**Go to File → New → Apex Class**

**Name it something like same as class name and paste this code:**

1)

public class ArrayAddition {

public static List<Integer> addArrays(List<Integer> arr1, List<Integer> arr2) {

List<Integer> result = new List<Integer>();

if (arr1.size() != arr2.size()) {

System.debug('Error: Arrays are not of the same size.');

return result;

}

for (Integer i = 0; i < arr1.size(); i++) {

result.add(arr1[i] + arr2[i]);

}

return result;

}

}

**run the method from Execute Anonymous Window like this:**

List<Integer> a = new List<Integer>{10, 20, 30};

List<Integer> b = new List<Integer>{1, 2, 3};

List<Integer> result = ArrayAddition.addArrays(a, b);

System.debug(result); // Should print (11, 22, 33)

**2) matrix addition**

public class MatrixOperations {

public static List<List<Integer>> addMatrices(List<List<Integer>> a, List<List<Integer>> b) {

List<List<Integer>> result = new List<List<Integer>>();

for (Integer i = 0; i < a.size(); i++) {

List<Integer> row = new List<Integer>();

for (Integer j = 0; j < a[0].size(); j++) {

row.add(a[i][j] + b[i][j]);

}

result.add(row);

}

return result;

}

}

**run the method from Execute Anonymous Window like this:**

List<List<Integer>> a = new List<List<Integer>>{

new List<Integer>{1, 2},

new List<Integer>{3, 4}

};

List<List<Integer>> b = new List<List<Integer>>{

new List<Integer>{5, 6},

new List<Integer>{7, 8}

};

System.debug(MatrixOperations.addMatrices(a, b));

**3) Temperature converter**

public class TemperatureConverter {

public static Decimal celsiusToFahrenheit(Decimal c) {

return (c \* 9 / 5) + 32;

}

public static Decimal fahrenheitToCelsius(Decimal f) {

return (f - 32) \* 5 / 9;

}

}

**run the method from Execute Anonymous Window like this:**

Decimal fTemp = 98.6;

Decimal cTemp = 37.0;

System.debug('Fahrenheit to Celsius: ' + TemperatureConverter.fahrenheitToCelsius(fTemp));

System.debug('Celsius to Fahrenheit: ' + TemperatureConverter.celsiusToFahrenheit(cTemp));

**4) FibonacciSeries**

public class FibonacciSeries {

public static List<Integer> generateFibonacci(Integer n) {

List<Integer> fib = new List<Integer>();

if (n >= 1) fib.add(0);

if (n >= 2) fib.add(1);

for (Integer i = 2; i < n; i++) {

fib.add(fib[i - 1] + fib[i - 2]);

}

return fib;

}

}

**Run from anonymous window**

System.debug(FibonacciSeries.generateFibonacci(10)); // First 10 terms

**5)prime number checking**

public class PrimeChecker {

public static Boolean isPrime(Integer num) {

if (num <= 1) return false;

for (Integer i = 2; i <= Math.floor(Math.sqrt(Decimal.valueOf(num))); i++) {

if (Math.mod(num, i) == 0) {

return false;

}

} return true;

}

}

**Anonymous window**

System.debug('Is 29 Prime? ' + PrimeChecker.isPrime(29));

**6) Create Apex class for calculator**

public class ScientificCalculator {  
     
    public Decimal num1 { get; set; }  
    public Decimal num2 { get; set; }  
    public Decimal result { get; set; }  
    public String operation { get; set; }  
    public String errorMessage { get; set; }  
  
    public ScientificCalculator() {  
        num1 = 0;  
        num2 = 0;  
        result = 0;  
        operation = '';  
        errorMessage = '';  
    }  
  
    public void calculate() {  
        errorMessage = '';  
        result = null;  
  
