1. Create a program that generates a random password with a specified length.

Write a TypeScript class that represents a person. The class should have properties for the person's name, age, and email address. It should also have a method that returns a greeting message.

Create an interface called Product with properties for name, price, and description. Create an array of Product objects and display their information in a table.

Implement a generic class **Stack** that can push and pop elements. Include methods to check if the stack is empty. Export the Stack class as a module so it can be imported and used in other files.

1. Create a program that takes a sentence and returns the longest word in the sentence.

Create a webpage that has a table with a column of checkboxes. When the user checks a checkbox, highlight the corresponding row in the table.

Write a TypeScript program that implements a basic calculator. The program should ask the user for two numbers and an operator (+, -, \*, /) and then perform the requested operation. Use interface

Create a class called Shape with properties for width and height. Add methods to the class that calculate the area and perimeter of the shape.

Create a generic class called Stack that implements a stack data structure. The class should have a generic type parameter that specifies the type of values that can be stored in the stack. The class should have the following methods:

• push(value: T): void - adds a value to the top of the stack

• pop(): T | undefined - removes and returns the value at the top of the stack, or returns undefined if the stack is empty

• peek(): T | undefined - returns the value at the top of the stack without removing it, or returns undefined if the stack is empty

• isEmpty(): boolean - returns true if the stack is empty, or false if it contains one or more values

Export the Stack class as a module so it can be imported and used in other files.

1. Write a TypeScript program that takes an array of numbers and returns a new array with only the even numbers from the original array.

Create a webpage that has a button and a div. When the button is clicked, load a new image into the div.

Write a TypeScript class that represents a car. The class should have properties for the make, model, and year of the car. It should also have methods for starting and stopping the car

Create a class about categories. Create a method that displays the list of categories and count of elements in the category. Export the Category class as a module so it can be imported and used in other files.

1. Create a class called Animal that extends an interface with properties for name and sound. Add a method to the class that returns a string containing the animal's name and sound.

Create a webpage that has a form with a select dropdown and a button. When the button is clicked, get the value of the selected option in the dropdown and display a message on the page based on the selected value.

Create an interface called Book with properties for title, author, and publisher. Create an array of Book objects and display their information in a list, sorted by author name in ascending order.

Create a generic class called Queue that implements a queue data structure. The class should have a generic type parameter that specifies the type of values that can be stored in the queue. The class should have the following methods:

• enqueue(value: T): void - adds a value to the back of the queue

• dequeue(): T | undefined - removes and returns the value at the front of the queue, or returns undefined if the queue is empty

1. Write a TypeScript class that represents a bank account. The class should have properties for the account number, account holder name, and account balance. It should also have methods for depositing and withdrawing money from the account.

Implement a generic function that takes two arrays and merges them into a single array.

Create a webpage that has a button and a paragraph of text. When the button is clicked, toggle the visibility of the paragraph.

Create an interface called Todo with properties for title, description, and completed. Create an array of Todo objects and display their information in a table.

1. Create a class called Vehicle with properties for make, model, and year. Add methods to the class that return the vehicle's make and model as a string.

Create a generic class called Stack that implements a stack data structure. The class should have a generic type parameter that specifies the type of values that can be stored in the stack. The class should have the following methods:

•peek(): T | undefined - returns the value at the top of the stack without removing it, or returns undefined if the stack is empty

• isEmpty(): boolean - returns true if the stack is empty, or false if it contains one or more values

Export the Stack class as a module so it can be imported and used in other files.

Create a webpage that has an input field and a button. When the button is clicked, use the value of the input field to search for a specific element on the page and highlight it.

1. Create a webpage that has a form with two input fields (username and password) and a submit button. When the submit button is clicked, validate that the username and password fields are not empty. If they are empty, display an error message below the form. If they are not empty, display a success message.

Implement a generic class Stack that can push and pop elements. Include methods to check if the stack is empty.

Create an interface Book with properties like title, author, and pages. Write a function that initializes a book object.

Create a program that takes an array of strings and returns a new array with only the unique strings.

1. Create a webpage that has a table with a column of checkboxes. When the user checks a checkbox, highlight the corresponding row in the table.

Create a generic function that takes in an input and returns the middle element.

Create a class called Person with properties for firstName, lastName, and email. Add a method to the class that returns the person's full name in the format "lastName, firstName".

Create an interface called Employee with properties for name, title, and salary. Create an array of Employee objects and display their information in a table, sorted by salary in descending order.

1. Write a function that takes an array of strings and returns a new array containing only the strings with more than 5 characters.

Create a module that exports a class Car with properties like make, model, and a method startEngine.

Implement a generic function that takes an array and returns the last element of the array.

Define an interface Person with name and age properties. Create a class Student that implements this interface.

1. Create a webpage that has a button and a div. When the button is clicked, create a new element (e.g. a paragraph) and append it to the div.

Create a class called Vehicle with properties for make, model, and year. Add methods to the class that return the vehicle's make and model as a string.

Create an interface called Book with properties for title, author, and publisher. Create an array of Book objects and display their information in a list, sorted by author name in ascending order.

Create a generic class called Queue that implements a queue data structure. The class should have a generic type parameter that specifies the type of values that can be stored in the queue. The class should have the following methods:

• enqueue(value: T): void - adds a value to the back of the queue

• dequeue(): T | undefined - removes and returns the value at the front of the queue, or returns undefined if the queue is empty