**ANSWERS**

**What term describes a reserved area in the memory of a computer program that can store input or output data?**

* a) Memory block
* b) Variable
* c) Data hub
* d) Storage slot
* **Answer:** b) Variable

**Which operator is used to assign a value to a variable in Java?**

* a) =
* b) +
* c) -
* d) \*
* **Answer:** a) =

**Which of the following Java methods is used to print a string with formatting?**

* a) System.out.print();
* b) System.out.printf();
* c) System.out.println();
* d) System.out.nextLine();
* **Answer:** b) System.out.printf();

**Which loop statement in Java executes the loop without indexing?**

* a) for loop
* b) while loop
* c) do-while loop
* d) for-each loop
* **Answer:** d) for-each loop

**What is the purpose of the break statement in Java?**

* a) To skip the current iteration of a loop and continue with the next iteration
* b) To exit from the loop or switch statement
* c) To terminate the program immediately
* d) To restart the loop from the beginning
* **Answer:** b) To exit from the loop or switch statement

**What does the void keyword indicate in a method declaration?**

* a) It indicates that the method does not take any parameters
* b) It indicates that the method does not return any value
* c) It indicates that the method is static
* d) It indicates that the method is public
* **Answer:** b) It indicates that the method does not return any value

**Which access modifier is placed automatically?**

* a) public
* b) private
* c) protected
* d) default
* **Answer:** d) default

**What does data hiding refer to in the context of encapsulation?**

* a) Making all class data public
* b) Making the attribute and functions private
* c) Allowing unrestricted access to class data
* d) Encapsulating class data in a separate class
* **Answer:** b) Making the attribute and functions private

**What is the purpose of the finally block in a Java try-catch statement?**

* a) To handle exceptions
* b) To execute code regardless of exceptions
* c) To skip the rest of the code
* d) To terminate the program
* **Answer:** b) To execute code regardless of exceptions

**Which Java keyword is used to create an instance of a class?**

* a) new
* b) create
* c) instance
* d) object
* **Answer:** a) new

**What is the concept of inheritance in OOP?**

* a) Creating a new class from an existing class
* b) Creating multiple objects from a single class
* c) Creating a class with multiple constructors
* d) Creating a class with multiple methods
* **Answer:** a) Creating a new class from an existing class

**Which Java method is used to convert a string to uppercase?**

* a) toLowerCase()
* b) toUpperCase()
* c) toString()
* d) substring()
* **Answer:** b) toUpperCase()

**What is the purpose of the super keyword in Java?**

* a) To access the parent class
* b) To access the child class
* c) To access the current class
* d) To access the grandparent class
* **Answer:** a) To access the parent class

**Which Java operator is used to check if two objects are equal?**

* a) ==
* b) !=
* c) ><
* d) !==
* **Answer:** a) ==

**What is the concept of polymorphism in OOP?**

* a) Method overriding
* b) Method overloading
* c) Method hiding
* d) Method encapsulation
* **Answer:** a) Method overriding

**What is the purpose of “this” keyword in Java?**

* a) To access the current object
* b) To access the parent object
* c) To access the child object
* d) To access the grandparent object
* **Answer:** a) To access the current object

**Which Java exception is thrown when a null value is encountered?**

* a) NullPointerException
* b) NullReferenceException
* c) NullValueException
* d) NullObjectException
* **Answer:** a) NullPointerException

**What is the concept of abstraction in OOP?**

* a) Hiding implementation details
* b) Showing implementation details
* c) Creating multiple objects
* d) Creating multiple classes
* **Answer:** a) Hiding implementation details

**Which Java keyword is used to declare an abstract class?**

* a) abstract
* b) interface
* c) class
* d) extends
* **Answer:** a) abstract

**What is the concept of encapsulation in OOP?**

* a) Hiding data and methods
* b) Showing data and methods
* c) Creating multiple objects
* d) Creating multiple classes
* **Answer:** a) Hiding data and methods

**Which Java keyword is used to apply/make use of an interface?**

* a) interface
* b) abstract
* c) implements
* d) extends
* **Answer:** c) implements

**Which Java exception is thrown when a program attempts to access an array with an invalid index?**

* a) ArrayIndexOutOfBoundsException
* b) NullPointerException
* IndexOutOfBoundsException
* BoundsException
* **Answer:** a) ArrayIndexOutOfBoundsException

**Which Java exception is thrown when an array is accessed with an illegal index?**

* a) ArrayIndexOutOfBoundsException
* b) NullPointerException
* c) IndexOutOfBoundsException
* d) BoundsException
* **Answer:** c) IndexOutOfBoundsException

**Which Java keyword is used to declare a static method?**

* a) static
* b) public
* c) private
* d) final
* **Answer:** a) static

**What is the concept of inheritance in OOP?**

* a) Creating a new class from an existing class
* b) Creating multiple objects from a single class
* c) Creating a class with multiple constructors
* d) Creating a class with multiple methods
* **Answer:a**

**Which Java keyword is used to declare an interface?**

* a) interface
* b) abstract
* c) class
* d) extends
* **Answer:** a) Interface

**What is the purpose of the continue statement in Java?**

* a) To skip a loop iteration
* b) To exit a loop
* c) To continue a loop iteration
* d) To restart a loop
* **Answer:** c) to continue a loop iteration

**Which Java exception is thrown when a program attempts to divide by zero?**

* **a) ArithmeticException**
* b) NullPointerException
* c) DivideByZeroException
* d) MathException
* **Answer:** a) ArithemeticException

**SECTION B**

**1.Define inheritance in java. Describe a real-life scenario on how it can be defined.**

Inheritance is a fundamental concept in object-oriented programming (OOP) that allows one class to inherit the properties and methods of another class. The class that inherits the properties is called the **subclass** and the class whose properties are inherited is called the **superclass**. Consider a real-life scenario of a company with different types of employees. Let's define a base class called Employee and two derived classes FullTimeEmployee and PartTimeEmployee that inherit from Employee.

