ANGULAR

1 Develop Angular JS program that allows user to input their first name and last name and display their full name. Note: The default values for first name and last name may be included in the program.

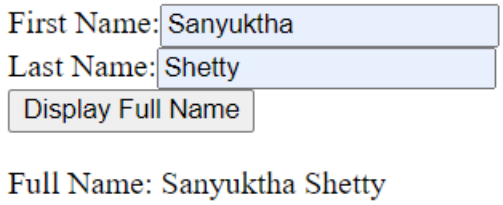
→

app.component.ts

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

@Component({
  selector: 'app-root',
  standalone: true,
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css'],
})
export class AppComponent {
  firstName: string = '';
  lastName: string = '';
  fullName: string = '';

  displayFullName(firstName: string, lastName: string): void {
    this.fullName = `${firstName} ${lastName}`;
  }
}
```

app.component.html

```
<div>
  <label for="firstName">First Name:</label>
  <input id="firstName" type="text" placeholder="Enter first name" #firstNameInput>

  <br>

  <label for="lastName">Last Name:</label>
  <input id="lastName" type="text" placeholder="Enter last name" #lastNameInput>

  <br>

  <button (click)="displayFullName(firstNameInput.value, lastNameInput.value)">Display
Full Name</button>

  <br>

  <p>Full Name: {{ fullName }}</p>
</div>
```



2 Develop an Angular application that displays a list of shopping items. Allow users to add and remove items from the list using directives and controllers.Note: The default values of items may be included in the program.

→

app.component.ts

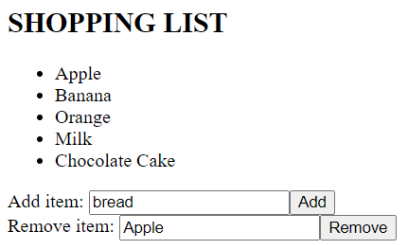
```
import { CommonModule } from '@angular/common';
import { Component } from '@angular/core';

@Component({
  selector: 'app-root',imports:[CommonModule],
  standalone: true,
  templateUrl: './app.component.html',
  styleUrl: './app.component.css'
})
export class AppComponent {
  shoppingItems:String[]=['Apple','Banana' ,'Orange','Milk','Chocolate Cake'];
  addItem(item:String):void{
    if(item &&! this .shoppingItems.includes(item)){
      this.shoppingItems.push(item);
    }

}

removeItem(item:String):void{
  const index=this.shoppingItems.indexOf(item);
  if (index> -1){
    this.shoppingItems.splice(index,1)

}
}
}
```

app.component.html

```
<!-- app.component.html -->

<div>
  <h2>SHOPPING LIST</h2>
  <ul>
    <li *ngFor="let item of shoppingItems">{{ item }}</li>
  </ul>
</div>

<div>
  <label for="newItem">Add item: </label>
  <input id="newItem" #newItemInput placeholder="Enter a new item" />
  <button (click)="addItem(newItemInput.value)">Add</button>
</div>

