MAKALAH

TENTANG SORTING

Disusun oleh :

Nama	: Muhammad Ajra Zemima Muda
NIM	: 2023903430022
Kelas	: TRKJ 1.C
Jurusan	: Teknologi Informasi dan Komputer
Program Studi	: Teknologi Rekayasa Komputer Jaringan
Dosen Pengajar	: Indrawati, SST. MT

JURUSAN TEKNOLOGI INFORMASI KOMPUTER

PRODI TEKNOLOGI REKAYASA KOMPUTER JARINGAN

POLITEKNIK NEGERI LHOKSEUMAWE

TAHUN AJARAN 2023/202

KATA PENGANTAR

Puji syukur kami panjatkan ke hadirat Allah SWT, yang telah memberikan rahmat, hidayah, dan karunia-Nya sehingga kami dapat menyelesaikan makalah ini dengan judul "Shorting". Makalah ini disusun dalam rangka memenuhi tugas mata kuliah Struktur Data, dengan tujuan untuk memberikan pemahaman yang mendalam tentang salah satu struktur data penting, yaitu Shorting, serta bagaimana dalam pemrograma n bahasa C.

Makalah ini bertujuan untuk menjelaskan konsep dasar dari pengurutan data, mengeksplorasi beberapa algoritma pengurutan yang umum digunakan, dan mengungkap penerapan praktis dari proses sorting dalam pemrograman komputer. Melalui pemahaman mendalam tentang sorting, diharapkan pembaca dapat mengoptimalkan kinerja program, merancang solusi yang lebih efisien, dan memahami pentingnya konsep ini dalam pemrosesan dan analisis data. Dengan demikian, makalah ini diharapkan dapat memberikan wawasan yang berharga dalam dunia yang semakin tergantung pada kapasitas pemrosesan dan manajemen data yang efisien.

Semoga makalah ini dapat memberikan wawasan yang berguna dan mendalam tentang penggunaan Sorting dalam pemrograman dengan bahasa C. Terima kasih atas perhatian dan semangat untuk terus mengeksplorasi dunia yang kaya ini melalui pengembangan perangkat lunak.

DAFTAR ISI

BAB I

PENDAHULUAN

A. Latar Belakang

Sorting adalah suatu proses yang umumnya diterapkan dalam pemrograman komputer untuk mengurutkan elemen-elemen dalam suatu kumpulan data. Tujuan dari pengurutan ini adalah agar data dapat disusun dalam urutan tertentu, seperti urutan numerik atau alfabet, sehingga memudahkan pencarian, analisis, dan manipulasi data. Sorting menjadi salah satu konsep dasar dalam ilmu komputer dan pemrograman, dan terdapat berbagai algoritma yang dikembangkan untuk menyelesaikan tugas ini dengan efisien.

B. Tujuan

Memberikan pemahaman yang mendalam mengenai konsep sorting, termasuk kebutuhan dan kegunaan pengurutan dalam pemrograman komputer.Menjelaskan beberapa algoritma pengurutan yang umum digunakan, seperti Bubble Sort, Insertion Sort, Selection Sort, Merge Sort, dan Quick Sort. Tujuannya adalah memberikan gambaran tentang cara kerja masing-masing algoritma, kelebihan, dan kelemahan yang dapat dipertimbangkan dalam pemilihan algoritma sesuai kebutuhan.

BAB II

PEMBAHASAN

A. Pengertian Shorting

Sorting merupakan teknik untuk mengurutkan data yang acak hingga bisa tersusun rapi dari terkecil ke terbesar atau sebaliknya. Sorting dalam bahasa C adalah proses pengaturan atau pengurutan elemen-elemen dalam sebuah kumpulan data, seperti array, sehingga elemen-elemen tersebut disusun sesuai dengan urutan tertentu. Urutan tersebut dapat berupa urutan numerik (ascending atau descending) atau urutan berdasarkan kriteria tertentu, seperti urutan alfabet dalam kasus string.

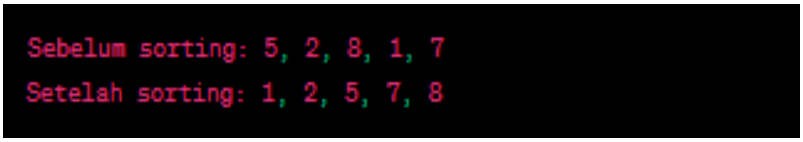
Sorting bertujuan untuk menyusun data agar dapat diakses dan dikelola dengan lebih efisien. Proses ini sangat penting dalam pemrograman karena memungkinkan pencarian data dengan lebih cepat, meningkatkan kinerja program, dan memfasilitasi analisis data.

Dalam bahasa C, sorting dapat diimplementasikan menggunakan berbagai algoritma pengurutan seperti Bubble Sort, Insertion Sort, Selection Sort, Merge Sort, dan Quick Sort. Pilihan algoritma bergantung pada kebutuhan aplikasi dan karakteristik data yang diurutkan. Setelah proses sorting selesai, data dapat diakses secara terurut, mempermudah operasi-operasi selanjutnya pada data tersebut. Sorting menjadi salah satu konsep dasar dalam ilmu komputer dan pemrograman, dan terdapat berbagai algoritma yang dikembangkan untuk menyelesaikan tugas ini dengan efisien.

Terdapat 2 macam pengurutan data pada sorting yaitu:Berdasarkan ascending (kecil ke besar) Dan berdasarkan Descending (besar ke kecil).

1. Ascending (Kecil ke Besar)

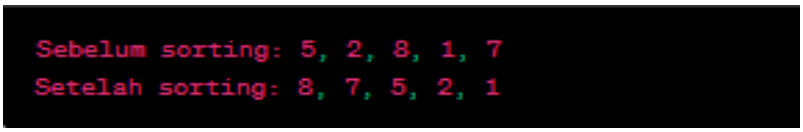
Dalam pengurutan ascending, elemen-elemen data disusun dari nilai terkecil ke nilai terbesar. Misalnya, jika kita mengurutkan sebuah array angka secara ascending, elemen-elemen diatur dari angka terkecil hingga yang terbesar.Contoh pengurutan ascending dari array angka:



Gambar 2. 1.Contoh Ascending

2. Descending (Besar ke Kecil)

Dalam pengurutan descending, elemen-elemen data disusun dari nilai terbesar ke nilai terkecil. Sebagai contoh, jika kita mengurutkan array angka secara descending, elemen-elemen diatur dari angka terbesar hingga yang terkecil.Contoh pengurutan descending dari array angka:



Gambar 2. 2.Contoh Descending

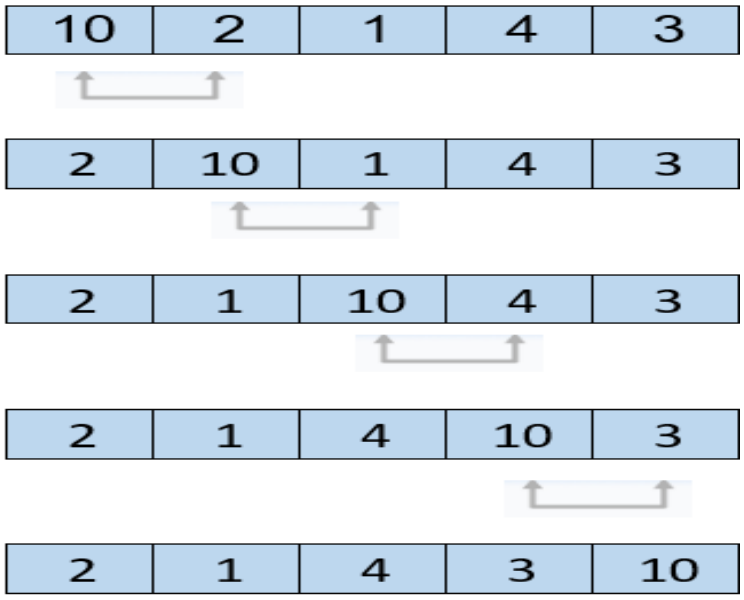
Pilihan antara ascending dan descending bergantung pada kebutuhan aplikasi atau persyaratan spesifik. Misalnya, ketika kita ingin menampilkan data dari yang terendah ke yang tertinggi, kita akan menggunakan ascending, sementara descending digunakan ketika kita ingin menyajikan data dari yang tertinggi ke yang terendah.