**2.Define abstraction in java. Describe a real-life scenario on how it can be defined.**

Abstraction is a fundamental concept in object-oriented programming (OOP) that focuses on exposing only the necessary details and functionalities of an object while hiding the implementation details. It allows developers to define abstract classes and interfaces to create a blueprint for other classes to follow. Consider a scenario of a payment system in an online shopping platform. There are different types of payment methods such as credit card, PayPal, and bank transfer. By using abstraction, we can focus on the essential features of the payment system while hiding the complex implementation details.

**3.Define polymorphism in java. Describe a real life scenario on how it can be defined.**

Polymorphism is a core concept in object-oriented programming (OOP) that refers to the ability of a single interface to represent different underlying forms (data types). In Java, polymorphism allows methods to do different things based on the object it is acting upon, even though they share the same name. Polymorphism is mainly of two types:

1. Compile-time Polymorphism (Method Overloading): This is achieved by having multiple methods with the same name but different parameter lists within the same class.
2. Runtime Polymorphism (Method Overriding): This is achieved when a subclass has a method with the same name and signature as a method in its superclass.

Imagine a zoo where different animals make different sounds.

* **Parent Class: Animal**
  + Method: makeSound()
* **Child Classes: Dog, Cat, Cow**
  + **Dog**: makeSound() -> "Bark"
  + **Cat**: makeSound() -> "Meow"
  + **Cow**: makeSound() -> "Moo"

Here, we can call the makeSound() method on different animal objects, and each will produce its unique sound.

**4.Define encapsulation in java . Describe a real-life scenario on how it can be defined as access modifier.**

**Encapsulation** is one of the fundamental principles of OOP which is the mechanism of wrapping the data and the code acting on the data (methods) together as a single unit. In encapsulation, the variables of a class are hidden from other classes and can be accessed only through the methods of their current class. This helps to protect the internal state of the object and only allows modifications through well-defined interfaces.

Imagine a system where employee details need to be managed, but some details should be protected from direct access.

* Class: Employee
  + Private Variables: name, salary
  + Public Methods: getName(), setName(String name), getSalary(), setSalary(double salary)

Encapsulation ensures that the name and salary variables are protected from direct access and modification, maintaining the integrity of the employee's data by using well-defined methods to access and update these fields.

**5. Explain the term "IS-A relationship" in the context of inheritance in java.**

In the context of inheritance in Java, the term "IS-A relationship" refers to a hierarchical connection between classes where one class (the subclass or derived class) is a specialized version of another class (the superclass or base class). This relationship indicates that the subclass inherits properties and behaviors (fields and methods) from the superclass. Essentially, the subclass "is a" type of the superclass. the extends keyword is used to establish an inheritance relationship. When a class extends another class, it inherits all non-private fields and methods from the superclass. For example, if class Dog extends class Animal, a Dog object can be referred to as an Animal object.

**6. Differentiate between method hiding and method overriding.**

| **Feature** | **Method Overriding** | **Method Hiding** |
| --- | --- | --- |
| **Method Type** | Instance methods | Static methods |
| **Polymorphism** | Supports runtime polymorphism | Does not support runtime polymorphism |
| **Determination** | Determined by the actual object type at runtime | Determined by the reference type at compile time |
| **Annotation** | @Override annotation used | No specific annotation |
| **Purpose** | To provide specific implementation in subclass | To define a new static method in subclass |

**7. Differentiate between abstract class and interfaces.**

|  |  |  |
| --- | --- | --- |
| **Feature** | **abstract class** | **interfaces** |
| |  | | --- | | **Definition** | | A class that cannot be instantiated and can contain both abstract methods (without implementation) and concrete methods (with implementation). | A contract that defines a set of methods that implementing classes must provide, but without any method implementations. |
| |  | | --- | | **Purpose** |  |  | | --- | |  | | To provide a common base class with shared code and abstract methods for subclasses. | To define a contract that multiple classes can implement, ensuring they all provide certain methods. |
| |  | | --- | | **Method Implementation** | | |  | | --- | | Can have both abstract and concrete methods. | | |  | | --- | | Can only have abstract methods (in some languages like Java 8+, can have default and static methods). | |
| |  | | --- | | **Inheritance** | | |  | | --- | | A class can inherit only one abstract class due to single inheritance. | | |  | | --- | | A class can implement multiple interfaces, allowing for multiple inheritance. | |
| |  | | --- | | **Examples** |  |  | | --- | |  | | abstract class Shape { ... } | interface Drawable { ... } |

**8. Differentiate between public and private access modifiers.**

|  |  |  |
| --- | --- | --- |
| **Feature** | **Private access modifier** | **Public access modifier** |
| Accessibility | The member (class, method, field, etc.) is accessible from any other class, anywhere in the application. | The member is accessible only within the class it is defined. |
| Scope | The member is accessible only within the class it is defined. | |  | | --- | | Local (available only within the same class). |  |  | | --- | |  | |
| Inheritance | Public members are inherited by subclasses and can be accessed through inheritance. | Private members are not inherited by subclasses; they are only accessible within the original class. |
| Encapsulation | Offers less encapsulation as the members are exposed to all other classes. | Provides strong encapsulation, keeping the details hidden from other classes. |
| Example | public int age; | private int age; |

**SECTION C**

Write short notes on:

#### **1.Method and Function**

* **Method**: In Java, a method is a block of code within a class that performs a specific task and can be called upon to execute. Methods can return a value and can take parameters.
* **Function**: While the term "function" is often used interchangeably with "method" in other programming languages, in Java, functions are typically referred to as methods since they are always associated with objects (except for static methods).

#### **2.Checked Exception**

* **Checked Exception**: Checked exceptions are exceptions that are checked at compile-time. These exceptions must be either caught or declared in the method signature using the throws keyword. Examples include IOException, SQLException, etc.