<div>
  <label for="removeItem">Remove item: </label>
```

```
  <input id="removeItem" #removeItemInput placeholder="Enter item to remove" />
  <button (click)="removeItem(removeItemInput.value)">Remove</button>
</div>
```

Output:



3 Develop a simple Angular calculator application that can perform basic mathematical operations (addition, subtraction, multiplication, division) based on user input.

→

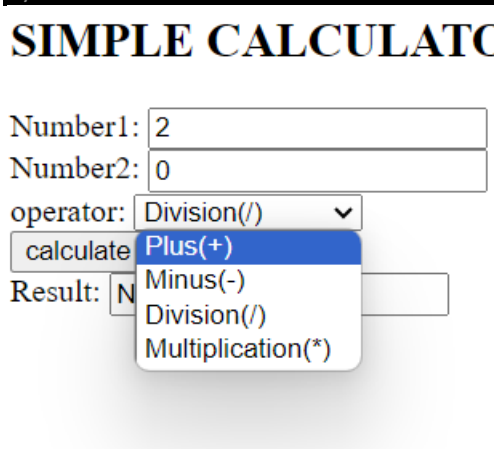
app.component.ts

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

@Component({
  selector: 'app-root',
  standalone: true,
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
})
export class AppComponent {
  result: number = 0;

  calculate(num1: string, num2: string, operator: string): void {
    const parsedNum1 = parseFloat(num1);
    const parsedNum2 = parseFloat(num2);

    switch (operator) {
      case '+':
        this.result = parsedNum1 + parsedNum2;
        break;
      case '-':
        this.result = parsedNum1 - parsedNum2;
        break;
      case '*':
        this.result = parsedNum1 * parsedNum2;
        break;
      case '/':
        this.result = parsedNum2 !== 0 ? parsedNum1 / parsedNum2 :NaN;
        break;
      default:
        this.result = 0;
        break;
    }
```

```
  }
}
```

app.component.html

```
<div>
  <h2>SIMPLE CALCULATOR</h2>
  <label for ="num1">Number1: </label>
  <input id="num1" #num1Input type="number"/>
<br/>
<label for="num2">Number2: </label>
<input id="num2" #num2Input type="number"/>
<br/>
<label for="operator">operator: </label>
<select id="operator" #operatorInput>
  <option value="+">Plus(+)</option>
  <option value="-">Minus(-)</option>
  <option value="/">Division(/)</option>
  <option value="*">Multiplication(*)</option>
</select>
<br/>
<button (click)="calculate(num1Input.value,
num2Input.value,operatorInput.value)">calculate</button>
<br/>
<label for="result">Result: </label>
<input id="result" readonly [value]="result"/>
</div>
```



4 Write an Angular application that can calculate factorial and compute square based on given user input.

→

app.component.ts

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

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

```
 result: any = "";
 factorial(num: number): number {
  if (num == 1) {
    return 1;
  } else {
    return num * this.factorial(num - 1);
  }
 }
 calculateFactorial(number: any) {
  this.result = this.factorial(number);
 }
 calculateSquare(number: any) {
   this.result = parseInt(number) * parseInt(number);
 }
}
```

app.component.html

```
<div>
  <h1>CALCULATE FACTORIAL AND SQUARE</h1>
  <label for="numberInput">Enter the number: </label>
  <input type="number" id="numberInput" name="numberInput" #numberInput>
  <button (click)="calculateFactorial(numberInput.value)">Factorial</button>
  <button (click)="calculateSquare(numberInput.value)">Square</button>
  <p>Result: {{result}}</p>
</div>
```

CALCULATE FACTORIAL AND SQUARE

Enter the number: 5 Factorial Square

Result: 120

CALCULATE FACTORIAL AND SQUARE

Enter the number: 8 Factorial Square

Result: 64

5 Develop Angular application that displays a details of students and their CGPA. Allow users to read the number of students and display the count. Note: Student details may be included in the program.

→

app.component.ts

```
import { Component } from '@angular/core';
import { CommonModule } from '@angular/common';


@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css'],
  standalone: true,
  imports:[CommonModule],
```

```
})
export class AppComponent {
  students: { name: string; USN: string; CGPA: string }[] = [
    { name: 'Sanyuktha', USN: '4MT21CS145', CGPA: '8.94' },
    { name: 'Swayam', USN: '4MT21CS146', CGPA: '8.54' },
    { name: 'Sanjana', USN: '4MT21CS035', CGPA: '9.01' },
    { name: 'veeraj', USN: '4MT21CS194', CGPA: '7.51' },
    { name: 'leela', USN: '4MT21CSo22', CGPA: '6.78' },
  ];

  displayedStudents: { name: string; USN: string; CGPA: string }[] = [];
  studentCount: number = 0;
  displayStudents(limit: string) {
    this.displayedStudents = this.students.slice(0, parseInt(limit));
    this.studentCount = this.displayedStudents.length;
}
}
```

app.component.html

```
<div>
  <h1>Student Details</h1>

  <label for="limitInput">Display Limit:</label>
  <input id="limitInput" type="number" placeholder="Enter limit" #displayLimit />

  <button (click)="displayStudents(displayLimit.value)">Display Students</button>

  <h3>Displayed Students</h3>
  <ul>
    <li *ngFor="let student of displayedStudents">
      {{ student.name }} ({{ student.USN }}) - CGPA: {{ student.CGPA }}
    </li>
  </ul>

  <h3>Student Count: {{ studentCount }}</h3>
</div>
```

