

JAVA PROGRAM

Task 4

1. Write a Java program that performs the following operations on a given string: find its length, convert it to uppercase, extract a substring, and replace a character.

program

```
import java.util.Scanner;

public class StringOperations{
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter the original string:");
        String originalString = scanner.nextLine();

        int length = originalString.length();
        System.out.println("Length of the string: " + length);

        String upperCaseString = originalString.toUpperCase();
        System.out.println("Uppercase string: " + upperCaseString);

        System.out.println("Enter the starting index for substring:");
        int startIndex = scanner.nextInt();
        System.out.println("Enter the ending index for substring:");
        int endIndex = scanner.nextInt();
        scanner.nextLine();
        String substring;
        try {
            substring = originalString.substring(startIndex, endIndex);
            System.out.println("Extracted substring: " + substring);
        } catch (StringIndexOutOfBoundsException e) {
            System.out.println("Invalid indices for substring extraction.");
        }

        System.out.println("Enter the character to be replaced:");
```

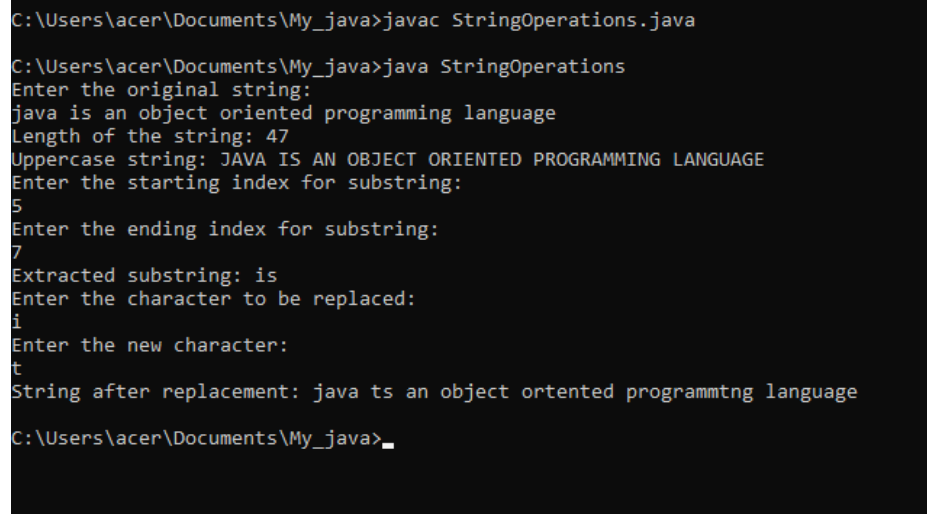
```

        char oldChar = scanner.nextLine().charAt(0);
        System.out.println("Enter the new character:");
        char newChar = scanner.nextLine().charAt(0);
        String replacedString = originalString.replace(oldChar, newChar);
        System.out.println("String after replacement: " + replacedString);

        scanner.close();
    }
}

```

output



```

C:\Users\acer\Documents\My_java>javac StringOperations.java
C:\Users\acer\Documents\My_java>java StringOperations
Enter the original string:
java is an object oriented programming language
Length of the string: 47
Uppercase string: JAVA IS AN OBJECT ORIENTED PROGRAMMING LANGUAGE
Enter the starting index for substring:
5
Enter the ending index for substring:
7
Extracted substring: is
Enter the character to be replaced:
i
Enter the new character:
t
String after replacement: java ts an object ortented programtng language
C:\Users\acer\Documents\My_java>_

```

2. Write a Java program to parse a string into different primitive data types using wrapper class methods like `parseInt`, `parseDouble`, `parseBoolean`, etc., and convert primitive types to strings using `valueOf`

program

```

public class ParseAndConvert {
    public static void main(String[] args) {
        String intString = "1111";
        String doubleString = "22.34";
        String booleanString = "true";
        String floatString = "33.44";
    }
}

