

# **LAPORAN PRAKTIKUM**

## **“Pertemuan ke-5: Algoritma Decrease & Conquer”**

Diajukan untuk memenuhi salah satu praktikum Mata Struktur Data Informatika yang di  
ampu oleh:

Dwi Normawati, S.T., M.Eng.



Disusun Oleh:

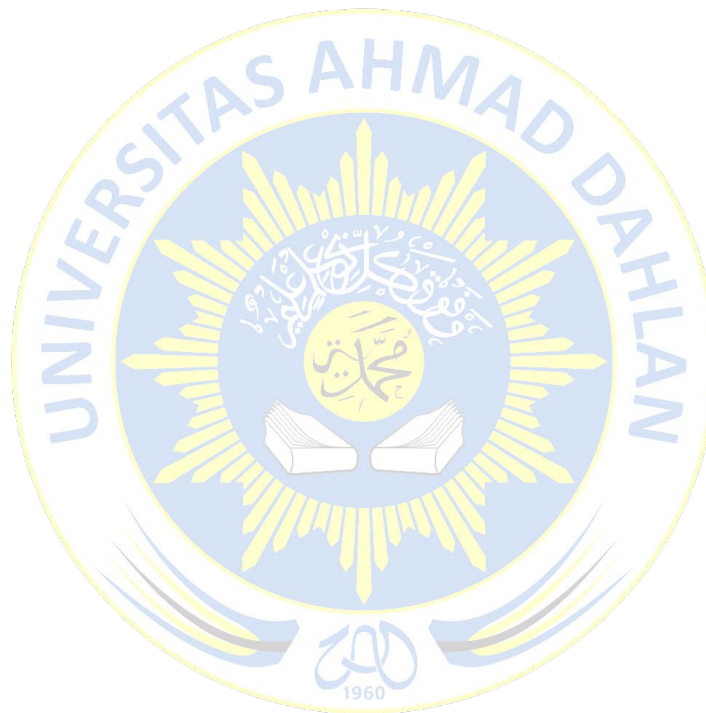
Mohammad Farid Hendianto 2200018401

A / Selasa 07.00– 08.15 Lab. Komputasi Dasar

**PROGRAM STUDI INFORMATIKA  
UNIVERSITAS AHMAD DAHLAN  
FAKULTAS TEKNOLOGI INDUSTRI  
TAHUN 2024**

## DAFTAR ISI

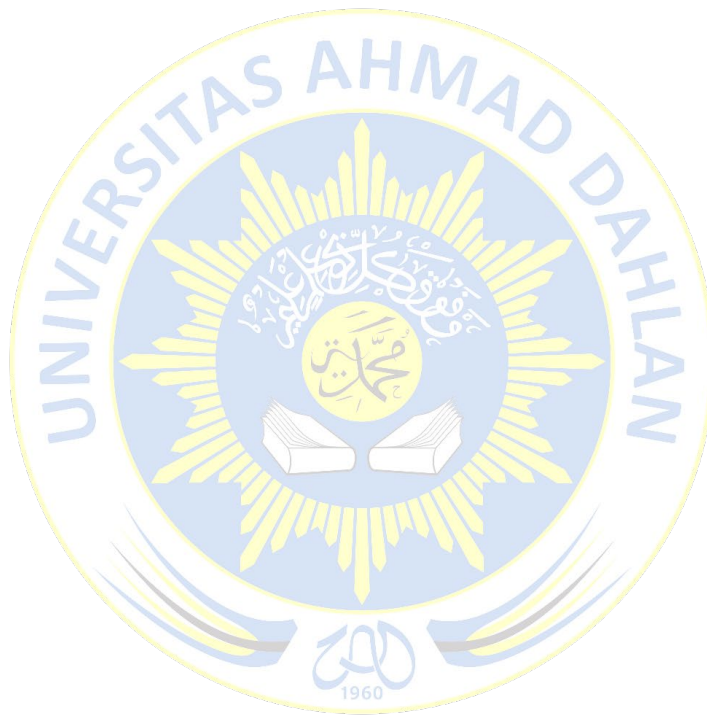
PRETEST .....	3
LANGKAH PRAKTIKUM.....	6
POST TEST .....	9



## PRETEST

1. Jelaskan 2 tahapan Decrease & Conquer
2. Sebutkan dan jelaskan 3 varian decrease & conquer
3. Jelaskan perbedaan algoritma decrease and conquer dengan divide and conquer!
4. Sebutkan algoritma selain selection sort yang menggunakan Teknik decrease and conquer!