Student Details

Display Limit: 5 Display Students

Displayed Students

- Sanyuktha (4MT21CS145) - CGPA: 8.94
- Swayam (4MT21CS146) - CGPA: 8.54
- Sanjana (4MT21CS035) - CGPA: 9.01
- veeraj (4MT21CS194) - CGPA: 7.51
- leela (4MT21CSo22) - CGPA: 6.78

Student Count: 5

6 Develop an Angular program to create a simple to-do list application. Allow users to add, edit, and delete tasks.Note: The default values for tasks may be included in the program.

→

app.component.ts

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

// Import necessary Angular modules
import { CommonModule } from '@angular/common';

@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css'],
  standalone: true,
  imports:[CommonModule],

})
export class AppComponent {
  tasks: string[] = ['Get vegetables', 'Go To Gym', 'complete notes'];
  editingTaskIndex: number | null = null;

  addTask(newTask: string) {
    if (newTask.trim() !== '') {
      this.tasks.push(newTask);
    }
  }

  editTask(index: number) {
    this.editingTaskIndex = index;
  }

  updateTask(editedTask: string) {
    if (this.editingTaskIndex !== null) {
      this.tasks[this.editingTaskIndex] = editedTask;
      this.editingTaskIndex = null;
    }
  }

  deleteTask(index: number) {
    this.tasks.splice(index, 1);
  }
  cancelEdit() {
    this.editingTaskIndex = null;
  }
}
```

app.component.html

```
<div>
  <h1>To-Do List</h1>

  <ul>
```

```
    <li *ngFor="let task of tasks; let i = index">
      {{ task }}
      <button (click)="editTask(i)">Edit</button>
      <button (click)="deleteTask(i)">Delete</button>
    </li>
  </ul>

  <div *ngIf="editingTaskIndex !== null">
    <label for="editTaskInput">Edit Task:</label>
    <input id="editTaskInput" type="text" #editedTaskInput />
    <button (click)="updateTask(editedTaskInput.value)">Update Task</button>
    <button (click)="cancelEdit()">Cancel</button>
  </div>

  <div>
    <label for="newTaskInput">New Task:</label>
    <input id="newTaskInput" type="text" #newTaskInput />
    <button (click)="addTask(newTaskInput.value)">Add Task</button>
  </div>
</div>
```

To-Do List

- Get vegetables Edit Delete
- Go To Gym Edit Delete
- complete notes Edit Delete
- washing clothes Edit Delete

New Task: washing clothes Add Task

7 Write an AngularJS program to create a simple CRUD application (Create, Read, Update, and Delete) for managing users.

→

app.component.ts

```
import { CommonModule } from '@angular/common';
import { Component } from '@angular/core';

@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css'],
  standalone: true,
  imports: [CommonModule],
})
export class AppComponent {
  users: { id: number; name: string; email: string }[] = [
    { id: 1, name: 'John Doe', email: 'john@example.com' },
    { id: 2, name: 'Jane Doe', email: 'jane@example.com' },
    { id: 3, name: 'Bob Smith', email: 'bob@example.com' },
```

