

Jobsheet 8
Praktikum Struktur Data



Dosen pengampu : Randi Proska Sandra, S.Pd, M.Sc

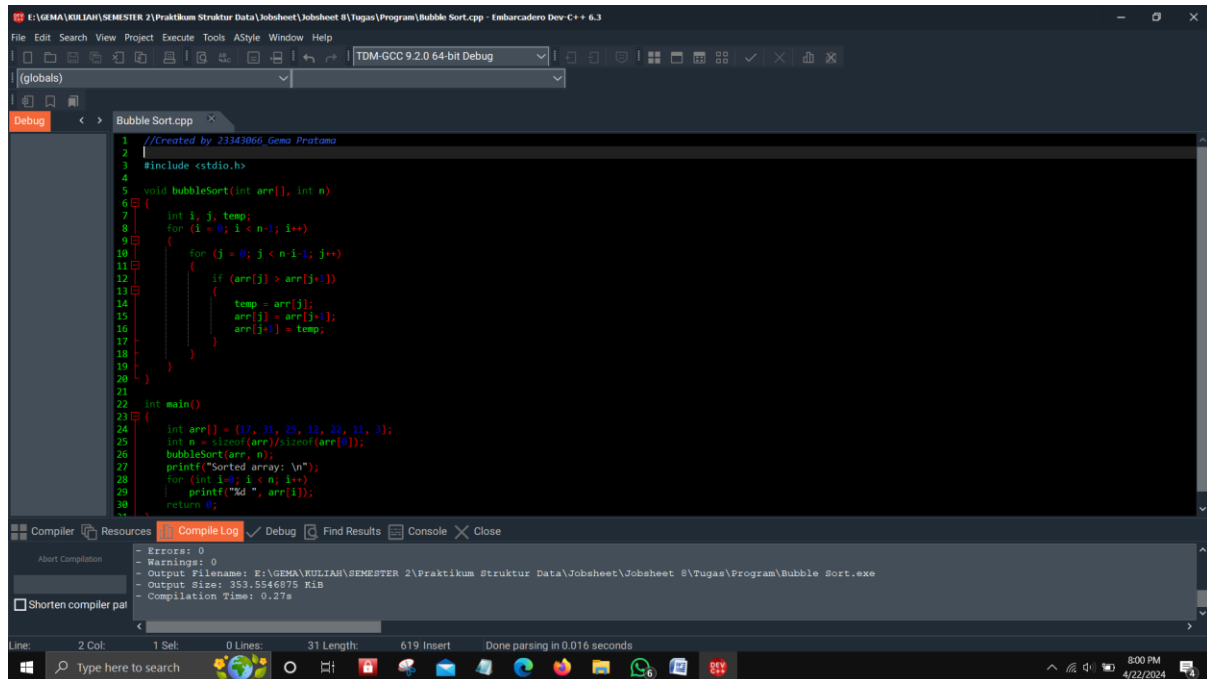
Kode Kelas : 202323430158

Disusun Oleh :

Gema Pratama Mahadi Putera
23343066

PROGRAM STUDI INFORMATIKA (NK)
FAKULTAS TEKNIK
UNIVERSITAS NEGERI PADANG
2024

1. Bubble Sort :

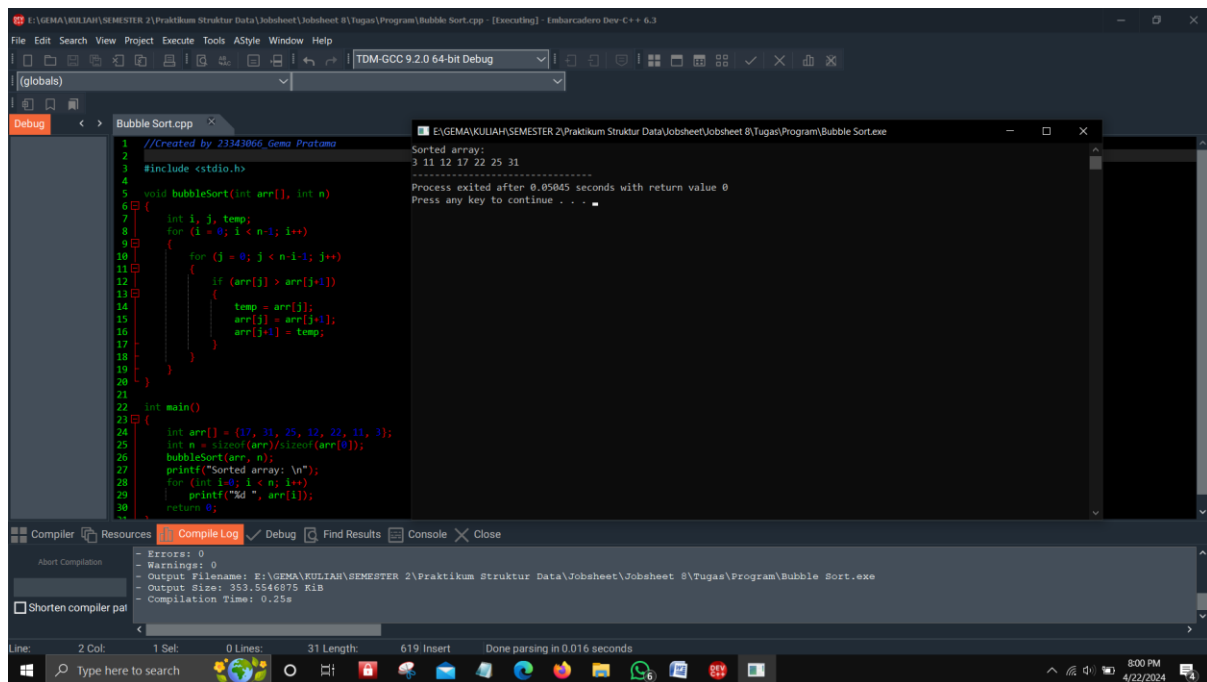


```
1 //Created by 23343066_Gema Pratama
2
3 #include <stdio.h>
4
5 void bubbleSort(int arr[], int n)
6 {
7     int i, j, temp;
8     for (i = 0; i < n-1; i++)
9     {
10         for (j = 0; j < n-i-1; j++)
11         {
12             if (arr[j] > arr[j+1])
13             {
14                 temp = arr[j];
15                 arr[j] = arr[j+1];
16                 arr[j+1] = temp;
17             }
18         }
19     }
20 }
21
22 int main()
23 {
24     int arr[] = {17, 31, 25, 12, 22, 11, 3};
25     int n = sizeof(arr)/sizeof(arr[0]);
26     bubbleSort(arr, n);
27     printf("Sorted array: \n");
28     for (int i=0; i < n; i++)
29     {
30         printf("%d ", arr[i]);
31     }
32     return 0;
33 }
```