```

```

        String byteString = "123";
int intValue = Integer.parseInt(intString);
    System.out.println("Parsed integer value: " + intValue);
    String intToString = String.valueOf(intValue);
    System.out.println("Converted back to string: " + intToString);
    double doubleValue = Double.parseDouble(doubleString);
    System.out.println("Parsed double value: " + doubleValue);
    String doubleToString = String.valueOf(doubleValue);
    System.out.println("Converted back to string: " + doubleToString);
    boolean booleanValue = Boolean.parseBoolean(booleanString);
    System.out.println("Parsed boolean value: " + booleanValue);
    String booleanToString = String.valueOf(booleanValue);
    System.out.println("Converted back to string: " + booleanToString);
    float floatValue = Float.parseFloat(floatString);
    System.out.println("Parsed float value: " + floatValue);
    String floatToString = String.valueOf(floatValue);
    System.out.println("Converted back to string: " + floatToString);
byte byteValue = Byte.parseByte(byteString);
    System.out.println("Parsed byte value: " + byteValue);
    String byteToString = String.valueOf(byteValue);
    System.out.println("Converted back to string: " + byteToString);
    }
}

```

output

```

C:\Users\acer\Documents\My_java>javac ParseAndConvert.java
C:\Users\acer\Documents\My_java>java ParseAndConvert
Parsed integer value: 1111
Converted back to string: 1111
Parsed double value: 22.34
Converted back to string: 22.34
Parsed boolean value: true
Converted back to string: true
Parsed float value: 33.44
Converted back to string: 33.44
Parsed byte value: 123
Converted back to string: 123
C:\Users\acer\Documents\My_java>_

```

3. Write a Java program to sort an array of integers in ascending order using a sorting algorithm of your choice

program

```
import java.util.Scanner;

public class QuickSort {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter the number of elements:");
        int n = scanner.nextInt();
        int[] array = new int[n];
        System.out.println("Enter the elements of the array:");
        for (int i = 0; i < n; i++) {
            array[i] = scanner.nextInt();
        }
        quickSort(array, 0, n - 1);
        System.out.println("Sorted array in ascending order:");
        for (int i : array) {
            System.out.print(i + " ");
        }
    }

    public static void quickSort(int[] array, int low, int high) {
        if (low < high) {
            int pi = partition(array, low, high);
            quickSort(array, low, pi - 1);
            quickSort(array, pi + 1, high);
        }
    }

    public static int partition(int[] array, int low, int high) {
        int pivot = array[high];
        int i = (low - 1);

        for (int j = low; j < high; j++) {
            if (array[j] <= pivot) {
                i++;
                int temp = array[i];
                array[i] = array[j];
                array[j] = temp;
            }
        }

        int temp = array[i + 1];
        array[i + 1] = array[high];
```

```

        array[high] = temp;

        return i + 1;
    }
}

```

output

```

C:\Users\acer>cd C:\Users\acer\Documents\My_java
C:\Users\acer\Documents\My_java>javac QuickSort.java
C:\Users\acer\Documents\My_java>java QuickSort
Enter the number of elements:
6
Enter the elements of the array:
23 43 21 8 9 2
Sorted array in ascending order:
2 8 9 21 23 43
C:\Users\acer\Documents\My_java>

```

4. Use an ArrayList to store the list of books. Each book should have attributes such as title, author, ISBN, and price. Implement functionalities to add new books, remove existing books, and display all books in the library.

program

```

import java.util.ArrayList;

class Book {
    private String title;
    private String author;
    private String isbn;
    private double price;

    public Book(String title, String author, String isbn, double price) {
        this.title = title;
        this.author = author;
    }
}

