PERTEMUAN 9
SORTING (LANJUT 2)

A. TUJUAN PEMBELAJARAN

Setelah menyelesaikan pertemuan ini, mahasiswa mampu mempraktekkan:

1. Shell Sort
2. Merge Sort
3. Insertion Sort

B. URAIAN MATERI

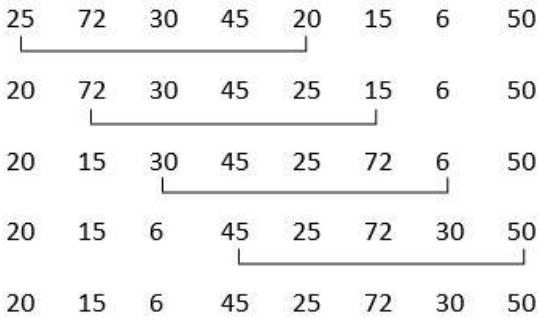
1. Shell Sort

Cara ini dinamai menurut pembuatnya dengan metode Bubble Sort, hanya saja perbandingannya bukan antara dua bilangan yang berurutan, melainkan antara dua bilangan dengan jarak tertentu. Perbandingan total dilakukan sampai nilai jarak sama dengan 1 Jika jaraknya 1, metode Sortir Shell sama dengan metode Bubble Sort.

Contoh: Kami memiliki array 8 elemen yang diurutkan dalam urutan menaik menggunakan metode Sortir Shell: 25, 72, 30, 45, 20, 15, 6, 59. Urutan langkah-langkah penyortiran adalah sebagai berikut.

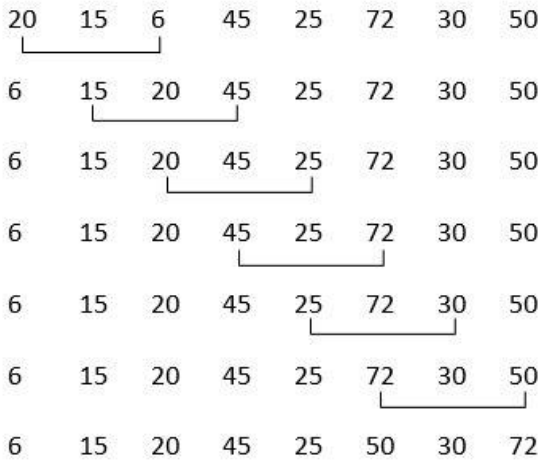
Untuk prtama kali nilai jarak (K) = 8 adalah div 2 = 4 dan kita buat pencacah C = 0. Kemudian dilakukan proses perbandingan secara terurut antara Value [0] dan Value [4], jika Value [0]> Value [4] tukar tempat dan jika tidak dilanjutkan dengan Nilai [1] dngan Nilai [5], jika Nilai [1]> Nilai [5] tukar tempat dan jika demikian jangan lanjutkan ke Nilai [2] dengan Nilai [6], jika Nilai [2]> Nilai [6] lalu tukar tempat dan jika tidak dilanjutkan ke Nilai [3] dngan Nilai [7], jika Nilai [3]> Nilai [7] maka tukar tmpat, Nilai C = 1 .

Tabel 9.4Langkah Pengurutan Shell Sort-1



Karna nilai K masih sama dngan 4 dan nilai C = 1, maka jalankan lagi proses perbadingan dngan mengghitung nilai jarak (K) sama dngan 4 div 2 = 2 dan nilai C = 0 Skarang proses prbandingan dilakukan scara berurutan antara nilai] dengan Nilai [2], jika Nilai [0]> Nilai [2] maka bertukar tempat dan jika tidak selanjutnya ke Nilai [1] dengan Nilai [3], jika Nilai [1]> Nilai [3] lalu tukar tempat dan jika tidak, lanjutkan ke Nilai [2] dengan Nilai [4], jika Nilai [2]> Nilai [4] maka tukar tempat dan jika tidak pergi lanjutkan ke Value [3] dengan Value [5], jika Value [3]> Value [5] maka lakukan prtukaran dan jika tidak lanjutkan ke Value [4] dengan Nilai [6], jika Value [4]> Value [6], kmudian mlakukan prtukaran dan jika tidak, lanjutkan ke Value [5] dengan Value [7], Jika Value [5]> Value [7], maka lakukan prtukaran dan nilai C = 1.

