

Tugas Jobsheet 8



Dosen pengampu : Randi Proska Sandra, M.Sc

Kode Kelas : 202323430158

Disusun Oleh :

**Fajrul Huda Ash Shiddiq
23343063**

**PROGRAM STUDI INFORMATIKA (NK)
FAKULTAS TEKNIK
UNIVERSITAS NEGERI PADANG
2023**

Bubble Sort

1. Source Code

```
//Created by_Fajrul Huda Ash Shiddiq_23343063
#include <stdio.h>

void bubbleSort(int arr[], int n) {
    int i, j, temp;
    for (i = 0; i < n-1; i++) {
        for (j = 0; j < n-i-1; j++) {
            if (arr[j] > arr[j+1]) {
                // Tukar elemen jika elemen saat ini lebih besar
                // dari elemen berikutnya
                temp = arr[j];
                arr[j] = arr[j+1];
                arr[j+1] = temp;
            }
        }
    }
}

int main() {
    int arr[] = {6, 34, 25, 12, 27, 1, 50};
    int n = sizeof(arr)/sizeof(arr[0]);
    bubbleSort(arr, n);
    printf("Array yang sudah diurutkan: \n");
    for (int i = 0; i < n; i++)
        printf("%d ", arr[i]);
    printf("\n");
    return 0;
}
```

2. Screenshot

The screenshot displays a C++ program implementing the bubble sort algorithm. The code is written in a dark-themed IDE. The program defines a `bubbleSort` function that takes an array `arr` and its size `n` as parameters. It uses nested loops to compare adjacent elements and swap them if they are in the wrong order. The `main` function initializes an array `arr` with the values `{6, 34, 25, 12, 27, 1, 50}`, calls `bubbleSort`, and prints the sorted array.

```
1 //Created by Fajrul Huda Ash Shiddiq_23343063
2 #include <stdio.h>
3
4 void bubbleSort(int arr[], int n) {
5     int i, j, temp;
6     for (i = 0; i < n-1; i++) {
7         for (j = 0; j < n-i-1; j++) {
8             if (arr[j] > arr[j+1]) {
9                 // Tukar elemen jika elemen saat ini lebih besar dari elemen berikutnya
10                temp = arr[j];
11                arr[j] = arr[j+1];
12                arr[j+1] = temp;
13            }
14        }
15    }
16 }
17
18 int main() {
19     int arr[] = {6, 34, 25, 12, 27, 1, 50};
20     int n = sizeof(arr)/sizeof(arr[0]);
21     bubbleSort(arr, n);
22     printf("Array yang sudah diurutkan: \n");
23     for (int i = 0; i < n; i++) {
24         printf("%d ", arr[i]);
25     }
26     printf("\n");
27     return 0;
28 }
```

The console output shows the sorted array: `1 6 12 25 27 34 50`. The process exited after 6.352 seconds with a return value of 0.

3. Penjelasan

a. Fungsi 'void bubbleSort':

- Fungsi ini merupakan implementasi dari algoritma Bubble Sort.
- Parameter `arr` adalah array yang akan diurutkan.
- Parameter `n` adalah jumlah elemen dalam array.
- Fungsi ini tidak mengembalikan nilai (void).
- Algoritma melakukan iterasi melalui array sebanyak `n-1` kali, dimana `n` adalah jumlah elemen dalam array.
- Pada setiap iterasi, algoritma membandingkan dua elemen berdekatan dalam array.

- Jika elemen pertama lebih besar dari elemen kedua, mereka ditukar.
- Proses ini diulangi hingga tidak ada lagi pertukaran yang perlu dilakukan.
- Fungsi ini melakukan pengurutan array arr secara ascending menggunakan algoritma Bubble Sort.

b. Fungsi main():

- Fungsi utama dari program.
- Array arr yang akan diurutkan dideklarasikan dan diinisialisasi.
- Ukuran array dihitung dengan membagi ukuran total array dengan ukuran elemen tunggal array.
- Fungsi bubbleSort() dipanggil untuk mengurutkan array arr.
- Hasil pengurutan dicetak ke layar.

