**TUGAS PRAKTIKUM STRUKTUR DATA**

**JOBSHEET 9**

****

**Dosen Pengampu:**

**Randi Proska Sandra, S.Pd, M.Sc**

**Kode Kelas:**

**202323430158**

**Disusun Oleh:**

**Wahyu Abdil Afif**

**23343085**

**PROGRAM STUDI INFORMATIKA (NK)**

**FAKULTAS TEKNIK**

**UNIVERSITAS NEGERI PADANG**

**2024**

1. Selection Sort:

Source code:

// created by wahyu abdil afif\_23343085

#include <stdio.h>

void selectionSort(int arr[], int n)

{

int i, j, min\_idx;

for (i = 0; i < n-1; i++)

{

min\_idx = i;

for (j = i+1; j < n; j++)

{

if (arr[j] < arr[min\_idx])

{

min\_idx = j;

}

}

int temp = arr[min\_idx];

arr[min\_idx] = arr[i];

arr[i] = temp;

}

}

int main()

{

int arr[] = {17, 31, 25, 12, 22, 11, 3};

int n = sizeof(arr)/sizeof(arr[0]);

selectionSort(arr, n);

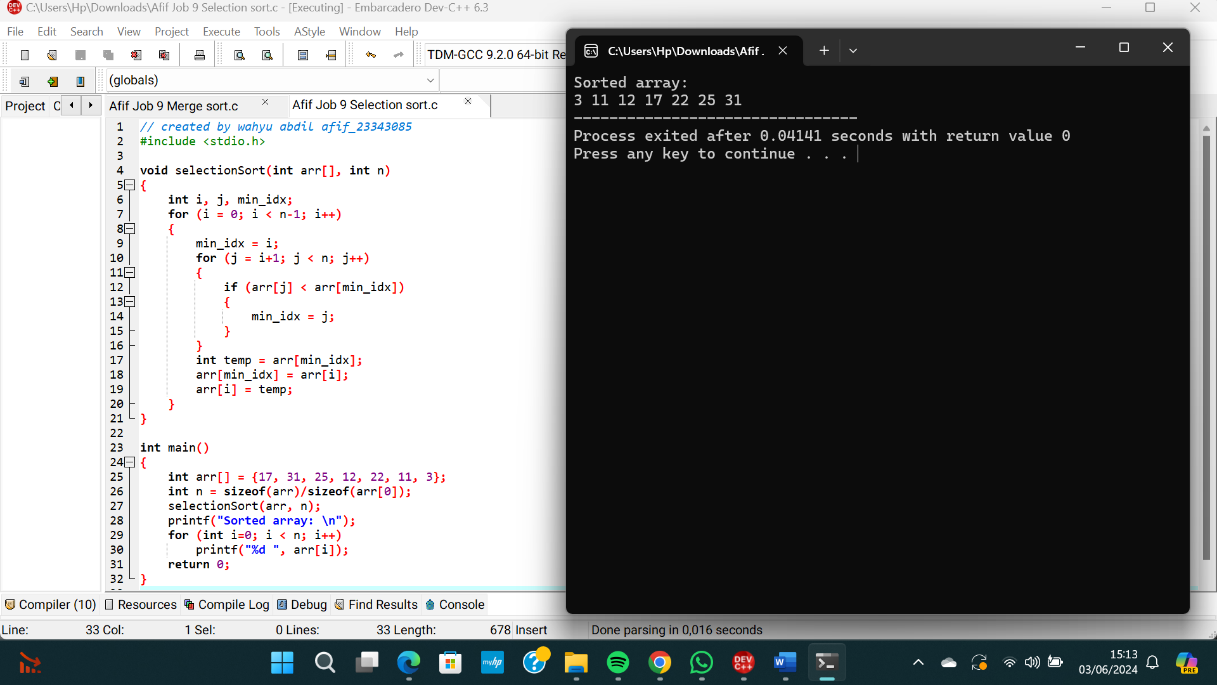
printf("Sorted array: \n");

for (int i=0; i < n; i++)

printf("%d ", arr[i]);

return 0;