#### **3.Unchecked Exception**

* **Unchecked Exception**: Unchecked exceptions are exceptions that occur at runtime and are not checked at compile-time. They are subclasses of RuntimeException. Examples include NullPointerException, ArrayIndexOutOfBoundsException, etc.

#### **4.Access Modifier**

* **Access Modifier**: Access modifiers in Java determine the visibility and accessibility of classes, methods, and variables. The main access modifiers are:
  + public: Accessible from any other class.
  + protected: Accessible within the same package and by subclasses.
  + default (no modifier): Accessible only within the same package.
  + private: Accessible only within the same class.

#### **5."this" keyword**

* **this** keyword: The this keyword in Java refers to the current instance of the class. It is used to access instance variables, methods, and constructors.

#### **6."super" keyword**

* **super** keyword: The super keyword in Java is used to refer to the immediate parent class object. It is used to access parent class methods and constructors.

#### **7.For loop**

* **For loop**: A for loop in Java is used to iterate over a range of values. It consists of an initialization, a termination condition, and an increment/decrement operation.

java

Copy code

for (int i = 0; i < 10; i++) {

// Code to execute

}

#### **8.While loop**

* **While loop**: A while loop in Java repeatedly executes a block of statements as long as a specified condition is true.

java

Copy code

while (condition) {

// Code to execute

}

#### **9.Do-While loop**

* **Do-While loop**: A do-while loop in Java is similar to a while loop, but it guarantees that the loop body will be executed at least once.

java

Copy code

do {

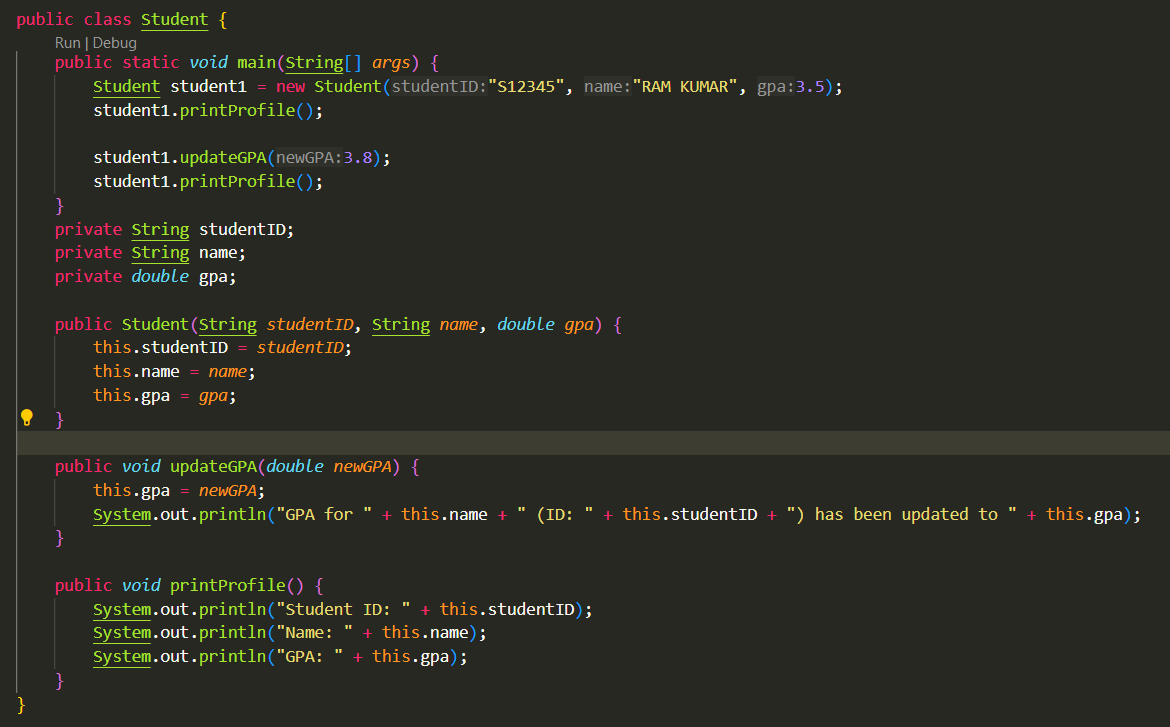
// Code to execute

} while (condition);

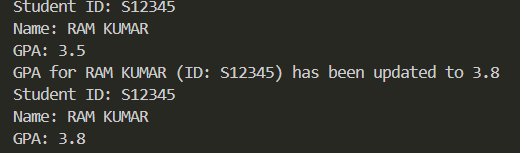
\

**SECTION D**

1. Create a Student class to simulate a student's profile. The class should have the following attributes: student ID, name, and GPA. Include methods to updateGPA() and printProfile(). Implement functionality to create new student profiles, update GPAs, and print student profiles.



