PENUKARAN UANG DAN PEMAMPATAN DATA MENGGUNAKAN ALGORITMA GREEDY

Anggota Kelompok:
Muhammad Faruq Al-Fauzi S. (221011015)
Muhammad Aldi Alfatih (221011044)
Nirmalasari Rodito Sulnas (221011022)

IK22-D

Teori Algiritma Greedy

Algoritma greedy merupakan salah satu dari sekian banyak algoritma yang biasa digunakan.
algoritma greedy ini merupakan algoritma yang di gunakan untuk memecahkan persoalan optimal meskipun hasilnya tidak selalu merupakan solusi optimal. prinsip utumanya ialah mengambil sebnayk mungkin apa yang dapat di peroleh sekarang

Kelebihan dan Kelemahan Algiritma Greedy

1. kelebihan

- Memudahkan diimplementasikan pada persoalan persoalanyang ada.
- Efisien, karena algoritma ini selalu berusaha memilih hasil yang optimal

2. Kekurangan

- Tidak ada penanganan kasus jika titik tujuan tidak di temukan
- tidak dapat melakukan pemantauaan parameter yang melibatkan lebih dari satu perangkat

KELOMPOK 8

Masalah Penukaran Uang

Cara Kerja Algoritma Greedy pada Masalah Penukaran Uang

Rumus dasar:

Total Uang = (d1 * x1) + (d2 * x2) + ... + (dn * xn)

- Nilai Uang yang Ditukar: A
- d1, d2, ..., dn: Himpunan koin.
- x1, x2, ..., xn: Himpunan solusi.

2. Syarat:

- Nilai Uang yang Ditukar >= A: Total uang harus sama dengan atau lebih besar dari jumlah yang perlu ditukar (A).
- x1, x2, ..., xn >= 0: Jumlah uang tidak boleh negatif.

3. Mencari solusi:

Ada beberapa metode untuk mencari solusi dari rumus ini, seperti:

- Metode coba-coba: Mencoba berbagai kombinasi nilai x1, x2, ..., xn hingga memenuhi syarat dan menghasilkan Total Uang Kembalian minimum.
- Metode pemrograman linier: Menggunakan software pemrograman linier untuk mencari solusi optimal.

Obyektif persoalan adalah

Minimisasi $F = \sum_{i=1}^{n} x_i$ (fungsi objektif)

dengan kendala $\sum_{i=1}^{n} d_i x_i = A$

Cara Kerja Algoritma Greedy pada Masalah Penukaran Uang

Contoh:

Misalkan kamu ingin menukar A = Rp. 50,- dan kasir memiliki pecahan C = Rp. 5,-, Rp. 10,-, Rp. 15,-, Rp. 25,- , Rp. 30,-

Total Uang = (5 * x1) + (10 * x2) + (15 * x3) + (25 * x4)+ (30 * x5)

Syarat:

- Total Uang Kembalian >= 50: 5x1 +10x2 + `15x3 + 25x4 + 30x5 >= 50
- x1, x2, x3, x4, x5 >= 0: Jumlah uang tidak boleh negatif.

Solusi:

Salah satu solusi untuk masalah ini adalah x1 = 1, x2 = 0, x3 = 1, x4 = 0, x5 = 1. Artinya, kasir hanya perlu memberikan 1 lembar uang Rp. 50,- sebagai kembalian.

PENYELESAIAN DENGAN EXHAUSTIVE SEARCH

sSeperti contoh soal di atas, kita akan mengerjakan menggunakn penyelesaain Exhaustive search nyaa

kemungkinan kombinasi koin untuk menghasilkan Rp. 50,-. Tabel tersebut menunjukkan:

- Kolom pertama: Pecahan koin (5, 10, 15, 25, 30).
- Kolom kedua: Jumlah koin yang digunakan untuk setiap pecahan.
- Kolom ketiga: Total nilai koin untuk setiap kombinasi.

Penjelasan:

Misalkan kita ingin menukar Rp. 50,- dengan menggunakan koin-koin yang tersedia (5, 10, 15, 25, 30). Kita dapat menggunakan rumus di atas untuk menghitung total nilai koin untuk setiap kombinasi yang mungkin.