```

```

        this.isbn = isbn;
        this.price = price;
    }
    public String getTitle() {
        return title;
    }

    public void setTitle(String title) {
        this.title = title;
    }

    public String getAuthor() {
        return author;
    }

    public void setAuthor(String author) {
        this.author = author;
    }

    public String getIsbn() {
        return isbn;
    }

    public void setIsbn(String isbn) {
        this.isbn = isbn;
    }

    public double getPrice() {
        return price;
    }

    public void setPrice(double price) {
        this.price = price;
    }

    @Override
    public String toString() {
        return "Title: " + title + ", Author: " + author + ", ISBN: " + isbn + ", Price: $"
    }
}

class Library {
    private ArrayList<Book> books;

    public Library() {
        books = new ArrayList<>();
    }
}

```

```

    }

    public void addBook(Book book) {
        books.add(book);
    }

    public boolean removeBook(String isbn) {
        for (Book book : books) {
            if (book.getIsbn().equals(isbn)) {
                books.remove(book);
                return true;
            }
        }
        return false;
    }

    public void displayBooks() {
        if (books.isEmpty()) {
            System.out.println("No books in the library.");
        } else {
            for (Book book : books) {
                System.out.println(book);
            }
        }
    }
}

public class LibraryBooks {
    public static void main(String[] args) {
        Library library = new Library();

        library.addBook(new Book("1984", "joseph", "1234567890", 150));
        library.addBook(new Book("To Kill a Mockingbird", "Harper Lee", "0987654321", 200));
        library.addBook(new Book("1977", "George peter", "7864523791475", 100));
        System.out.println("All books in the library:");
        library.displayBooks();

        boolean removed = library.removeBook("1234567890");
        if (removed) {
            System.out.println("\nBook removed successfully.");
        } else {
            System.out.println("\nBook not found.");
        }

        System.out.println("\nAll books in the library after removal:");
        library.displayBooks();
    }
}

```

```
    }
}
```

output

```
C:\Users\acer\Documents\My_java>javac LibraryBooks.java
C:\Users\acer\Documents\My_java>java LibraryBooks
All books in the library:
Title: 1984, Author: Joseph, ISBN: 1234567890, Price: $150.0
Title: To Kill a Mockingbird, Author: Harper Lee, ISBN: 0987654321, Price: $200.0
Title: 1977, Author: George peter, ISBN: 7864523791475, Price: $100.0
Book removed successfully.
All books in the library after removal:
Title: To Kill a Mockingbird, Author: Harper Lee, ISBN: 0987654321, Price: $200.0
Title: 1977, Author: George peter, ISBN: 7864523791475, Price: $100.0
C:\Users\acer\Documents\My_java>_
```

5. Use a HashSet to manage the unique genres available in the library Ensure that new genres can be added without duplicating existing genres

program

```
import java.util.HashSet;
import java.util.Set;

public class Genres {
    private Set<String> genres;

    public Genres() {
        genres = new HashSet<>();
    }

    public boolean addGenre(String genre) {
        return genres.add(genre);
    }

    public boolean containsGenre(String genre) {
        return genres.contains(genre);
    }

    public boolean removeGenre(String genre) {
        return genres.remove(genre);
    }
}
```



```

    public Set<String> listGenres() {
        return genres;
    }

    public static void main(String[] args) {
        Genres genres = new Genres();
        genres.addGenre("Science Fiction");
        genres.addGenre("Fantasy");
        genres.addGenre("Mystery");

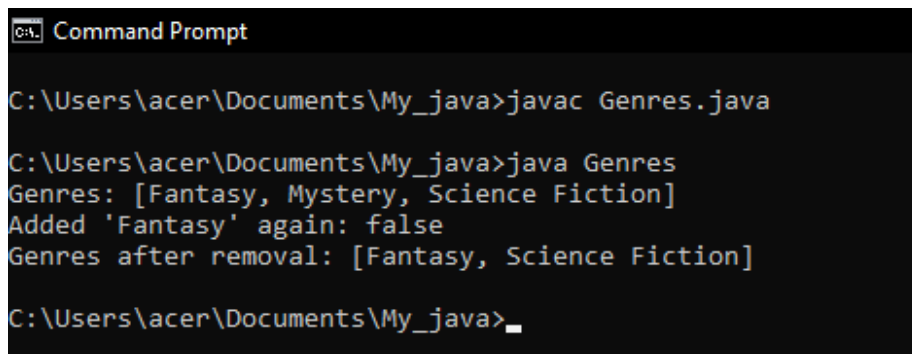
        System.out.println("Genres: " + genres.listGenres());

        boolean added = genres.addGenre("Fantasy");
        System.out.println("Added 'Fantasy' again: " + added);

        genres.removeGenre("Mystery");
        System.out.println("Genres after removal: " + genres.listGenres());
    }
}