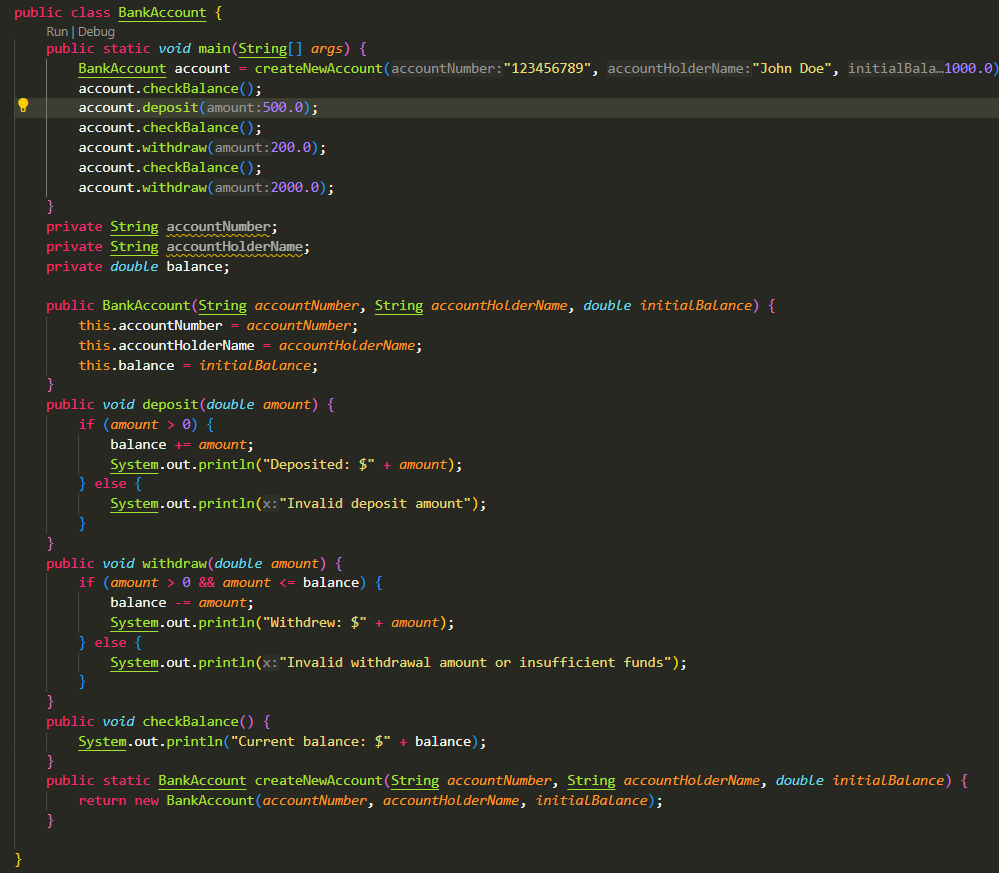
**Output:**



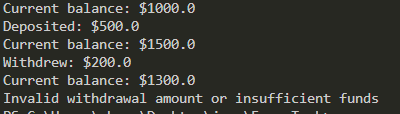
1. Create a BankAccount class to simulate a bank account. The class should have the following attributes: account number, account holder name, and balance. Include methods to deposit(), withdraw(), and checkBalance(). Implement functionality to:

-Create new accounts -Deposit money into the account

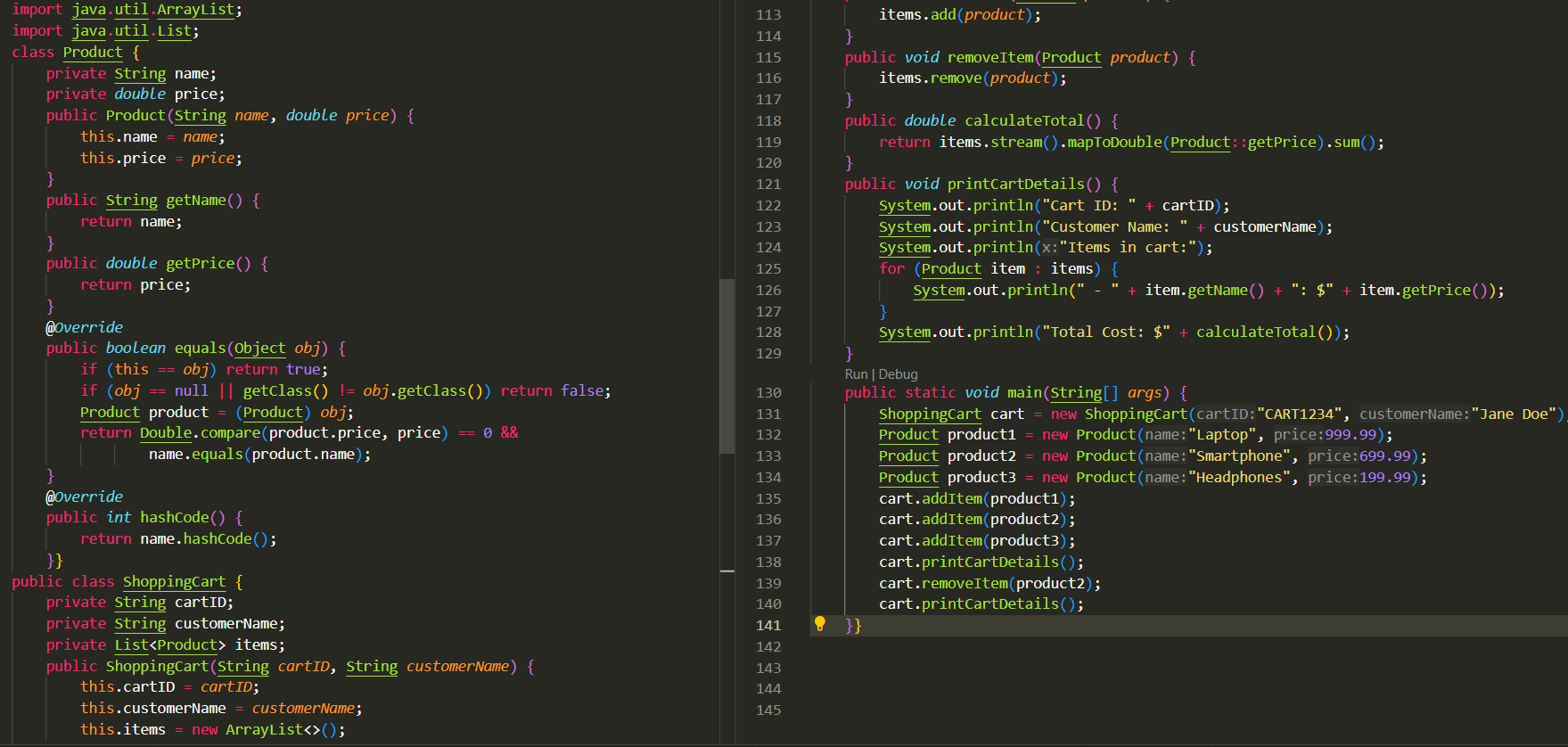
-Withdraw money -Check the account balance.