Ada berbagai macam algoritma pengurutan (sorting algorithms) yang dapat digunakan untuk mengurutkan data. Berikut adalah beberapa tipe sorting algorithma Yang ada.

B. Babble Short

Bubble sort (metode gelembung) adalah metode/algoritma pengurutan dengan dengan cara melakukan penukaran data dengan tepat disebelahnya secara terus menerus sampai bisa dipastikan dalam satu iterasi tertentu tidak ada lagi perubahan. Jika tidak ada perubahan berarti data sudah terurut. Disebut pengurutan gelembung karena masing-masing kunci akan dengan lambat menggelembung ke posisinya yang tepat.Bubble sort menggunakan teknik iterasi. mengurutkan data dalam urutan kecil ke besar (*Ascending*).

a. Iterasi Pertama (Bandingkan dan Tukar)

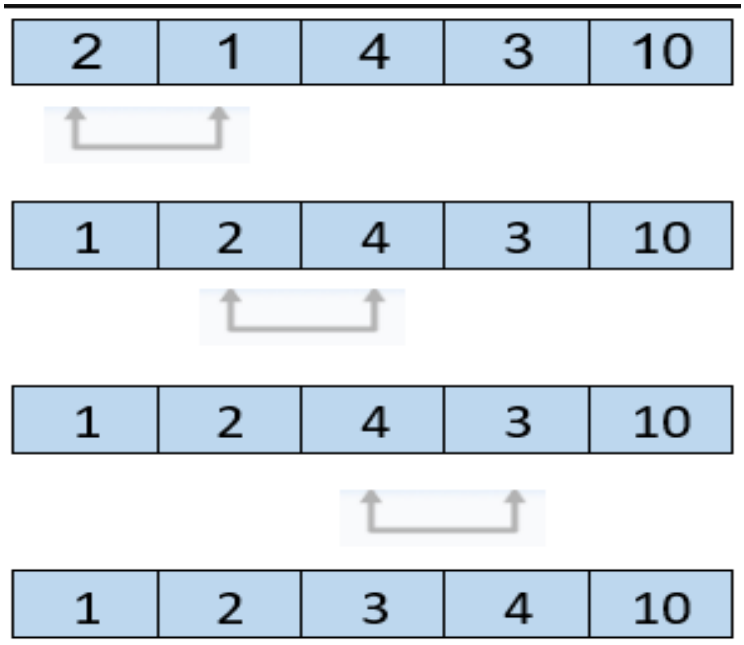


Gambar 2. 3.Contoh Iterasi Pertama

Penjelasan :

- Mulai dari indeks pertama, bandingkan data pertama dan kedua.
- Jika data pertama lebih besar dari data kedua, mereka ditukar. Karena contoh tersebut mengurutkan data berdasarkan kecil ke besar (ascending).
- Sekarang, bandingkan data kedua dan ketiga. Tukarkan jika tidak berurutan.
- Proses di atas berlangsung hingga elemen terakhir.

b. Iterasi Kedua (Melanjutkan proses iterasi dari iterasi pertama)



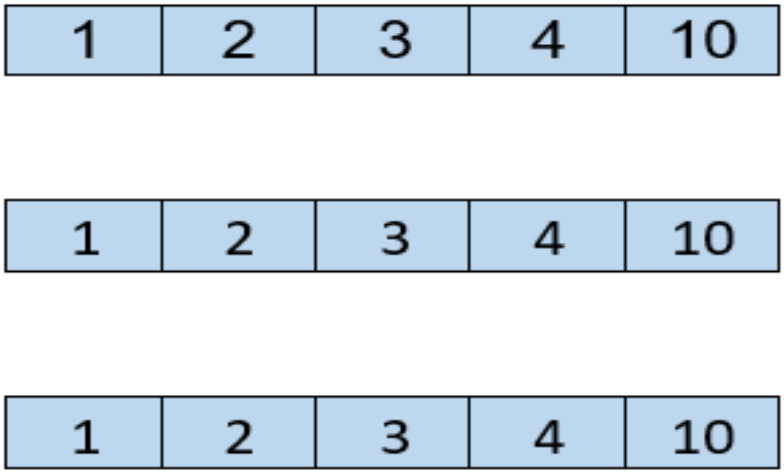
Gambar 2. 4.Contoh Iterasi Kedua

Penjelasan :

- Proses iterasi kedua melanjutkan hasil dari iterasi pertama.
- Proses pertukaran datanya pun sama saja. Jika data yang dibandingkan sudah benar maka tidak perlu terjadi pertukaran

c. Iterasi Ketiga

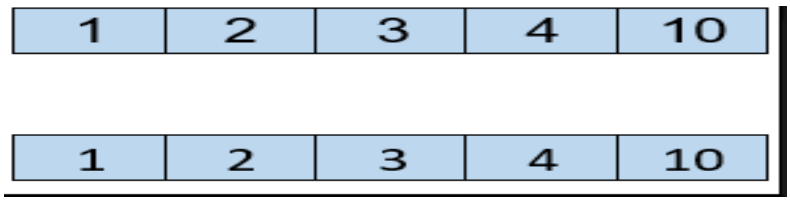
- Walaupun sudah ke sorting iterasi tetap berlanjut
- Karena iterasi menyesuaikan banyaknya data (contoh soal : 10 2 1 4 3).
- Karena ada 5 data maka harus 5-1 = 4 kali iterasi.
- Intinya pada iterasi membandingkan 2 data terus menerus.



Gambar 2. 5.Contoh Iterasi Ketiga

d. Iterasi Keempat

Iterasi keempat penjelasannya sama seperti dengan iterasi ketiga, yaitu banyaknya iterasi bergantung pada banyaknya data.



Gambar 2. 6.Contoh Iterasi Keempat

Kurang lebih seperti itu teori dari penggunaan bubble sort. Kamu dapat mengurutkan data dari kecil ke besar (ascending) atau dari besar ke kecil (descending). Prosesnya sama saja yaitu membandingkan dua buah data secara terus menerus.

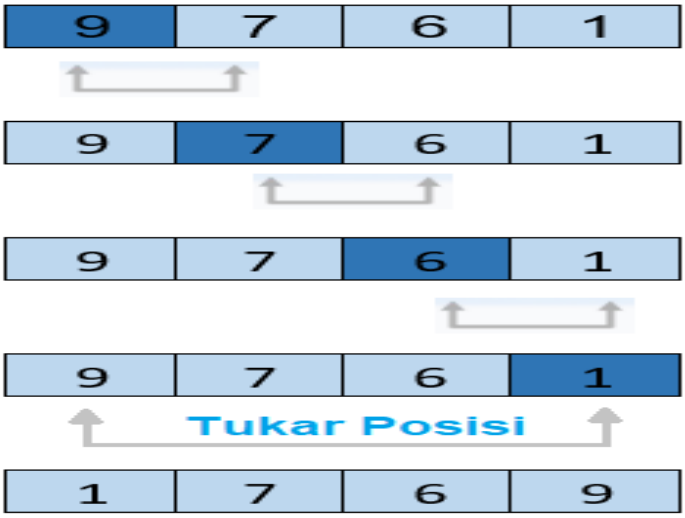
C. Selection Short

Selection sort menggunakan perbandingan data, selection sort akan membandingkan data 1 dengan lainnya dan jika menemukan data yang terkecil maka data tersebut menjadi acuannya.