```
  ];

  newUser: { name: string; email: string } = { name: '', email: '' };
  editingUser: { id: number; name: string; email: string } | null = null;

  addUser(name: string, email: string) {
    if (name.trim() !== '' && email.trim() !== '') {
      const newUser = {
        id: this.users.length + 1,
        name: name,
        email: email,
      };
      this.users.push(newUser);
      this.newUser = { name: '', email: '' };
    }
  }

  editUser(user: { id: number; name: string; email: string }) {
    this.editingUser = { ...user };
  }

  updateUser(name: string, email: string) {
    if (this.editingUser) {
      const index = this.users.findIndex((user) => user.id === this.editingUser!.id);
      if (index !== -1) {
        this.users[index].name = name;
        this.users[index].email = email;
        this.editingUser = null;
      }
    }
  }

  cancelEdit() {
    this.editingUser = null;
  }

  deleteUser(id: number) {
    this.users = this.users.filter((user) => user.id !== id);
  }
}
```

app.component.html

```
<div>
  <h1>User Management</h1>

  <ul>
    <li *ngFor="let user of users">
      {{ user.name }} ({{ user.email }})
      <button (click)="editUser(user)">Edit</button>
      <button (click)="deleteUser(user.id)">Delete</button>
    </li>
  </ul>
```

```
  <div>
    <label for="name">Name:</label>
    <input id="name" type="text" #newName />
  </div>

  <div>
    <label for="email">Email:</label>
    <input id="email" type="text" #newEmail />
  </div>

  <button (click)="addUser(newName.value, newEmail.value)">Add User</button>

  <div *ngIf="editingUser">
    <h2>Edit User</h2>

    <div>
      <label for="editName">Name:</label>
      <input id="editName" type="text" #editName />
    </div>

    <div>
      <label for="editEmail">Email:</label>
      <input id="editEmail" type="text" #editEmail />
    </div>

    <button (click)="updateUser(editName.value, editEmail.value)">Update User</button>
    <button (click)="cancelEdit()">Cancel</button>
  </div>
</div>
```

8 Develop Angular program to create a login form, with validation for the username and password fields.

→

app.component.ts

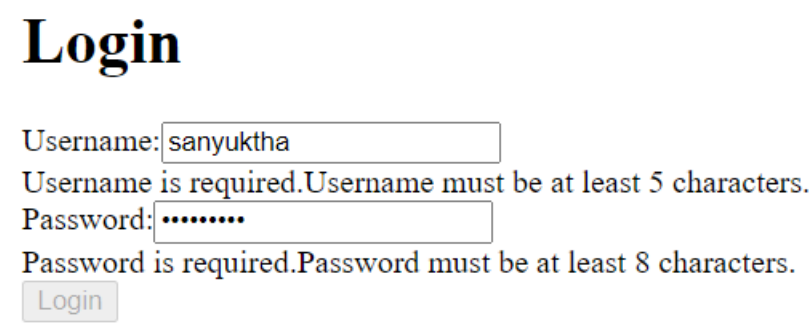
```
import { Component } from '@angular/core';
import { CommonModule } from '@angular/common';
import { FormBuilder, FormGroup, Validators } from '@angular/forms';
@Component({
  selector: 'app-root',
  standalone: true,
  imports: [CommonModule],
  templateUrl: './app.component.html',
```

```
  styleUrl: './app.component.css',
})
export class AppComponent {
  username: string = '';
  password: string = '';
  usernameInput: string = '';
  passwordInput: string = '';
  usernameInputTouched: boolean = false;
  passwordInputTouched: boolean = false;

  get usernameInputInvalid(): boolean {
    return this.usernameInputTouched && (this.usernameRequired || this.usernameMinlength);
  }

  get passwordInputInvalid(): boolean {
    return this.passwordInputTouched && (this.passwordRequired || this.passwordMinlength);
  }

  get usernameRequired(): boolean {
    return this.username.trim() === '';
  }

  get usernameMinlength(): boolean {
    return this.username.length < 5;
  }

  get passwordRequired(): boolean {
    return this.password.trim() === '';
  }

  get passwordMinlength(): boolean {
    return this.password.length < 8;
  }

  isValidForm(): boolean {
    return !this.usernameInputInvalid && !this.passwordInputInvalid;
  }

  login() {
    this.usernameInputTouched = true;
    this.passwordInputTouched = true;

    if (this.isValidForm()) {
      console.log('Username:', this.username);
      console.log('Password:', this.password);

