

Scripting Lab Assignment

Name : Pradipti Chettri

Section –A

Registration number –202011506

Semester –5

Create a calculator app using Angular which is capable of performing following operations:

1. Addition of two numbers
2. Subtraction of two numbers
3. Multiplication of two numbers
4. Division of two numbers
5. Factorial of a number
6. Checking if a given number is Prime or not

We have the root component app-root, its child component calculator, and calculator's child component calculator-keys.

Index.html

```
<!doctype html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <title>CalcApp</title>
  <base href="/">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="icon" type="image/x-icon" href="favicon.ico">
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.1/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-F3w7mX95PdgyTmZZMECAngseQB83DfGTowi0iMjiWaeVhAn4FJkqJByhZMI3AhiU" crossorigin="anonymous">
</head>
<body>
  <center><h1>Calculator app</h1></center>
  <app-root></app-root>
  <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.1/dist/js/bootstrap.bundle.min.js" integrity="sha384-/bQdsTh/da6pkI1MST/rWKFNjaCP5gBSY4sEBT38Q/9RBh9AH40zEOg7Hlq2THRZ" crossorigin="anonymous"></script>
</body>
</html>
```

App Component app-component.html

```
<app-calculator></app-calculator>
```

app-component.ts

```
import { Component } from '@angular/core';
@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
}) export class
AppComponent {
  title =
'calc-app';
}
```

app.module.ts

```
import { NgModule } from '@angular/core'; import {
BrowserModule } from '@angular/platform-browser';
import { AppComponent } from './app.component'; import {
CalculatorComponent } from './calculator/calculator.component'; import
{ CalculatorKeysComponent } from './calculator-
keys/calculatorkeys.component';

@NgModule({
  declarations: [
    AppComponent,
    CalculatorComponent,
    CalculatorKeysComponent
  ],
  imports: [
    BrowserModule
  ],
  providers: [],
  bootstrap: [AppComponent]
})
export class AppModule { }
```

Calculator Component calculator.component.html

```
<div class="calculator">
  <app-calculator-keys></app-calculator-keys>
</div>
```

calculator.component.spec.ts

```
import { ComponentFixture, TestBed } from '@angular/core/testing';
import { CalculatorComponent } from
'./calculator.component';
describe('CalculatorComponent', () => { let
component: CalculatorComponent; let fixture:
ComponentFixture<CalculatorComponent>;
  beforeEach(async () => { await
TestBed.configureTestingModule({
declarations: [ CalculatorComponent ]
})
  .compileComponents();
}); beforeEach(() => { fixture =
TestBed.createComponent(CalculatorComponent);
component = fixture.componentInstance;
fixture.detectChanges();
});
  it('should create', () => {
expect(component).toBeTruthy();
});
});
```

Calculator.component.ts

```
import { Component, OnInit } from '@angular/core';

@Component({ selector:
'app-calculator',
  templateUrl: './calculator.component.html',
styleUrls: ['./calculator.component.css']
})
export class CalculatorComponent {
}
```

calculator.component.css

```
.calculator { border: 1px solid #ccc; border-
radius: 5px; position: relative; top: 50%;
left: 33%; width: 400px;
}
```

Calculator-Keys Component

calculator-keys.component.html

```
<input type="text" class="calculator-screen" [value]="currentNumber"
disabled> <div class="calculator-keys">
  <button type="button" (click) = "getfacto()" class="operator" >!</button>
  <button type="button" (click) = "getPrime()" class="operator" >Prime</button>
  <button type="button" (click) = "getOperation('+')" class="operator" value="
+">+</button>
  <button type="button" (click) = "getOperation('-
')" class="operator" value="-">-</button>
  <button type="button" (click) = "getOperation('*')" class="operator" value="
*">x</button>
  <button type="button" (click) = "getOperation('/')" class="operator" value="
/">/</button>

  <button type="button" (click) = "getNumber('7')" value="7">7</button>
  <button type="button" (click) = "getNumber('8')" value="8">8</button>
<button type="button" (click) = "getNumber('9')" value="9">9</button>

  <button type="button" (click) = "getNumber('4')" value="4">4</button>
  <button type="button" (click) = "getNumber('5')" value="5">5</button>
<button type="button" (click) = "getNumber('6')" value="6">6</button>

  <button type="button" (click) = "getNumber('1')" value="1">1</button>
  <button type="button" (click) = "getNumber('2')" value="2">2</button>
<button type="button" (click) = "getNumber('3')" value="3">3</button>

  <button type="button" (click) = "getNumber('0')" value="0">0</button>
<button type="button" (click) = "getDecimal()" class="decimal" value=".">.</
button>
  <button type="button" (click) = "clear()" class="all-clear"
value="allclear">AC</button>

  <button type="button" (click) = "getOperation('=')" class="equalsign"
value="=">=</button>
</div>
```

