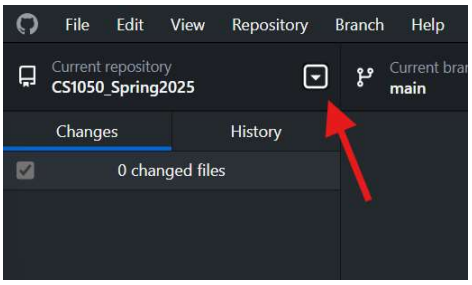
Computer Science 1 Documentation

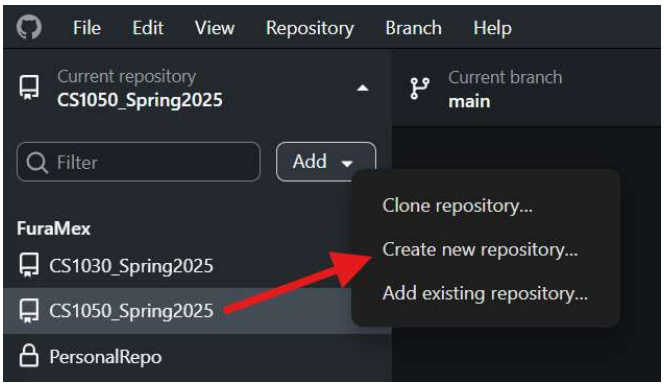
GitHub

Creating a new repo

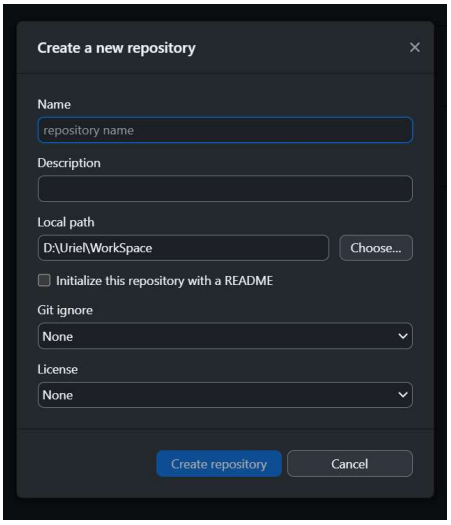
Click the dropdown button at the top left screen



Click the "Add" button and press "create new repository"

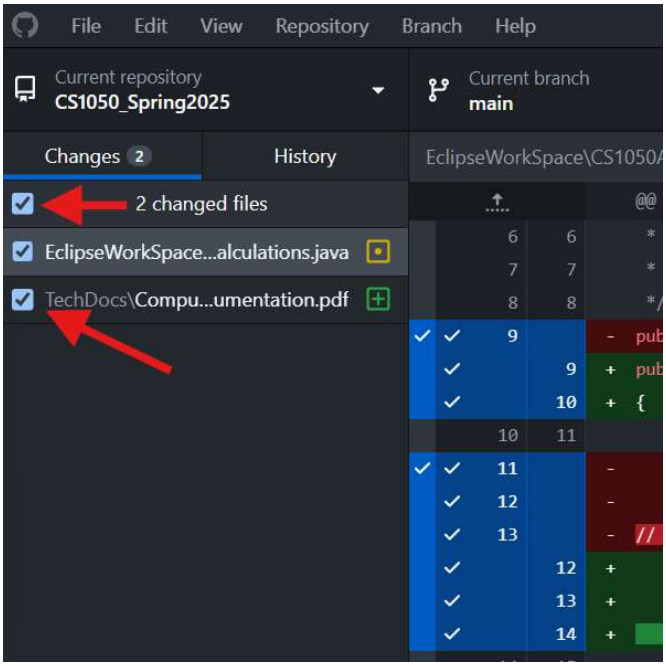


Choose a name for your repo, choose the local path, and select the git ignore and license

