**codingame solution 2**  
  
  
  
export const appRoutes: Routes = [

{ path: 'menu1', children:

[{ path: '', outlet: 'primary', component: Menu1Component },

{ path: '', outlet: 'footer', component: FooterAComponent }]},

{ path: 'menu2' , children:

[{ path: '', outlet: 'primary', component: Menu2Component },

{ path: '', outlet: 'footer', component: FooterBComponent }]}

]

@NgModule({

imports: [RouterModule.forRoot(appRoutes, {useHash: false})],

exports: [RouterModule]

})

export class AppRoutingModule {

}

<nav>

<a id="menu1" routerLink="/menu1">Menu1</a>

</nav>

<nav>

<a id="menu2" routerLink="/menu2">Menu2</a>

</nav>

<br>

<router-outlet></router-outlet>

<br>

<footer-component id="footer"></footer-component>

// Angular 15.x​​​​​​‌​‌​​​‌‌‌​‌‌‌‌​​​​‌​‌‌‌​‌ code import { Component, Input, NgModule, Injectable } from '@angular/core'; import {APP\_BASE\_HREF} from '@angular/common'; import { RouterModule, Routes, Router } from '@angular/router'; @Component({ selector:'footer-a-component', template: `

You are at footer A.

` }) export class FooterAComponent { } @Component({ selector:'footer-b-component', template: `

You are at footer B

` }) export class FooterBComponent { } @Component({ selector:'menu1-component', template: `

You are at menu 1.

` }) export class Menu1Component { } @Component({ selector:'menu2-component', template: `

You are at menu 2.

` }) export class Menu2Component { } @Component({ selector: 'footer-component', template: `` }) export class FooterWrapperComponent { } export const appRoutes: Routes = [ { path: 'menu1',children:[ {path:'',outlet:'primary',component:Menu1Component}, {path:'',outlet:'footer',component:FooterAComponent}, ]}, { path: 'menu2',children:[ {path:'',outlet:'primary',component:Menu2Component}, {path:'',outlet:'footer',component:FooterBComponent}, ]}, ] // #region Preview @Component({ template: ` Menu1 Menu2

` }) export class PreviewComponent { } // #endregion Preview // #region Module declaration - Do not Change @NgModule({ imports: [RouterModule.forRoot(appRoutes, {useHash: false})], exports: [RouterModule] }) export class AppRoutingModule { } @NgModule({ imports: [ RouterModule, AppRoutingModule ], declarations: [ PreviewComponent, Menu1Component, Menu2Component, FooterWrapperComponent, FooterAComponent, FooterBComponent ], entryComponents: [PreviewComponent], providers: [{provide: APP\_BASE\_HREF, useValue : '/'}] }) export class PreviewModule { } // #endregion Module declaration

**problem 3 min user interface**

type User = AuthenticatedUser | AnonymousUser;

**FruitComponent**

@Component({

selector:'fruit-component',

template: `

<div id="fruitComp">You have selected the {{name}}</div>

`

})

export class FruitComponent {

public name: any = '';

constructor(private route: ActivatedRoute) {

this.route.params.subscribe(params => {

this.name = params['name'];

});

}

}

export const appRoutes: Routes = [

{ path: 'fruit/:name', component: FruitComponent },

]

// #region Preview

@Component({

template: `

<nav>

<a id="menuBanana" routerLink="/fruit**/banana**" [queryParams]="{name: 'banana'}">

Link to banana.

</a>

</nav>

<nav>

<a id="menuOrange" routerLink="/fruit**/orange**" [queryParams]="{name: 'orange'}">

Link to orange.

</a>

</nav>

<nav>

<a id="menuStrawberry" routerLink="/fruit**/strawberry**" [queryParams]="{name: 'strawberry'}">

Link to strawberry.

</a>

</nav>

<br>

<router-outlet></router-outlet>

`

})

On considère la variable person ayant l'interface suivante :

{firstName: string, lastName: string)

Elle est initialisée par :

this.person = {firstName: 'First', lastName: 'Last'}

On vous demande d'ajouter des transformations, applicables à toutes variables

person implémentant l'interface ci-dessus.

Comportement des transformations:

• Si le paramètre du pipe name

est (truel, les propriétés

firstName

et

lastName

seront retournées dans un ordre inversé (ex : Last First). Si le paramètre est

false

ou non fourni, l'ordre reste le même.

• Si le paramètre du pipe dotifier

est

\*same', les propriétés firstName

et

lastName

seront retournées sans modification (ex : First Last)]. Si le paramètre

est "dotify' ou n'est pas fourni, un point est ajouté après chaque caractère de

la chaîne, y compris après l'espace séparant les deux propriétés. (ex :

F.i.r.s.t.

.L.a.s.t.

// Angular 15.x​​​​​​‌​‌​‌‌‌​‌​​​​​​​​‌‌​‌‌​​‌ code import { Component, NgModule } from '@angular/core'; // Fix classes Name and Dotifier export class Name { } export class Dotifier { } // #region Preview @Component({ template: `{{ person | name | dotifier }}` }) export class PreviewComponent { public person = {firstName: 'First', lastName:'Last'}; } // #endregion Preview // #region Module declaration - Do not Change @NgModule({ declarations: [PreviewComponent, Name, Dotifier], entryComponents: [PreviewComponent] }) export class PreviewModule { } // #endregion Module dimport { Component, NgModule, Pipe, PipeTransform } from '@angular/core';

// Fix classes Name and Dotifier

export class Name { }

export class Dotifier { }

// Custom pipe for reversing names

@Pipe({

name: 'name'

})

export class NamePipe implements PipeTransform {

transform(value: any, reverse: boolean = false): string {

if (reverse) {

return `${value.lastName} ${value.firstName}`;

} else {

return `${value.firstName} ${value.lastName}`;

}

}

}

// Custom pipe for dotifying names

@Pipe({

name: 'dotifier'

})

export class DotifierPipe implements PipeTransform {

transform(value: any, type: string = 'dotify'): string {

if (type === 'same') {

return `${value.firstName} ${value.lastName}`;

} else {

let dottedString = '';

for (let char of `${value.firstName} ${value.lastName}`) {

dottedString += char + '.';

}

return dottedString;

}

}

}

// #region Preview

@Component({

template: `{{ person | name:true | dotifier:'dotify' }}`

})

export class PreviewComponent {

public person = { firstName: 'First', lastName: 'Last' };

}

// #endregion Preview

// #region Module declaration - Do not Change

@NgModule({

declarations: [PreviewComponent, NamePipe, DotifierPipe],

entryComponents: [PreviewComponent]

})

export class PreviewModule { }

// #endregion Module declarationeclaration