**Jawaban:**



## LEMBAR JAWABAN PRE-TEST DAN POST-TEST PRAKTIKUM

Nama: Mohammad Farid H. NIM: 2200018401	Asisten: Paraf Asisten:	Tanggal: 1 May 2024 Nilai:
--	----------------------------	-------------------------------

1. Decrease and conquer terdiri dari dua tahapan:
  - 1) Decrease: Mereduksi persoalan menjadi beberapa persoalan yang lebih kecil. Biasanya menjadi dua sub-persoalan. Proses ini mirip dengan tahap 'divide' pada divide and conquer, namun berbeda dalam hal jumlah sub-persoalan yang dihasilkan dan proses.
  - 2) Conquer: Setelah persoalan telah direduksi, tahap 'conquer' memproses satu sub-persoalan secara rekursif hingga mendapatkan solusinya. Berbeda dengan Divide and Conquer yang memproses semua sub-persoalan dan menggabungkan hasilnya.
2. Tiga varian decrease & conquer:
  - Decrease by a constant: Ukuran instances persoalan direduksi sebesar konstanta yang sama setiap iterasi algoritma. Biasanya konstanta = 1.
  - Decrease by a constant factor: Ukuran instances persoalan direduksi sebesar factor konstanta yang sama setiap iterasi algoritma. Biasanya factor konstanta = 2.
  - Decrease by a variable size: Ukuran instances persoalan direduksi bervariasi pada setiap iterasi algoritma.
3. Perbedaan algoritma decrease and conquer dengan Divide and conquer:
  - Decrease and Conquer: Mereduksi persoalan menjadi beberapa sub-persoalan yang lebih kecil dan hanya memproses satu sub-persoalan tersebut. Tidak ada tahap 'combine' dalam algoritma ini.
  - Divide and Conquer: Persoalan menjadi sub-persoalan, memproses semua sub-persoalan, dan menggabungkan semua solusi setiap sub-persoalan. Proses ini melibatkan tahap 'combine' untuk menggabungkan hasil dari semua sub-persoalan.

4. Algoritma selain Selection sort yang menggunakan teknik Decrease and Conquer:

- Insertion sort: Algoritma ini juga menggunakan teknik decrease and conquer. Pada setiap iterasi, algoritma ini mengambil satu elemen dari data yang belum diurutkan dan menempatkannya pada posisi yang tepat didalam data yang sudah diurutkan.
- DFS (Depth-First Search): Algoritma ini digunakan untuk menjelajahi atau mencari elemen dalam graf. DFS menggunakan teknik decrease and conquer dengan cara mengunjungi satu cabang terlebih dahulu sebelum kembali dan mengunjungi cabang lain.
- Binary search: Algoritma ini mencari elemen tertentu dalam array yang sudah diurutkan dengan cara membagi array menjadi dua bagian dan hanya mencari satu bagian yang mungkin mengandung elemen yang dicari.