}



Penjelasan:

* Algoritma selection sort bekerja dengan mencari elemen terkecil dalam array pada setiap iterasi.
* Elemen terkecil kemudian ditukar dengan elemen pada posisi awal array yang belum diurutkan.
* Proses ini diulang sampai semua elemen diurutkan.
* Prinsip utama selection sort adalah memilih elemen terkecil satu per satu dan menempatkannya di posisinya yang benar.
* Selection sort adalah pilihan yang baik untuk data berukuran kecil atau untuk skenario di mana kesederhanaan dan kemudahan implementasi menjadi prioritas. Namun, untuk data berukuran besar, disarankan menggunakan algoritma sorting yang lebih efisien seperti merge sort atau quicksort.

1. Merge Sort:

Source code:

// created by wahyu abdil afif\_23343085

#include <stdio.h>

void merge(int arr[], int l, int m, int r)

{

int i, j, k;

int n1 = m - l + 1;

int n2 = r - m;

int L[n1], R[n2];

for (i = 0; i < n1; i++)

L[i] = arr[l + i];

for (j = 0; j < n2; j++)

R[j] = arr[m + 1 + j];

i = 0;

j = 0;

k = l;

while (i < n1 && j < n2)

{

if (L[i] <= R[j])

{

arr[k] = L[i];

i++;

} else {

arr[k] = R[j];

j++;

}

k++;

}

while (i < n1)

{

arr[k] = L[i];

i++;

k++;

}

while (j < n2)

{

arr[k] = R[j];

j++;

k++;

}

}

void mergeSort(int arr[], int l, int r)

{

if (l < r)

{

int m = l + (r - l) / 2;

mergeSort(arr, l, m);

mergeSort(arr, m + 1, r);

merge(arr, l, m, r);

}

}

int main()

{

int arr[] = {17, 31, 25, 12, 22, 11, 3};

int arr\_size = sizeof(arr)/sizeof(arr[0]);

mergeSort(arr, 0, arr\_size - 1);

