TEST-7

JAVA PROGRAMING

CSA0961

1.

```
class BankAccount {
    private String accountNumber;
    private double balance;

    public BankAccount(String
accountNumber, double initialBalance) {
        this.accountNumber =
accountNumber;
        this.balance = initialBalance;
    }

    public void deposit(double amount) {
        if (amount > 0) {
```

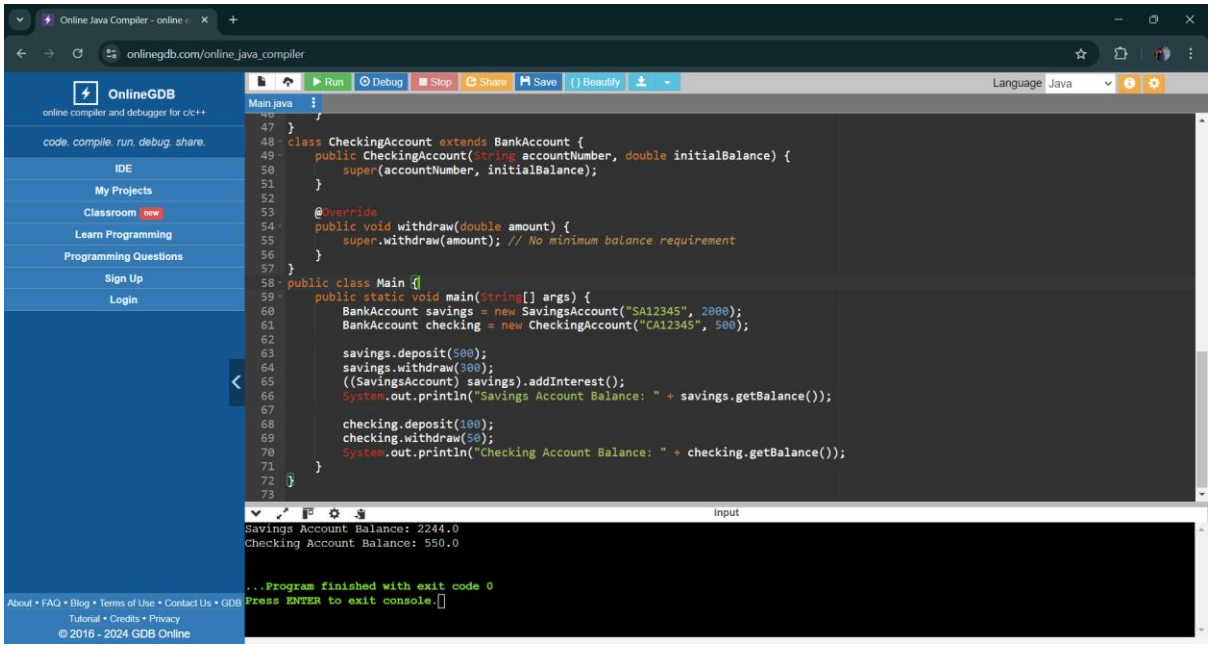
```
            balance += amount;
        }
    }

    public void withdraw(double amount) {
        if (amount > 0 && amount <=
balance) {
            balance -= amount;
        }
    }

    public double getBalance() {
        return balance;
    }
}
```

```
class SavingsAccount extends
BankAccount {
    private static final double
MIN_BALANCE = 1000;
    private static final double
INTEREST_RATE = 0.02;

    public SavingsAccount(String
accountNumber, double initialBalance) {
        super(accountNumber, initialBalance);
        if (initialBalance < MIN_BALANCE) {
            throw new
IllegalArgumentException("Initial balance
must be at least " + MIN_BALANCE);
        }
    }
    public void addInterest() {
```

```
        double interest = getBalance() *
INTEREST_RATE;
        deposit(interest);
    }

    @Override
    public void withdraw(double amount) {
        if (getBalance() - amount >=
MIN_BALANCE) {
            super.withdraw(amount);
        } else {
            System.out.println("Withdrawal
denied! Minimum balance requirement
not met.");
        }
    }
}
```

```
class CheckingAccount extends
BankAccount {
    public CheckingAccount(String
accountNumber, double initialBalance) {
        super(accountNumber, initialBalance);
    }
    @Override
    public void withdraw(double amount) {
        super.withdraw(amount); // No
minimum balance requirement
    }
}
public class Main {
    public static void main(String[] args) {
        BankAccount savings = new
SavingsAccount("SA12345", 2000);
```

```
        BankAccount checking = new
CheckingAccount("CA12345", 500);

        savings.deposit(500);
        savings.withdraw(300);
        ((SavingsAccount)
savings).addInterest();
        System.out.println("Savings Account
Balance: " + savings.getBalance());

        checking.deposit(100);
        checking.withdraw(50);
        System.out.println("Checking Account
Balance: " + checking.getBalance());
    }
}
```