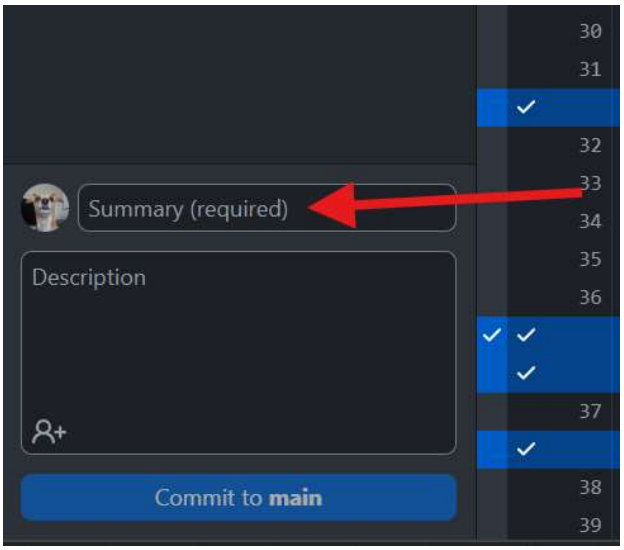


Committing and pushing a version

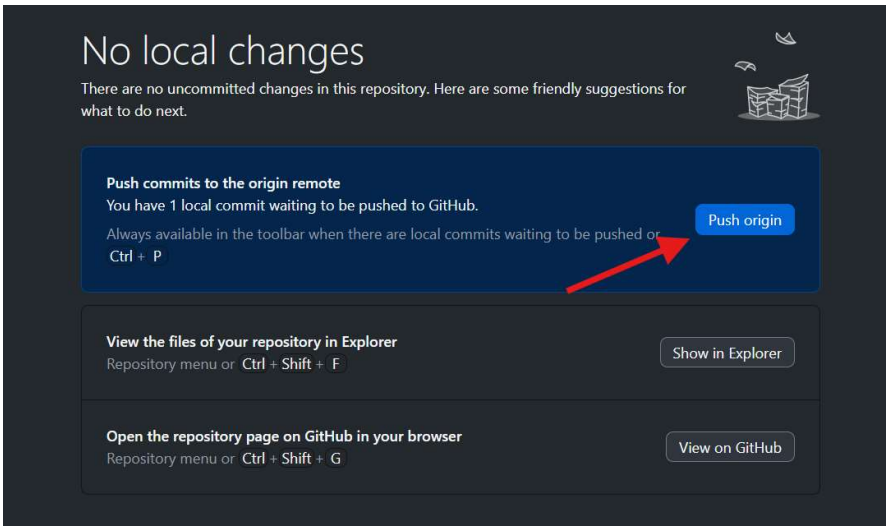
When a change is made to your repo, select which changes you wish to commit on the left side



Add a descriptive summary to the "summary" field, then click "commit to main"



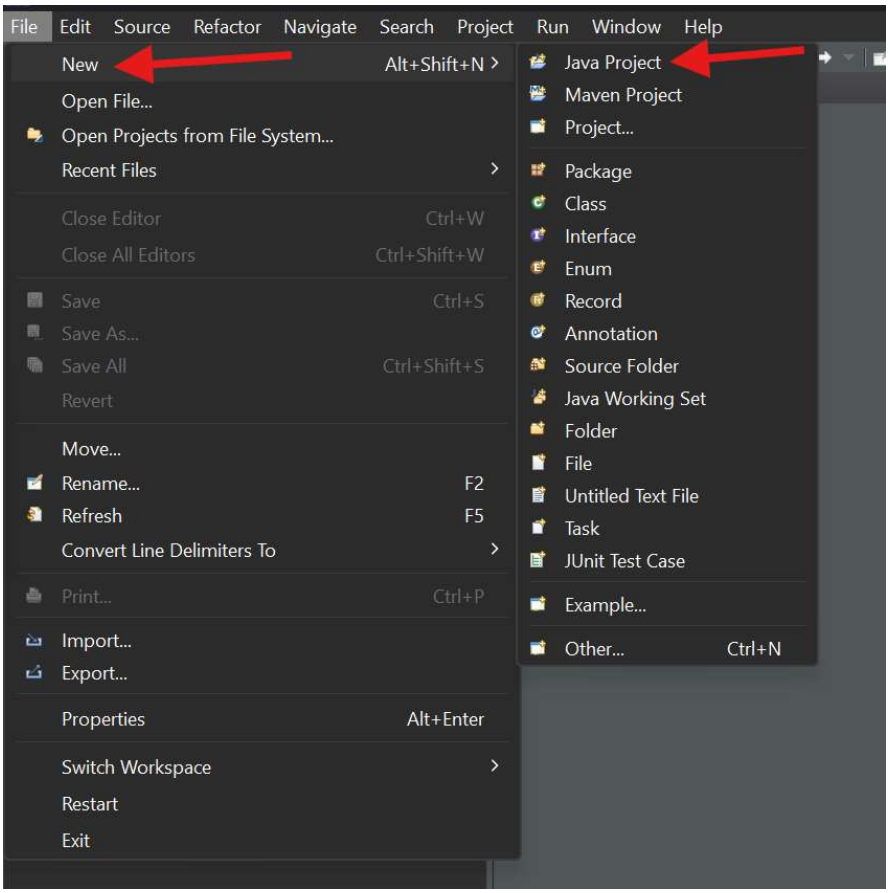
A local version was created but that version is only on your computer, not the cloud. To push it to the cloud, select "push origin"



