STRUKTUR DATA – TI.20.B.1
TEKNIK INFORMATIKA – UNIVERSITAS PELITA BANGSA
TUGAS PERTEMUAN – 4

Nama : Febro Herdyanto
NIM : 312010043
Mata Kuliah : Struktur Data
Dosen : Candra Naya,S.Kom.,M.Kom

SOAL :

1. Carilah 3 metode sorting lainnya dan tuliskan dalam paper beserta source code, cara dan analisis dan tiap-tiap metode sorting yang ada!
   Buatlah semua procedure-procedure yang ada di atas dalam program utuh!

JAWABAN :

Metode Sorting

a. **Shell Sort**

Shell sort merupakan algoritma yang pertama-tama mengurutkan elemen jauh satu sama lain dan secara berturut-turut mengurangi interval antara elemen yang akan diurutkan. Ini adalah versi umum dari semacam penyisipan. Dalam Shell Sort, elemen pada interval tertentu diurutkan. Interval antar elemen dikurangi secara bertahap berdasarkan urutan yang digunakan. Performa jenis shell bergantung pada jenis urutan yang digunakan untuk larik masukan yang diberikan.

Proses sorting :

Pertama-tama adalah menentukan jarak mula-mula dari data yang akan dibandingkan, yaitu N / 2. Data pertama dibandingkan dengan data dengan jarak N / 2. Apabila data pertama lebih besar dari data ke N / 2 tersebut maka kedua data tersebut ditukar. Kemudian data kedua dibandingkan dengan jarak yang sama yaitu N / 2. Demikian seterusnya sampai seluruh data dibandingkan sehingga semua data ke-j selalu lebih kecil daripada data ke-(j + N / 2).

Pada proses berikutnya, digunakan jarak (N / 2) / 2 atau N / 4. Data pertama dibandingkan dengan data dengan jarak N / 4. Apabila data pertama lebih besar dari data ke N / 4 tersebut maka kedua data tersebut ditukar. Kemudian data kedua dibandingkan dengan jarak yang sama yaitu N / 4. Demikianlah seterusnya hingga seluruh data dibandingkan sehingga semua data ke-j lebih kecil daripada data ke-(j + N / 4).

Pada proses berikutnya, digunakan jarak (N / 4) / 2 atau N / 8. Demikian seterusnya sampai jarak yang digunakan adalah 1.

Algoritma metode Shell dapat dituliskan sebagai berikut :

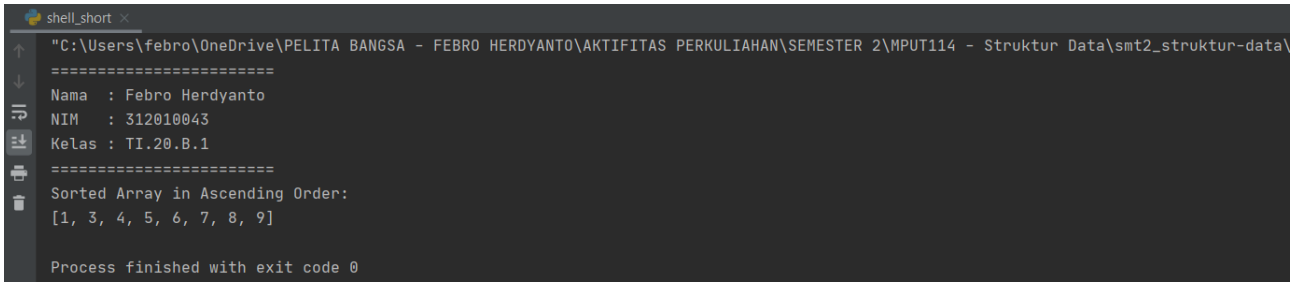
- Jarak = N
- Selama (Jarak > 1) kerjakan baris 3 sampai dengan 9
- Jarak = Jarak / 2. Sudah = false
- Kerjakan baris 4 sampai dengan 8 selama Sudah = false
- Sudah = true
- j = 0
- Selama (j < N – Jarak) kerjakan baris 8 dan 9
- Jika (Data[j] > Data[j + Jarak] maka tukar Data[j], Data[j + Jarak].
  Sudah = true
- 9. j = j + 1

```
shellSort(array, size)
for interval i <- size/2n down to 1
for each interval "i" in array
sort all the elements at interval "i"
end shellSort
```

```
print('========================')
print('Nama  : Febro Herdyanto')
print('NIM   : 312010043')
print('Kelas : TI.20.B.1')
print('========================')
def shellSort(array, n):
    interval = n // 2
    while interval > 0:
        for i in range(interval, n):
            temp = array[i]
            j = i
            while j >= interval and array[j - interval] > temp:
                array[j] = array[j - interval]
                j -= interval
            array[j] = temp
        interval //= 2
data = [9, 8, 3, 7, 5, 6, 4, 1]
size = len(data)
shellSort(data, size)
print('Sorted Array in Ascending Order:')
print(data)
```

