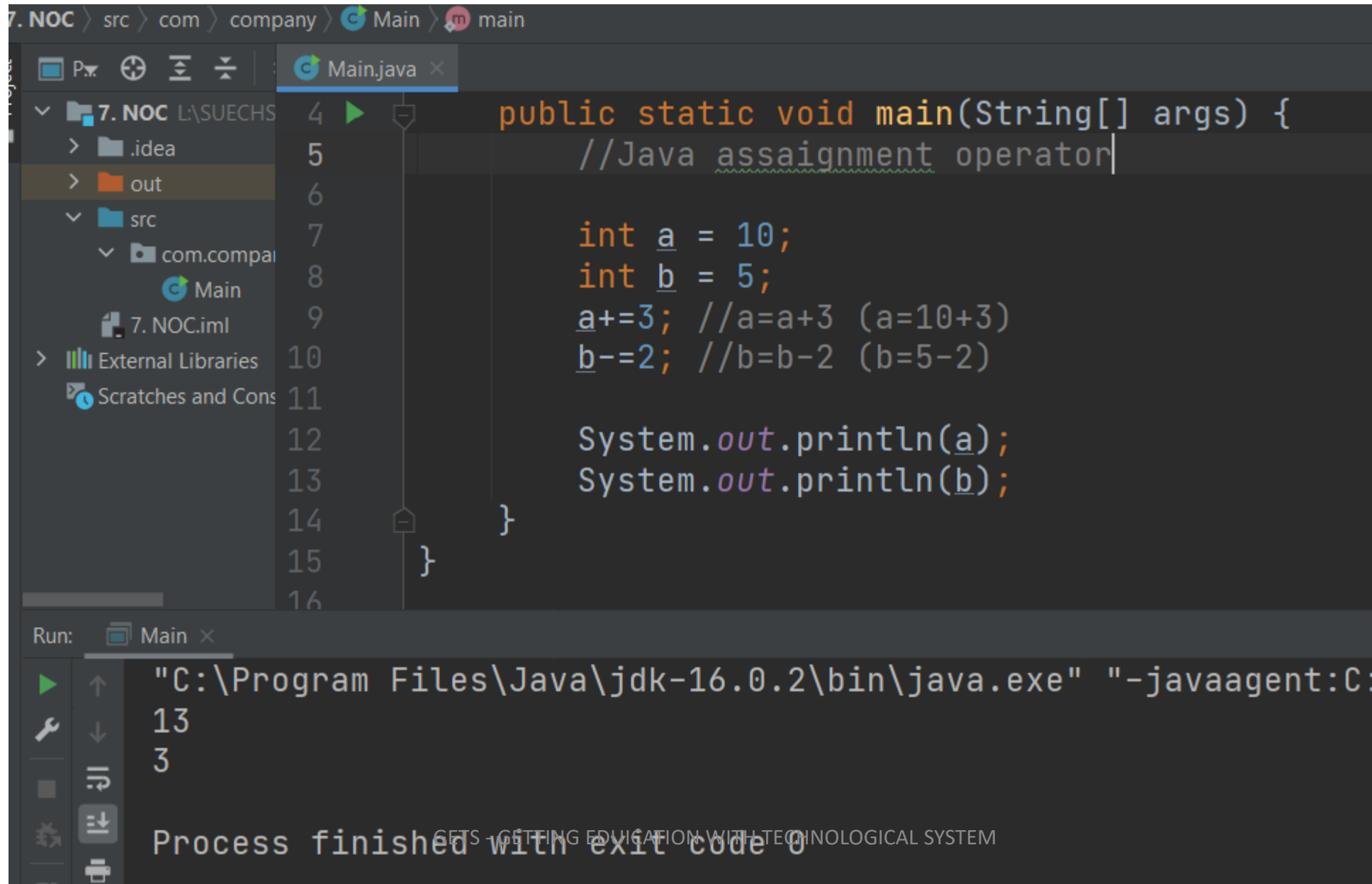
The screenshot displays the IntelliJ IDEA IDE. The left sidebar shows the project structure for '7. NOC', with the 'src' directory expanded to show 'com.company' and 'Main.java'. The main editor window shows the following Java code:

```
1 package com.company;
2
3 public class Main {
4     public static void main(String[] args) {
5         //Java Ternary Operator use
6
7         int a = 10;
8         int b = 5;
9         int minimum = (a<b)?a:b;
10
11
12         System.out.println(minimum); //if 1st condition is true then it print the minimum num
13     }
```