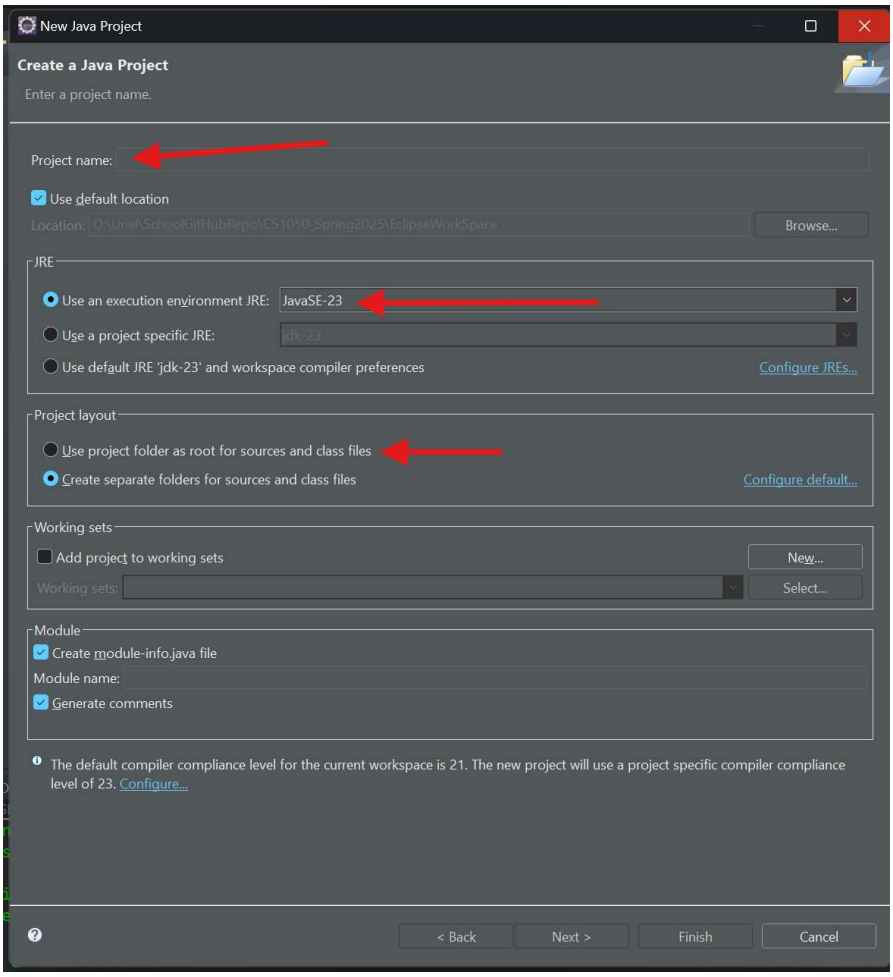
Eclipse

Creating a new project

Click "file" navigate to "new" then click "java project"

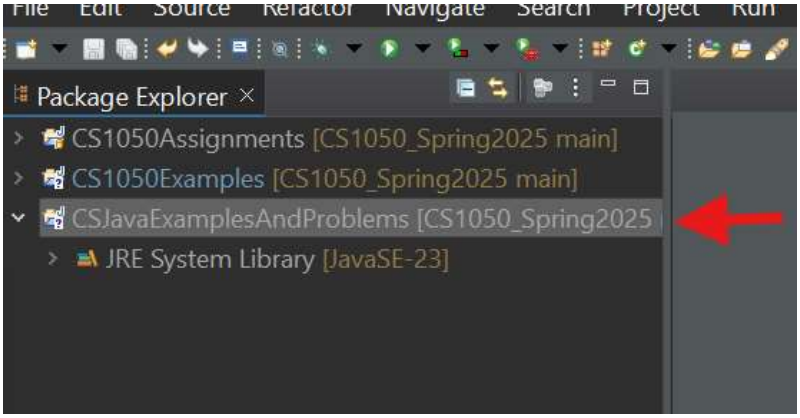


Give the project a new name, make sure the JRE is the newest version, select "Use project folder as root for sources and class files" in the options. Once done click "finish"