## LANGKAH PRAKTIKUM

1. Ketik program selection sort dibawah ini, kemudian lakukan Run program!

Source Kodingan:

```
// C++ program for implementation of selection sort
#include <bits/stdc++.h>
using namespace std;

void swap(int *xp, int *yp) {
    int temp = *xp;
    *xp = *yp;
    *yp = temp;
}

void selectionSort(int arr[], int n) {
    int i, j, min_idx;

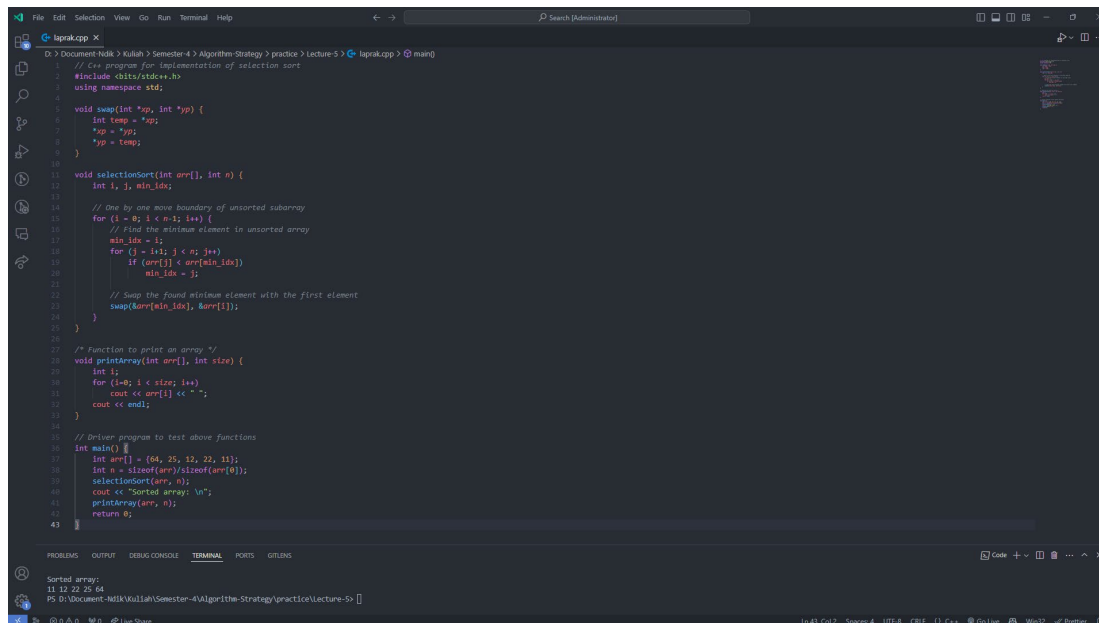
    // One by one move boundary of unsorted subarray
    for (i = 0; i < n-1; i++) {
        // Find the minimum element in unsorted array
        min_idx = i;
        for (j = i+1; j < n; j++)
            if (arr[j] < arr[min_idx])
                min_idx = j;

        // Swap the found minimum element with the first element
        swap(&arr[min_idx], &arr[i]);
    }
}

/* Function to print an array */
void printArray(int arr[], int size) {
    int i;
    for (i=0; i < size; i++)
        cout << arr[i] << " ";
    cout << endl;
}

// Driver program to test above functions
int main() {
```

```
int arr[] = {64, 25, 12, 22, 11};
int n = sizeof(arr)/sizeof(arr[0]);
selectionSort(arr, n);
cout << "Sorted array: \n";
printArray(arr, n);
return 0;
}
```



Gambar kodingan beserta hasil eksekusi

2. Kemudian lakukan eksekusi program!

Hasil Eksekusi

```
PS D:\Document-Ndik\Kuliah\Semester-4\Algorithm-Strategy\practice\Lecture-5> cd "d:\Document-Ndik\Kuliah\Semester-4\Algorithm-Strategy\practice\Lecture-5\"; if ($?) { g++ -Ofast -funroll-loops -march=native -static "laprak.cpp" -o "laprak.exe" -lpthread -lm }; if ($?) { & ".\laprak.exe" }
Sorted array:
11 12 22 25 64
PS D:\Document-Ndik\Kuliah\Semester-4\Algorithm-Strategy\practice\Lecture-5>
```

3. Analisis hasil output:

Misalkan tabel berisi elemen-elemen berikut:

64 25 12 22 11



### Selection sort (Decrease by a constant)

Langkah-langkah pengurutan dengan Selection Sort:

- Iterasi 1 ( $i = 0$ ):

**Mencari Minimum:** Elemen terkecil (**11**) ditemukan pada indeks **4**.

**Penukaran:** Elemen pada indeks 0 (**64**) ditukar dengan elemen terkecil:

**11 25 12 22 64**

- Iterasi 2 ( $i = 1$ ):

**Mencari Minimum:** Elemen terkecil (**12**) ditemukan pada indeks **2**.

**Penukaran:** Elemen pada indeks **1** (**25**) ditukar dengan elemen terkecil:

**11 12 25 22 64**

- Iterasi 3 ( $i = 2$ ):

**Mencari Minimum:** Elemen terkecil (**22**) ditemukan pada indeks **3**.

**Penukaran:** Elemen pada indeks **2** (**25**) ditukar dengan elemen terkecil:

**11 12 22 25 64**

- Iterasi 4 ( $i = 3$ ):

**Mencari Minimum:** Elemen terkecil (**25**) sudah berada pada posisinya.

**Tidak ada penukaran.**

- Hasil Akhir (Array Terurut):

**11 12 22 25 64**

Hasilnya akan sama dengan output kodingan

```
($?) { & ".\laprak.e
Sorted array:
11 12 22 25 64
PS D:\Document-Ndik\K
```



## POST TEST

Buatlah output seperti berikut!

```

Masukkan jumlah data yang akan di urutkan : 7
=====
Masukkan data ke-1 : 8
Masukkan data ke-2 : 15
Masukkan data ke-3 : 13
Masukkan data ke-4 : 7
Masukkan data ke-5 : 1
Masukkan data ke-6 : 3
Masukkan data ke-7 : 2
=====
Langkah ke-1: 8 15 13 7 1 3 2
Langkah ke-2: 1 15 13 7 8 3 2
Langkah ke-3: 1 2 13 7 8 3 15
Langkah ke-4: 1 2 3 7 8 13 15
Langkah ke-5: 1 2 3 7 8 13 15
Langkah ke-6: 1 2 3 7 8 13 15
=====
Sorted array:
1 2 3 7 8 13 15

```

Jawab:

Test unit (copy kan ke dalam input kodingan)

7
8
15

13
7
1
3
2

Source code:

```
#include <bits/stdc++.h>
using namespace std;

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;

        swap(arr[min_idx], arr[i]);

        cout << "Langkah ke-" << i+1 << ": ";
        for (int k = 0; k < n; k++) {
            cout << arr[k] << " ";
        }
        cout << endl;
    }
}

int main() {
    int n;
    cout << "Masukkan jumlah data yang akan di urutkan: ";
    cin >> n;
    int arr[n];

    cout << "=====" << endl;
    for (int i = 0; i < n; i++) {
        cout << "Masukkan data ke-" << i+1 << ": ";
        cin >> arr[i];
    }
    cout << "=====" << endl;
```

```

selectionSort(arr, n);

cout << "===== " << endl;

cout << "Sorted array: " << endl;

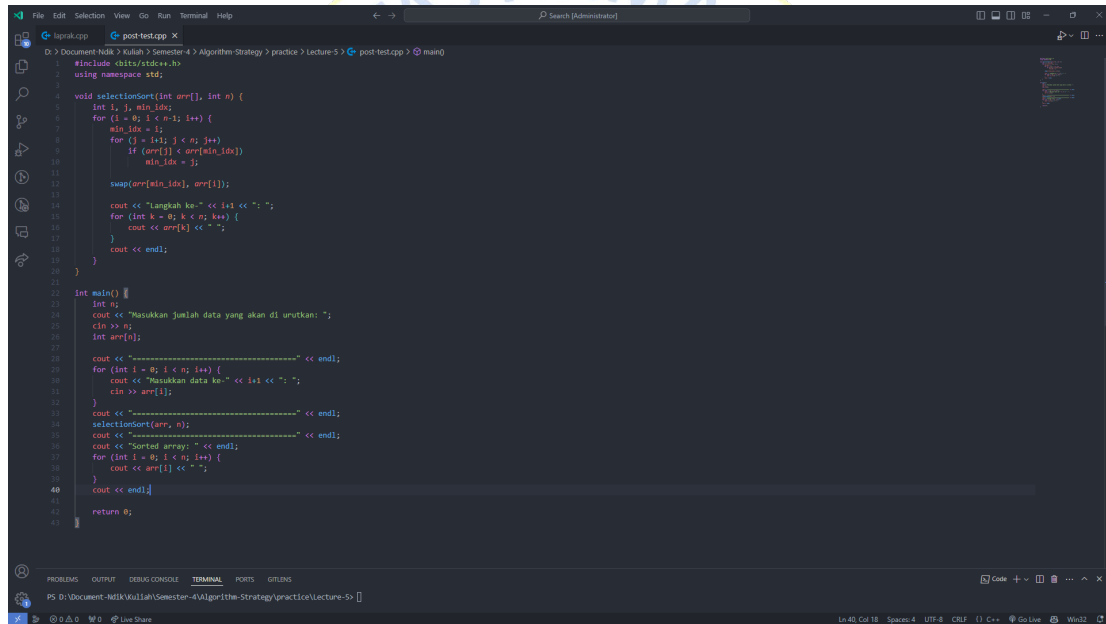
for (int i = 0; i < n; i++) {
    cout << arr[i] << " ";
}

cout << endl;

return 0;
}