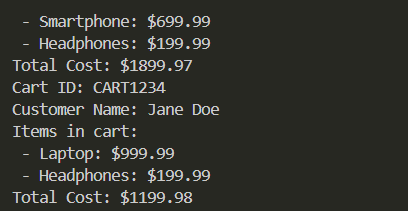


Output:



3.Create a ShoppingCart class to simulate an online shopping cart. The class should have the following attributes: cart ID, customer name, and items (a list of products). Include methods to addItem(), removeItem(), and calculateTotal(). Implement functionality to create new shopping carts, add/remove items, and calculate the total cost of items in the cart.

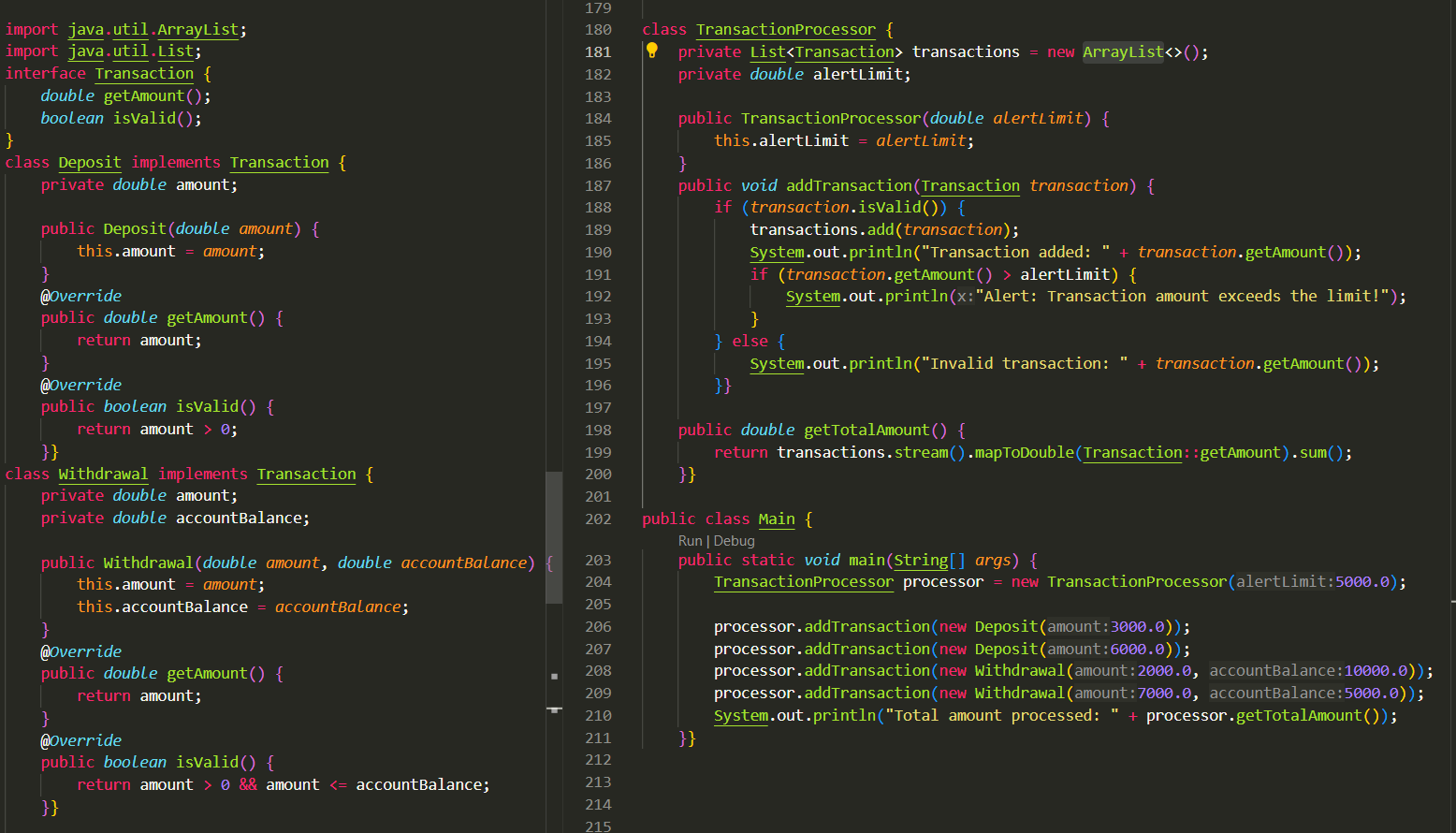




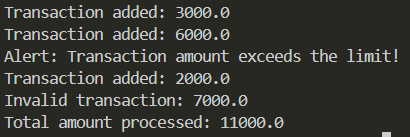
4. Design and implement a Java program that simulates a financial transaction system. Your system should be capable of processing different types of transactions, such as deposits and withdrawals, and alerting if any transactions exceed certain limits. Use interfaces to define common behaviors for transactions and implement these behaviors in different transaction classes.