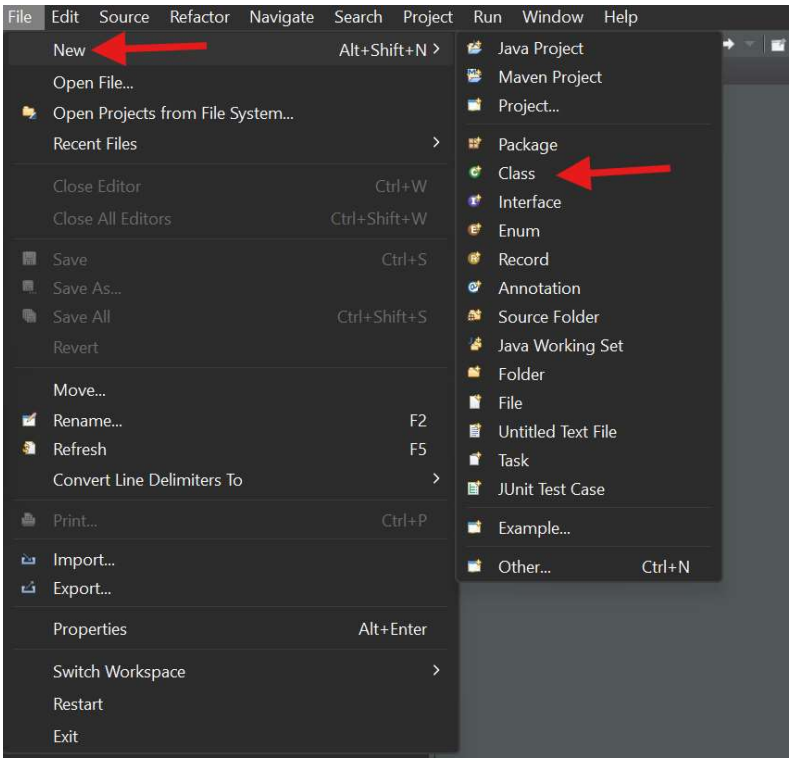


Creating a new class

Select the project you wish to add the class to

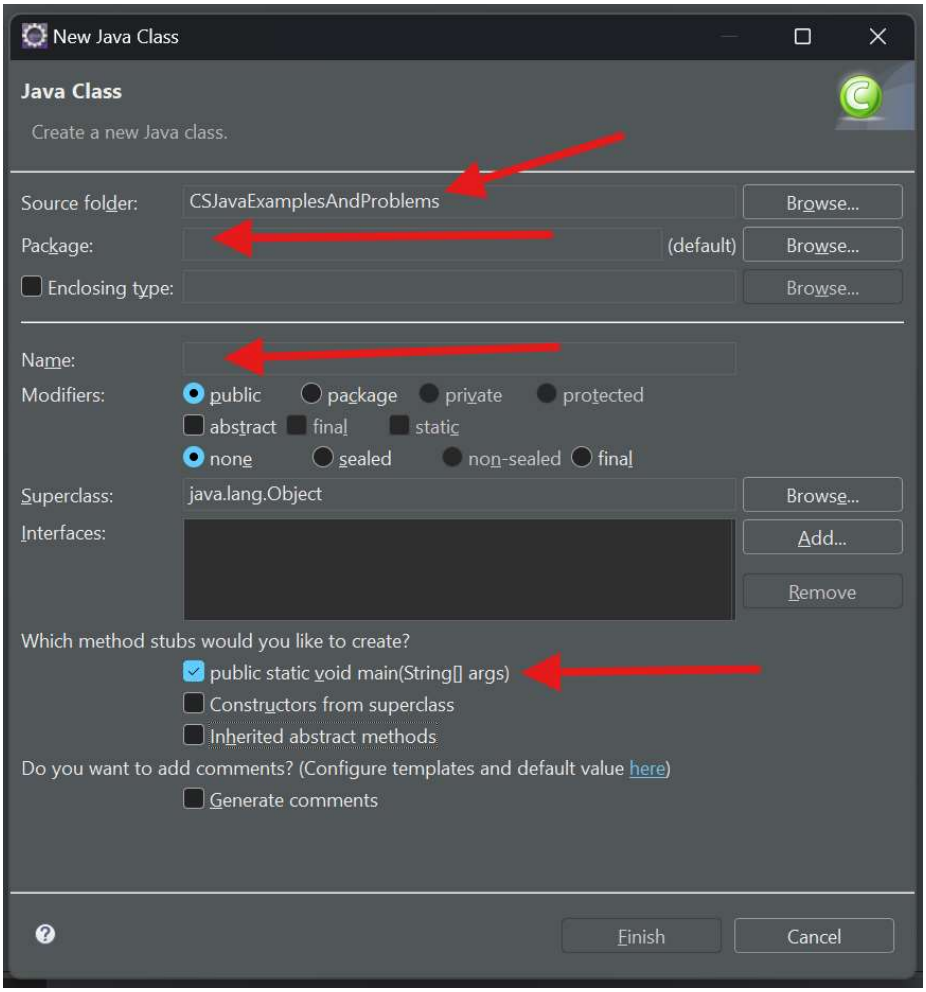


Click "file" navigate to "new", then click "class"



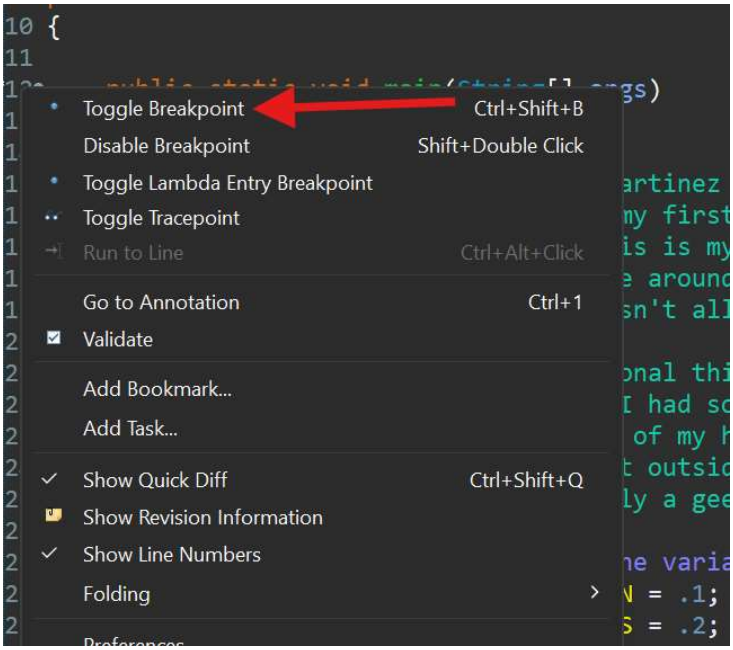
Do the following:

- Source folder should match the folder you clicked on
- The "package" field should be empty
- Give the class a name, remember it needs to start with a capital
- Public static void should be selected