Tabel 9.5Langkah Pengurutan Shell Sort-2



Karna K masih sama dngan 2 dan nilai C = 1, maka jlankan lagi proses prbandingan dngan mnghitung nilai jarak (K) sama dngan 2 div 2 = 1 dan nilai C = 0 skarang proses prbandingan dilakkukan scara brurutan antara nilai [0] Nilai [1], Nilai [1] Nilai [2], Nilai [2] Nilai [3], Nilai [3] Nilai [4], Nilai [4] Nilai [5], Nilai [5] dengan Nilai [6] dan Nilai [6]> Nilai [7], maka nilai C = 1 diproleh hasil prtukaran setlah dilakukan prbandingan sbagai brikut.

6 15 20 25 45 30 50 72

Andaikan K tlah berNlai sama dngan satu, tapi Nilai C masih berNillai sama dngan 1 maka prbandingan harus diulang kmbali dngan mmberi Nilai C=0 dan Nilai K sama sperti sbelumnya yaitu sama dengan 1. Stelah dilakukan prbandingan maka hasillnya dapat diproleh sperti brikut:

6 15 20 25 30 45 50 72

Krena Nillai C tidak lagi brubah mnjadi 1 maka prbandingan tlah brakhir dan hasil trakhir mrupakan hasil pngurutan yang diharrapkan.

Contoh: Kami memiliki array 8 elemen yang diurutkan dalam urutan menurun menggunakan metode Sortir Shell: 25, 72, 30, 45, 20, 15, 6, 50. Urutan langkah-langkah penyortiran adalah sebagai brikut.

Dngan cara yang sama unttuk pngurutan scara mnaik, dngan mngganti tanda > mnjadi < maka akan diproleh elmen aray yang trurut scara menurun.

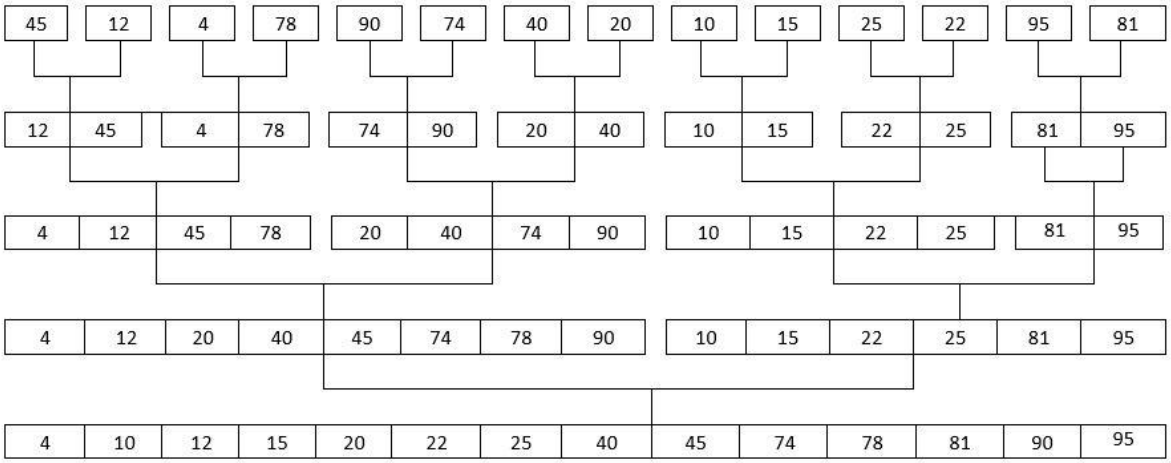
2. Merge Sort

Metode ini memnfatkan ketraturan yang diproleh dngan mengabungkan dua larik brsama. Sbuah aray niilai dengan N elmen (niliai [0..N-1]) diangap trdiri dari N aray masiing-msing trdiri dari satu elmen. Untuk pasangan aray yang berdkatan, kita gabungkan shingga mendapattkan N / 2 aray, masiing-msing memliki 2 elmen (jika N ganjil, ada aray dengan 1 elmen). Selama proses penggabungan, posisi disesuaikan sehingga elemen yang lebih kecil ditempatkan di posiisi awal (untuk mengurutkan naik) dan elemen yang lebih besar di posisi awal (untuk mengurutkan menurun). Kmudian lakukan concatenation untuk stiap pasang array seperti diatas, shingga didapatkan N / 2 aray yang masiing-msing mmiliki 4 elmen. Kami akan melanjutkan langkah ini

sampai kami memiliki array yang telah diurutkan.

