

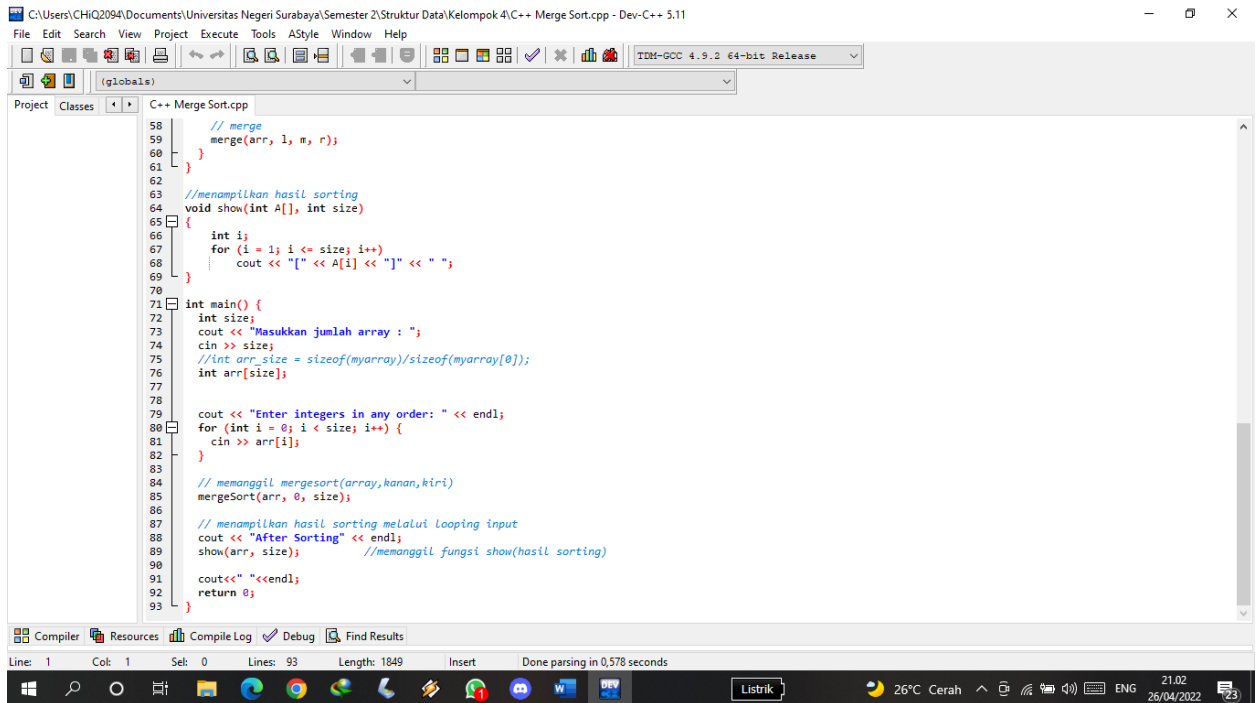
Azkie Kamila_21091397027

Tugas Individu Struktur Data (26/04/2022)

1. Program C++ Merge Sort

The image displays two screenshots of a C++ Merge Sort program being developed in the Dev-C++ IDE. The top screenshot shows the initial part of the code, including the inclusion of `<iostream>`, the use of the `std` namespace, and the declaration of variables `arr`, `l`, `m`, and `r`. It also shows the initialization of a temporary array `temp` of size 5. The bottom screenshot shows the continuation of the code, including the recursive `mergeSort` function and the `main` function. The `main` function prompts the user to input an array and its size, then calls `mergeSort` and `show` to display the sorted array.

```
1 #include <iostream>
2 using namespace std;
3
4 // deklarasi variabel
5 void merge(int arr[], int l, int m, int r) {
6     int i = l;
7     int j = m + 1;
8     int k = l;
9
10    // membuat temp array
11    int temp[5];
12
13    while (i <= m && j <= r) {
14        if (arr[i] <= arr[j]) {
15            temp[k] = arr[i];
16            i++;
17            k++;
18        } else {
19            temp[k] = arr[j];
20            j++;
21            k++;
22        }
23    }
24
25    // menyalin elemen yang tersisa di babak kedua jika ada
26    while (i <= m) {
27        temp[k] = arr[i];
28        i++;
29        k++;
30    }
31
32    // menyalin elemen yang tersisa di babak kedua jika ada
33    while (j <= r) {
34        temp[k] = arr[j];
35        j++;
36        k++;
37    }
38
39    // menyalin temp array ke array asli
40    for (int p = l; p <= r; p++) {
41        arr[p] = temp[p];
42    }
43
44    // l untuk indeks sebelah kiri dan r untuk indeks sebelah kanan di subarray dari array untuk disortir
45    void mergeSort(int arr[], int l, int r) {
46        if (l < r) {
47            // mencari titik tengah
48            int m = (l + r) / 2;
49
50            // Rekrusif mergesort bagian pertama dan kedua
51            mergeSort(arr, l, m);
52            mergeSort(arr, m + 1, r);
53
54            // merge
55            merge(arr, l, m, r);
56        }
57    }
58
59    //menampilkan hasil sorting
60    void show(int A[], int size)
61    {
62        int i;
63        for (i = 1; i <= size; i++)
64            cout << "[" << A[i] << "]" << " ";
65    }
66
67    int main() {
68        int size;
69        cout << "Masukkan jumlah array : ";
70    }
```