Contohnya kali ini kita akan mengerjakan teori dari selection sort.Contoh soal diketahui terdapat 4 buah data, dengan angka 9,7,6,1. Urutkanlah data tersebut berdasarkan ascending.

Langkah 1 :

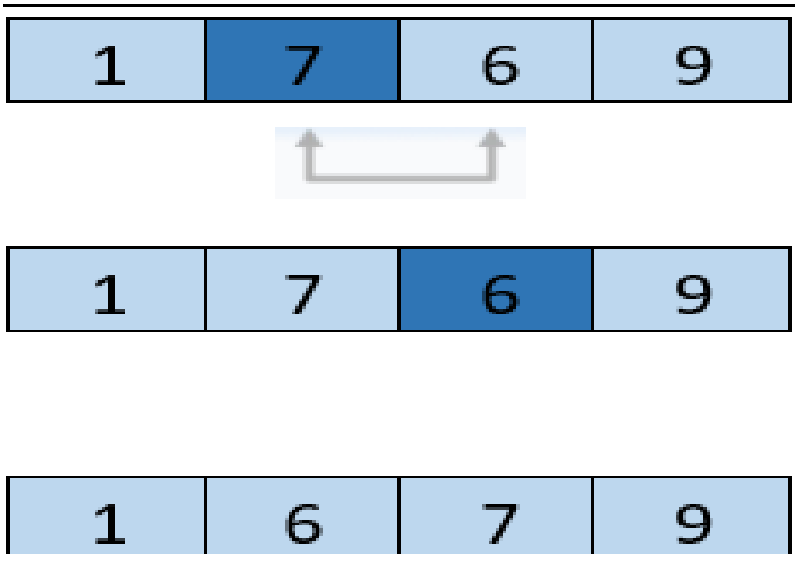
- Gunakanlah data pertama yaitu 9 sebagai acuannya, lalu bandingkan dengan data kedua yaitu 7. Karena 7 < 9 maka sekarang 7 yang menjadi acuannya.
- Bandingkan angka 7 dengan data selanjutnya yaitu 6. Karena 6 < 7 maka angka 6 menjadi acuannya.
- Kemudian bandingkan angka 6 dengan data selanjutya yaitu 1. Karena 1 < 6 maka angka 1 menjadi acuannya.
- Lalu tukar angka 1 dengan angka 9 yang menjadi angka acuan awal.



Gambar 2. 7.Langkah Ke 1 Selection Short

Langkah 2 :

- Bandingkan data kedua yaitu 7 dengan data selanjutnya yaitu 6. Karena 6 < 7 maka angka 6 menjadi acuannya.
- Kemudian tukar angka 6 dan 7

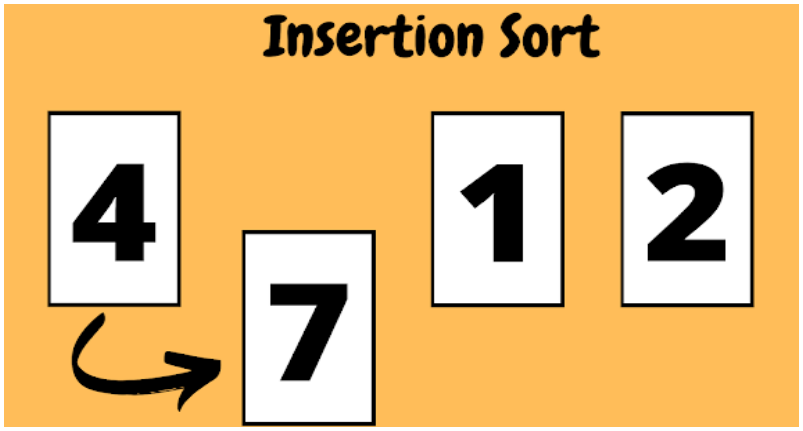


Gambar 2. 8.Langkah Ke 2 Selection Short

D. Insertion Short

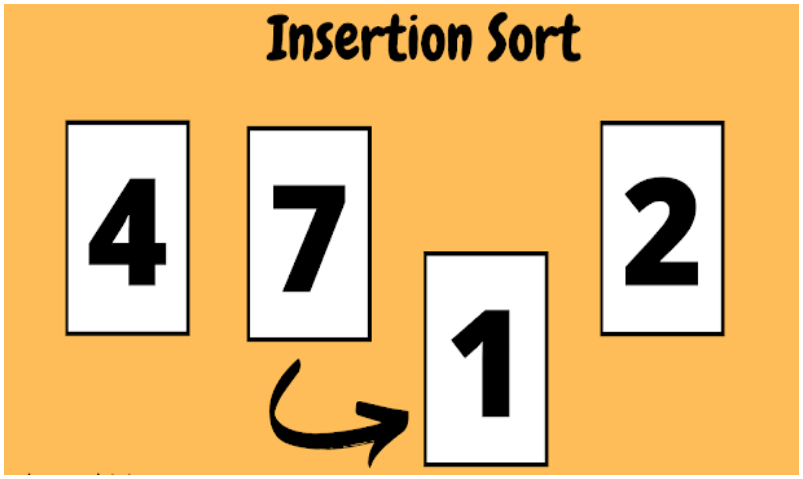
Insertion sort merupakan teknik sorting dengan cara menyisipkan atau memasukan setiap elemen secara berulang berulang.Konsep insertion sort bisa diibaratkan sebuah kartu. Untuk lebih jelasnya perhatikan gambar berikut.

Contohnya kita melakukan sort berdasarkan kecil ke besar (ascending) 4 buah data yaitu {4,7,1,2}.Langkah pertama bandingkan nomor 4 dengan 7, karena 4 < 7 maka tidak perlu diubah.



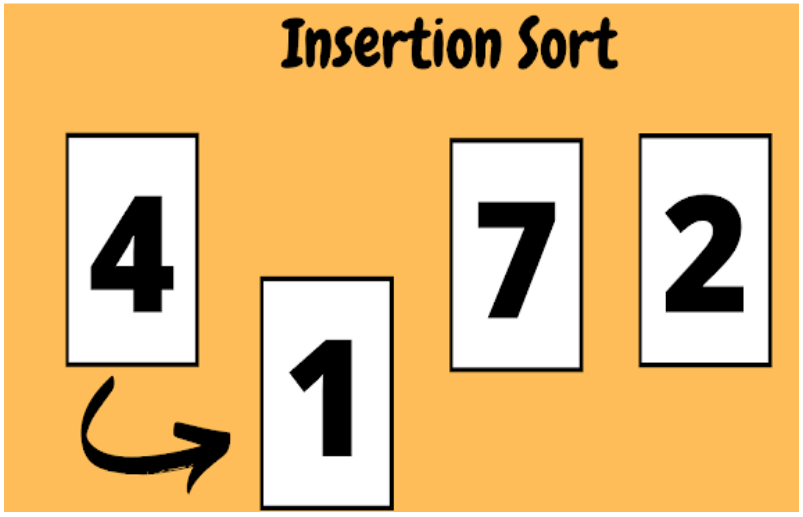
Gambar 2. 9.Langkah Ke 1 Insertion Short

Langkah kedua bandingkan nomor 7 dengan 1, karena 7 > 1 maka ubah posisi.