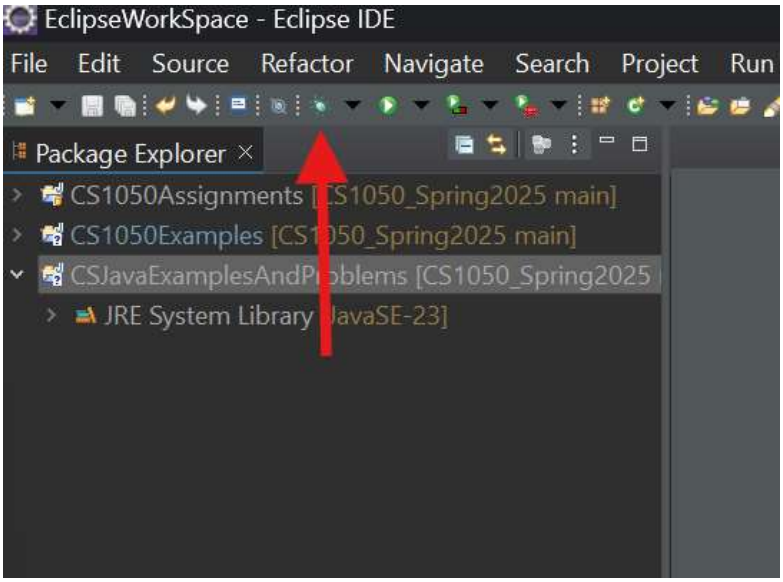


Using the debugger

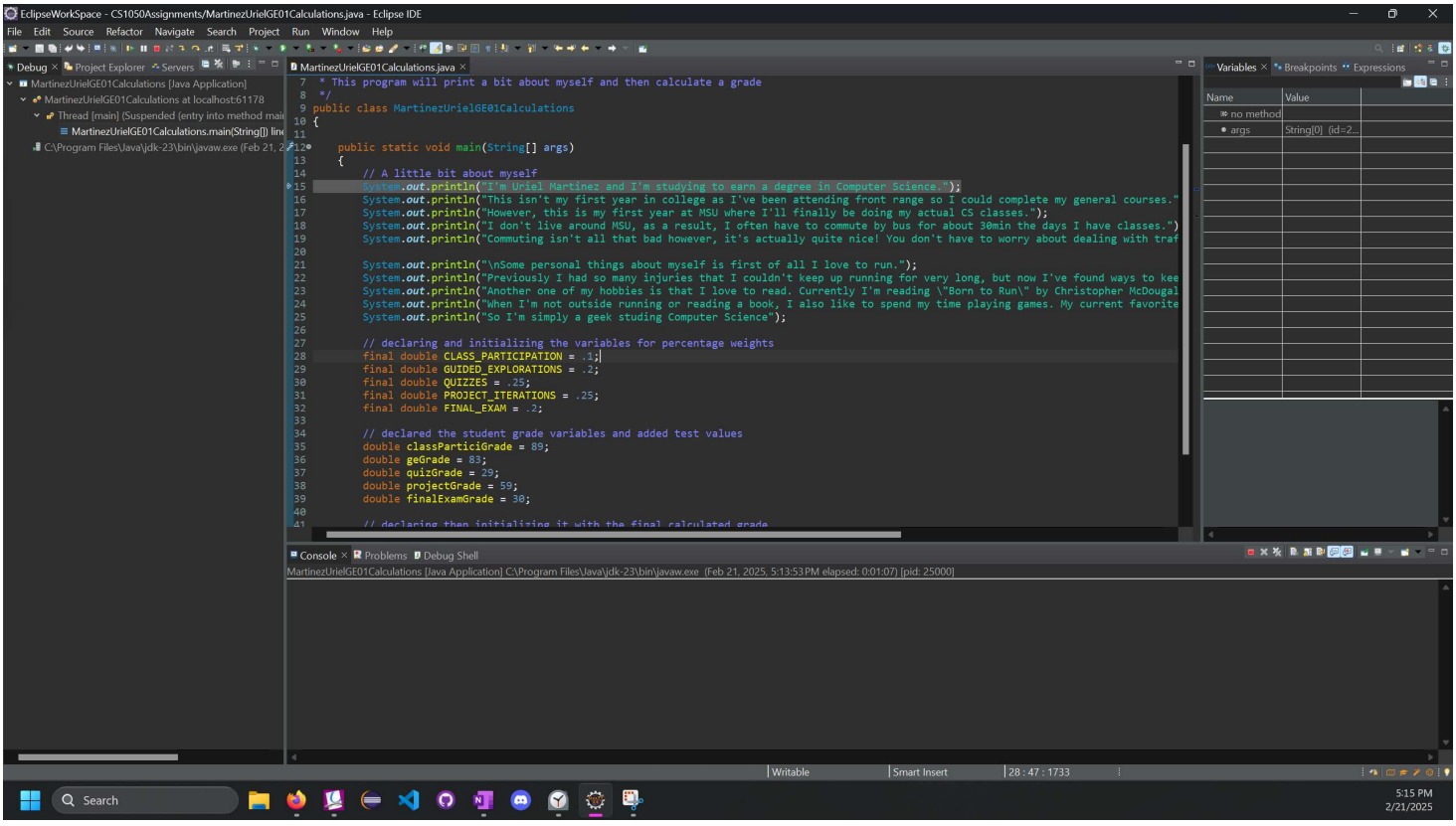
To start, right click the line you wish to debug and click "toggle breakpoint". This will stop the program at the desired line



