LAB 8 ANP-C7781

Solve following questions:

1. Write a program that tries to access an element outside the bounds of an array and handles the ArrayIndexOutOfBoundsException by printing a user-friendly message.

Program :-

package Day07;

public class ArrayAccessExample {

public static void main(String[] args) {

// Create an integer array with elements from 1 to 5

int[] numbers = { 1, 2, 3, 4, 5 };

try {

// Access an element at index 2 and print it

System.***out***.println("Accessing element at index 2: " + numbers[2]);

// Try to access an element at index 10, which is outside the array bounds

System.***out***.println("Accessing element at index 10: " + numbers[10]); // This will cause an exception

} catch (ArrayIndexOutOfBoundsException e) {

// If an ArrayIndexOutOfBoundsException occurs, print a user-friendly message

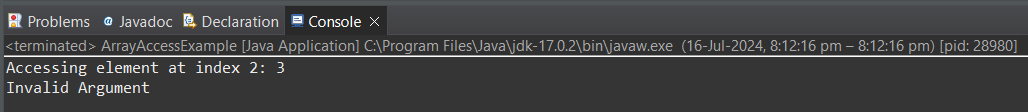
System.***out***.println("Invalid Argument");

}

}

}

Output :-



1. Write a program that attempts to divide a number by zero and handles the ArithmeticException by printing a message that division by zero is not allowed.

Program :-

package Day07;

public class DivisionByZeroExample {

public static void main(String[] args) {

// Declare the numerator and denominator variables

int numer = 10;

int denom = 0;

try {

// Attempt to divide the numerator by the denominator, which is 0

int result = numer / denom; // This will cause an ArithmeticException

System.***out***.println("Result: " + result);

} catch (ArithmeticException e) {

// If an ArithmeticException occurs (due to division by zero), print an error

// message

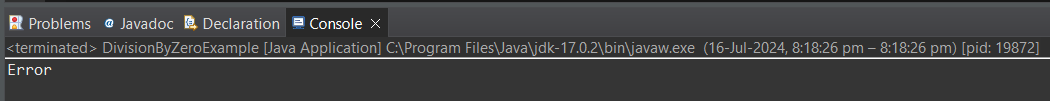
System.***out***.println("Error");

}

}

}

Output :-



1. Write a Java program that reads an integer input from the user and throws an IllegalArgumentException if the input is negative. Display an appropriate message when the exception is caught.

Program :-

package Day07;

import java.util.Scanner;

public class NegativeInputExample {

public static void main(String[] args) {

// Create a Scanner object to read input from the user

Scanner scanner = new Scanner(System.***in***);

try {

// Prompt the user to enter a positive integer

System.***out***.print("Enter a positive integer: ");

int userInput = scanner.nextInt();

// Check if the user input is negative

if (userInput < 0) {

// If the input is negative, throw an IllegalArgumentException with a custom

// error message

throw new IllegalArgumentException("Error: Negative input is not allowed.");

}

// If the input is positive, print the user's input

System.***out***.println("You entered: " + userInput);

} catch (IllegalArgumentException e) {

// If an IllegalArgumentException is thrown, catch it and print the error

// message

System.***out***.println(e.getMessage());

} finally {

// Close the Scanner resource

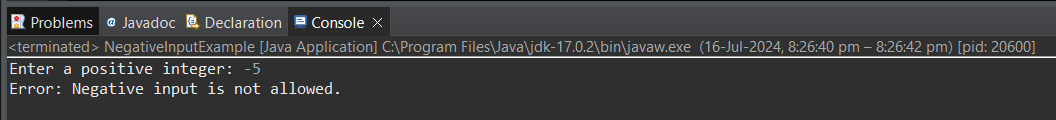
scanner.close();

}

}

}

Output :-



1. Define a custom exception called InvalidAgeException. Write a Java program that throws this exception if the age provided is less than 18. Handle the exception and display an appropriate message.

Program :-

package Day07;

import java.util.Scanner;

public class CustomExceptionExample {

public static void main(String[] args) {

// Create a Scanner object to read input from the user

Scanner scanner = new Scanner(System.***in***);

try {

// Prompt the user to enter their age

System.***out***.print("Enter your age: ");

int age = scanner.nextInt();

// Check if the age is less than 18

if (age < 18) {

// If the age is less than 18, throw a custom InvalidAgeException with a custom

// error message

throw new InvalidAgeException("Error: You are not eligible to enter.");

}

// If the age is 18 or above, print a message indicating eligibility

System.***out***.println("You are eligible to enter.");

} catch (InvalidAgeException e) {

// Catch the custom InvalidAgeException and print the error message

System.***out***.println(e.getMessage());

} finally {

// Close the Scanner resource

scanner.close();

}

}

}

// Define a custom exception class called InvalidAgeException

class InvalidAgeException extends Exception {

// Provide a constructor that takes a custom error message

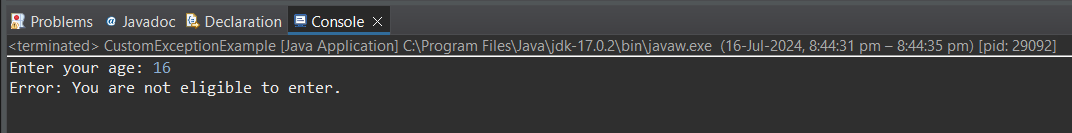
public InvalidAgeException(String message) {

super(message);

}

}

Output :-



5 Write a Java program that has a method to validate a user's email address. The method should throw a custom exception InvalidEmailException if the email does not contain @ and .. Handle the exception in the main method.

Program :-

package Day07;

import java.util.Scanner;

public class EmailValidationExample {

public static void main(String[] args) {

// Create a Scanner object to read input from the user

Scanner scanner = new Scanner(System.***in***);

try {

// Prompt the user to enter an email address

System.***out***.print("Enter your email address: ");

String email = scanner.nextLine();

// Call the validateEmail() method to validate the email address

*validateEmail*(email);

// If the email is valid, print a message

System.***out***.println("Valid email address: " + email);

} catch (InvalidEmailException e) {

// If an InvalidEmailException is thrown, catch it and print the error message

System.***out***.println(e.getMessage());

} finally {

// Close the Scanner resource

scanner.close();

}

}

public static void validateEmail(String email) throws InvalidEmailException {

// Check if the email contains the '@' and '.' characters

if (!email.contains("@") || !email.contains(".")) {

// If the email is invalid, throw a custom InvalidEmailException with a custom

// error message

throw new InvalidEmailException("Error: Invalid email address format.");

}

}

}

// Define a custom exception class called InvalidEmailException

class InvalidEmailException extends Exception {

// Provide a constructor that takes a custom error message

public InvalidEmailException(String message) {

super(message);

}

}

Output:-