PENYELESAIAN DENGAN EXHAUSTIVE SEARCH

ontoh:

Kombinasi 1:

- 5 koin 10 rupiah (x2 = 5)
- Total nilai koin: (10 * 5) = Rp. 50,-

Kombinasi 2:

- 2 koin 25 rupiah (x4 = 2)
- Total nilai koin: (25 * 2) = Rp. 50,-

Kombinasi 3:

- 1 koin 25 rupiah (x4 = 1)
- 1 koin 15 rupiah (x3 = 1)
- 1 koin 10 rupiah (x2 = 1)
- Total nilai koin: (25 * 1) + (15 * 1) + (10 * 1) = Rp. 50,-

Pemampatan Data dengan Algoritma Huffman

Kompresi dengan algoritma Huffman dilakukan dengan cara:

1. Hitung frekuensi kemunculan setiap simbol.
2. Pilih dua buah simbol dengan frekuensi terkecil, lalu gabungkan dalam satu tangkai
3. Ulangi langkah kedua hingga tidak ada lagi tangkai yyang dapat di gabungkan.

Misalkan, terdapat sebuah pesan : ABABAAAADDDCCCFBB
pesan berukuran **17 byte**(termasuk spasi)

- **Pertama, kita akan menghitung kemunculan setiap karakter:**

Simbol	Kemunculan
A	6
B	4
C	3
D	3
F	1

- **Pilih dua buah simbol dengan frekuensi terkecil, yaitu simbol F dan D, lalu gabungkan**

Simbol	Kemunculan
A	6
B	4
C	3
(D,F)	4

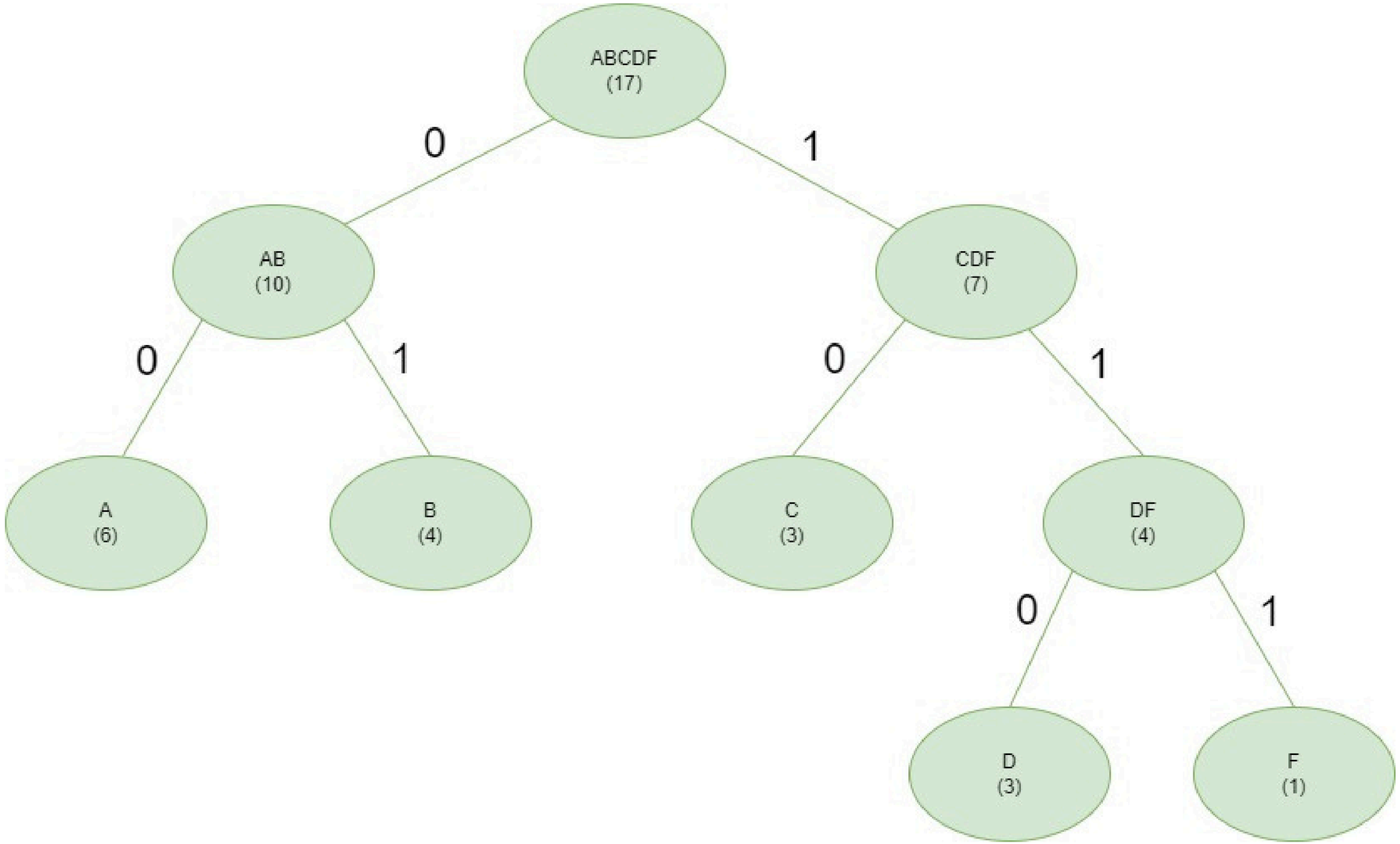
- **Pilih kembali dua buah simbol dengan frekuensi terkecil, lalu gabungkan.**