```

Kodingan di vs code:

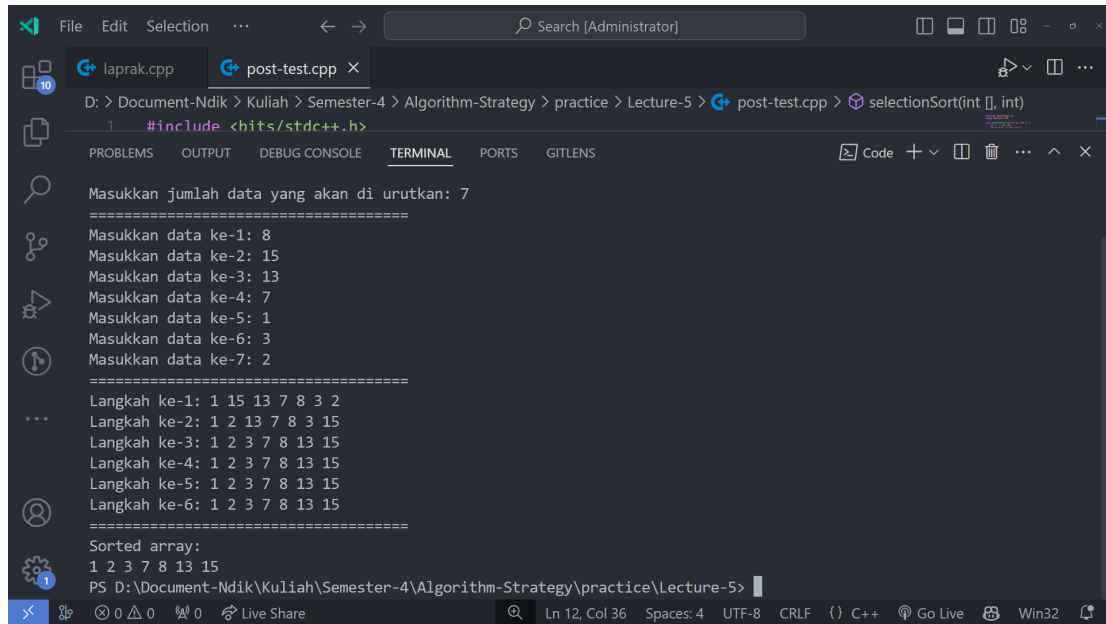


```

1  #include <bits/stdc++.h>
2  using namespace std;
3
4  void selectionSort(int arr[], int n) {
5      int i, j, min_idx;
6      for (i = 0; i < n-1; i++) {
7          min_idx = i;
8          for (j = i+1; j < n; j++)
9              if (arr[j] < arr[min_idx])
10                 min_idx = j;
11
12             swap(arr[min_idx], arr[i]);
13
14             cout << "Langkah ke-" << i+1 << ": ";
15             for (int k = 0; k < n; k++) {
16                 cout << arr[k] << " ";
17             }
18             cout << endl;
19         }
20     }
21
22     int main() {
23         int n;
24         cout << "Masukkan jumlah data yang akan di urutkan: ";
25         cin >> n;
26         int arr[n];
27
28         cout << "===== " << endl;
29         for (int i = 0; i < n; i++) {
30             cout << "Masukkan data ke-" << i+1 << ": ";
31             cin >> arr[i];
32         }
33         cout << "===== " << endl;
34         selectionSort(arr, n);
35         cout << "Sorted array: " << endl;
36         for (int i = 0; i < n; i++) {
37             cout << arr[i] << " ";
38         }
39         cout << endl;
40         return 0;
41     }