```

output



```

C:\Users\acer\Documents\My_java>javac Genres.java

C:\Users\acer\Documents\My_java>java Genres
Genres: [Fantasy, Mystery, Science Fiction]
Added 'Fantasy' again: false
Genres after removal: [Fantasy, Science Fiction]

C:\Users\acer\Documents\My_java>_

```

6. Use a HashMap to map ISBN numbers to books for quick lookup Implement functionalities to add, update, and retrieve book details using ISBN.

program

```
import java.util.HashMap;
```

```

import java.util.Map;

public class Library06 {
    private Map<String, Book> books;

    public Library06() {
        books = new HashMap<>();
    }

    public void addOrUpdateBook(String isbn, String title, String author) {
        Book book = new Book(isbn, title, author);
        books.put(isbn, book);
    }

    public Book getBookByIsbn(String isbn) {
        return books.get(isbn);
    }

    public Book removeBookByIsbn(String isbn) {
        return books.remove(isbn);
    }

    public Map<String, Book> listBooks() {
        return books;
    }

    public static void main(String[] args) {
        Library06 library = new Library06();

        library.addOrUpdateBook("978-0135166307", "Effective Java", "Joshua Bloch");
        library.addOrUpdateBook("978-0596009205", "Head First Java", "Kathy Sierra");

        Book book = library.getBookByIsbn("978-0135166307");
        System.out.println("Retrieved Book: " + book);

        library.addOrUpdateBook("978-0135166307", "Effective Java", "Joshua Bloch (Updated)");

        book = library.getBookByIsbn("978-0135166307");
        System.out.println("Updated Book: " + book);

        library.removeBookByIsbn("978-0596009205");
        System.out.println("Books after removal: " + library.listBooks());
    }
}

class Book {

```

```

private String isbn;
private String title;
private String author;

public Book(String isbn, String title, String author) {
    this.isbn = isbn;
    this.title = title;
    this.author = author;
}

public String getIsbn() {
    return isbn;
}

public void setIsbn(String isbn) {
    this.isbn = isbn;
}

public String getTitle() {
    return title;
}

public void setTitle(String title) {
    this.title = title;
}

public String getAuthor() {
    return author;
}

public void setAuthor(String author) {
    this.author = author;
}

@Override
public String toString() {
    return "Book{ISBN='" + isbn + "', Title='" + title + "', Author='" + author + "'}";
}
}

```

```
C:\Users\acer\Documents\My_java>javac Library06.java
C:\Users\acer\Documents\My_java>java Library06
Retrieved Book: Book{ISBN='978-0135166307', Title='Effective Java', Author='Joshua Bloch'}
Updated Book: Book{ISBN='978-0135166307', Title='Effective Java', Author='Joshua Bloch (Updated)'}
Books after removal: {978-0135166307=Book{ISBN='978-0135166307', Title='Effective Java', Author='Joshua Bloch (Updated)'}}
```

output

7. Implement a custom exception called `ProductNotFoundException` that is thrown when a product is not found in the inventory. Use `try`, `catch`, `finally`, `throw`, and `throws` to handle exceptions appropriately

