20CS2035L–Object Oriented Programming (Lab)–B11 REG.NO:URK22AI1048

|  |  |
| --- | --- |
| **Ex. No. 7** | **EXCEPTION HANDLING** |
| **Date of Exercise** | 21-09-2023 |

# Aim :

To write a java program to create a menu driven program in java to perform the operations on an integer queue and to create custom expetations to deal.

# Procedure :

* 1. Start the program.
  2. Create the required classes.
  3. Create custom expectation class.
  4. Insert queue operation to understand full or not.
  5. Create queue operable interface and implement .
  6. Give the input and get the output.
  7. Stop the program.

# Program :

import java.util.LinkedList; import java.util.Queue; import java.util.Scanner;

class NonIntegerValueException extends Exception { public NonIntegerValueException(String message) {

super(message);

}

}

class QueueFullException extends Exception { public QueueFullException(String message) {

1

20CS2035L–Object Oriented Programming (Lab)–B11 REG.NO:URK22AI1048

super(message);

}

}

class QueueEmptyException extends Exception { public QueueEmptyException(String message) {

super(message);

}

}

interface QueueOperations {

void insert(int value) throws QueueFullException;

int remove() throws QueueEmptyException, NonIntegerValueException; void display();

}

class IntegerQueue implements QueueOperations { private Queue<Integer> queue;

private int maxSize;

public IntegerQueue(int maxSize) { this.maxSize = maxSize; this.queue = new LinkedList<>();

}

@Override

public void insert(int value) throws QueueFullException { if (queue.size() >= maxSize) {

throw new QueueFullException("Queue is full. Cannot insert.");

}

queue.add(value);

}

@Override

public int remove() throws QueueEmptyException, NonIntegerValueException { if (queue.isEmpty()) {

throw new QueueEmptyException("Queue is empty. Cannot remove.");

}

int value = queue.poll();

if (value < Integer.MIN\_VALUE || value > Integer.MAX\_VALUE) {

throw new NonIntegerValueException("Non-integer value found in the queue.");

}

return value;

}

2

20CS2035L–Object Oriented Programming (Lab)–B11 REG.NO:URK22AI1048

@Override

public void display() {

System.out.println("Queue elements: " + queue);

}

}

public class Main {

public static void main(String[] args) { Scanner scanner = new Scanner(System.in);

System.out.print("Enter the maximum size of the queue: "); int maxSize = scanner.nextInt();

QueueOperations integerQueue = new IntegerQueue(maxSize);

while (true) { System.out.println("\nMenu:"); System.out.println("1. Insert"); System.out.println("2. Remove"); System.out.println("3. Display"); System.out.println("4. Quit"); System.out.print("Enter your choice: "); int choice = scanner.nextInt();

try {

switch (choice) { case 1:

System.out.print("Enter an integer to insert: "); int valueToInsert = scanner.nextInt(); integerQueue.insert(valueToInsert);

break; case 2:

int removedValue = integerQueue.remove(); System.out.println("Removed element: " + removedValue); break;

case 3:

integerQueue.display(); break;

case 4:

System.out.println("Exiting program."); scanner.close();

System.exit(0);

3

20CS2035L–Object Oriented Programming (Lab)–B11 REG.NO:URK22AI1048

default:

System.out.println("Invalid choice. Please try again.");

}

} catch (QueueFullException | QueueEmptyException | NonIntegerValueException e) { System.err.println("Error: " + e.getMessage());

}

}

}

}

# Output Screenshot :

**Result :**

The above program has been successfully executed and verified.

4

20CS2035L–Object Oriented Programming (Lab)–B11 REG.NO:URK22AI1048

# Aim :

To write a java program to create a menu drive to automate the ATM operations by demonstrating the concepts of interfaces.

# Procedure :

* 1. Start the program.
  2. Create a class called Invaliedpinexpen
  3. Use the superkeys in the program .
  4. Use the words catch,try,final,etc in the program.
  5. End the program.