    }
  }
}
```

app.component.html

```
<div>
  <h1>Login</h1>
```

```
  <div>
    <label for="usernameInput">Username:</label>
    <input id="usernameInput" type="text" #usernameInputRef required minlength="5"
#usernameInput />
    <div *ngIf="usernameInputTouched && usernameInputInvalid">
      <span *ngIf="usernameRequired">Username is required.</span>
      <span *ngIf="usernameMinlength">Username must be at least 5 characters.</span>
    </div>
  </div>

  <div>
    <label for="passwordInput">Password:</label>
    <input id="passwordInput" type="password" #passwordInputRef required minlength="8"
#passwordInput />
    <div *ngIf="passwordInputTouched && passwordInputInvalid">
      <span *ngIf="passwordRequired">Password is required.</span>
      <span *ngIf="passwordMinlength">Password must be at least 8 characters.</span>
    </div>
  </div>

  <button (click)="login()" [disabled]="!isValidForm()">Login</button>
</div>
```



9 Create an Angular application that displays a list of employees and their salaries. Allow users to search for employees by name and salary. Note: Employee details may be included in the program.

→

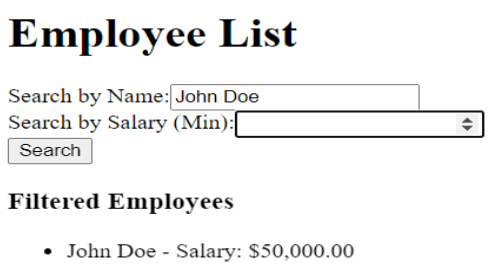
app.component.ts

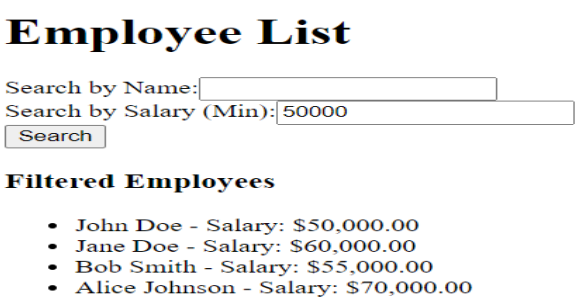
```
import { Component } from '@angular/core';
import { CommonModule } from '@angular/common';
@Component({
  selector: 'app-root',
  standalone:true,
  imports: [CommonModule],
  templateUrl: './app.component.html',
  styleUrl: './app.component.css',
})
export class AppComponent {
  employees: { name: string; salary: number }[] = [
    { name: 'John Doe', salary: 50000 },
    { name: 'Jane Doe', salary: 60000 },
    { name: 'Bob Smith', salary: 55000 },
```

```
    { name: 'Alice Johnson', salary: 70000 },
    { name: 'Charlie Brown', salary: 48000 },
  ];

  filteredEmployees: { name: string; salary: number }[] = [];

  searchEmployees(name: string, salary: string) {
    this.filteredEmployees = this.employees.filter((employee) =>
      this.matchesSearchCriteria(employee, name, salary)
    );
  }

  private matchesSearchCriteria(employee: { name: string; salary: number }, name: string,
salary: string): boolean {
    const nameMatch = !name || employee.name.toLowerCase().includes(name.toLowerCase());
    const salaryMatch = salary === '' || employee.salary >= +salary;
    return nameMatch && salaryMatch;
  }
}
```

app.component.html

```
<div>
  <h1>Employee List</h1>

  <div>
    <label for="nameInput">Search by Name:</label>
    <input id="nameInput" type="text" #nameInput />
  </div>

  <div>
    <label for="salaryInput">Search by Salary (Min):</label>
    <input id="salaryInput" type="number" #salaryInput />
  </div>