        try {  
            if (operation == 'add') {  
                result = num1 + num2;  
            } else if (operation == 'subtract') {  
                result = num1 - num2;  
            } else if (operation == 'multiply') {  
                result = num1 \* num2;  
            } else if (operation == 'divide') {  
                if (num2 == 0) {  
                    errorMessage = 'Cannot divide by zero!';  
                } else {  
                    result = num1 / num2;  
                }  
            } else if (operation == 'power') {  
                result = Math.pow(Double.valueOf(num1), Double.valueOf(num2));  
            } else if (operation == 'sqrt') {  
                if (num1 < 0) {  
                    errorMessage = 'Cannot find square root of a negative number!';  
                } else {  
                    result = Math.sqrt(Double.valueOf(num1));  
                }  
            } else if (operation == 'log') {  
                if (num1 <= 0) {  
                    errorMessage = 'Logarithm undefined for zero or negative numbers!';  
                } else {  
                    result = Math.log(Double.valueOf(num1)) / Math.log(10);  
                }  
            } else if (operation == 'ln') {  
                if (num1 <= 0) {  
                    errorMessage = 'Natural log undefined for zero or negative numbers!';  
                } else {  
                    result = Math.log(Double.valueOf(num1));  
                }  
            } else if (operation == 'abs') {  
                result = Math.abs(num1);  
            } else if (operation == 'exp') {  
                result = Math.exp(Double.valueOf(num1));  
            } else if (operation == 'tenPower') {  
                result = Math.pow(10, Double.valueOf(num1));  
            } else if (operation == 'floor') {  
                result = Math.floor(Double.valueOf(num1));  
            } else if (operation == 'ceil') {  
                result = Math.ceil(Double.valueOf(num1));  
            } else if (operation == 'round') {  
                result = Math.round(Double.valueOf(num1));  
            } else if (operation == 'sin') {  
                result = Math.sin(num1 \* Math.PI / 180);  
            } else if (operation == 'cos') {  
                result = Math.cos(num1 \* Math.PI / 180);  
            } else if (operation == 'tan') {  
                result = Math.tan(num1 \* Math.PI / 180);  
            } else if (operation == 'cot') {  
                if (Math.tan(num1 \* Math.PI / 180) == 0) {  
                    errorMessage = 'Cotangent undefined!';  
                } else {  
                    result = 1 / Math.tan(num1 \* Math.PI / 180);  
                }  
            } else if (operation == 'sec') {  
                if (Math.cos(num1 \* Math.PI / 180) == 0) {  
                    errorMessage = 'Secant undefined!';  
                } else {  
                    result = 1 / Math.cos(num1 \* Math.PI / 180);  
                }  
            } else if (operation == 'csc') {  
                if (Math.sin(num1 \* Math.PI / 180) == 0) {  
                    errorMessage = 'Cosecant undefined!';  
                } else {  
                    result = 1 / Math.sin(num1 \* Math.PI / 180);  
                }  
            } else if (operation == 'factorial') {  
                if (num1 < 0 || num1 != Math.floor(num1)) {  
                    errorMessage = 'Factorial is only defined for non-negative integers!';  
                } else if (num1 > 20) {  
                    errorMessage = 'Factorial is too large to compute!';  
                } else {  
                    result = factorial(Integer.valueOf(num1));  
                }  
            } else {  
                errorMessage = 'Invalid Operation';  
            }  
        } catch (Exception e) {  
            errorMessage = 'Error: ' + e.getMessage();  
        }  
    }  
  
    public static Integer factorial(Integer n) {  
        if (n == 0 || n == 1) return 1;  
        return n \* factorial(n - 1);  
    }  
}

**Create visualforce page for UI**

<apex:page controller="ScientificCalculator">

<apex:form >

<apex:outputPanel style="max-width: 500px; margin: auto; text-align: center;">

<apex:pageBlock title="Scientific Calculator">

<apex:pageBlockSection columns="1">

<apex:outputLabel value="Enter Number 1:"/>

<apex:inputText value="{!num1}" styleClass="input-box"/>

<apex:outputLabel value="Enter Number 2 (if required):"/>

<apex:inputText value="{!num2}" styleClass="input-box"/>

<apex:outputLabel value="Select Operation:"/>

<apex:selectList value="{!operation}" size="1" styleClass="select-box">

<apex:selectOption itemValue="add" itemLabel="Addition (+)"/>

<apex:selectOption itemValue="subtract" itemLabel="Subtraction (-)"/>

<apex:selectOption itemValue="multiply" itemLabel="Multiplication (×)"/>

<apex:selectOption itemValue="divide" itemLabel="Division (÷)"/>

<apex:selectOption itemValue="power" itemLabel="Power (x^y)"/>

<apex:selectOption itemValue="sqrt" itemLabel="Square Root (√x)"/>

<apex:selectOption itemValue="log" itemLabel="Logarithm (log₁₀)"/>

<apex:selectOption itemValue="ln" itemLabel="Natural Log (ln)"/>

<apex:selectOption itemValue="abs" itemLabel="Absolute Value (|x|)"/>

<apex:selectOption itemValue="exp" itemLabel="Exponential (e^x)"/>

<apex:selectOption itemValue="tenPower" itemLabel="10^x"/>

<apex:selectOption itemValue="floor" itemLabel="Floor (⌊x⌋)"/>

<apex:selectOption itemValue="ceil" itemLabel="Ceiling (⌈x⌉)"/>

<apex:selectOption itemValue="round" itemLabel="Round"/>

<apex:selectOption itemValue="sin" itemLabel="Sine (sin x)"/>

<apex:selectOption itemValue="cos" itemLabel="Cosine (cos x)"/>

<apex:selectOption itemValue="tan" itemLabel="Tangent (tan x)"/>

<apex:selectOption itemValue="cot" itemLabel="Cotangent (cot x)"/>

<apex:selectOption itemValue="sec" itemLabel="Secant (sec x)"/>

<apex:selectOption itemValue="csc" itemLabel="Cosecant (csc x)"/>

<apex:selectOption itemValue="factorial" itemLabel="Factorial (x!)"/>

</apex:selectList>

<apex:commandButton value="Calculate" action="{!calculate}" styleClass="calc-button"/>

<apex:outputPanel rendered="{!errorMessage != ''}">

<p class="error-message">{!errorMessage}</p>

</apex:outputPanel>

<apex:outputPanel rendered="{!result != null}">

<p class="result-text">Result: <strong>{!result}</strong></p>

</apex:outputPanel>

</apex:pageBlockSection>

</apex:pageBlock>

</apex:outputPanel>

</apex:form>

</apex:page>

**You can preview a Visualforce page in Salesforce Setup → Visualforce Pages → [Preview],**