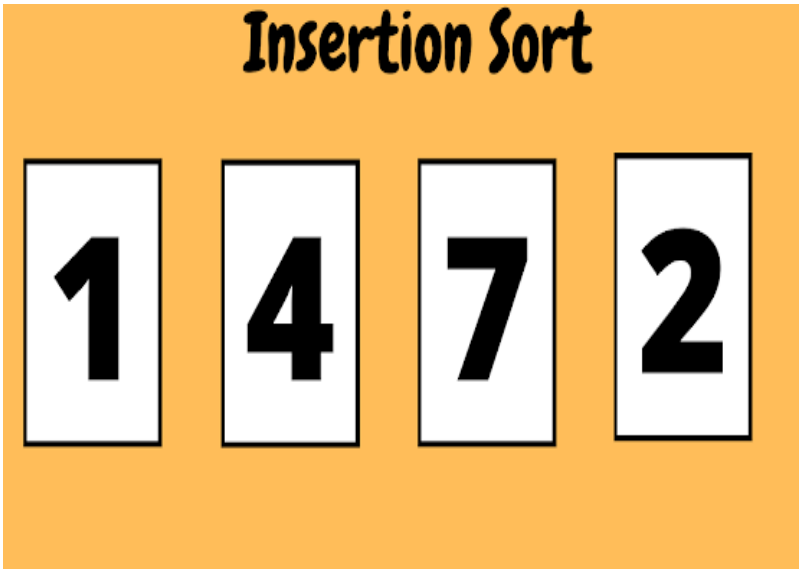
Gambar 2. 10.Langkah Ke 2 Insertion Short

Langkah ketiga bandingkan nomor 4 dengan 1, karena 4 > 1 maka ubah posisi.



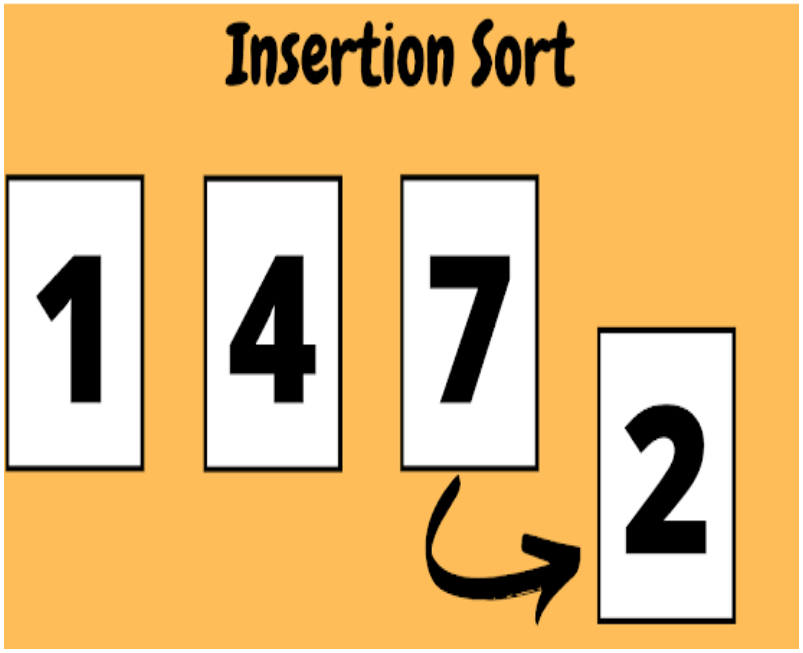
Gambar 2. 11.Langkah Ke 3 Insertion Short

Langkah keempat cari angka berikutnya yang belum terurut.



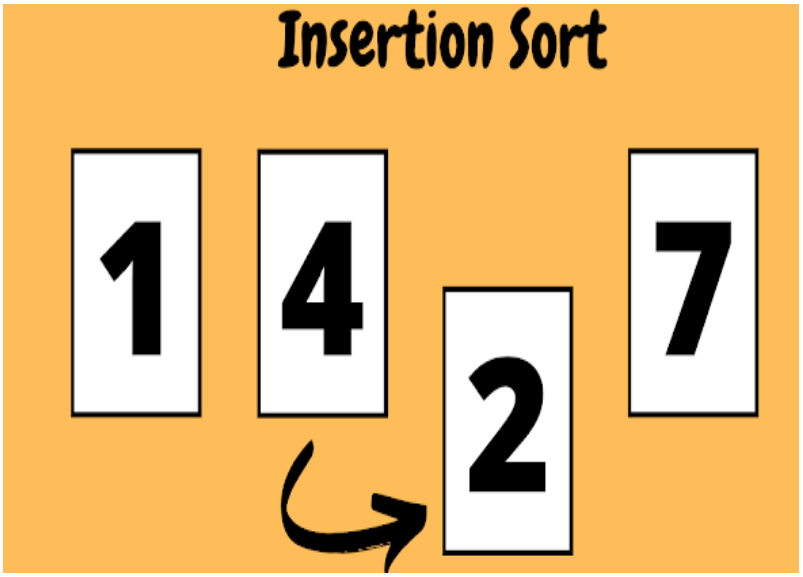
Gambar 2. 12.Langkah Ke 4 Insertion Short

Langkah kelima bandingkan nomor 7 dengan 2, karena 7 > 2 maka ubah posisi.



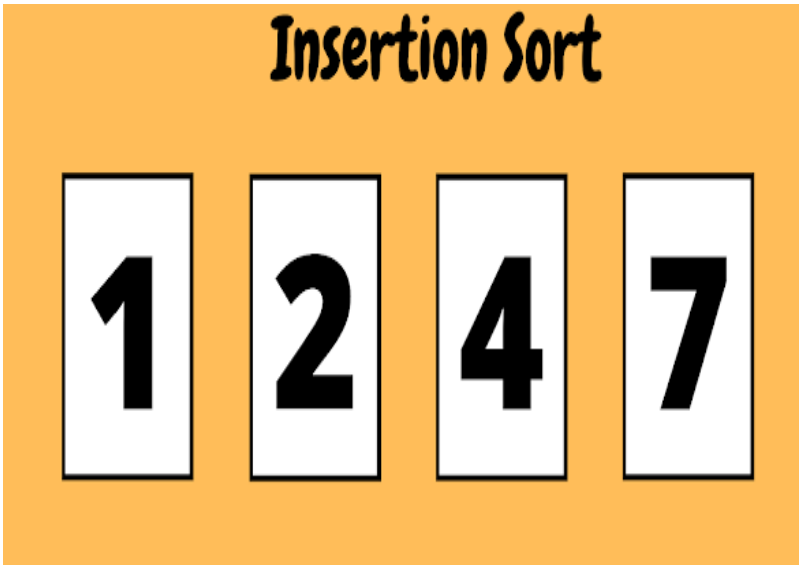
Gambar 2. 13.Langkah Ke 5 Insertion Short

Langkah keenam bandingkan nomor 4 dengan 2, karena 4 > 2 maka ubah posisi.



Gambar 2. 14.Langkah Ke 6 Insertion Short

Jika data sudah berurut maka tidak ada yang perlu dibandingkan dan sorting berhasil.