Compiler: TDM-GCC 9.2.0 64-bit Debug

Errors: 0
Warnings: 0
Output Filename: E:\GEMA\KULIAH\SEMESTER 2\Praktikum Struktur Data\Jobsheet 8\Tugas\Program\Bubble Sort.exe
Output Size: 353.5546875 KiB
Compilation Time: 0.27s

Line: 2 Col: 1 Sel: 0 Lines: 31 Length: 619 Insert Done parsing in 0.016 seconds



```
1 //Created by 23343066_Gema Pratama
2
3 #include <stdio.h>
4
5 void bubbleSort(int arr[], int n)
6 {
7     int i, j, temp;
8     for (i = 0; i < n-1; i++)
9     {
10         for (j = 0; j < n-i-1; j++)
11         {
12             if (arr[j] > arr[j+1])
13             {
14                 temp = arr[j];
15                 arr[j] = arr[j+1];
16                 arr[j+1] = temp;
17             }
18         }
19     }
20 }
21
22 int main()
23 {
24     int arr[] = {17, 31, 25, 12, 22, 11, 3};
25     int n = sizeof(arr)/sizeof(arr[0]);
26     bubbleSort(arr, n);
27     printf("Sorted array: \n");
28     for (int i=0; i < n; i++)
29     {
30         printf("%d ", arr[i]);
31     }
32     return 0;
33 }
```

Compiler: TDM-GCC 9.2.0 64-bit Debug

Errors: 0
Warnings: 0
Output Filename: E:\GEMA\KULIAH\SEMESTER 2\Praktikum Struktur Data\Jobsheet 8\Tugas\Program\Bubble Sort.exe
Output Size: 353.5546875 KiB
Compilation Time: 0.25s

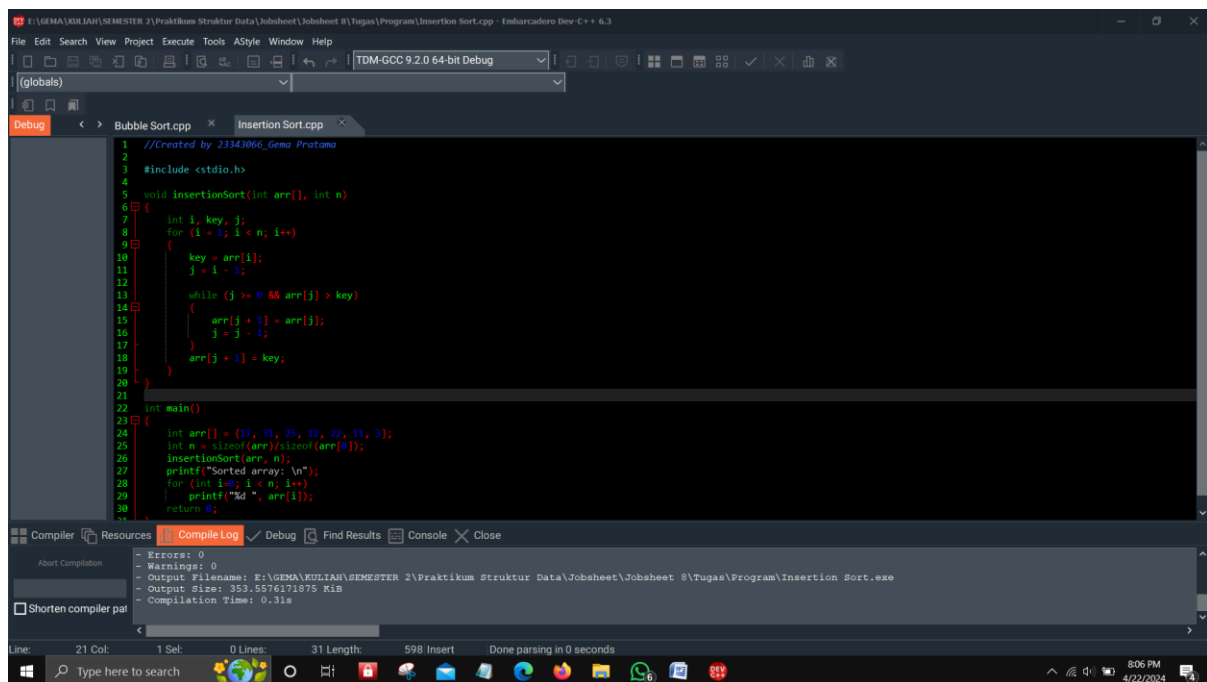
Line: 2 Col: 1 Sel: 0 Lines: 31 Length: 619 Insert Done parsing in 0.016 seconds

