EXPERIMENT NO:-07

AIM:- Implementation of solution of Knapsack problem using Greedy method.

```
import java.util.Scanner;

public class Knapsack {
  public static int knapsack(int[] weights, int[] values, int capacity)
{
    int n = weights.length;
    int[] dp = new int[capacity + 1];

    for (int i = 0; i < n; i++) {
      for (int w = capacity; w >= weights[i]; w--)
      {
        dp[w] = Math.max(dp[w], values[i] + dp[w - weights[i]]);
      }
    }
    return dp[capacity];
  }

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

    System.out.print("Enter the number of items: ");
    int n = scanner.nextInt();

    int[] weights = new int[n];
    int[] values = new int[n];

    System.out.println("Enter weights of items:");
    for (int i = 0; i < n; i++)
    {
      weights[i] = scanner.nextInt();
    }

    System.out.println("Enter values of items:");
    for (int i = 0; i < n; i++)
    {
      values[i] = scanner.nextInt();
    }

    System.out.print("Enter the capacity of the knapsack: ");
```

```
        int capacity = scanner.nextInt();

        int maxValue = knapsack(weights, values, capacity);
        System.out.println("Maximum value: " + maxValue);

        scanner.close();
    }
}
```

OUTPUT:-

```
"C:\Program Files\Java\jdk-22\bin\java.exe" "-javaagent:C:\Users\Ankit raj\IntelliJ IDEA Community Edition 2023.2.4
Enter the number of items: 5
Enter weights of items:
8 9 6 7 11
Enter values of items:
3 6 8 5 6
Enter the capacity of the knapsack: 20
Maximum value: 14

Process finished with exit code 0
```

EXPERIMENT NO:-06

AIM:- Implementation of Merge Sorting technique using Divide and Conquer approach.

```
import java.util.Arrays;
import java.util.Scanner;

public class MergeSort {
  public static void mergeSort(int[] arr, int left, int right)

 {
    if (left < right) {
      int mid = left + (right - left) / 2;
      mergeSort(arr, left, mid);
      mergeSort(arr, mid + 1, right);
      merge(arr, left, mid, right);
    }
  }

  private static void merge(int[] arr, int left, int mid, int right)

  {
    int[] temp = new int[right - left + 1];
    int i = left, j = mid + 1, k = 0;

    while (i <= mid && j <= right) {
      temp[k++] = arr[i] <= arr[j] ? arr[i++] : arr[j++];
    }
    while (i <= mid) temp[k++] = arr[i++];
    while (j <= right) temp[k++] = arr[j++];

    for (int p = 0; p < temp.length; p++) {
      arr[left + p] = temp[p];
    }
  }

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

    System.out.print("Enter the number of elements: ");
    int n = scanner.nextInt();

    int[] arr = new int[n];
    System.out.println("Enter the elements of the array:");
    for (int i = 0; i < n; i++) {
      arr[i] = scanner.nextInt();
```

```
    }

    mergeSort(arr, 0, arr.length - 1);
    System.out.println("Sorted array: " + Arrays.toString(arr));

  }
}
```

OUTPUT:-

```
"C:\Program Files\Java\jdk-22\bin\java.exe" "-javaagent:C:\Users\Ankit raj\IntelliJ IDEA Community Edition 2023.2.4"
Enter the number of elements: 8
Enter the elements of the array:
12 8 6 14 18 9 13 17
Sorted array: [6, 8, 9, 12, 13, 14, 17, 18]

Process finished with exit code 0
```

EXPERIMENT NO:-05

AIM:- Implementation of Quick Sorting technique using Divide and Conquer approach.

```
import java.util.Arrays;
import java.util.Scanner;

public class QuickSort {
  public static void quickSort(int[] arr, int low, int high)

  {
    if (low < high)

    {
      int pivotIndex = partition(arr, low, high);
      quickSort(arr, low, pivotIndex - 1);
      quickSort(arr, pivotIndex + 1, high);
    }
  }

  private static int partition(int[] arr, int low, int high)

  {
    int pivot = arr[high];
    int i = low - 1;
    for (int j = low; j < high; j++)

     {
      if (arr[j] <= pivot) {
        i++;
        swap(arr, i, j);
      }
    }
    swap(arr, i + 1, high);
    return i + 1;
  }

  private static void swap(int[] arr, int i, int j)

  {
    int temp = arr[i];
    arr[i] = arr[j];
    arr[j] = temp;
  }

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

```
        System.out.print("Enter the number of elements: ");
        int n = scanner.nextInt();

        int[] arr = new int[n];
        System.out.println("Enter the elements of the array:");
        for (int i = 0; i < n; i++) {
            arr[i] = scanner.nextInt();
        }

        quickSort(arr, 0, arr.length - 1);
        System.out.println("Sorted array: " + Arrays.toString(arr));

    }
}
```

OUTPUT:-

```
"C:\Program Files\Java\jdk-22\bin\java.exe" "-javaagent:C:\Users\Ankit raj\IntelliJ IDEA Community Edition 2023.2.4"
Enter the number of elements: 8
Enter the elements of the array:
12 22 9 13 7 17 12 18
Sorted array: [7, 9, 12, 12, 13, 17, 18, 22]

Process finished with exit code 0
```