Gambar 2. 15.Langkah Ke 7 Insertion Short

E. Advanced Sort

Pengertian advanced sort" bisa bervariasi tergantung pada konteksnya, karena istilah ini mungkin memiliki makna yang berbeda di berbagai domain atau bidang. Secara umum, "advanced sort" dapat merujuk pada pendekatan pengurutan yang lebih kompleks atau canggih dibandingkan dengan metode pengurutan dasar. Berikut adalah beberapa interpretasi yang mungkin sesuai:

- Optimasi Kinerja: Dalam beberapa konteks, "advanced sort" bisa merujuk pada teknik pengurutan yang dioptimalkan untuk kinerja tertentu, seperti pengurutan data yang besar atau pengurutan dalam lingkungan paralel.
- Penanganan Kasus Khusus: Pengurutan dapat menjadi lebih canggih dengan menangani kasus khusus tertentu. Misalnya, kemampuan untuk mengatasi data yang sudah hampir terurut atau data yang memiliki pola tertentu.
- Custom Sorting: "Advanced sort" juga dapat mengacu pada kemampuan untuk mengurutkan data berdasarkan kriteria khusus atau menggunakan fungsi perbandingan yang disesuaikan sesuai dengan kebutuhan aplikasi.
- Stability: Menjamin stabilitas dalam pengurutan, yang berarti elemen-elemen yang memiliki nilai yang sama akan tetap dalam urutan relatif yang sama setelah pengurutan. Algoritma pengurutan yang stabil dianggap lebih canggih dalam beberapa situasi.
- Adaptabilitas: Kemampuan algoritma pengurutan untuk beradaptasi dengan jenis data atau situasi tertentu. Ini bisa mencakup penggunaan strategi yang berbeda tergantung pada sifat data yang diurutkan.

Penting untuk dicatat bahwa "advanced sort" bukanlah istilah yang spesifik dan standar dalam dunia pemrograman atau ilmu komputer. Jika Anda bertanya dalam konteks tertentu, lebih banyak informasi atau detail tentang aplikasi atau lingkungan di mana "advanced sort" digunakan akan membantu memberikan definisi yang lebih tepat.

F. Quick Sort

Quick Sort merupakan suatu algoritma pengurutan data yang menggunakan teknik pemecahan data menjadi partisi-partisi, sehingga metode ini disebut juga dengan nama partition exchange sort. Untuk memulai irterasi pengurutan, pertama-tama sebuah elemen dipilih dari data, kemudian elemen-elemen data akan diurutkan diatur sedemikian rupa.

Algoritma ini mengambil salah satu elemen secara acak (biasanya dari tengah) yang disebut dengan pivot lalu menyimpan semua elemen yang lebih kecil di sebelah kiri pivot dan semua elemen yang lebih besar di sebelah kanan pivot. Hal ini dilakukan secara rekursif terhadap elemen di sebelah kiri dan kanannya sampai semua elemen sudah terurut.

1. Algoritma Quick Sort.

- Pilih satu elemen secara acak sebagai pivot
- Pindahkan semua elemen yang lebih kecil ke sebelah kiri pivot dan semua elemen yang lebih besar ke sebelah kanan pivot. Elemen yang nilainya sama bisa disimpan di salah satunya.
- Lakukan sort secara rekursif terhadap sub-array sebelah kiri dan kanan pivot

2. Tips Pemilihan Pivot.

Dalam algoritma quick sort, pemilihan pivot adalah hal yang menentukan apakah algoritma quick sort tersebut akan memberikan performa terbaik atau terburuk. Berikut beberapa cara pemilihan pivot :

- Pivot adalah elemen pertama, elemen terakhir, atau elemen tengah tabel. Cara ini hanya bagus jika elemen tabel tersusun secara acak, tetapi tidak bagus jika elemen tabel semula sudah terurut. Misalnya, jika elemen tabel semula menurun, maka semua elemen tabel akan terkumpul di upatabel kanan.
- Pivot dipilih secara acak dari salah satu elemen tabel. Cara ini baik,

tetapi mahal, sebab memerlukan biaya (cost) untuk pembangkitan prosedur acak. Lagi pula, itu tidak mengurangi kompleksitas waktu algoritma.

- Pivot adalah elemen median tabel. Cara ini paling bagus, karena hasil partisi menghasilkan dua bagian tabel yang berukuran seimbang (masing masing ≈ n/2 elemen). Cara ini memberikan kompleksitas waktu yang minimum. Masalahnya, mencari median dari elemen tabel yang belum terurut adalah persoalan tersendiri. Algoritma Quick Sort terdiri dari dua prosedur, yaitu prosedur PARTISI dan prosedur QUICKSORT.

3. Keunggulan Quick sort

- Secara umum memiliki kompleksitas O(n log n).
- Algoritmanya sederhana dan mudah diterapkan pada berbagai bahasa pemrograman dan arsitektur mesin secara efisien.
- Dalam prakteknya adalah yang tercepat dari berbagai algoritma pengurutan dengan perbandingan, seperti mergesort dan heapsort.
- Melakukan proses langsung pada input (in-place) dengan sedikit tambahan memori

4. Kekurangan Quick sort

- Sedikit kesalahan dalam penulisan program membuatnya bekerja tidak beraturan (hasilnya tidak benar atau tidak pernah selesai).
- Memiliki ketergantungan terhadap data yang dimasukkan, yang dalam kasus terburuk memiliki kompleksitas O(n2).
- Secara umum bersifat tidak stable, yaitu mengubah urutan input dalam hasil akhirnya (dalam hal inputnya bernilai sama).
- Pada penerapan secara rekursif (memanggil dirinya sendiri) bila terjadi kasus terburuk dapat menghabiskan stack dan memacetkan program.
- Pada bahasa pemrograman, quicksort ada dalam pustaka stdlib.h untuk bahasa C, dan class TList dan TStringList dalam Delphi

(Object Pascal) maupun FreePascal.

G. Merge Short

Dikutip dari laman Programiz, merge sort merupakan algoritma program komputer sorting atau penggabungan yang didasarkan pada prinsip Devide and Conquer Algorithm.Jenis algoritma ini dibagi menjadi beberapa sub masalah. Jadi, setiap sub masalah tersebut akan diselesaikan secara individual. Nantinya, sub masalah digabungkan untuk mencari solusi akhir.

Sederhananya, proses Merge Sort dalam pemrograman adalah membagi masalah menjadi beberapa bagian kemudian mengurutkan setiap bagian. Selanjutnya, menggabungkan bagian yang telah diurutkan kembali menjadi satu.Menurut laman Geeks For Geeks, keuntungan utama dari jenis algoritma Merge Sort ialah memiliki kompleksitas waktu dalam menyelesaikan masalah secara relatif cepat.

