```
/*Task 9.
[This keyword + class + constructor + inheritance + oops + object]
Problem Statement:
You are tasked with implementing a simple banking system using JavaScript classes. Your system should have
the following features:

1.BankAccount Class:
This class should represent a generic bank account. It should have the following properties and methods:
oProperties:
owner: The owner's name.
balance: The current balance in the account.
*/
class BankAccount {
    constructor (owner,balance){
        this.owner = owner;
        this.balance = balance;
    }
```

```
// Methods:
// deposit(amount): Adds the specified amount to the account balance.
deposit(amount) {
        if (amount > 0) {
            this.balance += amount;
            console.log(`${amount} deposited. New balance is ${this.balance}.`);
        } else {
            console.log("Deposit amount must be positive.");
        }
    }

// withdraw(amount): Subtracts the specified amount from the account balance. Make sure to check if the
withdrawal amount is not greater than the current balance.
 withdraw(amount){
        if(amount>0 && amount <= this.balance){
            this.balance-=amount;
            console.log(`${this.owner} withdrw amount`);
        }
        else {
            console.log("withdrawl amunt is greater than balance and it is not positive");
        }
    }
```

```
}
// 2.SavingsAccount Class (Inheritance):
// This class should inherit from the BankAccount class and have the following additional features:
// oProperties:
class SavingsAccount extends BankAccount {
  constructor(owner, balance = 0, interestRate = 0) {
    super(owner, balance);
    this.interestRate = interestRate;
  }
```

```
// interestRate: The annual interest rate for the savings account.
```

```
// oMethods:
// addInterest(): Adds interest to the account balance based on the annual interest rate.
addInterest() {
    const interest = this.balance * this.interestRate / 100;
    this.balance += interest;
    console.log(`${this.owner} earned $${interest} interest. New balance: $${this.balance}`);
  }
}
// 3.CheckingAccount Class (Inheritance):
// class CheckingAccount extends BankAccount {
class CheckingAccount extends BankAccount {
  constructor(owner, balance = 0, overdraft = 0) {
    super(owner, balance);
    this.overdraft = overdraft;
  }
```

```
  withdraw(amount) {
    if (amount > 0 && amount <= this.balance + this.overdraft) {
      this.balance -= amount;
      console.log(`${this.owner} withdrew $${amount}. New balance: $${this.balance}`);
    } else {
      console.log('Withdrawal amount exceeds balance and overdraft limit or is not positive');
    }
  }
}
// Task:
// 1.Implement the BankAccount, SavingsAccount, and CheckingAccount classes as described above.
// 2.Create instances of each class and demonstrate the functionality of their methods by performing
various deposit, withdrawal, and interest addition operations.
// 3.Test the inheritance relationship by using the instanceof operator to verify that instances of
SavingsAccount and CheckingAccount are also instances of BankAccount.
const bankAccount = new BankAccount('mani', 1000);
bankAccount.deposit(500);
bankAccount.withdraw(200);
```

```
const savingsAccount = new SavingsAccount('manoj', 2000, 5);
savingsAccount.deposit(500);
savingsAccount.withdraw(200);
savingsAccount.addInterest();
```

```
const checkingAccount = new CheckingAccount('Chinna', 500, 200);
checkingAccount.deposit(300);
checkingAccount.withdraw(1000);
checkingAccount.withdraw(50);
```

```
// Test inheritance relationship
console.log(savingsAccount instanceof BankAccount); // true
console.log(checkingAccount instanceof BankAccount); // true
console.log(savingsAccount instanceof SavingsAccount); // true
console.log(checkingAccount instanceof CheckingAccount); // true
```