MỤC TIÊU:

Kết thúc bài thực hành này bạn có khả năng

- ✓ Sử dụng các loại dữ liệu nâng cao: Intersection Type, Type Guard, Discriminated unions.
- ✓ Sử dụng Generic

NỘI DUNG

Bài 1: Intersection Type

Thực hiện bài tập ví dụ

```
type Admin = {
  name: string;
  privileges: string[];
};

type Employee = {
  name: string;
  startDate: Date;
};

type ElevatedEmployee = Admin & Employee;
```

Bài 2: Type guard

Thực hiện bài tập ví dụ

```
class Car {
  drive() {
    console.log('Driving...');
  }
}

class Truck {
  drive() {
    console.log('Driving a truck...');
  }

  loadCargo(amount: number) {
    console.log('Loading cargo ...' + amount);
  }
}

type Vehicle = Car | Truck;

const v1 = new Car();
const v2 = new Truck();

function useVehicle(vehicle: Vehicle) {
  vehicle.drive();
  if (vehicle instanceof Truck) {
    vehicle.loadCargo(1000);
  }
}
```

Bài 3: Discriminated unions

Thực hiện bài tập ví dụ

```
interface Bird {
  type: 'bird';
  flyingSpeed: number;
}

interface Horse {
  type: 'horse';
  runningSpeed: number;
}

type Animal = Bird | Horse;

function moveAnimal(animal: Animal) {
  let speed;
  switch (animal.type) {
    case 'bird':
      speed = animal.flyingSpeed;
      break;
    case 'horse':
      speed = animal.runningSpeed;
  }
  console.log('Moving at speed: ' + speed);
}
```

Bài 4: Generic

Generic

```
function merge<T extends object, U extends object>(objA: T, objB: U) {
  return Object.assign(objA, objB);
}

const mergedObj = merge({ name: 'Max', hobbies: ['Sports'] }, { age: 30 });
console.log(mergedObj);
```

Generic interface

```
interface Lengthy {
  length: number;
}

function countAndDescribe<T extends Lengthy>(element: T): [T, string] {
  let descriptionText = 'Got no value.';
  if (element.length === 1) {
    descriptionText = 'Got 1 element.';
  } else if (element.length > 1) {
    descriptionText = 'Got ' + element.length + ' elements.';
  }
  return [element, descriptionText];
}
```

Generic class

```
class DataStorage<T extends string | number | boolean> {
  private data: T[] = [];

  addItem(item: T) {
    this.data.push(item);
  }

  removeItem(item: T) {
    if (this.data.indexOf(item) === -1) {
      return;
    }
    this.data.splice(this.data.indexOf(item), 1); // -1
  }

  getItems() {
    return [...this.data];
  }
}

const textStorage = new DataStorage<string>();
textStorage.addItem('Max');
textStorage.addItem('Manu');
textStorage.removeItem('Max');
console.log(textStorage.getItems());

const numberStorage = new DataStorage<number>();
```

Bài 5: LEARNING TYPESCRIPT

Tạo simple project có cấu trúc sau:

- Tên project: LEARNING_TYPESCRIPT
- Project gồm: `src/app.ts (biên dịch đến dist/app.js), index.html, style.css, tsconfig.json`

index.html

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <link rel="stylesheet" href="public/style.css" />
    <link href="http://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css" rel="stylesheet"
    <script src="http://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js"></script>
    <script src="http://cdnjs.cloudflare.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>
    <title>Learning TypeScript</title>
  </head>
  <body>
    <main>
      <h1>Typed Pokedex</h1>
      <div id="login">

      </div>
      <div id="app">
      </div>
    </main>
    <script src="public/js/app.js"></script>
  </body>
</html>
```

app.ts

Hiển thị form như hình vào id=”demo”

Bài 4

Giảng viên cho thêm

*** Yêu cầu nộp bài:

SV nén file (*hoặc share thư mục google drive*) bao gồm các yêu cầu đã thực hiện trên, nộp LMS đúng thời gian quy định của giảng viên. KHÔNG NỘP BÀI COI NHƯ KHÔNG CÓ ĐIỂM.

--- Hết ---