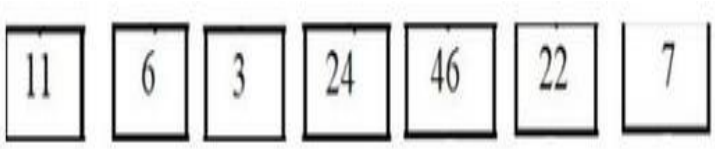
Program algoritma ini termasuk populer di kalangan programmer karena mengurutkan data set besar dan relatif efisien serta mudah diimplementasikan.Merge sort sering dipakai bersamaan dengan algoritma lain seperti quicksort dengan tujuan meningkatkan kinerja keseluruhan dari rutinitas pengurutan yang dijalankan. Lantas seperti apa contoh merge sort? Simak rinciannya pada pejabaran berikut ini.:

1. Contoh Merge Sort dalam Sistem Pemrograman

Untuk memahami cara kerja Merge Sort dalam sistem pemrograman, berikut contohnya yang dikutip dari laman educba.com.Pada contoh ini array atau larik kode yang diberikan adalah 11, 6, 3, 24, 46, 22, dan 7. Cara kerja Merge Sort larik kode tersebut dibagi menjadi beberapa sub-array. Nantinya, setiap sub diselesaikan secara terpisah. Berikut caranya:

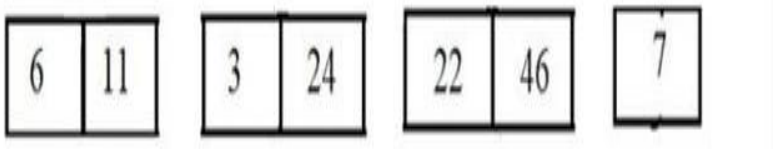
- Bagi array dari 7 elemen kode di atas seeprti di bawah ini.
- Array 1 adalah 11, 6, 3, dan 24. Array 2 adalah 46, 22, dan 7.
- Bagi kedua larik tersebut menjadi dua, yakni 11 dan 6, 3 dan 4, 46 dan 22, kemudian 7.

- Bagi lagi array di atas menjadi nilai tunggal hingga tidak dapat lagi dibagi seperti gambar di bawah ini.



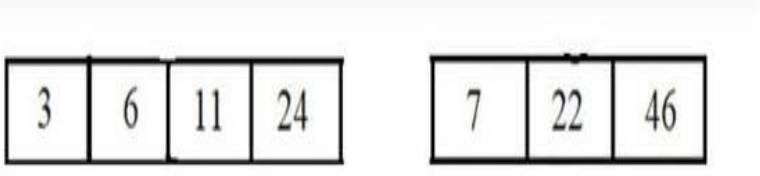
Gambar 2. 16

- Bandingkan elemen di setiap daftar dan gabungkan dengan cara mengurutkan ke dalam daftar lain. Misalnya, bandingkan 11 dan 6, dan urutannya dibalik.
- Bandingkan 3 dan 24, mereka berada dalam urutan yang benar. Selanjutnya, bandingkan lagi 46 dan 22 dan masukkan 22 lalu 46. Selanjutnya, pertahankan 7 apa adanya.



Gambar 2. 17

- Setelah itu, bagikan daftar dua nilai data yang berurutan dalam daftar yang terurut seperti gambar di bawah ini.

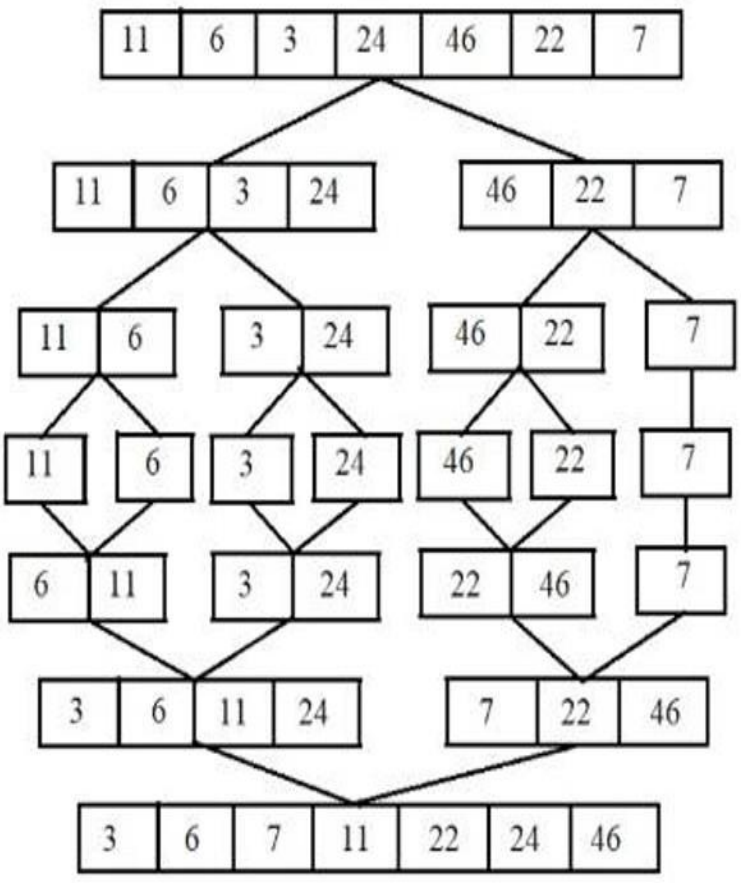


Gambar 2. 18

- Kemudian daftar akan terlihat seperti ini setelah penggabungan terakhir.

Gambar 2. 19

Jika digabung dari beberapa sub array, hasil akhirnya akan seperti pada gambar berikut.



Gambar 2. 20.Contoh Ascending

Perlu diperhatikan, setiap bahasa pemrograman memiliki cara penulisan yang berbeda. Namun, sistem yang dipakai untuk menyelesaikan Merge Sort menggunakan sistem yang sama.Demikian contoh merge sort dalam sistem pemrograman. Semoga informasi di atas bermanfaat.

H. Radix Sort

Radix Sort adalah algortima atau metode pengurutan (sorting) tanpa pembandingan dengan kata lain, sorting Non-Comparasion sort dimana dalam prosesnya tidak melakukan perbandingan antar data. Kata radix bermakna harafiah posisi dalam angka.Di mana sederhananya, dalam representasi desimal, radix adalah digitnya. Dalam implementasinya, Radix Sort merupakan algoritma pengurutan yang cepat, mudah, dan sangat efektif. Namun banyak yangmengira bahwa algoritma radix memiliki banyak batasan di mana untuk kasus-kasus tertentu tidak dapat dilakukan dengan algoritma ini, seperti pengurutan bilangan pecahan dan bilangan negative.

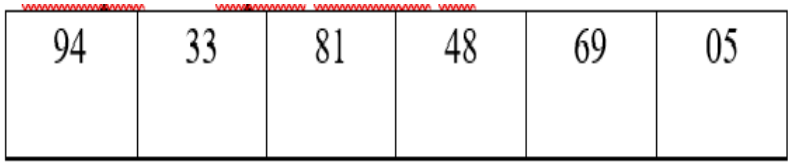
Berdasarkan urutan pemrosesan radixnya, Radix Sort terbagi 2 macam, yaitu:

- LSD (Least Significant Digit), di mana pemrosesan dimulai dari radix yang paling tidak signifikan. Sorting dilakukan dengan cara mengurutkan nilai-nilai input berdasarkan digit terjahir ke digit pertama.
- MSD (Most Significant Digit), di mana pemrosesan dimulai dari radix yang paling signifikan. Sorting dilakukan dengan cara mengurutkan nilai-nilai input berdasarkan digit pertama, lalu dilanjutkan lagi berdasarkan radix keuda dan seterusnya.

Cara kerja dalam artikel jurnal yang ditulis sabarudi et al, Proses dasar Radix Sort adalah mengkategorikan data-data menjadi sub kumpulan data sesuai dengan nilai radix-nya (kategori tertentu), dimana dalam tiap kategorinya dilakukan pengklasifikasian lagi dan seterusnya sesuai dengan kebutuhan mengkonkatenasinya, dan subkategori-subkategori tersebut digabungkan kembali, yang secara dilakukan hanya dengan metode sederhana concatenation.