Create an interface named Transaction with at least two methods:

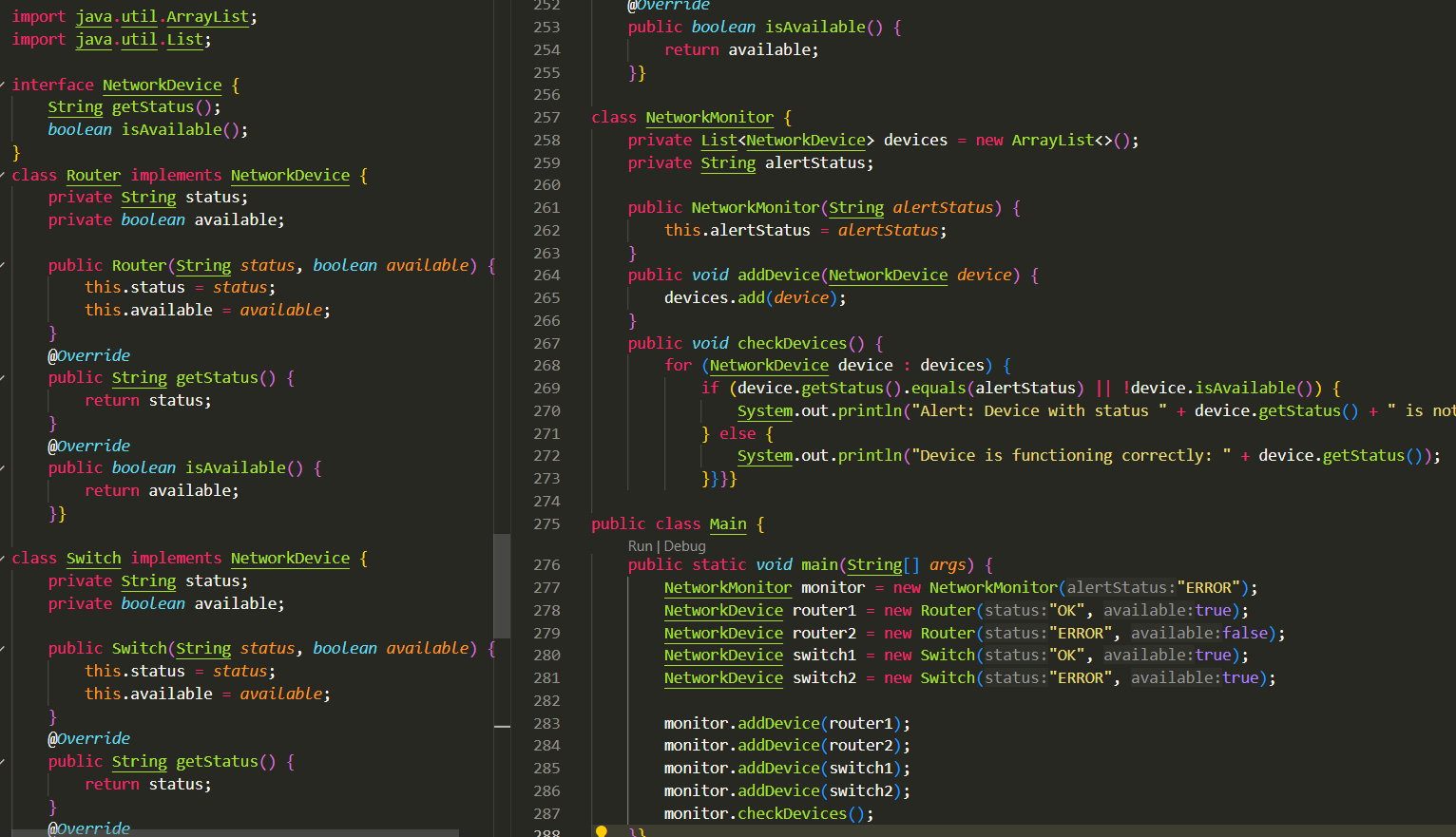
* getAmount() which returns the amount of the transaction as a double
* isValid() which returns a boolean indicating if the transaction is valid based on certain criteria.

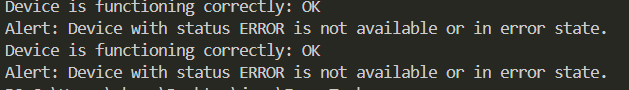


Output:



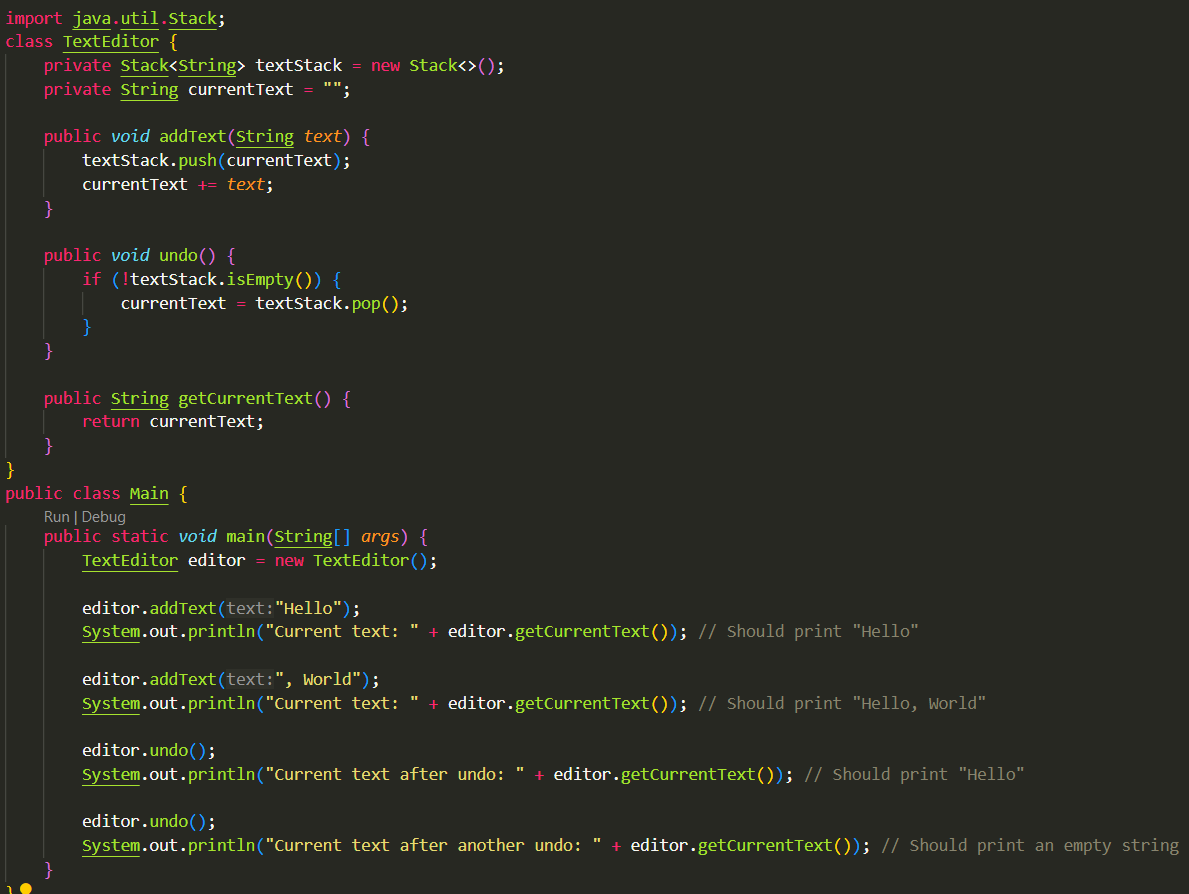
5.Design and implement a Java program that simulates a network monitoring system. Your system should be capable of monitoring different types of network devices, such as routers and switches, and alerting if any devices are experiencing errors or downtime. Use interfaces to define common behaviors for network devices and implement these behaviors in different device classes. Create an interface named NetworkDevice with at least two methods: 1.getStatus() which returns a String indicating the current status of the device 2.isAvailable() which returns a boolean indicating if the device is currently available and functioning correctly Implement the NetworkDevice interface in at least three separate classes, Router and Switch. Each class should have a constructor that sets specific properties for the device and specific logic to determine its status and availability.

