

NAMA : FIFIT SYAFAATY

NIM : 21091397001

KELAS : A

TUGAS STRUKTUR DATA

LAPORAN INDIVIDU (RADIX SORT)

- **PENGERTIAN**

Radix Sort adalah algoritma atau bahkan strategi pengurutan (sorting) tanpa perbandingan bersama istilah satu lagi, sorting Non-Comparison type dimana berat prosesnya tidak mengadakan evaluasi antar informasi. Kata radix relevan harafiah penempatan berat angka.

Di mana sederhananya, berat lukisan desimal, radix adalah digitnya. Dalam implementasinya, Radix Sort merupakan algoritma pengurutan yang segera, nyaman, dan secara substansial membantu. Namun tak terhitung yang mengira bahwa algoritma radix {memiliki} tak terhitung pembatasan di mana untuk kasus-kasus tertentu tidak dapat dilakukan bersama algoritma ini, seperti pengurutan bilangan bagian dan bilangan unfavourable.

- **CARA KERJA**

Dalam posting buku harian yang ditulis sabarudi *et al*, Proses sederhana Radix Sort adalah mengkategorikan data-data akhirnya menjadi sub kumpulan informasi ideal bersama nilai pasar radix-nya (klasifikasi tertentu), dimana berat tiap kategorinya dilakukan pengklasifikasian bahkan lebih dan seterusnya ideal bersama butuh mengkonkatenasinya, dan subkategori-subkategori tersebut digabungkan dapatkan, yang secara dilakukan seluruhnya bersama strategi kecil *concatenation*.

- Buat kode C++ untuk seetiap sort!
- Codingan Radix sort

Page 1

```

1 //fifit syafaaty_21091397001
2
3 #include<iostream>
4 #include<list>
5 #include<cmath>
6 using namespace std;
7 void display(int *array, int size) {
8     for(int i = 0; i<size; i++)
9         cout << array[i] << " ";
10    cout << endl;
11}
12 void radixSort(int *arr, int n, int max) {
13    int i, j, m, p = 1, index, temp, count = 0;
14    list<int> pocket[10];
15    //radix bit, desimal 10
16    for(i = 0; i< max; i++) {
17        m = pow(10, i+1);
18        p = pow(10, i);
19        for(j = 0; j<n; j++) {
20            temp = arr[j]%m;
21            index = temp/p;
22            //index array
23            pocket[index].push_back(arr[j]);
24        }
25        count = 0;
26        for(i = 0; i<10; i++) {

```

Compilation results...

```

- Errors: 0
- Warnings: 0
- Output Filename: D:\STRUKTUR DATA\ fifit syafaaty_001_Radix sort.exe
- Output Size: 1,07254905700684 MiB
- Compilation Time: 1,33s

```

Page 2

```

29         arr[count] = *(pocket[j].begin());
30         pocket[j].erase(pocket[j].begin());
31         count++;
32     }
33 }
34
35 int main() {
36     int n, max;
37     cout << "masukkan jumlah array: ";
38     cin >> n;
39     cout << "maksimum array: ";
40     cin >> max;
41     int arr[n];
42     //buat array
43     cout << "isi array : " << endl;
44     for(int i = 0; i<n; i++) {
45         cin >> arr[i];
46     }
47     cout << "data yang belum disorting: ";
48     display(arr, n);
49     radixSort(arr, n, max);
50     cout << "data yang sudah disorting: ";
51     display(arr, n);
52 }
53

```

Compilation results...