# Program :

import java.util.Scanner;

class InvalidPinException extends Exception { public InvalidPinException(String message) {

super(message);

}

}

class InsufficientBalanceException extends Exception { public InsufficientBalanceException(String message) {

super(message);

}

}

interface ATMOperations { void checkBalance();

void deposit(double amount);

void withdraw(double amount) throws InsufficientBalanceException;

}

class ATM implements ATMOperations { private double balance;

private int pinAttempts;

public ATM(double initialBalance) { this.balance = initialBalance;

5

20CS2035L–Object Oriented Programming (Lab)–B11 REG.NO:URK22AI1048

this.pinAttempts = 0;

}

@Override

public void checkBalance() { System.out.println("Current Balance: $" + balance);

}

@Override

public void deposit(double amount) { balance += amount;

System.out.println("$" + amount + " deposited successfully.");

}

@Override

public void withdraw(double amount) throws InsufficientBalanceException { if (balance >= amount) {

balance -= amount;

System.out.println("$" + amount + " withdrawn successfully.");

} else {

throw new InsufficientBalanceException("Insufficient balance. Unable to withdraw $" + amount);

}

}

public boolean validatePin(int enteredPin) throws InvalidPinException { int correctPin = 1234; // Replace with your actual PIN

if (enteredPin == correctPin) {

pinAttempts = 0; // Reset PIN attempts on successful validation return true;

} else {

pinAttempts++;

if (pinAttempts >= 3) {

throw new InvalidPinException("Invalid PIN entered 3 times. Card is blocked.");

}

throw new InvalidPinException("Invalid PIN. Please try again.");

}

}

}

public class Main {

public static void main(String[] args) {

6

20CS2035L–Object Oriented Programming (Lab)–B11 REG.NO:URK22AI1048

ATM atm = new ATM(1000.0); // Initial balance Scanner scanner = new Scanner(System.in);

while (true) {

System.out.println("1. Check Balance"); System.out.println("2. Deposit"); System.out.println("3. Withdraw"); System.out.println("4. Exit"); System.out.print("Select an option: ");

int choice = scanner.nextInt(); switch (choice) {

case 1:

atm.checkBalance(); break;

case 2:

System.out.print("Enter the deposit amount: $"); double depositAmount = scanner.nextDouble(); atm.deposit(depositAmount);

break; case 3:

try {

System.out.print("Enter your PIN: "); int enteredPin = scanner.nextInt();

if (atm.validatePin(enteredPin)) { System.out.print("Enter the withdrawal amount: $"); double withdrawAmount = scanner.nextDouble(); atm.withdraw(withdrawAmount);

}

} catch (InvalidPinException e) { System.out.println(e.getMessage());

} catch (InsufficientBalanceException e) { System.out.println(e.getMessage());

}

break; case 4:

System.out.println("Thank you for using the ATM. Goodbye!"); System.exit(0);

default:

7

8

20CS2035L–Object Oriented Programming (Lab)–B11 REG.NO:URK22AI1048

System.out.println("Invalid option. Please try again."); break;

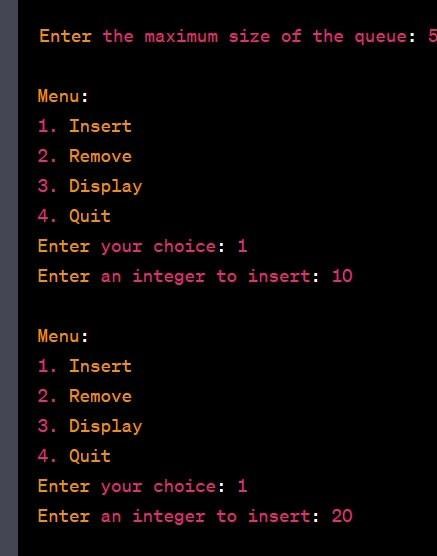
}

}

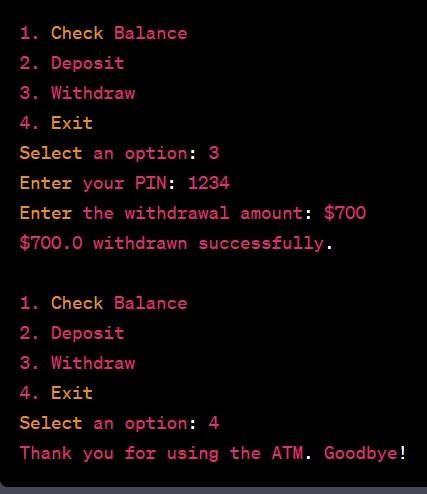
}

}

# Output Screenshot :



20CS2035L–Object Oriented Programming (Lab)–B11 REG.NO:URK22AI1048



**Result :**

The above program has been successfully executed and verified.

9