Jika kita memliki aray Nilai[0..13] = {45, 12, 4, 78, 90, 74, 40, 20, 10, 15, 25, 22, 95, 81} maka dngan mengunakan metode Merge Sort scara mnaik dapat dilkakukan eperti pada gambar.

Gambar 9.9 Array Nilai Secara Menaik



3. Insertion Sort

Insertion Sort adalah mengurutkan dengan memasukkan elmen aray pada posisi yang benar. Pncarian yang tepat dillakukan dngan mlakukan pncarian beruntun dallam larik. Algoritma pengurutan ini cocok untuk masalah memasukkan elemen baru ke dalam array yang sudah dipesan. Misalnya, dalam setumpuk kartu, kartu yang ditarik biasanya dimasukkan oleh pemain pada posisi yang benar sehingga penambahan kartu membuat semua kartu tetap teratur.

Misalkan kita memiliki array dengan N, urutan menaik menggunakan metode Insertion Sort adalah sebagai berikut:

Step -1: elemen pertama Nilai[0] diasumsikan telah sesuai tempatnya.

Step -2: ambil elemen kedua (Nilai [1]), temukan lokasi yang tepat dari Nilai [0..0] untuk Nilai [1]. Geser ke kanan jika Nilai [0..1] lebih besar (untuk urutan menaik) atau lebih kecil (untuk urutan menurun) daripada Nilai [1].

Step -3: ambil elmen ketiga (Nlai [2]), temukan lokas yang tepat dari Nilai [0..1] hingga Nilai [2]. Geser ke kanan jika Nilai [0..2] lbih bsar (untuk urutan menaik) atau lebih kecil (untuk urutan mnurun) daripada Nilai [2].

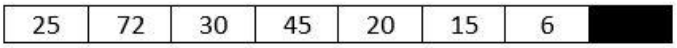
Step -4: ambil elmen keempat (Nlai [3]), tmukan lokas yang tpat dari Nilai [0..3] hingga Nilai Nilai [3]. Geser ke knan jika Nilai [0..2] lebi besar (untuk urutan menaik) atau lebih kecil (untuk urutan mnurun) daripada Nilai [3].

Step –N: bawa elmen ke N (Value [N]), cari lokas yang bnar pada Value [0..N-1] untuk Value Value [N]. Geser ke kanan jika Nilai [0..N-1] lebih besar (untuk urutan menaik) atau lbih kcil (untuk urutan mnurun) daripada Nilai [N].

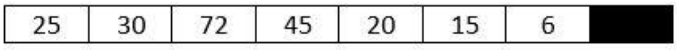
Contoh: Kami memiliki array 8 elemen yang diurutkan dalam urutan menaik menggunakan metode sisipkan sortir: 25, 72, 30, 45, 20, 15, 6, 50. Urutan langkah-langkah pengurutan adalah sebagai brikut:

Step -1: Nilai[0] disumsikan tlah trurut.

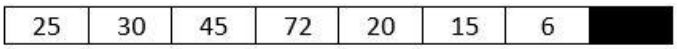
Step -2: Temukan lokassi yang benar untuk Niilai [1] pada Nilai [0..0]. Dalm hal ini, trnyata 72 tidak lebih besar dari 25, jadi tidak ada offset, jadi kita mendapatkan aray seperti ini:



Step -3: Temukan lokaasi yang benar untuk Niilai [2] pada Nilai [0..1]. Dlam hal ini trnyata lokasi yang benar adallah 1, sehingga Value [1] digaser ke knan shingga Value [2] berada di poisi pertama, shingga didapat larik sbagai brikut:

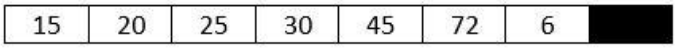


Step -4: Temukan lokkasi yng benar untuk Niilai [3] pada Nilai [0..2]. Dlam hal ini trnyata lokkasi yang benar adallah 2, sehingga Value [2] digesar ke knan sehingga Value [3] berada di posisi 3, shingga menghasilkan aray sebagai brikut:



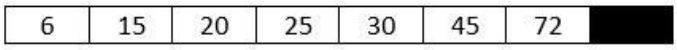
Step -5: Temukan lokasi yang benar untuk nilai [4] pada nilai [0..3]. Dalam hal ini ternyata letak yang benar adalah 0, kmudian nilai [0], nilai [1], nilai [2], nilai [3] digeser satu posisi ke kanan setiap kali shingga nilai [4] berada di posisi 0, jadi aray seperti brikut:

Step -6: Temukan lokasi yang benar untuk nilai [5] pada nilai [0..4]. Dalam hal ini ternyata letak yang benar adalah 0, maka Nilai [0], Nilai [1], Nilai [2], Nilai [3] dan Nilai [4] masing-masing bergeser satu posisi ke kanan, sehingga Nilai [6]

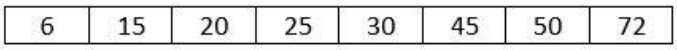


adalah posisi di 0, kita mendapatkan array brikut:

Step -7: Temukan lokasi yang benar untuk nilai [6] pada nilai [0..5]. Dalam hal ini ternyata letak yang benar adalah 0, maka Nilai [0], Nilai [1], Nilai [2], Nilai [3], Nilai [4] dan Nilai [5] masing-masing bergeser satu posisi ke kanan, sehingga Nilai [6] pada posisi 0, kita mendapatkan larik brikut:



Step -8: Temukan lokasi yang benar untuk nilai [7] dalam nilai [0..6]. Dalam hal ini ternyata letak yang benar adalah 6, sehingga Nilai [6] digeser satu posisinya ke kanan sehingga Value [7] berada pada posisi 6, diperoleh larik sebagai brikut:



Selesai. Dan aray tlah diurutkan dalam urutan menaik. Sebuah program untuk mengurutkan elmen aray dalam urutan menaik menggunakan metode Sortir Penyisipan.

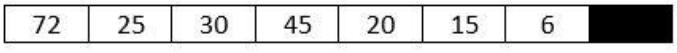
```
#include<iostream.h>
#include<conio.h>
#include<iomanip.h>
```

```
void main()
{
 int Nilai[20];
 int i, j, k, N;
 int temp;
 cout<<"Masukan Bilangan : ";
 cin>>N;
 for(i=0; i<N; i++)
 {
  cout<<"Elmen ke-"<<i<<" : ";
  cin>>Nilai[i];
 }
```

Contoh: Misallkan kiita memiliki range nilai, sbanyak 8 elmen diurutkan secara descending menggunakan metode insert sort: 25, 72, 30, 45, 20, 15, 6, 50. Urutan langkah sortir adalah sebagai brikut:

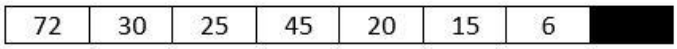
Step -1: Niilai[0] diasmsikan telah terurut

Step -2: Temukan lokasii yng benar utuk Nilai [2] pada Nilai [0..0]. Dalam hal ini trnyata lokasi yang benar adalah 0, sehingga Nilai [0] digeser satu posisi ke kanan dan Nilai [1] digeser ke 0, menghasilkan larik berikut:

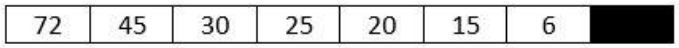


Step -3: Temukan lokasi yang benar untuk Nilai [2] pada Nilai [0..1]. Dalam hal ini trnyata lokasi yang benar adalah 1, sehingga Nilai [1] digeser satu posisinya

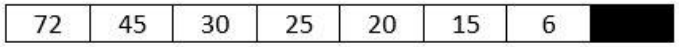
ke kanan shingga Nilai [2] berada di posisi pertama, shingga menghasilkan larik sebagai brikut:



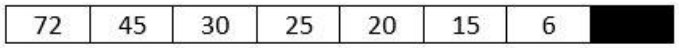
Step -4: Temukan lokasi yang benar untuk Nilai [3] pada Nilai [0..2]. Dalam hal ini trnyata letak yang benar adalah 1, kemudian Nilai [1] dan Nilai [2] digeser satu posisinya ke kanan sehingga Nilai [3] ada di posisi pertama, shingga menghasilkan larik sebagai brikut:



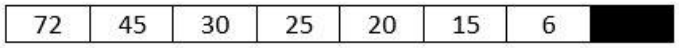
Step -5: Temukan lokasi yang benar untuk nilai [4] pada nilai [0..3]. Dalam kasus ini, trnyata lokasi persisnya adalah 4, tetapi krena tidak ada offset dengan sendirinya, lariknya terlihat seperti ini:



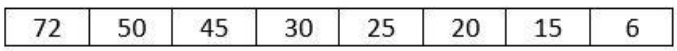
Step -6: Temukan lokasi yang benar untuk nilai [5] pada nilai [0..4]. Dalam hal ini trnyata lokasi persisnya adalah 5, tetapi tidak ada offset dengan sendirinya, sehingga lariknya terlihat seperti ini:



Step -7: Temukan lokasi yang benar untuk nilai [6] pada nilai [0..5]. Dalam hal ini, trnyata lokasi persisnya adallah 6, tetapi dengan sendirinya tidak ada offset, sehingga lariknya terlihat seperti ini:



Step -8: Temukan lokasi yang benar untuk nilai [7] dalam nilai [0..6]. Dalam hal ini trnyata letak yang benar adalah 1, kemudian Nilai [1], Nilai [2], Nilai [3], Nilai [4], Nilai [5], Nilai [6] digeser satu posisi ke kanan shingga Nilai [7] di posisi 1, kita mendapatkan array brikut:



Slesai. Dan aray tlah diurutkan dalam urutan mnaik. Sebuah metod untuk mengurutkan elmen aray dalam urutan menaik menggunakan metode Sortir Penyisipan.

C. SOAL LATIHAN/TUGAS

Latihan	Petunjuk Pengerjaan Tugas
Latihan 9	1. Jelaskan Konsep Metode Shell Sort, Marge Sort, dan Insertion Sort ? 2. Sebutkan dan Jelaskan Kelebihan dan Kekurangan dari Metode Shell Sort, Marge Sort dan Insertion Sort ? 3. Terdapat urutan data berikut : 12 7 9 10 13 15 16 1 tulislah urutan proses untuk mengurutkan data Dengan menggunakan “Shell Sort” ? 4. Terdapat urutan data berikut : 12 7 9 10 13 15 16 1 tulislah urutan proses untuk mengurutkan data Dengan menggunakan “Marge Sort” ? 5. Terdapat urutan data berikut : 12 7 9 10 13 15 16 1 tulislah urutan proses untuk mengurutkan data Dengan menggunakan “Insertion Sort” ?

D. REFERENSI

C and Data Structures by Practice by Ramesh Vasappanavara

Data Structures Program Design in C++ by KruseDordal, P. L. (2020). *An Introduction to Computer Network.* Chicago: Loyola University Chicago.

Forouzan, B. A. (2013). *Data Communications and Networking.* New York: McGraw-Hill.

Goralski, W. (2017). *The Illustrated Network.* Cambridge: Morgan Kaufmann.

Kurose, J. F., & Ross, K. W. (2017). *Computer Networking: A Top-down Approach.* Pearson.

Lowe, D. (2018). *Networking All-In-One.* Hoboken: John Wiley & Sons, Inc.

Peterson, L. L., & Davie, B. S. (2010). *Computer Networks.* Burlington: Kaufmann.

Sudiendro, H. (2013). *Teknik Dasar Telekomunikasi.* Jakarta: Kementrian Pendidikan & Kebudayaan.

Sukaridhoto, S. (2014). *Buku Jaringan Komputer I.* Surabaya: Politeknik Elektronika Negeri Surabaya (PENS).

Tanenbaum, A. S., & Wetherall, D. J. (2011). *Computer Networks.* Pearson Prentice Hall.