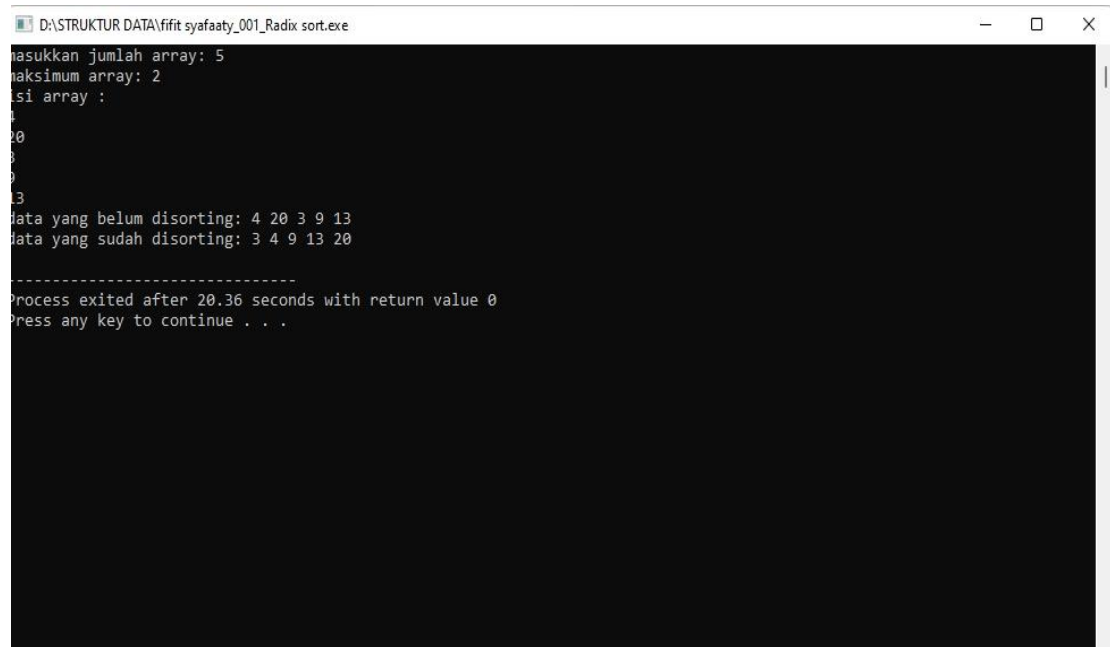
```

- Errors: 0
- Warnings: 0
- Output Filename: D:\STRUKTUR DATA\ fifit syafaaty_001_Radix sort.exe
- Output Size: 1,07254905700684 MiB
- Compilation Time: 1,33s

```

Line: 1 Col: 1 Sel: 0 Lineu: 53 Length: 1311 Inset Done parsing in 0.015 seconds

- Bukti hasil Run



```
D:\STRUKTUR DATA\fit syafaaty_001_Radix sort.exe
Masukkan jumlah array: 5
Maksimum array: 2
Isi array :
4
20
3
9
13
Data yang belum disorting: 4 20 3 9 13
Data yang sudah disorting: 3 4 9 13 20

-----
Process exited after 20.36 seconds with return value 0
Press any key to continue . . .
```

- Hitung jenis big O. jelaskan kenapa kompleksitasnya adalah yang anda temukan!

Jawab :

Pengurutan radix membutuhkan waktu $O(\ell \cdot (n + k))$ dan ruang $O(n + k)$, di mana n adalah jumlah item yang akan diurutkan, ℓ adalah jumlah digit dalam setiap item, dan k adalah jumlah nilai yang dapat dimiliki setiap digit.

Kompleksitas waktu ini berasal dari fakta bahwa kita memanggil pengurutan penghitungan satu kali untuk setiap digit ℓ dalam angka input, dan pengurutan penghitungan memiliki kompleksitas waktu $O(n + k)$

Kompleksitas ruang juga berasal dari pengurutan pencacahan, yang membutuhkan ruang $O(n + k)$ untuk menampung jumlah, indeks, dan vektor keluaran.

- Jelaskan kelebihan dan kekurangan sorting yang kalian buat dan bandingkan yang dibuat oleh teman kalian!

Jawab :

❖ Kelebihan radix sort:

-Algoritma sangat mangkus. Hal ini bisa dilihat dari kompleksitas waktu asimptotiknya yang ber ukuran kecil. Ini menyebabkan algoritma radix sort sangat efektif untuk data dalam jumlah yang berukuran besar sekalipun

-Untuk konsep algoritma mudah dipahami. Algoritma radix sort mengurutkan data berdasarkan digit, tidak melalui proses perbandingan yang cenderung sulit dipahami

❖ Kekurangan radix sort:

-realisasi program rumit

-kurang fleksibel untuk dipakai pada tipe data lain

