```
// 1. Arrow Function for Square
const square = (num) => num * num;

// Example usage:
let num = parseInt(prompt("Enter a number to find its square: "));
console.log(`Square of ${num} is ${square(num)}`);

// 2. Array Manipulations on Students' Ages
const ages = [19, 22, 19, 24, 20, 25, 26, 24, 25, 24];

// Sorting the array to find min and max
ages.sort((a, b) => a - b);
const minAge = ages[0];
const maxAge = ages[ages.length - 1];
console.log(`Min Age: ${minAge}, Max Age: ${maxAge}`);

// Finding the median age
let median;
if (ages.length % 2 === 0) {
  median = (ages[ages.length / 2 - 1] + ages[ages.length / 2]) / 2;
} else {
  median = ages[Math.floor(ages.length / 2)];
}
console.log(`Median Age: ${median}`);

// Finding the average age
const averageAge = ages.reduce((sum, age) => sum + age, 0) / ages.length;
console.log(`Average Age: ${averageAge}`);

// Finding the range of the ages
const ageRange = maxAge - minAge;
```

```
console.log(`Age Range: ${ageRange}`);

// Comparing (min - average) and (max - average)
const minDifference = Math.abs(minAge - averageAge);
const maxDifference = Math.abs(maxAge - averageAge);
console.log(`Min-Average Difference: ${minDifference}, Max-Average Difference: ${maxDifference}`);

// 3. Contact Information using a Map
const contacts = new Map();

// Adding contact details
contacts.set('John', {age: 30, email: 'john@example.com', location: 'New York'});
contacts.set('Jane', {age: 25, email: 'jane@example.com', location: 'Los Angeles'});

// Function to retrieve contact details by name
const getContactByName = (name) => {
  if (contacts.has(name)) {
    console.log(contacts.get(name));
  } else {
    console.log("Contact not found.");
  }
}

// Example usage:
getContactByName(prompt("Enter the name to get contact details: "));

// 4. Using Call Method with Introduce Function
const person1 = {
  name: 'John',
  age: 30
};
```

```
const person2 = {
   name: 'Jane',
   age: 25
};

function introduce() {
   console.log(`Hello, I'm ${this.name}, and I'm ${this.age} years old.`);
}

// Using call to introduce person2
introduce.call(person2);

// 5. Managing Unique Items with Set and Map
const uniqueNumbers = new Set([1, 2, 3, 4, 5]);
const squaresMap = new Map();

// Storing squares of unique numbers in the map
uniqueNumbers.forEach(num => {
   squaresMap.set(num, num * num);
});

// Printing unique numbers and their squares
squaresMap.forEach((square, num) => {
   console.log(`Number: ${num}, Square: ${square}`);
});

// 6. Display Info and Greet with Call, Apply, and Bind
// Function displayInfo
function displayInfo(name, role) {
   console.log(`Name: ${name}, Role: ${role}`);
}

// Using call
displayInfo.call(null, 'Alice', 'Developer');

// Using apply
displayInfo.apply(null, ['Bob', 'Designer']);

// Function greet with this context
function greet() {
  console.log(`Hello, ${this.name}`);
}

// Using bind to create a new function with specific context
const person = {name: 'Charlie'};
const boundGreet = greet.bind(person);
boundGreet();

// 7. Calculator Object and Discount Application
const calculator = {
  add: (a, b) => a + b,
  subtract: (a, b) => a - b,
  multiply: (a, b) => a * b,

  calculate: function(operation, a, b) {
    return this[operation](a, b);
  }
};

// Using call for addition
console.log(calculator.calculate.call(calculator, 'add', 10, 20));
```

```
// Using apply for multiplication
console.log(calculator.calculate.apply(calculator, ['multiply', 10, 20]));

const discountCalculator = {
  discountPercentage: 10,

  applyDiscount: function(amount) {
    return amount - (amount * this.discountPercentage / 100);
  }
};

// Using bind to create a new function calculateDiscount
const calculateDiscount = discountCalculator.applyDiscount.bind(discountCalculator);

// Example usage:
console.log(calculateDiscount(100));  // Discounted amount
```