calculator-keys.component.css

```
.calculator-screen {
width: 100%; font-size:
5rem; height: 100px;
border: none; background-
color: #252525; color:
#fff; text-align: right;
padding-right: 20px;
padding-left: 10px;
} button { height: 48px;
background-color: #fff;
border-radius: 3px; border:
1px solid #c4c4c4;
background-color: transparent;
font-size: 2rem; color: #333;
background-image: linear-
gradient(to bottom,transparent,transparent 50%,rgba(0,0,0,.04));
box-
shadow: inset 0 0 0 1px rgba(255,255,255,.05), inset 0 1px 0 0
rgba(255,255,255,.45), inset 0 -
1px 0 0 rgba(255,255,255,.15), 0 1px 0 0 rgba(255,255,255,.15); text-
shadow: 0 1px rgba(255,255,255,.4);
} button:hover {
background-color: #eaeaea;
}

.operator {
color: #337cac;
```

```

}

.all-clear {  background-
color: #f0595f;  border-
color: #b0353a;  color: #fff;
}

.all-clear:hover {
  background-color: #f17377;
}

.equal-sign {  background-
color: #2e86c0;  border-
color: #337cac;  color:
#fff;  height: 100%;
grid-area: 2 / 4 / 6 / 5;
}

.equal-sign:hover {
  background-color: #4e9ed4;
}

.calculator-keys {  display: grid;
grid-template-columns: repeat(4, 1fr);
grid-gap: 20px;  padding: 20px;
}

```

calculator-keys.component.specs.ts

```

import { ComponentFixture, TestBed } from '@angular/core/testing';

import { CalculatorKeysComponent } from './calculator-keys.component';

describe('CalculatorKeysComponent', () => {  let
component: CalculatorKeysComponent;  let fixture:
ComponentFixture<CalculatorKeysComponent>;
  beforeEach(async () =>
{
  await TestBed.configureTestingModule({
declarations: [ CalculatorKeysComponent ]  })

```

```

        .compileComponents();
    });    beforeEach(() => {        fixture =
 TestBed.createComponent(CalculatorKeysComponent);
    component = fixture.componentInstance;
    fixture.detectChanges();
    });    it('should create', () =>
    {
    expect(component).toBeTruthy();
    });
    });

```

Calculator-keys.component.ts

```

import { Component, OnInit } from '@angular/core';

@Component({    selector: 'app-calculator-keys',
    templateUrl: './calculator-keys.component.html',
    styleUrls: ['./calculator-keys.component.css']
}) export class
CalculatorKeysComponent{
    currentNumber = '0';    firstOperand=
    0;    operator = "";
    waitForSecondNumber = false;
    public getNumber(v: string){
        console.log(v);
        if(this.waitForSecondNumber)
        {
            this.currentNumber = v;
            this.waitForSecondNumber = false;
        }else{
            this.currentNumber === '0'? this.currentNumber = v: this.currentNumber
            + = v;

            }    }    getDecimal(){
    if(!this.currentNumber.includes('.')){
        this.currentNumber += '.';
    }    }
    getPrime(){