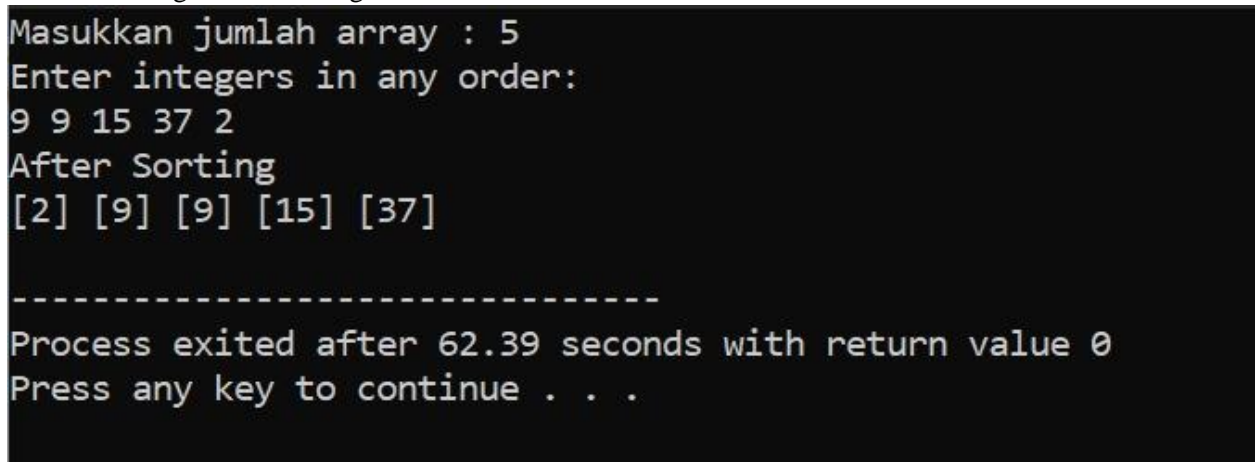


The screenshot shows a C++ program for Merge Sort in the Dev-C++ IDE. The code is as follows:

```
58 // merge
59 merge(arr, l, m, r);
60 }
61 }
62
63 //menampilkan hasil sorting
64 void show(int A[], int size)
65 {
66     int i;
67     for (i = 1; i <= size; i++)
68         cout << "[" << A[i] << "]" << " ";
69 }
70
71 int main() {
72     int size;
73     cout << "Masukkan jumlah array : ";
74     cin >> size;
75     //int arr_size = sizeof(myarray)/sizeof(myarray[0]);
76     int arr[size];
77
78     cout << "Enter integers in any order: " << endl;
79     for (int i = 0; i < size; i++) {
80         cin >> arr[i];
81     }
82
83     // memanggil mergesort(array,kanan,kiri)
84     mergeSort(arr, 0, size);
85
86     // menampilkan hasil sorting melalui looping input
87     cout << "After Sorting" << endl;
88     show(arr, size); //memanggil fungsi show(hasil sorting)
89
90     cout << " " << endl;
91     return 0;
92 }
93 }
```

The IDE status bar at the bottom indicates: Line: 1, Col: 1, Sek: 0, Lines: 93, Length: 1849, Insert, Done parsing in 0,578 seconds. The Windows taskbar at the bottom shows the date as 26/04/2022 and time as 21:02.

2. Hasil Run Program C++ Merge Sort



The screenshot shows the output of the Merge Sort program in a terminal window. The output is as follows:

```
Masukkan jumlah array : 5
Enter integers in any order:
9 9 15 37 2
After Sorting
[2] [9] [9] [15] [37]

-----
Process exited after 62.39 seconds with return value 0
Press any key to continue . . .
```

3. Kelebihan Merge Sort

- Dibanding dengan algoritma lain, merge sort terhitung cepat prosesnya.
- Dibanding dengan algoritma lain, merge sort termasuk algoritma yang sangat efisien dalam penggunaannya sebab setiap list selalu dibagi bagi menjadi list yang lebih kecil, kemudian digabungkan lagi sehingga tidak perlu melakukan banyak perbandingan.
- Cocok untuk sorting akses datanya lambat misalnya tape drive atau hard disk.
- Cocok untuk sorting data yang biasanya diakses secara sequentially (berurutan), misalnya linked list, tape drive, dan hard disk.

4. Kekurangan Merge Sort

- Merge Sort membutuhkan lebih banyak ruang pada memori daripada jenis sorting lainnya.