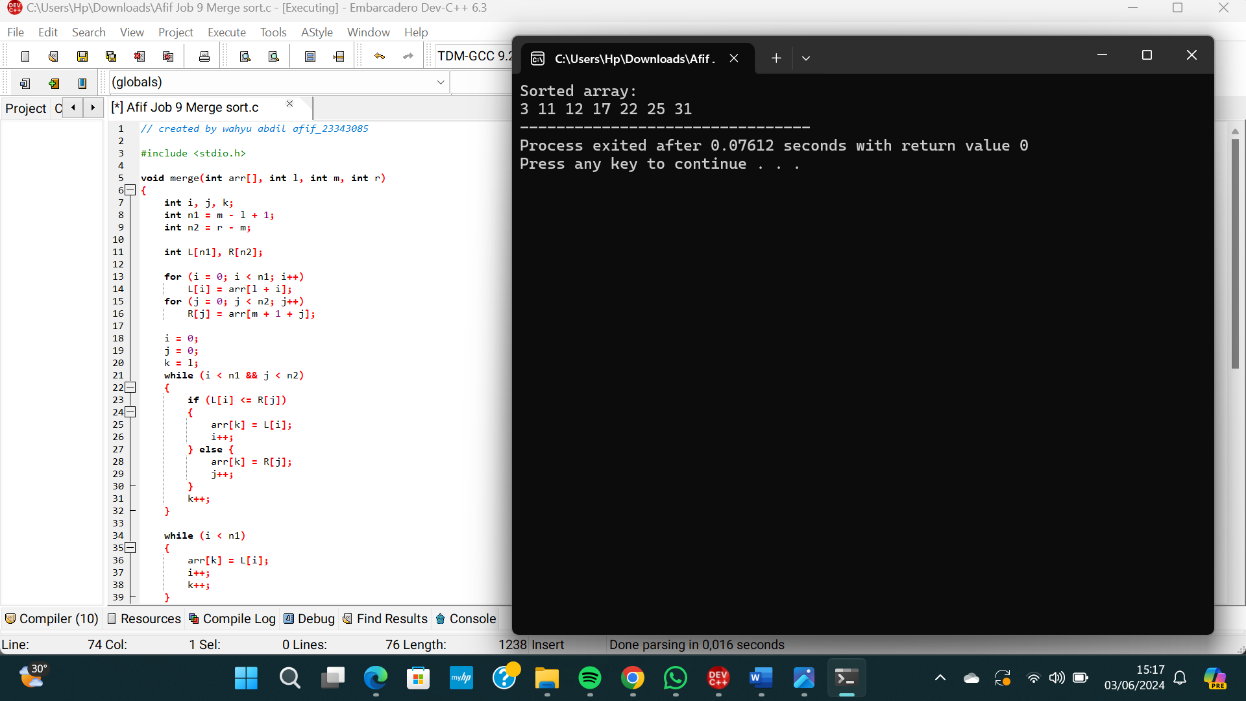
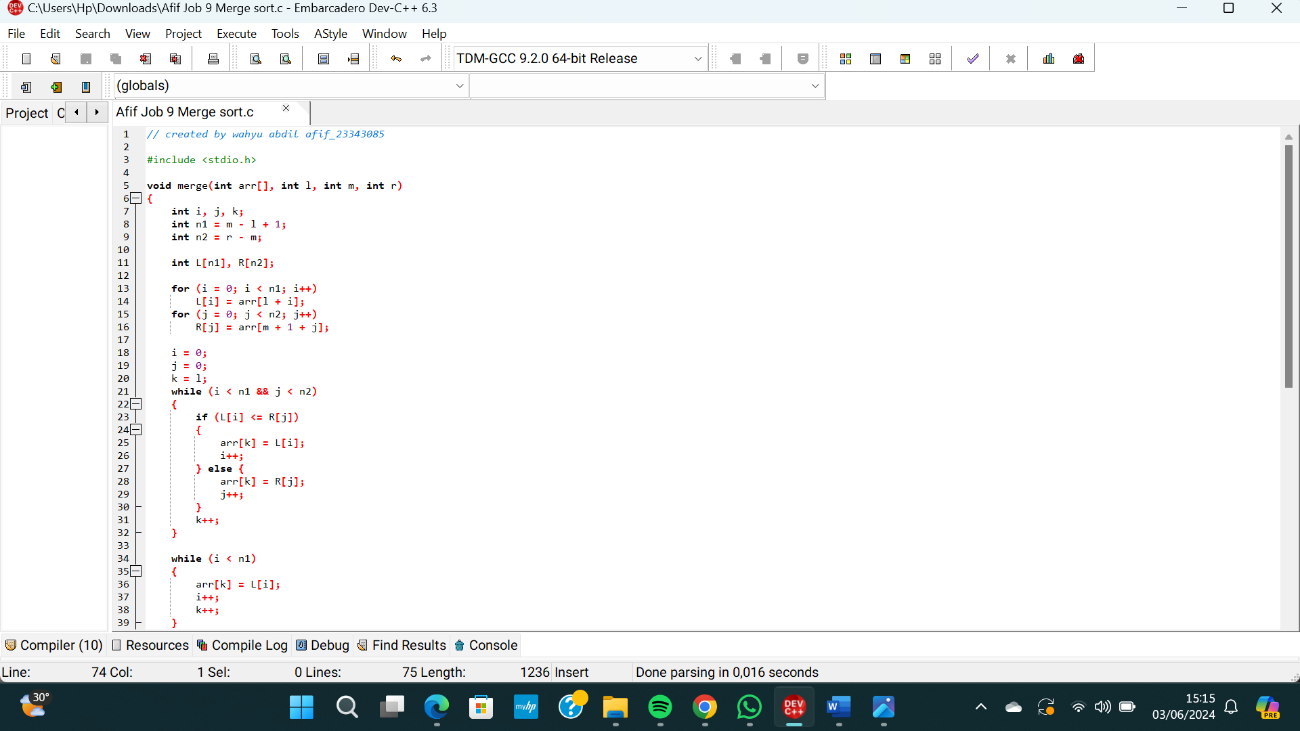
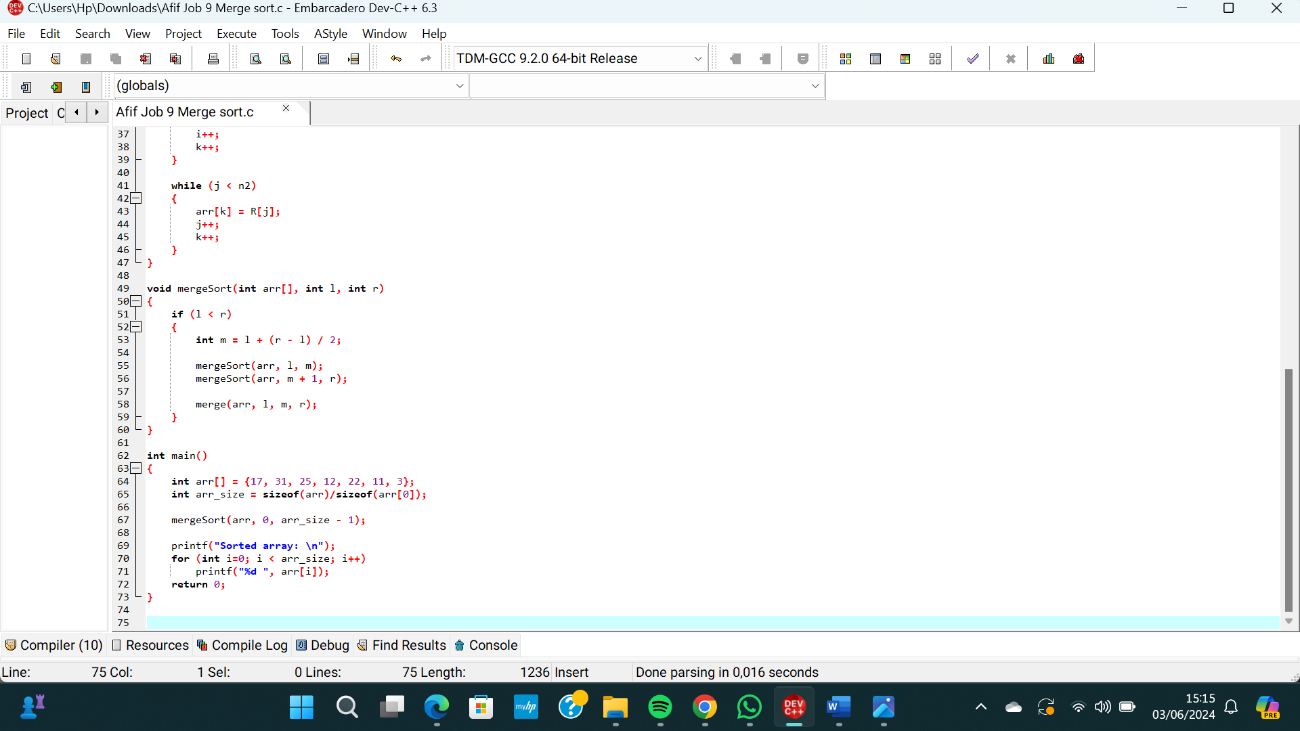
printf("Sorted array: \n");

for (int i=0; i < arr\_size; i++)

printf("%d ", arr[i]);

return 0;

}



Penjelasan:

* Metode merge sort menggunakan pendekatan rekursif untuk membagi array menjadi dua bagian hingga menjadi sub-array yang berisi satu elemen.
* Kemudian, sub-array tersebut digabungkan secara berurutan (merge) dengan membandingkan dan menggabungkan elemen-elemen dalam sub-array.
* Proses ini berlanjut hingga semua sub-array digabungkan kembali menjadi satu array yang terurut.
* Merge sort adalah pilihan yang baik untuk mengurutkan data berukuran besar karena efisiensinya. Namun, untuk data berukuran kecil, bubble sort atau selection sort mungkin lebih sederhana untuk diimplementasikan.