Below the code editor, the 'Run' tab is active, showing the execution command and output:

```
Run: Main x
"C:\Program Files\Java\jdk-16.0.2\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Communi
5
Process finished with exit code 0
```

4. Java Assignment operators | `=, +=, -=, *=, /=, %=, &=, ^=, |=, <<=, >>=, >>>=`



The screenshot displays an IDE window with a project named '7. NOC'. The file explorer on the left shows the project structure, including 'src' and 'out' folders. The main editor shows the file 'Main.java' with the following code:

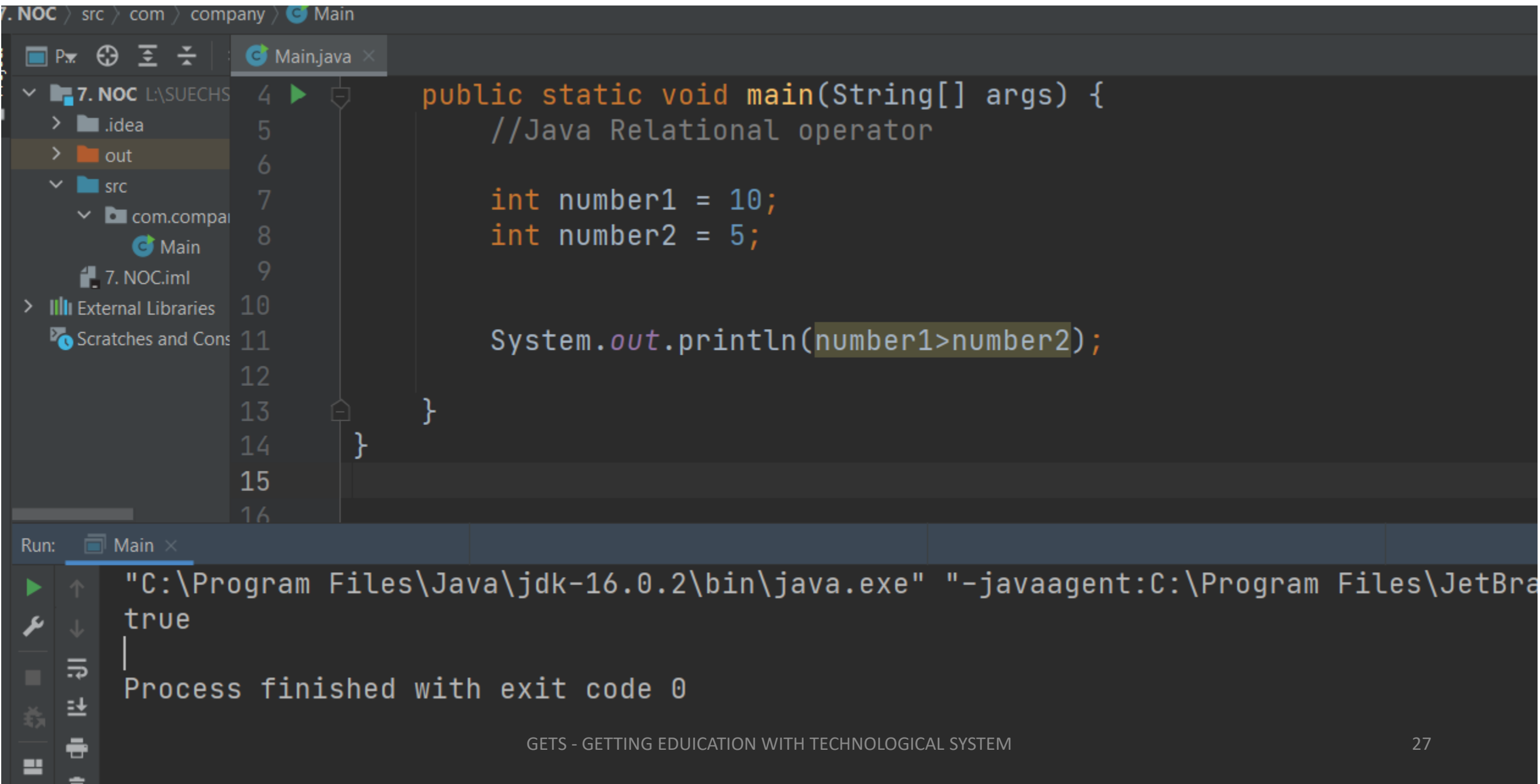
```
4 public static void main(String[] args) {  
5     //Java assignment operator  
6  
7     int a = 10;  
8     int b = 5;  
9     a+=3; //a=a+3 (a=10+3)  
10    b-=2; //b=b-2 (b=5-2)  
11  
12    System.out.println(a);  
13    System.out.println(b);  
14 }  
15  
16 }
```