Insertion Sort

1. Source Code

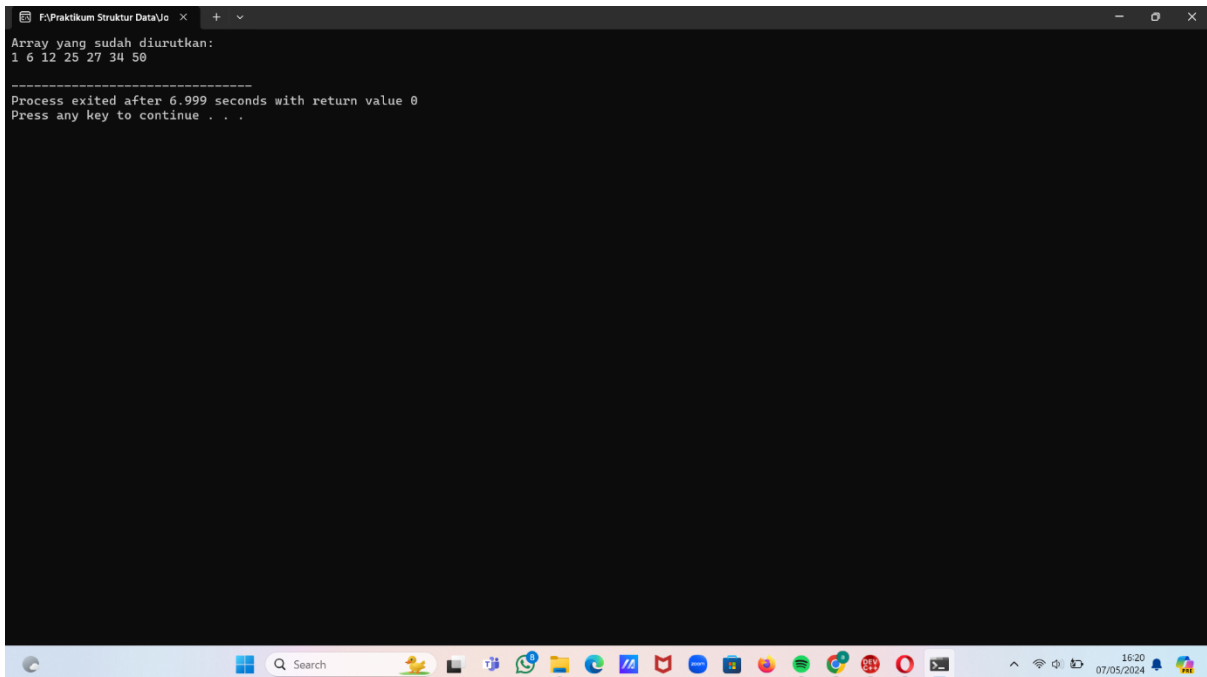
```
//Created by_Fajrul Huda Ash Shiddiq_23343063
#include <stdio.h>

void insertionSort(int arr[], int n) {
    int i, key, j;
    for (i = 1; i < n; i++) {
        key = arr[i];
        j = i - 1;

        /* Pindahkan elemen arr[0..i-1] yang lebih besar dari key
           ke satu posisi di depan posisinya saat ini */
        while (j >= 0 && arr[j] > key) {
            arr[j + 1] = arr[j];
            j = j - 1;
        }
        arr[j + 1] = key;
    }
}
```

```
int main() {  
    int arr[] = {6, 34, 25, 12, 27, 1, 50};  
    int n = sizeof(arr)/sizeof(arr[0]);  
    insertionSort(arr, n);  
    printf("Array yang sudah diurutkan: \n");  
    for (int i = 0; i < n; i++)  
        printf("%d ", arr[i]);  
    printf("\n");  
    return 0;  
}
```

2. Screenshot



```
F:\Praktikum Struktur Data\Jo  
Array yang sudah diurutkan:  
1 6 12 25 27 34 50  
-----  
Process exited after 6.999 seconds with return value 0  
Press any key to continue . . .
```

```
1 //Created by: Fajrut Huda Ash Shiddiq_23343063
2 #include <iostream>
3
4 void insertionSort(int arr[], int n) {
5     int i, key, j;
6     for (i = 1; i < n; i++) {
7         key = arr[i];
8         j = i - 1;
9
10        /* Pindahkan elemen arr[0..i-1] yang lebih besar dari key
11        ke satu posisi di depan posisinya saat ini */
12        while (j >= 0 && arr[j] > key) {
13            arr[j + 1] = arr[j];
14            j = j - 1;
15        }
16        arr[j + 1] = key;
17    }
18 }
19
20 int main() {
21     int arr[] = {6, 34, 25, 12, 27, 1, 50};
22     int n = sizeof(arr)/sizeof(arr[0]);
23     insertionSort(arr, n);
24     printf("Array yang sudah diurutkan: \n");
25     for (int i = 0; i < n; i++)
26         printf("%d ", arr[i]);
27     printf("\n");
28     return 0;
29 }
30
```

Compiler (1) | Resources | Compile Log | Debug | Find Results | Console | Close

Warnings: 0
Output Filename: F:\Praktikum Struktur Data\Jobsheet 8\Tugas\Project\InsertionSort.exe
Output size: 323,466320125 Kib
Compilation Time: 1,30s

Line: 21 Col: 44 Sel: 0 Lines: 30 Length: 757 Insert Done parsing in 0,015 seconds

3. Penjelasan

a. Fungsi 'void insertionSort':

- Fungsi ini merupakan implementasi dari algoritma Insertion Sort.
- Parameter arr adalah array yang akan diurutkan.
- Parameter n adalah jumlah elemen dalam array.
- Fungsi ini tidak mengembalikan nilai (void).
- Algoritma memisahkan array menjadi dua bagian: bagian yang terurut dan bagian yang belum terurut.
- Dimulai dari indeks kedua ($i = 1$), algoritma memilih elemen satu per satu dan membandingkannya dengan elemen-elemen sebelumnya dalam bagian yang sudah terurut.
- Jika elemen yang dipilih lebih kecil dari elemen sebelumnya, elemen tersebut digeser ke kanan untuk memberi ruang bagi elemen yang lebih besar.
- Proses ini diulangi hingga tidak ada lagi elemen yang harus disisipkan.
- Fungsi ini melakukan pengurutan array arr secara ascending menggunakan algoritma Insertion Sort.

b. Fungsi 'int main()'

- Fungsi utama dari program.
- Array arr yang akan diurutkan dideklarasikan dan diinisialisasi.
- Ukuran array dihitung dengan membagi ukuran total array dengan ukuran elemen tunggal array.
- Fungsi insertionSort() dipanggil untuk mengurutkan array arr.
- Hasil pengurutan dicetak ke layar.