STRUKTUR DATA – TI.20.B.1
TEKNIK INFORMATIKA – UNIVERSITAS PELITA BANGSA
TUGAS PERTEMUAN – 4

Nama : Febro Herdyanto
NIM : 312010043
Mata Kuliah : Struktur Data
Dosen : Candra Naya,S.Kom.,M.Kom



STRUKTUR DATA – TI.20.B.1
TEKNIK INFORMATIKA – UNIVERSITAS PELITA BANGSA
TUGAS PERTEMUAN – 4

Nama : Febro Herdyanto
NIM : 312010043
Mata Kuliah : Struktur Data
Dosen : Candra Naya,S.Kom.,M.Kom

b. Merger Sort

Merge Sort adalah algoritma Divide and Conquer. Ini membagi larik masukan dalam dua bagian, memanggil dirinya sendiri untuk dua bagian dan kemudian menggabungkan dua bagian yang diurutkan. Fungsi merge () digunakan untuk menggabungkan dua bagian. Penggabungan (arr, l, m, r) adalah proses kunci yang mengasumsikan bahwa arr [l..m] dan arr [m + 1..r] diurutkan dan menggabungkan dua sub-array yang diurutkan menjadi satu.

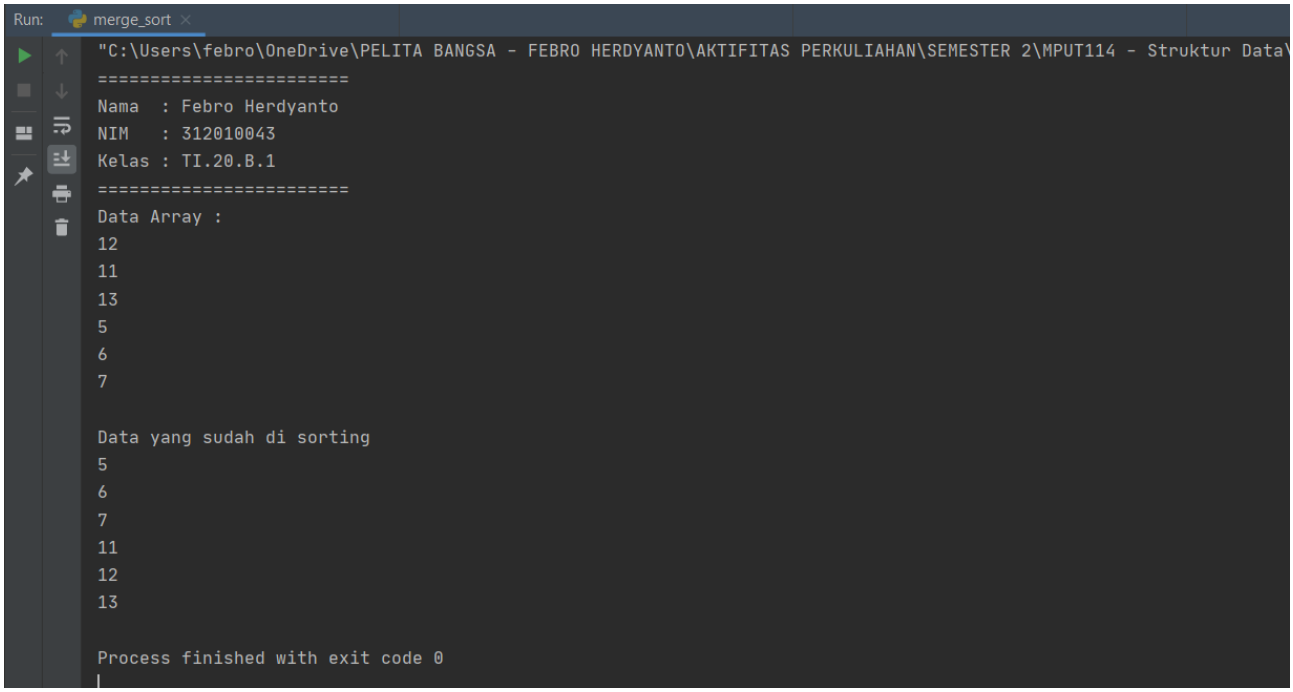
Ini adalah pendekatan rekursif untuk mengimplementasikan jenis gabungan. Langkah-langkah yang diperlukan untuk mendapatkan array yang diurutkan melalui metode ini dapat ditemukan di bawah ini:

- Daftar ini dibagi menjadi kiri dan kanan di setiap panggilan rekursif sampai diperoleh dua elemen yang berdekatan.
- Sekarang mulailah proses penyortiran. I dan j iterator melintasi dua bagian di setiap panggilan. K iterator melintasi seluruh daftar dan membuat perubahan di sepanjang jalan.
- Jika nilai pada i lebih kecil dari nilai pada j, kiri [i] ditetapkan ke slot myList [k] dan i bertambah. Jika tidak, maka kanan [j] dipilih.
- Dengan cara ini, nilai yang diberikan melalui k semuanya diurutkan.
- Di akhir putaran ini, salah satu bagian mungkin belum dilintasi sepenuhnya. Nilainya hanya ditetapkan ke slot yang tersisa dalam daftar.

Dan dengan itu, jenis gabungan telah diterapkan!

```
def merge(arr, l, m, r):
    n1 = m - l + 1
    n2 = r - m

    L = [0] * n1
    R = [0] * (n2)

    for i in range(0, n1):
        L[i] = arr[l + i]

    for j in range(0, n2):
        R[j] = arr[m + 1 + j]

    i = 0
    j = 0
    k = l
    while i < n1 and j < n2:
        if L[i] <= R[j]:
            arr[k] = L[i]
            i += 1
        else:
            arr[k] = R[j]
            j += 1
        k += 1
    while i < n1:
        arr[k] = L[i]
        i += 1
        k += 1
    while j < n2:
        arr[k] = R[j]
        j += 1
        k += 1

def mergeSort(arr, l, r):
    if l < r:
        m = (l + (r - 1)) // 2
        mergeSort(arr, l, m)
        mergeSort(arr, m + 1, r)
        merge(arr, l, m, r)

arr = [12, 11, 13, 5, 6, 7]
n = len(arr)
print("Data Array : ")
for i in range(n):
    print("%d" % arr[i]),

mergeSort(arr, 0, n - 1)
print("\nData yang sudah di sorting")
for i in range(n):
    print("%d" % arr[i]),
```

Nama : Febro Herdyanto
NIM : 312010043
Mata Kuliah : Struktur Data
Dosen : Candra Naya,S.Kom.,M.Kom



c. Quick Sort

Quick Sort adalah salah satu algoritma pengurutan data yang paling cepat, yaitu dengan membagi list menggunakan sebuah pivot. Quick Sort juga menggunakan rekursif dalam algoritmanya.
Misalnya jika pengurutannya secara ascending, data yang kurang dari pivot sudah ditentukan ditaruh disebelah kirinya pivot sedangkan data yang lebih besar dari pivot maka ditaruh disebelah kanan pivot. Setelah dipisah, bagian-bagian yang sudah dipisah melakukan pembagian seperti sebelumnya karena ini sehingga Quick Sort menggunakan rekursif. Begitu seterusnya sampai data terurut.

```
print('========================')
print('Nama  : Febro Herdyanto')
print('NIM   : 312010043')
print('Kelas : TI.20.B.1')
print('========================')

def quickSort(alist):
  quickSortHelper(alist,0,len(alist)-1)
def quickSortHelper(alist,start,end):
  if start >= end:
    return
  i,j = start, end
  pivot = (start + (end - start)//2)
  while i<=j:
    while(alist[i] < alist[pivot]):
      i+=1
    while(alist[j] > alist[pivot]):
      j-=1
    if(i<=j):
      alist[i],alist[j] = alist[j],alist[i]
      i+=1
      j-=1
  quickSortHelper(alist,start,j)
  quickSortHelper(alist,i,end)
  return alist

alist = [54,26,93,17,77,31,44,55,20]

print('Data sebelum Sort : ')
print(alist)
print()
print('Data yang sudah di Sort : ')
quickSort(alist)

print(alist)
```

STRUKTUR DATA – TI.20.B.1
TEKNIK INFORMATIKA – UNIVERSITAS PELITA BANGSA
TUGAS PERTEMUAN – 4

Nama : Febro Herdyanto
NIM : 312010043

Mata Kuliah : Struktur Data
Dosen : Candra Naya,S.Kom.,M.Kom