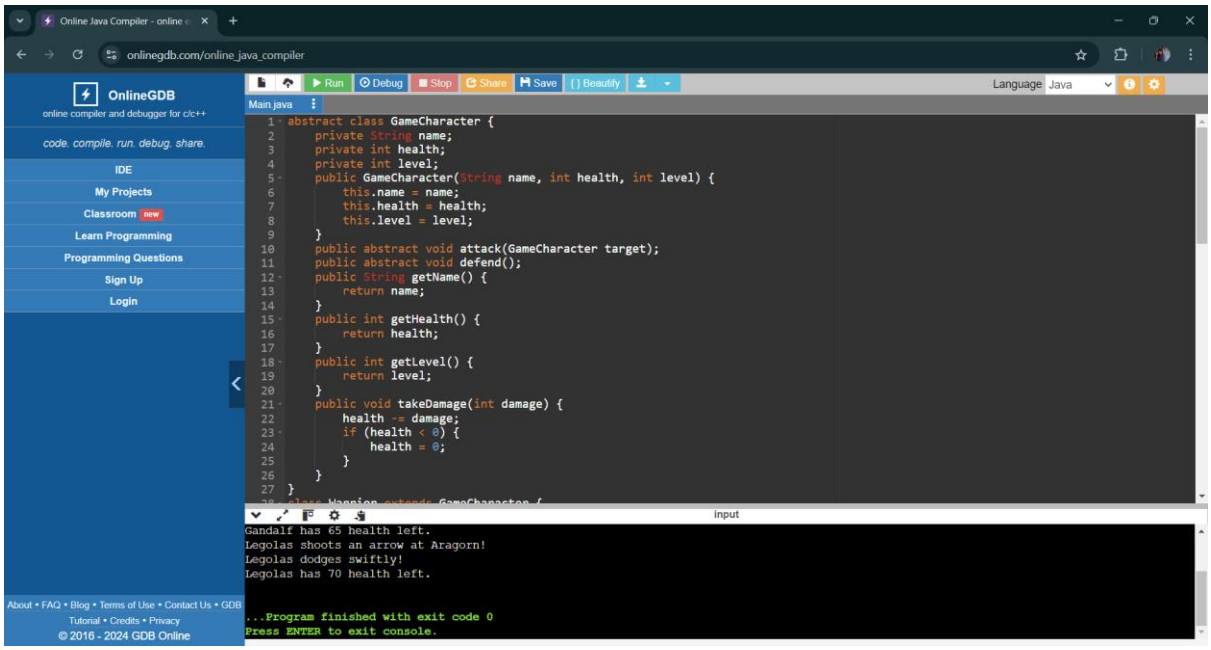
2.

```
public class Main {
    public static void main(String[] args) {
        GameCharacter warrior = new Warrior("Thor", 100, 1);
        GameCharacter mage = new Mage("Gandalf", 80, 1);
        GameCharacter archer = new Archer("Legolas", 90, 1);


        warrior.attack(mage);
```

```
        mage.defend();
        mage.attack(archer);
        archer.defend();
        archer.attack(warrior);
        warrior.defend();

        System.out.println(warrior.getName()
+ " Health: " + warrior.getHealth());
        System.out.println(mage.getName() +
" Health: " + mage.getHealth());
        System.out.println(archer.getName()
+ " Health
public class Main {
    public static void main(String[] args) {
        Warrior warrior = new
Warrior("Aragorn", 100, 5);
```

```
        Mage mage = new Mage("Gandalf",
80, 7);
        Archer archer = new Archer("Legolas",
90, 6);
        warrior.attack(mage);
        warrior.defend();
        System.out.println(warrior.getName()
+ " has " + warrior.getHealth() + " health
left.");
        mage.attack(archer);
        mage.defend();
        System.out.println(mage.getName() +
" has " + mage.getHealth() + " health
left.");
        archer.attack(warrior);
        archer.defend();
```

```
        System.out.println(archer.getName() + " has " + archer.getHealth() + " health left.");
    }
}
```