Sorted array:
3 11 12 17 22 25 31
Process exited after 0.05845 seconds with return value 0
Press any key to continue . . .

Penjelasan :

- Algoritma bubble sort bekerja dengan membandingkan elemen-elemen yang berdekatan dalam array.
- Jika elemen pertama lebih besar daripada elemen kedua, maka elemen-elemen tersebut ditukar.
- Proses ini diulang sampai semua elemen diurutkan.
- Prinsip utama bubble sort adalah menggelembungkan elemen terbesar ke atas array seperti gelembung dalam air.
- Bubble sort adalah pilihan yang baik untuk mengurutkan array kecil atau untuk tujuan pembelajaran karena kesederhanaannya. Namun, untuk array besar, disarankan menggunakan algoritma pengurutan yang lebih efisien seperti merge sort atau quicksort.

2. Insertion Sort :



The screenshot shows a C++ IDE with the following code for Insertion Sort:

```
1 //Created by 23343066_Gema Pratama
2
3 #include <stdio.h>
4
5 void insertionSort(int arr[], int n)
6 {
7     int i, key, j;
8     for (i = 1; i < n; i++)
9     {
10         key = arr[i];
11         j = i - 1;
12         while (j >= 0 && arr[j] > key)
13         {
14             arr[j + 1] = arr[j];
15             j = j - 1;
16         }
17         arr[j + 1] = key;
18     }
19 }
20
21
22 int main()
23 {
24     int arr[] = {17, 31, 25, 12, 22, 11, 3};
25     int n = sizeof(arr)/sizeof(arr[0]);
26     insertionSort(arr, n);
27     printf("Sorted array: \n");
28     for (int i=0; i < n; i++)
29         printf("%d ", arr[i]);
30     return 0;
31 }
```

The output window shows the following compilation results:

```
- Errors: 0
- Warnings: 0
- Output filename: E:\GEMA\KULIAH\SEMESTER 2\Praktikum Struktur Data\Jobsheet 8\Tugas\Program\Insertion Sort.exe
- Output Size: 353.5576171875 KiB
- Compilation Time: 0.31s
```

The screenshot shows a code editor with the following C++ code for Insertion Sort:

```
1 //Created by 23343066_Gema Pratama
2 #include <stdio.h>
3
4 void insertionSort(int arr[], int n)
5 {
6     for (int i = 1; i < n; i++)
7     {
8         int key = arr[i];
9         int j = i - 1;
10        while (j >= 0 && arr[j] > key)
11        {
12            arr[j + 1] = arr[j];
13            j = j - 1;
14        }
15        arr[j + 1] = key;
16    }
17 }
18
19 int main()
20 {
21     int arr[] = {17, 31, 25, 12, 22, 11, 3};
22     int n = sizeof(arr) / sizeof(arr[0]);
23     insertionSort(arr, n);
24     printf("Sorted array: \n");
25     for (int i = 0; i < n; i++)
26     {
27         printf("%d ", arr[i]);
28     }
29     return 0;
30 }
```

The console output shows the sorted array: 3 11 12 17 22 25 31. The process exited after 0.0429 seconds with return value 0.

Penjelasan:

- Algoritma insertion sort bekerja dengan memasukkan elemen satu per satu ke dalam array yang telah diurutkan.
- Elemen baru dibandingkan dengan elemen-elemen yang telah diurutkan sebelumnya.
- Jika elemen baru lebih kecil daripada elemen yang telah diurutkan, maka elemen-elemen yang telah diurutkan digeser ke kanan untuk memberikan ruang bagi elemen baru.
- Proses ini diulang sampai semua elemen dimasukkan ke dalam array.
- Prinsip utama insertion sort adalah menyisipkan elemen-elemen ke dalam array yang telah diurutkan seperti memasukkan kartu ke dalam tumpukan kartu yang sudah diurutkan.
- Insertion 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.