```



```

        const num = Number(this.currentNumber);    let flag =
0;    if(num < 2){        this.currentNumber = "Neither
Prime nor Composite"
    }    for (let k = 2; k < num;
k++){        if( num % k == 0){
flag =1;
        }    }    if(flag==0){
this.currentNumber = "Prime"
    }    else{
this.currentNumber = "Composite"
    } }    getfacto(){        const num=
Number(this.currentNumber);    let answer
= 1;    if (num == 0 || num == 1){
this.currentNumber= "1";
    }    else{        for(var i =
num; i >= 1; i--){            answer =
answer * i;
        }        this.currentNumber =
String(answer);
    }
}

private doCalculation(op:string , secondOp:number){
switch (op){        case '+':
    return this.firstOperand += secondOp;
case '-':
    return this.firstOperand -= secondOp;
case '*':
    return this.firstOperand *= secondOp;
case '/':
    return this.firstOperand /= secondOp;
case '=':
    return secondOp;
    }    return
secondOp;
}    public getOperation(op:
string){        console.log(op);
if(this.firstOperand === null){

```

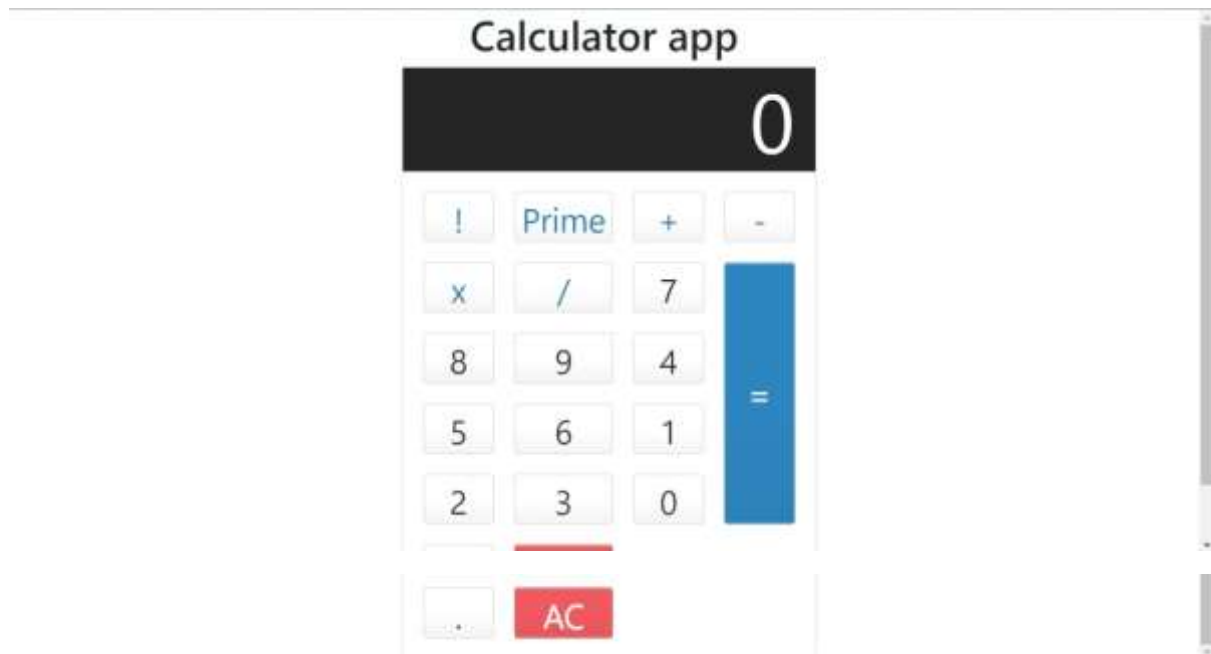
```

        this.firstOperand = Number(this.currentNumber);
    }else if(this.operator){        const result =
this.doCalculation(this.operator , Number(this.currentNu mber)) as number
this.currentNumber = String(result);        this.firstOperand = result;
    }        this.operator = op;
this.waitForSecondNumber = true;

console.log(this.firstOperand);
    }    public clear(){
this.currentNumber = '0';
this.firstOperand = 0;
this.operator = "";
this.waitForSecondNumber = false;
}
}

```

Screenshot of App



Screenshot of all installations taken in vs code