3.

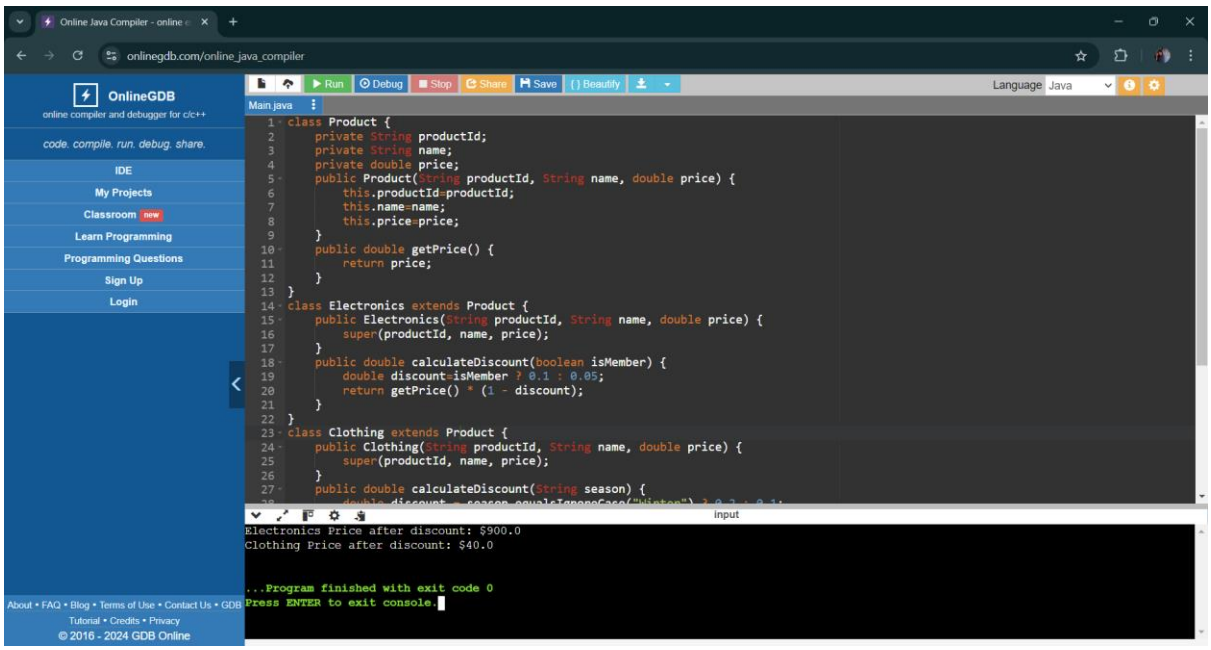
```
class Product {
    private String productId;
    private String name;
    private double price;
```

```
    public Product(String productId, String
name, double price) {
        this.productId = productId;
        this.name = name;
        this.price = price;
    }

    public double getPrice() {
        return price;
    }
}

class Electronics extends Product {
    public Electronics(String productId,
String name, double price) {
        super(productId, name, price);
    }
```

```
    public double
calculateDiscount(boolean isMember) {
        double discount = isMember ? 0.1 :
0.05;
        return getPrice() * (1 - discount);
    }
}

class Clothing extends Product {
    public Clothing(String productId, String
name, double price) {
        super(productId, name, price);
    }

    public double calculateDiscount(String
season) {
```

```
        double discount =
season.equalsIgnoreCase("Winter") ? 0.2 :
0.1;
        return getPrice() * (1 - discount);
    }
}
public class Main {
    public static void main(String[] args) {
        Electronics laptop = new
Electronics("E001", "Laptop", 1000);
        double laptopPriceAfterDiscount =
laptop.calculateDiscount(true);
        System.out.println("Electronics Price
after discount: $" +
laptopPriceAfterDiscount);
        Clothing shirt = new Clothing("C001",
"Shirt", 50);
```

```
        double shirtPriceAfterDiscount = shirt.calculateDiscount("Winter");
        System.out.println("Clothing Price after discount: $" + shirtPriceAfterDiscount);
    }
}
```