```

Output:



```

File Edit Selection ... Search [Administrator]
laprak.cpp post-test.cpp x
D: > Document-Ndik > Kuliah > Semester-4 > Algorithm-Strategy > practice > Lecture-5 > post-test.cpp > selectionSort(int [], int)
1 #include <bits/stdc++.h>

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS
Code + - - - - -

Masukkan jumlah data yang akan di urutkan: 7
=====
Masukkan data ke-1: 8
Masukkan data ke-2: 15
Masukkan data ke-3: 13
Masukkan data ke-4: 7
Masukkan data ke-5: 1
Masukkan data ke-6: 3
Masukkan data ke-7: 2
=====
Langkah ke-1: 1 15 13 7 8 3 2
Langkah ke-2: 1 2 13 7 8 3 15
Langkah ke-3: 1 2 3 7 8 13 15
Langkah ke-4: 1 2 3 7 8 13 15
Langkah ke-5: 1 2 3 7 8 13 15
Langkah ke-6: 1 2 3 7 8 13 15
=====
Sorted array:
1 2 3 7 8 13 15
PS D:\Document-Ndik\Kuliah\Semester-4\Algorithm-Strategy\practice\Lecture-5>

```

Perbedaan kodingan sebelumnya:

- Menambahkan step by step di dalam fungsi selection sort

Hal ini agar bisa mencetak langkah ke- (i+1) dan bagaimana isi array melakukan eksekusi step bystep

```
3
4 void selectionSort(int arr[], int n) {
5     int i, j, min_idx;
6     for (i = 0; i < n-1; i++) {
7         min_idx = i;
8         for (j = i+1; j < n; j++)
9             if (arr[j] < arr[min_idx])
10                min_idx = j;
11
12        swap(arr[min_idx], arr[i]);
13
14        cout << "Langkah ke-" << i+1 << ": ";
15        for (int k = 0; k < n; k++) {
16            cout << arr[k] << " ";
17        }
18        cout << endl;
19    }
20 }
21
```

Tambahkan solusi step by step agar mengetahui eksekusi di dalam array

- Input dinamis

Karena ini berbentuk array, diperlukan inisialisasi jumlah data kemudian baru masukkan data satu persatu dengan perulangan

```
21
22 int main() {
23     int n;
24     cout << "Masukkan jumlah data yang akan di urutkan: ";
25     cin >> n;
26     int arr[n];
27
28     cout << "===== " << endl;
29     for (int i = 0; i < n; i++) {
30         cout << "Masukkan data ke-" << i+1 << ": ";
31         cin >> arr[i];
32     }
33     cout << "===== " << endl;
```

- Perbedaan lainnya kurang lebih sama, saya tidak menginisialisasi fungsi swap (karena sudah ada di c++, jadi tidak perlu dibuat lagi fungsinya), dan menambahkan break menggunakan “=====”