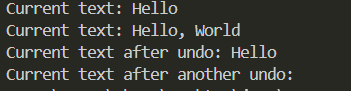
****



6.Design and implement a Java program that simulates the undo feature of a text editor using a stack. The program should support adding text to the editor and undoing changes. The stack should store strings, each representing a version of the text. Implement methods for:

* adding text to the editor: addText(String text)
* undoing changes: undo()
* getting the current text: getCurrentText()





**7.**Design and implement a simple Java application that prompts the user to enter a string within a specified length, for example, between 5 and 20 characters. Your application should throw and handle a custom exception if the user enters a string outside this length.

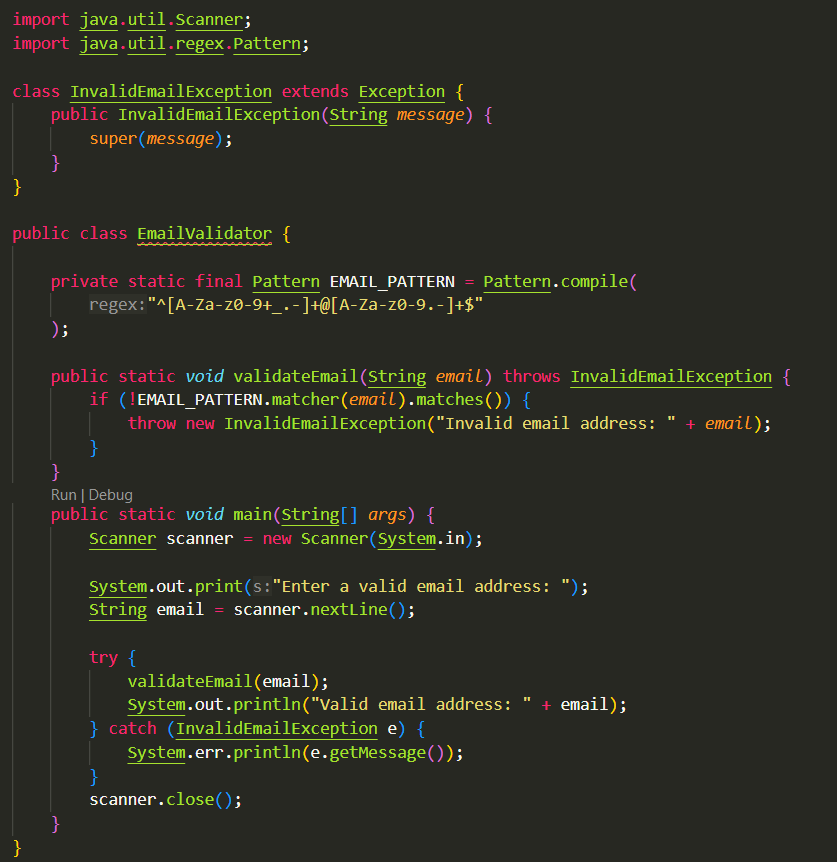
* InvalidStringLengthException: A custom exception that is thrown when the user enters a string with a length outside the valid range (5 to 20 characters).





8.Design and implement a simple Java application that prompts the user to enter a valid email address. Your application should throw and handle a custom exception if the user enters an invalid email address.

-InvalidEmailException: A custom exception that is thrown when the user enters an email address that does not match a valid email pattern (e.g., it should contain '@' and '.' characters).