program

```
import java.util.HashMap;
import java.util.Map;

class ProductNotFoundException extends Exception {
    public ProductNotFoundException(String message) {
        super(message);
    }
}

class Product {
    private String id;
    private String name;
    private double price;

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

    public String getId() {
        return id;
    }

    public String getName() {
        return name;
    }

    public double getPrice() {
        return price;
    }
}
```

```

        @Override
        public String toString() {
            return "Product{ID='" + id + "', Name='" + name + "', Price=" + price + "}";
        }
    }

    public class Inventory {
        private Map<String, Product> products;

        public Inventory() {
            products = new HashMap<>();
        }

        public void addProduct(Product product) {
            products.put(product.getId(), product);
        }

        public Product getProductById(String id) throws ProductNotFoundException {
            Product product = products.get(id);
            if (product == null) {
                throw new ProductNotFoundException("Product with ID " + id + " not found.");
            }
            return product;
        }

        public static void main(String[] args) {
            Inventory inventory = new Inventory();
            inventory.addProduct(new Product("988", "Laptop", 100000));
            inventory.addProduct(new Product("238", "phone", 16000));

            try {

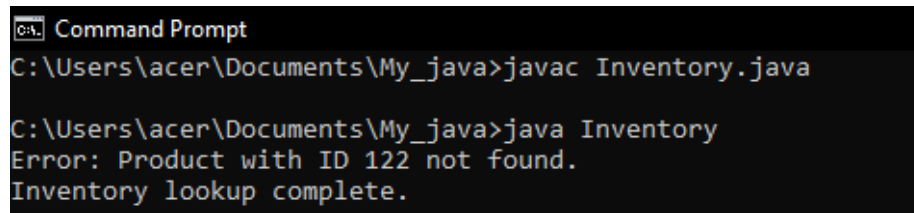
                Product product = inventory.getProductById("122");
                System.out.println("Product found: " + product);
            } catch (ProductNotFoundException e) {

                System.err.println("Error: " + e.getMessage());
            } finally {

                System.out.println("Inventory lookup complete.");
            }
        }
    }
}

```

output



```
C:\Users\acer\Documents\My_java>javac Inventory.java

C:\Users\acer\Documents\My_java>java Inventory
Error: Product with ID 122 not found.
Inventory lookup complete.
```

8. Use any one of the file handling to read employee records from a text file and write employee

program

```
import java.io.*;
import java.util.ArrayList;
import java.util.List;

class Employee {
    private String name;
    private int id;
    private String position;

    public Employee(String name, int id, String position) {
        this.name = name;
        this.id = id;
        this.position = position;
    }

    @Override
    public String toString() {
        return name + ", " + id + ", " + position;
    }

    public static Employee fromString(String employeeString) {
        String[] parts = employeeString.split(", ");
        return new Employee(parts[0], Integer.parseInt(parts[1]), parts[2]);
    }
}
```

```

public class EmployeeFileHandler {

    public static List<Employee> readEmployeesFromFile(String filename) throws IOException {
        List<Employee> employees = new ArrayList<>();
        try (BufferedReader reader = new BufferedReader(new FileReader(filename))) {
            String line;
            while ((line = reader.readLine()) != null) {
                employees.add(Employee.fromString(line));
            }
        }
        return employees;
    }

    public static void writeEmployeesToFile(String filename, List<Employee> employees) throws IOException {
        try (BufferedWriter writer = new BufferedWriter(new FileWriter(filename))) {
            for (Employee employee : employees) {
                writer.write(employee.toString());
                writer.newLine();
            }
        }
    }

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

            List<Employee> employees = readEmployeesFromFile("employees.txt");
            System.out.println("Employees read from file:");
            for (Employee employee : employees) {
                System.out.println(employee);
            }
            writeEmployeesToFile("new_employees.txt", employees);
            System.out.println("Employee records have been written to new_employees.txt.");
        } catch (IOException e) {
            System.err.println("An error occurred during file operations: " + e.getMessage());
        }
    }
}

```