Below the editor, the 'Run' tab shows the execution command and output:

```
Run: Main ×  
"C:\Program Files\Java\jdk-16.0.2\bin\java.exe" "-javaagent:C:  
13  
3  
Process finished with exit code 0
```

The output shows the values of 'a' and 'b' after the assignment operations: '13' and '3'.

7. Java Relational operators | `<, >, <=, >=, instanceof`



The screenshot displays an IDE with a project named '7. NOC'. The file explorer on the left shows the project structure, including 'src' and 'out' folders. The main editor window shows the code for 'Main.java'. The code defines a 'main' method that initializes two integers, 'number1' (10) and 'number2' (5), and prints the result of the comparison 'number1 > number2'. The output window at the bottom shows the command used to run the program, the output 'true', and the message 'Process finished with exit code 0'.

```
7. NOC > src > com > company > Main
Main.java x
7. NOC L:\SUECHS
> .idea
> out
src
  com.compai
    Main
  7. NOC.iml
External Libraries
Scratches and Cons

4 public static void main(String[] args) {
5     //Java Relational operator
6
7     int number1 = 10;
8     int number2 = 5;
9
10    System.out.println(number1>number2);
11
12
13 }
14 }
15
16

Run: Main x
"C:\Program Files\Java\jdk-16.0.2\bin\java.exe" "-javaagent:C:\Program Files\JetBra
true
Process finished with exit code 0
```

7. NOC > src > com > company > Main > main

Project
7. NOC L:\SUECHS
 > .idea
 > out
 > src
 > com.compai
 Main
 7. NOC.iml
External Libraries
Scratches and Cons