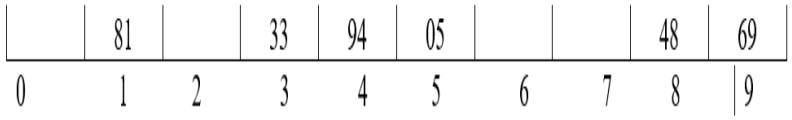
1. **Contoh Algoritma Radix Sort**

- Terdapat data sebagai berikut:



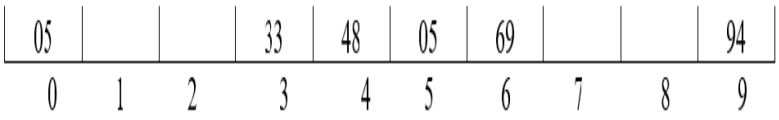
Gambar 2. 21

- Lalu buat bucket untuk menyimpan data sementara dari data diatas. Dengan mengurutkan dari radix (digit) atau digit satuan dan dimasukkan ke dalam bucket yang sesuai angka radix.



Gambar 2. 22

- Setalah itu tulis hasil sorting 81 33 94 05 48 69 Dan lanjutkan sorting dengan digit depanya (digit puluhan)



Gambar 2. 23

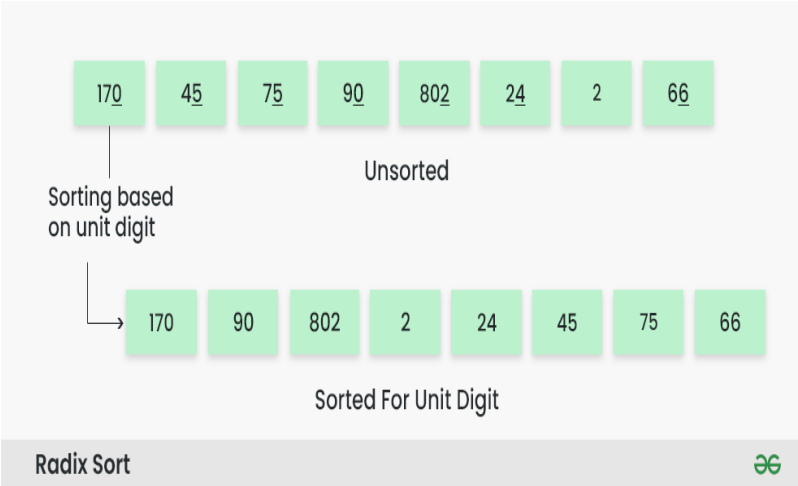
- Lalu dijadikan satu lagi dalam array05 33 48 05 69 94.Dan itulah hasil setelah diurutkan menggunakan radix sort. Jika masih ada digit depanya (digit ratusan) maka tetap lanjut, cara seperti sebelumnya.

2. Cara Kerja Algoritma Pengurutan Radix

Untuk melakukan pengurutan radix pada array [170, 45, 75, 90, 802, 24, 2, 66], kita ikuti langkah-langkah berikut:

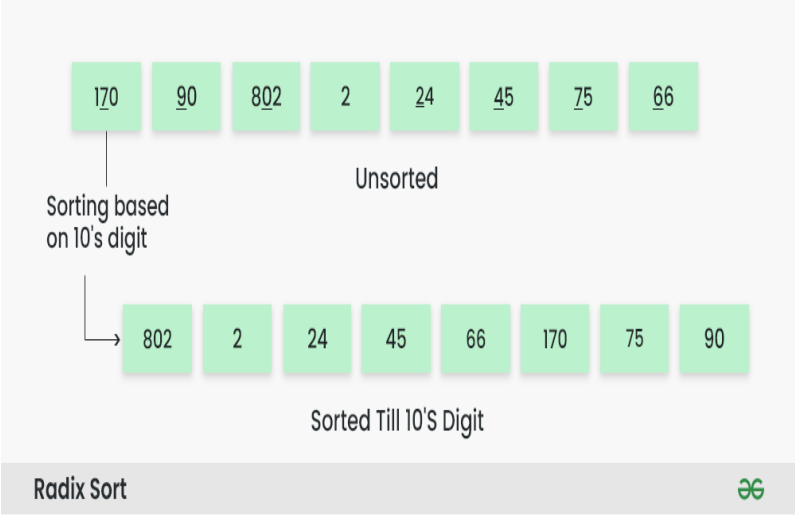
Langkah 1: Temukan elemen terbesar dalam array, yaitu 802. Ia memiliki tiga digit, jadi kita akan melakukan iterasi tiga kali, satu kali untuk setiap tempat yang signifikan.

Langkah 2: Urutkan elemen berdasarkan digit tempat satuan (X=0). Kami menggunakan teknik pengurutan yang stabil, seperti pengurutan penghitungan, untuk mengurutkan digit di setiap tempat penting



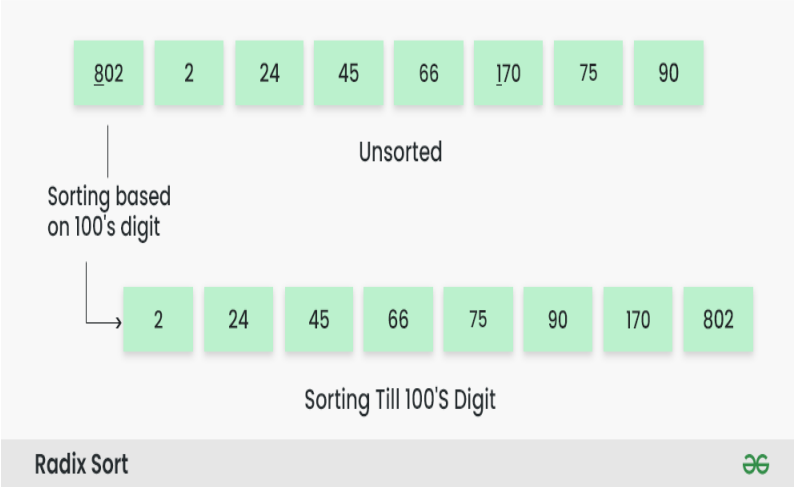
Gambar 2. 24.Langkah Ke 2

Langkah 3: Urutkan elemen berdasarkan angka tempat puluhan. Pengurutanberdasarkan tempat puluhan Melakukan penghitungan pada array berdasarkan angka tempat puluhan.Array yang diurutkan berdasarkan tempat puluhan adalah [802, 2, 24, 45, 66, 170, 75, 90].



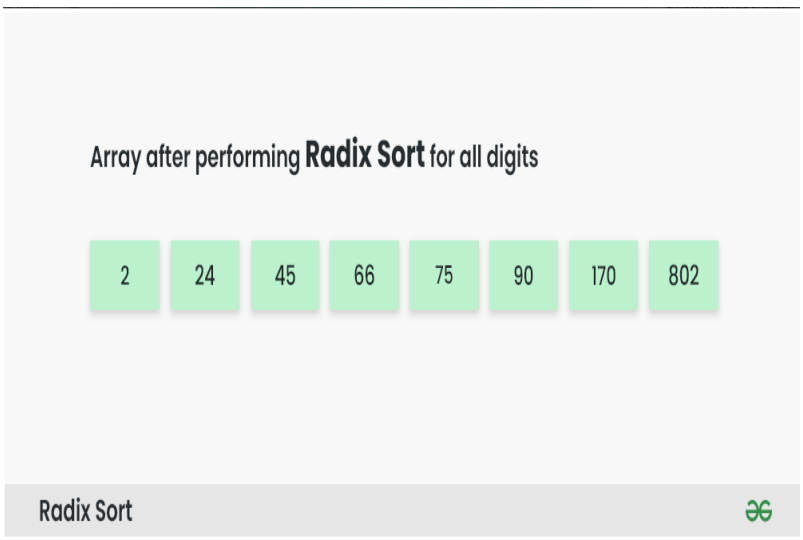
Gambar 2. 25.Langkah Ke 3

Langkah 4, Urutkan elemen berdasarkan ratusan digit tempat. Pengurutan berdasarkan tempat ratusan: Melakukan pengurutan penghitungan pada larik berdasarkan angka ratusan tempat. Array yang diurutkan berdasarkan tempat ratusan adalah [2, 24, 45, 66, 75, 90, 170, 802].