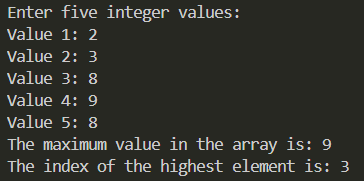




9. Write a program that prompts the user to enter five integer values and reads them into an array. The program should consist of the following methods:

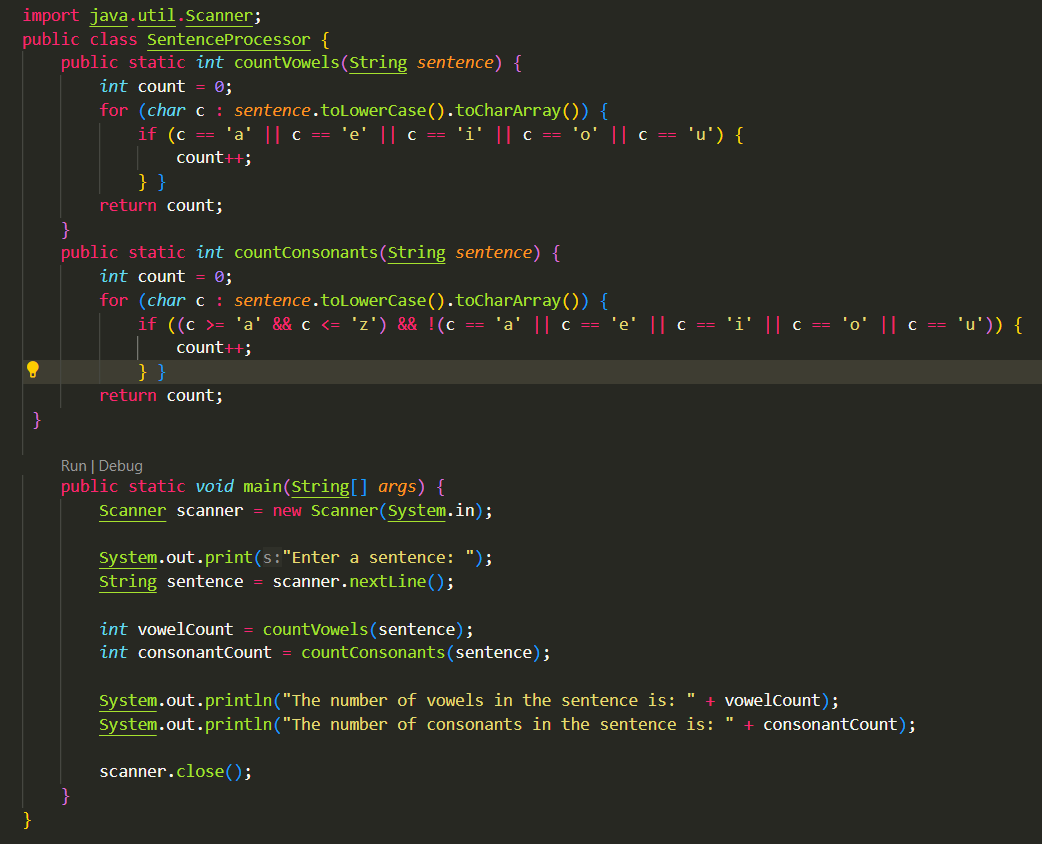
* + a method named max that reads an array of integer values and returns the maximum value
  + a method named indexOfHighest that returns the index of the highest element in an array of integers. If more than one such element exists, return the smallest index.
  + Invoke the above methods and demonstrate them in the program.

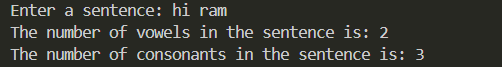




**10.**  Write a program that prompts the user to enter a sentence and reads it into a string. The program should consist of the following methods:

* + a method named countVowels that reads a string and returns the number of vowels in the string
  + a method named countConsonants that reads a string and returns the number of consonants in the string
  + Invoke the above methods and demonstrate them in the program.





Q.N.1.

The following is the correct syntax of while loop

while(int index = 0; index &lt;=5; index ++){

// body

}

(1 marks)

(Choose one option)

True

False

Q.N.2.

Do while checks the expression after the loop statement is executed

(1 marks)

(Choose one option)

False

True

Q.N.3.

Consider the following data.

double length = 20;

double breadth = 10;

Write a function that takes length and breadth and calculates and returns the perimeter of a

rectangle.

Q.N.4.

Write a program that displays the following table (note that 1 mile is 1.609 kilometers):

Miles Kilometers

1 1.609

2 3.218

… ….

9 14.481

10 16.090

Q.N.5.

Write short notes on Control Flow in Java / Java Control Statements. What are the 3 types of Java

Control Statements?

Q.N.6.

In Java there are basic of 6 operators, list down at least 5 of them with example.

Q.N.7.

Is the following syntax correct to make integer.

int intValue = &quot;12&quot;;

(1 marks)

(Choose one option)

False

True

Q.N.8.

Which is the correct way of making main function.

(1 marks)

(Choose one option)

public static void main(String[] args){}

void main(){}

private void static main(String[] args){}

public void static main(String[] args){}

Q.N.9.

Java is a/an \_\_\_\_\_\_\_\_\_\_\_\_\_\_ language

(1 marks)

(Choose one option)

Object Oriented

Scripting

Machine

Assembly

Q.N.10.

Which of the following is not a type of variable in Java?

(1 marks)

(Choose one option)

Global

Local

Instance

Static

Q.N.11.

Use a while loop for the following,

int index = 1;

Loop from 1 - 20

Increament the loop 2

If the value of index is 5, 7, 9 skip the iteration

If the value of index is 17 exit the loop

Print the index

Output:

1

3

11

13

15

Q.N.12.

What is the output of the following.

System.out.printf(&quot;Hello %s and %s, are you %d siblings?&quot;, &quot;John&quot;, &quot;Mark&quot;, 2);

(1 marks)

(Choose one option)

Syntax Error

Hello Mark and John, are you 2 siblings?

Hello John and Mark, are you 2 siblings?

Hello 2 and 2, are you 2 siblings?

Q.N.13.

What is the difference of prefix and postfix operator?

For eg:

++5;

5++;

Q.N.13.

What is the difference of prefix and postfix operator?

For eg:

++5;

5++;

Q.N.13.

What is the difference of prefix and postfix operator?

For eg:

++5;

5++;

Q.N.16.

JAVA is currently owned by which company?

(1 marks)

(Choose one option)

Oracle Corporation

SUN Corporation

Google

IBM

Q.N.17.

Java is an interpreted language. (No compiler)

(1 marks)

(Choose one option)

True

False