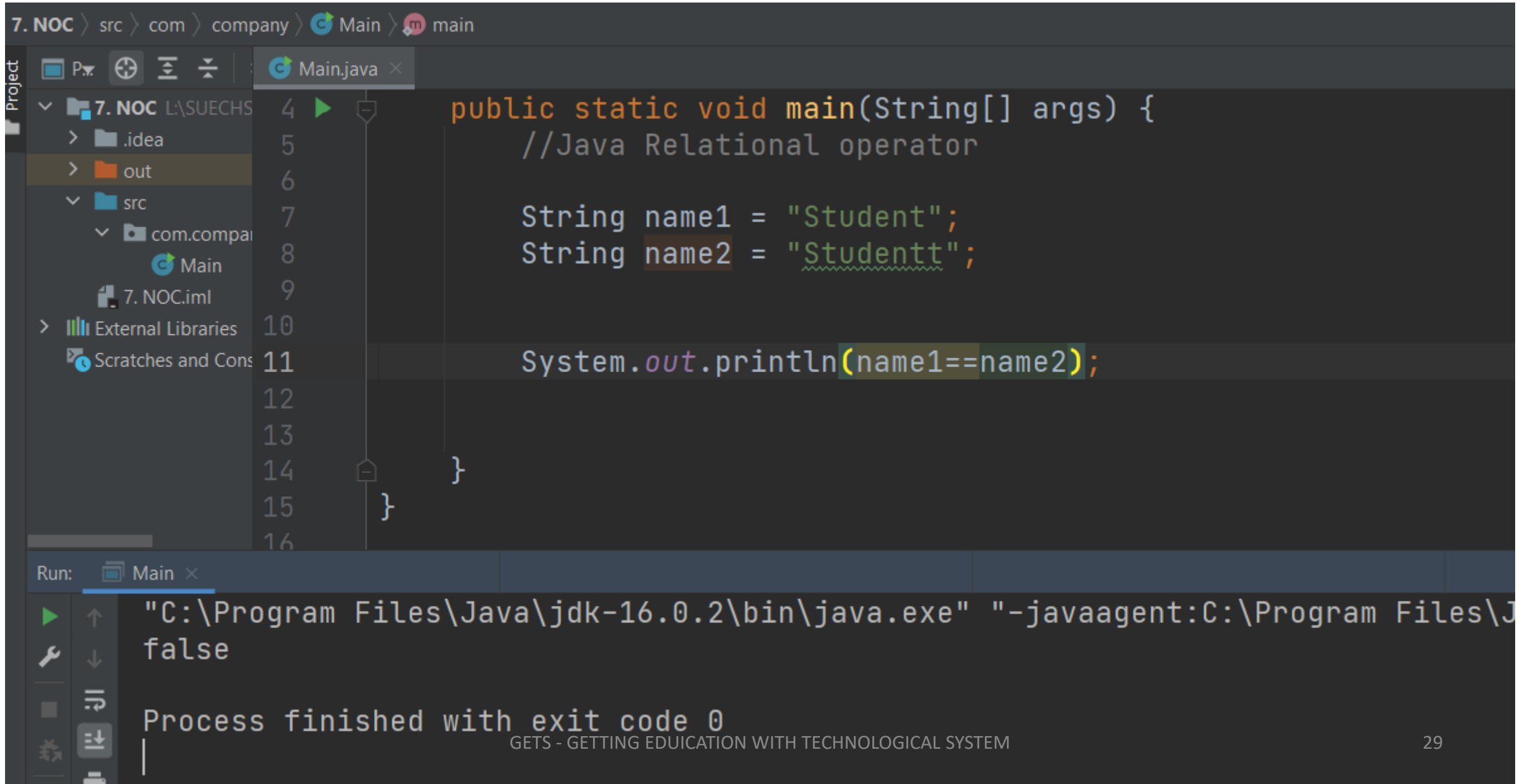
```
4 public static void main(String[] args) {  
5     //Java Relational operator  
6  
7     int number1 = 10;  
8     int number2 = 5;  
9  
10  
11     System.out.println(number1>number2);  
12     System.out.println(number1<number2);  
13     System.out.println(number1>=number2);  
14     System.out.println(number1<=number2);  
15  
16 }
```

Run: Main

```
"C:\Program Files\Java\jdk-16.0.2\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\Intelli  
true  
false  
true  
false  
  
Process finished with exit code 0  
|
```

7. Java Relational operators

String compression ==



The screenshot shows an IDE window with a project named '7. NOC'. The file explorer on the left shows the project structure: '7. NOC' (containing '.idea', 'out', 'src', and '7. NOC.iml') and 'External Libraries'. The 'Main.java' file is open in the editor. The code is as follows:

```
4 public static void main(String[] args) {  
5     //Java Relational operator  
6  
7     String name1 = "Student";  
8     String name2 = "Studentt";  
9  
10  
11     System.out.println(name1==name2);  
12  
13  
14 }  
15 }  
16
```

The output window at the bottom shows the command executed: `"C:\Program Files\Java\jdk-16.0.2\bin\java.exe" "-javaagent:C:\Program Files\J` and the output: `false`. Below the output, it says 'Process finished with exit code 0'.