Gambar 2. 26.LangkahKe 4

Langkah 5: Array sekarang diurutkan dalam urutan menaik.Array terakhir yang diurutkan menggunakan radix sort adalah [2, 24, 45, 66, 75, 90, 170, 802].



Gambar 2. 27.Langkah Ke 5

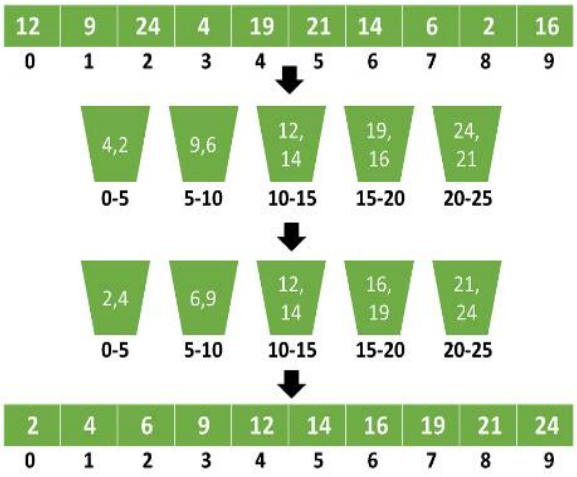
I. Bucket Sort

Bucket sort adalah teknik pengurutan yang melibatkan pembagian elemen ke dalam berbagai kelompok, atau keranjang. Ember ini dibentuk dengan mendistribusikan elemen secara merata. Setelah elemen dibagi ke dalam keranjang, elemen tersebut dapat diurutkan menggunakan algoritma pengurutan lainnya. Akhirnya, elemen-elemen yang telah diurutkan dikumpulkan bersama secara teratur.Algoritma pengurutan bucket adalah teknik pengurutan tingkat lanjut, berbeda dengan yang lain. Pengurutan keranjang dapat dianggap sebagai kerangka kolektif yang dibangun menggunakan berbagai algoritma pengurutan.

Misalnya, Anda dapat menggunakan salah satu jenis algoritme untuk mengurutkan elemen ke dalam keranjang dan jenis algoritme lainnya untuk mengurutkan elemen dalam keranjang.

Bucket Short , juga dikenal sebagai pengurutan bin, adalah algoritma pengurutan yang membagi elemen array menjadi beberapa keranjang. Buc ket kemudian diurutkan satu per satu, baik menggunakan algoritma pengurutan yang berbeda atau dengan menerapkan algoritma pengurutan bucket secara rekursif.

- Metode pengurutan keranjang adalah sebagai berikut:
- Buat array "ember" yang awalnya kosong
- Scatter: Menelusuri array asli, menempatkan setiap objek di keranjang yang sesuai
- Setiap ember yang tidak kosong harus disortir
- Gather: Mengembalikan semua elemen ke array asli setelah mengunjungi bucket secara berurutan



Gambar 2. 28.Contoh Bucket Short

J. Counting Short

Counting sort merupakan sebuah teknik pengurutan dengan cara menghitung jumlah kemunculan dari setiap data yang berada di dalam array. Pada algorithma ini kita harus membuat sebuah array penampung untuk menyimpan jumlah kemunculan data dimana ukuran dari array tersebut harus sejumlah range angka yang bisa di input oleh user. Algorithma ini dapat mengurutkan data dari besar ke kecil (Ascending) dan kecil ke besar (Descending). Algoritma ini tidak cocok untuk set data dengan jumlah besar karena kompleksitas dari algorithma ini adalah O(n+r) di mana n adalah jumlah item dan r adalah jarak dari inputan.

Berikut gambaran dari implementasi Counting Sort:Diketahui int arr[] = {5, 2, 3, 4, 1, 5, 6}. Misalkan kita hanya memiliki angka dengan range 0 – 10. Pertama-tama kita harus mempersiapkan sebuah array untuk untuk menyimpan jumlah data dalam array yang disebut int count[11] = {0}, maka akan membentuk array sebagai berikut:

total	0	0	0	0	0	0	0	0	0	0	0
index	0	1	2	3	4	5	6	7	8	9	10

Gambar 2. 29

1st Cycle:

(5, 2, 3, 4, 1, 5, 6) ->count[5]++;

total	0	0	0	0	0	1	0	0	0	0	0
index	0	1	2	3	4	5	6	7	8	9	10

Gambar 2. 30

2nd Cycle:

(5, 2, 3, 4, 1, 5, 6) ->count[2]++;

total	0	0	1	0	0	1	0	0	0	0	0
index	0	1	2	3	4	5	6	7	8	9	10

Gambar 2. 31

3rd Cycle:

(5, 2, 3, 4, 1, 5, 6) ->count[3]++;

total	0	0	1	1	0	1	0	0	0	0	0
index	0	1	2	3	4	5	6	7	8	9	10

Gambar 2. 32

4th Cycle:

(5, 2, 3, 4, 1, 5, 6) ->count[4]++;

total	0	0	1	1	1	1	0	0	0	0	0
index	0	1	2	3	4	5	6	7	8	9	10

Gambar 2. 33

5th Cycle:

(5, 2, 3, 4, 1, 5, 6) ->count[1]++;

total	0	1	1	1	1	1	0	0	0	0	0
index	0	1	2	3	4	5	6	7	8	9	10

Gambar 2. 34

6th Cycle:

(5, 2, 3, 4, 1, 5, 6) ->count[5]++;

total	0	1	1	1	1	2	0	0	0	0	0
index	0	1	2	3	4	5	6	7	8	9	10

Gambar 2. 35

7th Cycle:

(5, 2, 3, 4, 1, 5, 6) ->count[6]++;

total	0	1	1	1	1	2	1	0	0	0	0
index	0	1	2	3	4	5	6	7	8	9	10

Gambar 2. 36

Pada looping yang terakhir, kita sudah mendapatkan jumlah kemunculan dari setiap angka. Kita tinggal membuat sebuah looping untuk mengcheck apakah array untuk setiap nilai tersebut nilainya 0, jika tidak maka cetak angka data array tersebut.Output yang akan dihasilkan dari counting sort sebagi berikut: 1, 2, 3, 4, 5, 5, 6.

BAB III

PENUTUP

A. Kesimpulan

Dari penjelasan diatas bahwa sorting dibagi menjadi beberapa bagian, diantaranya : bubble sort, selection sort dan insertion sort. Yang memiliki kelebihan dan kekurangan tersendiri. Dari beberapa sorting tersebut memiliki tujuan yang sama, yaitu mengurutkan data untuk mempermudah pencarian dan data akan menjadi sistematis.

DAFTAR PUSTAKA

https://kukuhapriyanto.blogspot.com/2017/06/sorting-dalam-bahasa-c.html

https://socs.binus.ac.id/2019/12/27/counting-sort/

https://id.scribd.com/presentation/526332588/5-Advanced-Sorting

https://onophp.blogspot.com/2018/11/quick-sort-pengertian-agoritma-dan.html