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

public class ETOE {

    static int turnaroundtime = 0, length;
    static Scanner s = new Scanner(System.in);

    static int[] getTasks() {
        System.out.print("Number of Tasks: ");
        length = s.nextInt();
        int[] a = new int[length];
        for (int i = 0; i < length; i++) {
            System.out.print("Tasks #" + (i + 1) + "'s Burst Time: ");
            a[i] = s.nextInt();
        }
        return a;
    }

    static int[] sortArray(int[] NA) {
        int sorting = 0;
        while (sorting < NA.length) {
            if (sorting > 0) {
                quickSort(NA, 0, sorting);
                turnaroundtime++;
            }
            sorting++;
        }
        return NA;
    }

    static void quickSort(int arr[], int begin, int end) {
        if (begin < end) {
            int partitionIndex = partition(arr, begin, end);
            quickSort(arr, begin, partitionIndex - 1);
            quickSort(arr, partitionIndex + 1, end);
        }
    }

    static int partition(int arr[], int begin, int end) {
        int pivot = arr[end];
        int i = (begin - 1);
        for (int j = begin; j < end; j++) {
            if (arr[j] <= pivot) {
                i++;
```

```
            int swapTemp = arr[i];
            arr[i] = arr[j];
            arr[j] = swapTemp;
        }
    }
    int swapTemp = arr[i + 1];
    arr[i + 1] = arr[end];
    arr[end] = swapTemp;
    return i + 1;
}

static void executeTasks(int[] q1) {
    int tq, contextSwitch = 0, shortest = 0, longest = length - 1;
    int currenttasks = length - 2;
    int wT = 0;
    while (shortest <= longest) {
        shortest = 0;
        longest = length - 1;
        for (int i = 0; i < length; i++) {
            if (q1[shortest] == 0) {
                shortest++;
            } else {
                break;
            }
        }
        tq = q1[shortest];
        q1[shortest] = 0;
        turnaroundtime += tq;
        int lt = 0, bt = 0;
        if(shortest != longest){
            lt = q1[longest];
            bt = lt - tq;
            //lt = longest task, bt = burst time
            turnaroundtime += tq;
            q1[longest] = bt;
        }
        if (longest <= 0 || shortest >= q1.length - 1) {
            break;
        } else {
            shortest++;
            if (bt <= 0) {
                q1[longest] = 0;
                longest--;
                currenttasks--;
                wT += tq;
            } else {
```

```
                contextSwitch++;
            }
        }
        if (currenttasks > 0) {
            wT += (tq * currenttasks);
        }
        currenttasks--;
        sortArray(q1);
    }
    System.out.println("************************************************");
    System.out.println("TASKS: " + Arrays.toString(q1));
    System.out.println("Total Turnaround Time:\t\t" + (turnaroundtime) + " ms");
    System.out.println("Average Turnaround Time:\t" + (turnaroundtime / length) + " ms");
//          System.out.println("Waiting Time:\t\t\t"+wT+" ms");
    System.out.println("Average Waiting Time:\t\t" + (wT / length) + " ms");
    System.out.println("# of Context Switches:\t\t" + contextSwitch);
    System.out.println("************************************************");
  }