  <button (click)="searchEmployees(nameInput.value, salaryInput.value)">Search</button>

  <h3>Filtered Employees</h3>
  <ul>
    <li *ngFor="let employee of filteredEmployees">
      {{ employee.name }} - Salary: {{ employee.salary | currency: 'USD':'symbol':'1.2-2'
}}
    </li>
  </ul>
</div>
```





10 Create Angular application that allows users to maintain a collection of items. The application should display the current total number of items, and this count should automatically update as items are added or removed. Users should be able to add items to the collection and remove them as needed. Note: The default values for items may be included in the program.

→

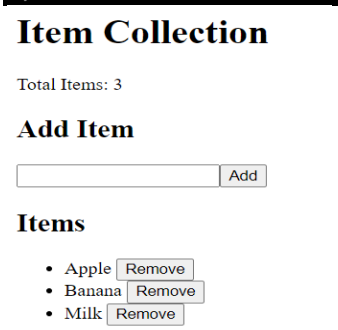
app.component.ts

```
import { Component } from '@angular/core';
import { CommonModule } from '@angular/common';

@Component({
  selector: 'app-root',
  standalone: true,
  imports: [CommonModule],
  templateUrl: './app.component.html',
  styleUrl: './app.component.css'
})
export class AppComponent {
  items: string[] = ['Apple', 'Banana', 'Milk']; // Default items
  itemCount: number = this.items.length;

  addItem(newItem: string): void {
    this.items.push(newItem);
    this.itemCount = this.items.length;
  }

  removeItem(index: number): void {
    this.items.splice(index, 1);
    this.itemCount = this.items.length;
  }
}
```

app.component.html

```
<div>
  <h1>Item Collection</h1>
  <p>Total Items: {{ itemCount }}</p>

  <h2>Add Item</h2>
  <input type="text" #newItem />
  <button (click)="addItem(newItem.value)">Add</button>

  <h2>Items</h2>
  <ul>
```

```
    <li *ngFor="let item of items; let i = index">
      {{ item }} <button (click)="removeItem(i)">Remove</button>
    </li>
  </ul>
</div>
```



11 Create Angular application to convert student details to Uppercase using angular filters. Note: The default details of students may be included in the program.

→

app.component.ts

```
import { Component } from '@angular/core';
import { CommonModule } from '@angular/common';

@Component({
  selector: 'app-root',
  standalone: true,
  imports: [CommonModule],
  templateUrl: './app.component.html',
  styleUrl: './app.component.css'
})
export class AppComponent {
  students: Student[] = [
    { name: 'John Doe', grade: 'A' },
    { name: 'Jane Doe', grade: 'B' },
    { name: 'Alice Smith', grade: 'C' }
    // Default student details
  ];
}
export interface Student {
  name: string;
  grade: string;
}
```

app.component.html

```

<div>
  <h1>Student Details</h1>

  <table>
    <thead>
```

```
      <tr>
        <th>Name</th>
        <th>Grade</th>
      </tr>
    </thead>
    <tbody>
      <tr *ngFor="let student of students">
        <td>{{ student.name | uppercase }}</td>
        <td>{{ student.grade | uppercase }}</td>
      </tr>
    </tbody>
  </table>
</div>
```

Student Details

Name	Grade
JOHN DOE	A
JANE DOE	B
ALICE SMITH	C

12 Create an Angular application that displays the date by using date filter parameters

→

app.component.ts

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

@Component({
  selector: 'app-root',
  standalone: true,
  templateUrl: './app.component.html',
  styleUrl: './app.component.css'
})
export class AppComponent {
  currentDate: Date = new Date();
}
```

app.component.html

```
<!-- app.component.html -->
<div>
  <h1>Current Date</h1>

  <p>{{ currentDate.toLocaleDateString('en-US', { weekday: 'long', year: 'numeric', month: 'long', day: 'numeric' }) }}</p>
</div>
```

Current Date

Tuesday, February 27, 2024