4.

```
class LibraryItem {
    private String title;
    private String author;
```

```
    private int year;

    public LibraryItem(String title, String
author, int year) {
        this.title = title;
        this.author = author;
        this.year = year;
    }

    public void checkOut() {
        System.out.println(title + " checked
out.");
    }

    public void checkIn() {
        System.out.println(title + " checked
in.");
```

```
    }

    public void displayInfo() {
        System.out.println("Title: " + title);
        System.out.println("Author: " +
author);
        System.out.println("Year: " + year);
    }
}

class Book extends LibraryItem {
    public Book(String title, String author,
int year) {
        super(title, author, year);
    }

    @Override
```

```
    public void displayInfo() {
        super.displayInfo();
        System.out.println("Type: Book");
    }
}

class DVD extends LibraryItem {
    public DVD(String title, String author, int
year) {
        super(title, author, year);
    }

    @Override
    public void displayInfo() {
        super.displayInfo();
        System.out.println("Type: DVD");
```

```
    }
}
public class Main {
    public static void main(String[] args) {
            Book book = new Book("1984",
"George Orwell", 1949);
        book.checkOut();
        book.displayInfo();
        book.checkIn();
        DVD dvd = new DVD("The Matrix",
"The Wachowskis", 1999);
        dvd.checkOut();
        dvd.displayInfo();
        dvd.checkIn();
    }
}
```

Online Java Compiler - online e

onlinegdb.com/online_java_compiler

OnlineGDB

online compiler and debugger for c/c++

code. compile. run. debug. share.

IDE

My Projects

Classroom new

Learn Programming

Programming Questions

Sign Up

Login

Run | Debug | Stop | Share | Save | { } Beautify | Language Java

Main.java

```java
class LibraryItem {
    private String title;
    private String author;
    private int year;
    public LibraryItem(String title, String author, int year) {
        this.title = title;
        this.author = author;
        this.year = year;
    }

    public void checkOut() {
        System.out.println(title + " checked out.");
    }

    public void checkIn() {
        System.out.println(title + " checked in.");
    }
    public void displayInfo() {
        System.out.println("Title: " + title);
        System.out.println("Author: " + author);
        System.out.println("Year: " + year);
    }
}
class Book extends LibraryItem {
    public Book(String title, String author, int year) {
        super(title, author, year);
    }
```

Input

```
Author: The Wachowskis
Year: 1999
Type: DVD
The Matrix checked in.


...Program finished with exit code 0
Press ENTER to exit console.
```

About • FAQ • Blog • Terms of Use • Contact Us • GDB Tutorial • Credits • Privacy