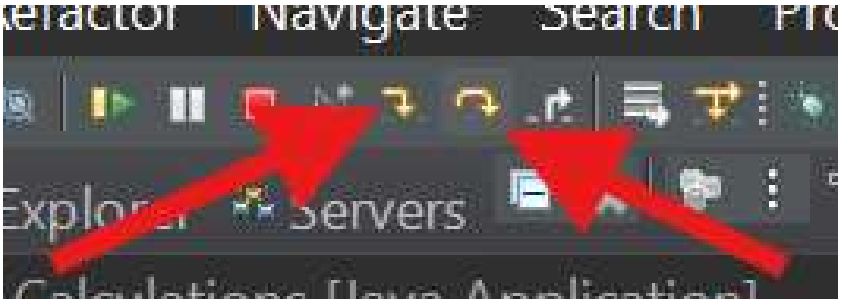
Click the icon that looks like a green bug



The debugger will begin and you'll see a screen that looks like this. The most important part of this screen is the right side window called "variables". This will tell you which values are being declared/initialized and other things that are happening



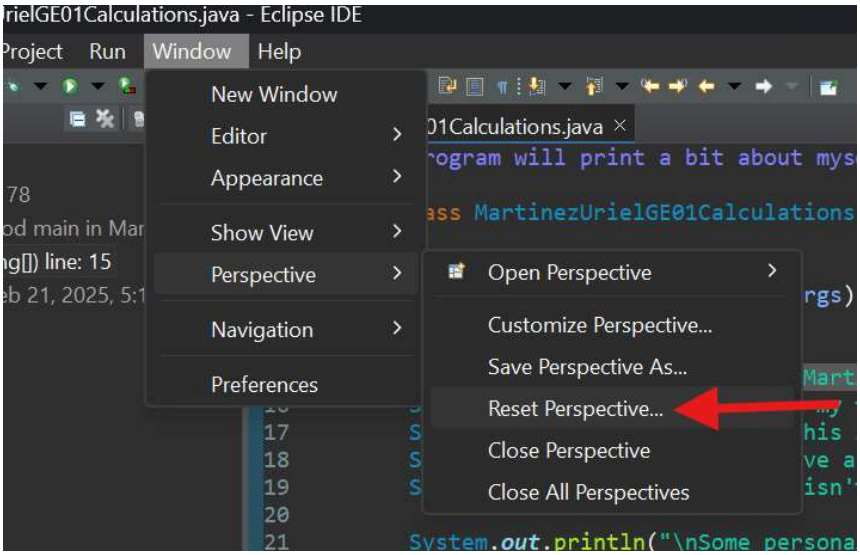
These 2 buttons will allow you to step over or step into code. It is not recommended to step into code such as "print", you probably don't need to see what's happening in their



Resetting your perspective

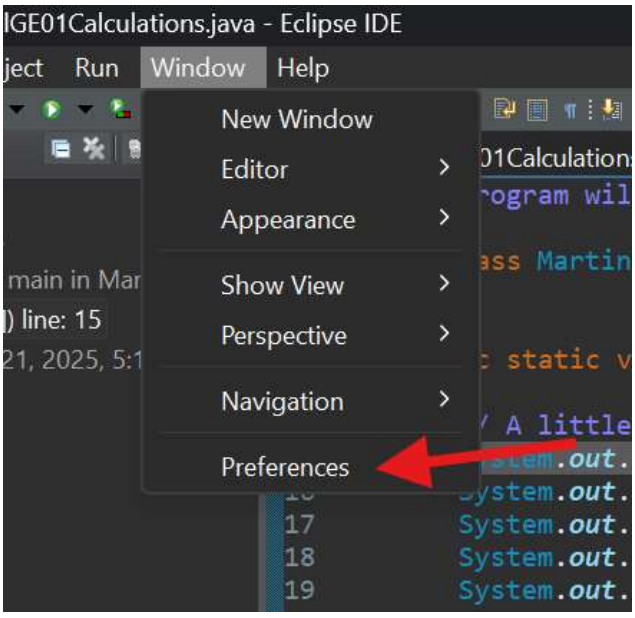
Resetting your perspective means to return to how it originally was setup in case you've accidentally gotten rid of a window

To do this, go to Window > Perspective > Reset Perspective

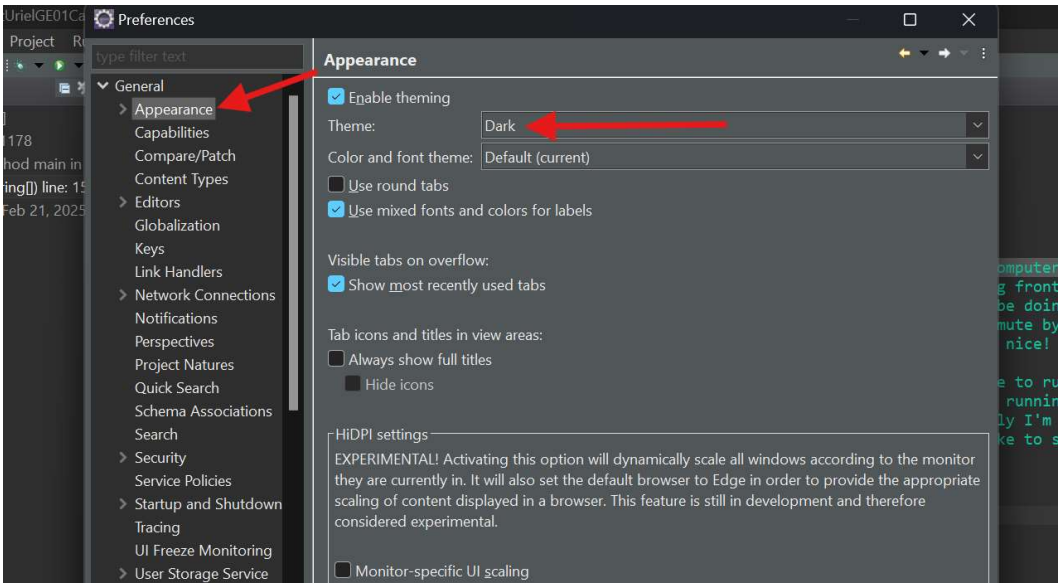


Changing the theme

To change the theme between light or dark theme or another one, click window > Preferences