  public static void main(String[] args) {
    while (true) {
      System.out.println("Enter new tasks? [Y/N]");
      int[] NA;
      char choice = s.next().charAt(0);
      if (choice == 'y' || choice == 'Y') {
        NA = getTasks();
      } else {
        break;
      }
      System.out.println("NA: " + Arrays.toString(NA));
      int[] a = sortArray(NA);
      executeTasks(a);
    }
  }
}
```

SJF

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

public class ETOESJF {

    static int turnaroundtime = 0, length, wT = 0, cs = 0;
    static Scanner s = new Scanner(System.in);

    static int[] getTasks() {
        System.out.print("Number of Tasks: ");
        length = s.nextInt();
        int[] a = new int[length];
        for (int i = 0; i < length; i++) {
            System.out.print("Tasks #" + (i + 1) + "'s Burst Time: ");
            a[i] = s.nextInt();
        }
        return a;
    }

    static int[] sortArray(int[] NA) {
        int sorting = 0;
        while (sorting < NA.length) {
            if (sorting > 0) {
                quickSort(NA, 0, sorting);
                turnaroundtime++;
            }
            sorting++;
        }
        return NA;
    }

    static void quickSort(int arr[], int begin, int end) {
        if (begin < end) {
            int partitionIndex = partition(arr, begin, end);
            quickSort(arr, begin, partitionIndex - 1);
            quickSort(arr, partitionIndex + 1, end);
        }
    }

    static int partition(int arr[], int begin, int end) {
        int pivot = arr[end];
        int i = (begin - 1);
```

```
    for (int j = begin; j < end; j++) {
        if (arr[j] <= pivot) {
            i++;
            int swapTemp = arr[i];
            arr[i] = arr[j];
            arr[j] = swapTemp;
        }
    }
    int swapTemp = arr[i + 1];
    arr[i + 1] = arr[end];
    arr[end] = swapTemp;
    return i + 1;
}

static void executeTasks(int[] q1) {
    for (int i = 0; i < q1.length; i++) {
        turnaroundtime += q1[i];
        wT += (q1[i]*(q1.length-(1+i)));
        q1[i] = 0;
        cs++;
        sortArray(q1);
        System.out.println(wT);
    }

    System.out.println("************************************************");
    System.out.println("TASKS: " + Arrays.toString(q1));
    System.out.println("Total Turnaround Time:\t\t" + (turnaroundtime) + " ms");
    System.out.println("Average Turnaround Time:\t" + (turnaroundtime / length) + " ms");
    System.out.println("Average Waiting Time:\t\t"+(wT/q1.length)+" ms");
    System.out.println("# of Context Switches:\t\t"+cs);
    System.out.println("************************************************");
}

public static void main(String[] args) {
    while (true) {
        System.out.println("Enter new tasks? [Y/N]");
        int[] NA;
        char choice = s.next().charAt(0);
        if (choice == 'y' || choice == 'Y') {
            NA = getTasks();
        } else {
            break;
        }
        System.out.println("NA: " + Arrays.toString(NA));
        int[] a = sortArray(NA);
        executeTasks(a);
```

```
    }
  }

}
```

Round Robin

```
import java.util.Arrays;
import java.util.Scanner;
public class ETOERR {

        static int executionCount = 0;
        static int TQ = 0;
    static Scanner s = new Scanner(System.in);

    static int[] getTasks(){
        System.out.print("Number of Tasks: ");
        int length = s.nextInt();
        int[] a = new int[length];
        for (int i = 0; i < length; i++) {
            System.out.print("Tasks #"+(i+1)+"'s Burst Time: ");
            a[i] = s.nextInt();
        }
        return a;
    }

        static void executeTasks(int[] q1, int TQ) {

                int i = 0;
                int completedTask = 0;

                while(completedTask != q1.length) {

                        if (q1[i] != 0) {

                                if(q1[i] < TQ) {

                                        q1[i] = 0;
                                        completedTask++;
                                        executionCount+=TQ;

                                } else {

                                        q1[i] -= TQ;
                                        executionCount+=TQ;

                                        if(q1[i] <= 0)
                                                completedTask++;
                                }
```

```
                        System.out.println("-- After: " + Arrays.toString(q1));

                    }

                    if (i >= q1.length-1)
                        i = 0;
                    else
                        i++;

                }

                System.out.println("************************************************");
                System.out.println("Execution Time: " + (executionCount) + "ms");
                System.out.println("************************************************");

        }

        public static void main(String[] args) {
    while(true){
        System.out.print("Enter new tasks? [Y/N]: ");
        int[] NA;
        char choice = s.next().charAt(0);
        if(choice == 'y' || choice == 'Y') {
            NA = getTasks();
            System.out.print("What is the give Time Quantum (TQ): ");
            TQ = s.nextInt();
        } else
            break;
        System.out.println("NA: " + Arrays.toString(NA));
        executeTasks(NA, TQ);
    }
        }
}
```