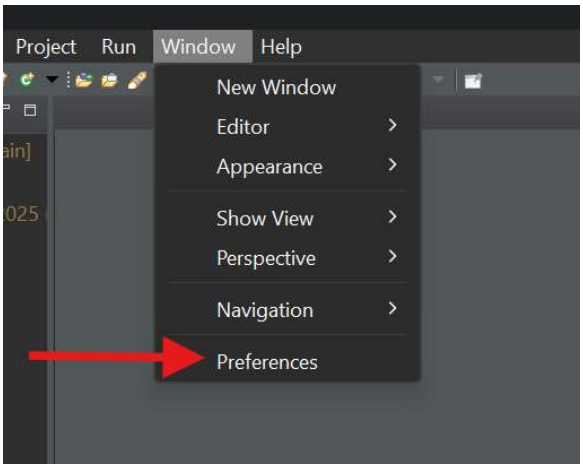


Click on General > Appearance, then select the theme from the drop down menu

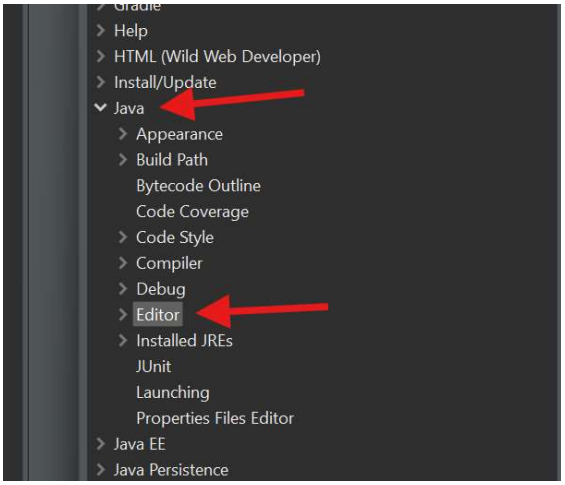


Changing the color of the comments

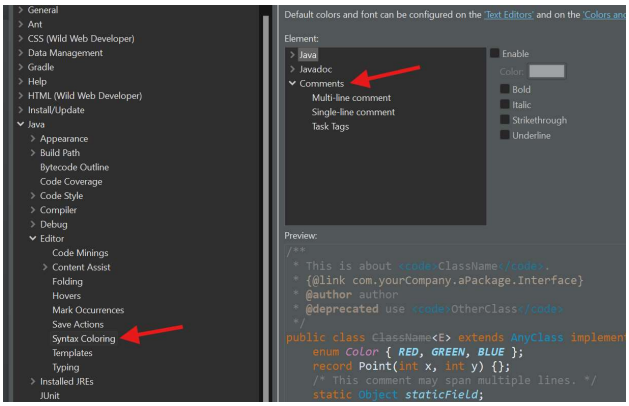
Go to "Window" and select "Preferences"



Click on "Java" then navigate to "Editor"



Click on "Syntax Coloring" then Open the "comments" dropdown



Then simply change comments to what you wish. Or change them back to default if the color becomes a problem

Module 1

Operating System

Software that runs on your computer to manage the computer's basic functions such as running programs or controlling devices

Some examples of some common operating systems

Software

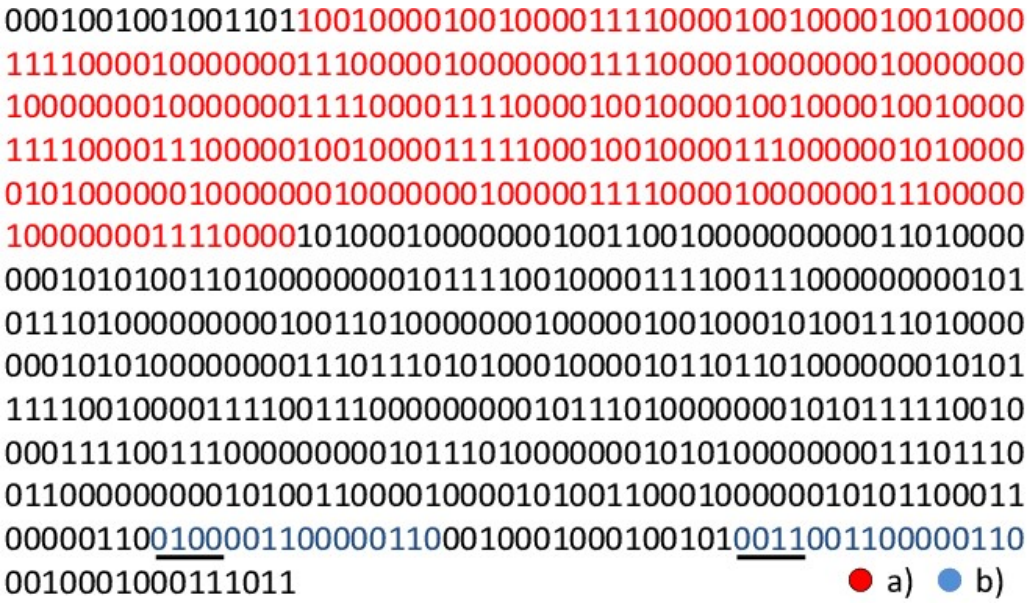
A set of instructions that enable the computer to solve a problem or perform a task

Source Program or Code

A program written in a high-level programming language

Machine Language

Consists of strings of 0's and 1's that instruct the computer to perform simple operations



Assembly languages

Uses English-like abbreviations to represent elementary operations

```
    .globl  "_*add_forty_two<Int32>:Int32"
    .align  4, 0x90
"_*add_forty_two<Int32>:Int32":
    .cfi_startproc
    pushq %rbp
Ltmp1992:
    .cfi_def_cfa_offset 16
Ltmp1993:
    .cfi_offset %rbp, -16
    movq  %rsp, %rbp
Ltmp1994:
    .cfi_def_cfa_register %rbp
    addl  $42, %edi
    movl  %edi, %eax
    popq  %rbp
    retq
    .cfi_endproc
```

High-Level Languages

Uses single statements that are English

new.py > ...

```python
number_1 = input('Enter your first number: ')
operator = input('operator: ')
number_2 = input('Enter your second number: ')

if (operator == '+'):
    print("Rusult:", number_1+number_2)
elif(operator == '-'):
    print("Rusult:", number_1-number_2)
elif(operator == '*'):
    print("Rusult:", number_1*number_2)
elif(operator == '/'):
    print("Rusult:", number_1/number_2)
else:
    print('Try again')
```

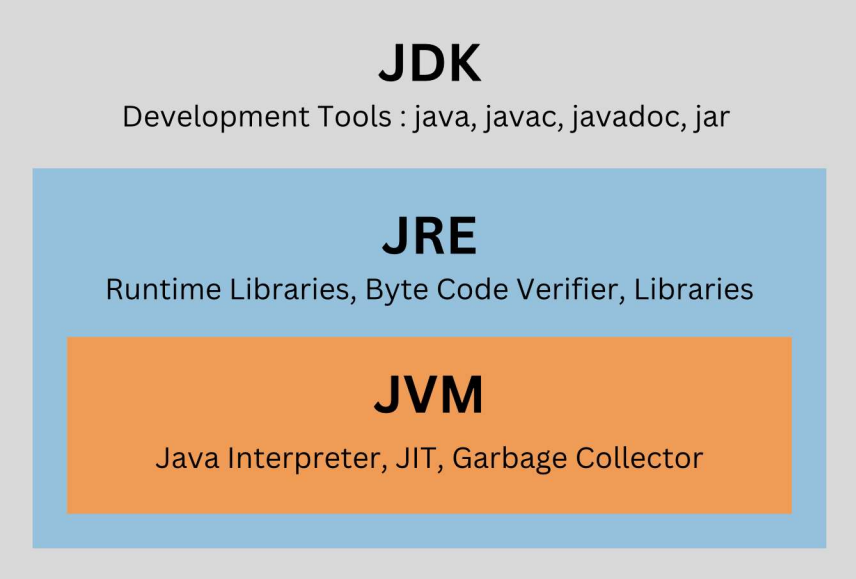
Java API

Contains predefined classes and interfaces for developing java programs

- Also known as a library

Java Development Kit

Software for running and compiling java programs



Key Words

Words that have special meaning in the programming language

Operators

Characters that have special meaning int the programming language

Identifiers

Names in the programming language that are defined by the programmer

Module 2

Concatenation

Combining 2 or more values in a print statement or variable

```
String firstName = "John";
String lastName = "Doe";
System.out.println(firstName + " " + lastName);
```

Multi-way If
An If statement that checks one variable for different conditions

Nested If
An if statement that contains another if statement within it

Java Programming
Printing
Displays information in the developer console

```
System.out.print("value");
```

Print New Line Statement
Prints the information it was given then moves the cursor to the next line. Simply adds "ln" to the statement

```
System.out.println("value");
```

Formatting Print Statements
Allows you to combine values of different datatypes in to one print statement with formatting

```
System.out.printf("value", "args");
```

If statements
Allows the code to take different paths depending on what conditions are met

```
if (condition > 0)
{
    // code to execute if true
}
else if (condition < 0)
{
    // code to execute if true
}
else
{
    // "fallback" code is nothing is true
}
```

Switch Statements
Runs the case depending on what expression it is given

```
switch(expression)
{
  case 1:
    // code block
    break;

  case 2:
    // code block
    break;

  default:
    // code block
}
```

Using the scanner object

In order to use the scanner object, you first must import the scanner class, create a new scanner object, then call it

```
import java.util.Scanner;
```

```
Scanner myObj = new Scanner(System.in);  // Create a Scanner object
System.out.println("Enter username");
```