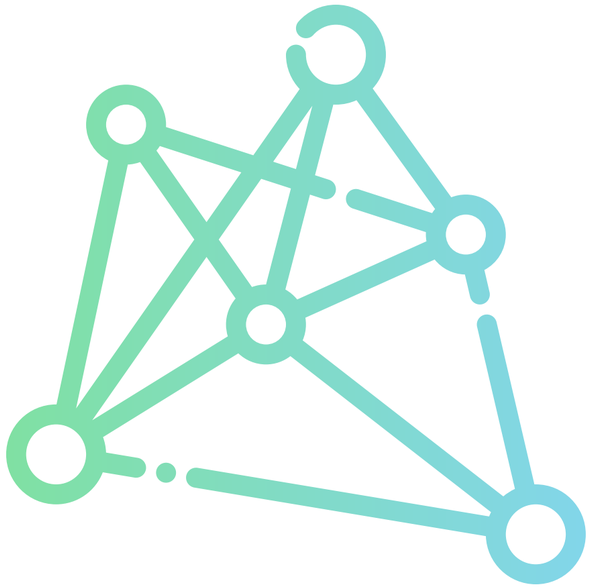
Simbol	Kemunculan
A	6
B	4
(C(D,F))	7

Simbol	Kemunculan
(A,B)	10
(C(D,F))	7

Simbol	Kemunculan
(A,B), (C(D,F))	17

Pembentukan pohon Huffman:





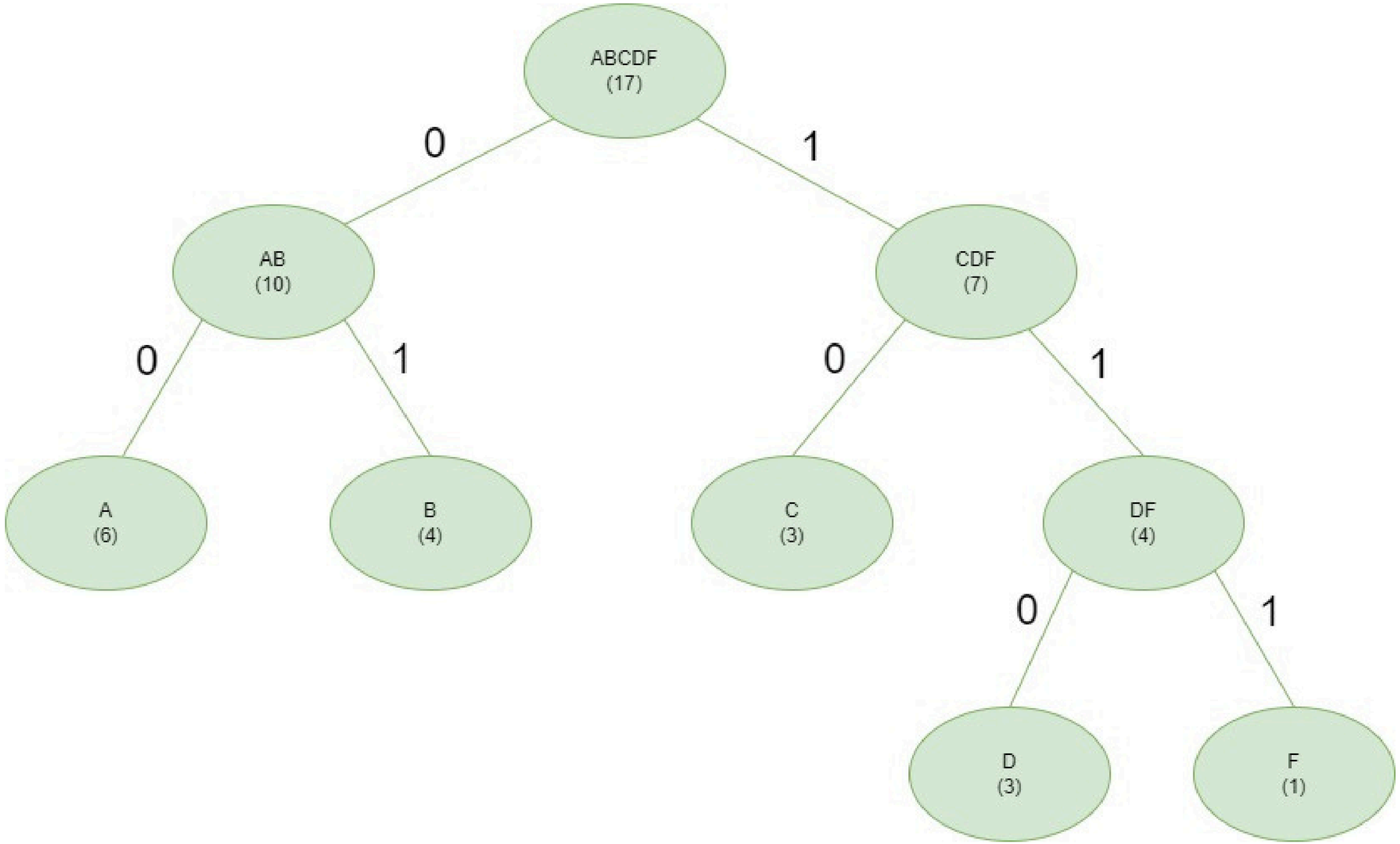
ENCODING

Encoding adalah cara menyusun *string* biner dari teks yang ada

Langkah-langkah untuk men-*encodeng* suatu string biner adalh sebagai berikutL

1. Tentukan karakter yang akan di-*encoding*
2. Mulai dari akar, baca setiap bit yang ada pada cabang yang bersesuaian sampai ketemu daun dimana karakter itu berada
3. Ulangi langkah 2 sampai seluruh karakter di-*encoding*

Pembentukan pohon Huffman:



Berdasarkan tabel diatas, maka “ABABAAAADDDCCCFBB” dapat kita kodekan menjasi seperti berikut:

00010001000000001101101101010101110101

Data hasil hasil kompresi berukuran **38 bit = 4 byte (1 byte = 8 bit)** dengan demikian, kita telah menghemat sebanyak **13 byte (76%)**

RumusPerhitungan Rasio Kompresi:

(100% - Ukuran setelah di kompresi) X 100% (dibulatkan)
Ukuran sebelum di konfersi

Merge Sort

Sorting

Sorting merupakan merupakan proses mengurutkan data sesuai aturan tertentu:

Ascending: dari terkecil sampai terbesar
Descending: dari terbesar sampai terkecil

Tujuan sorting adalah untuk mempercepat proses pencarian (searching) data

Sorting

Tekcik pengurutan sederhana:

- Bubble short
- Selection sort
- Insertion sort

Teknik penggurutan lanjut:

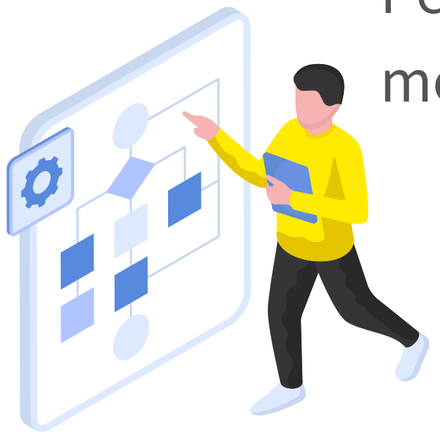
- Quick sort
- Merge sort

Merge sort

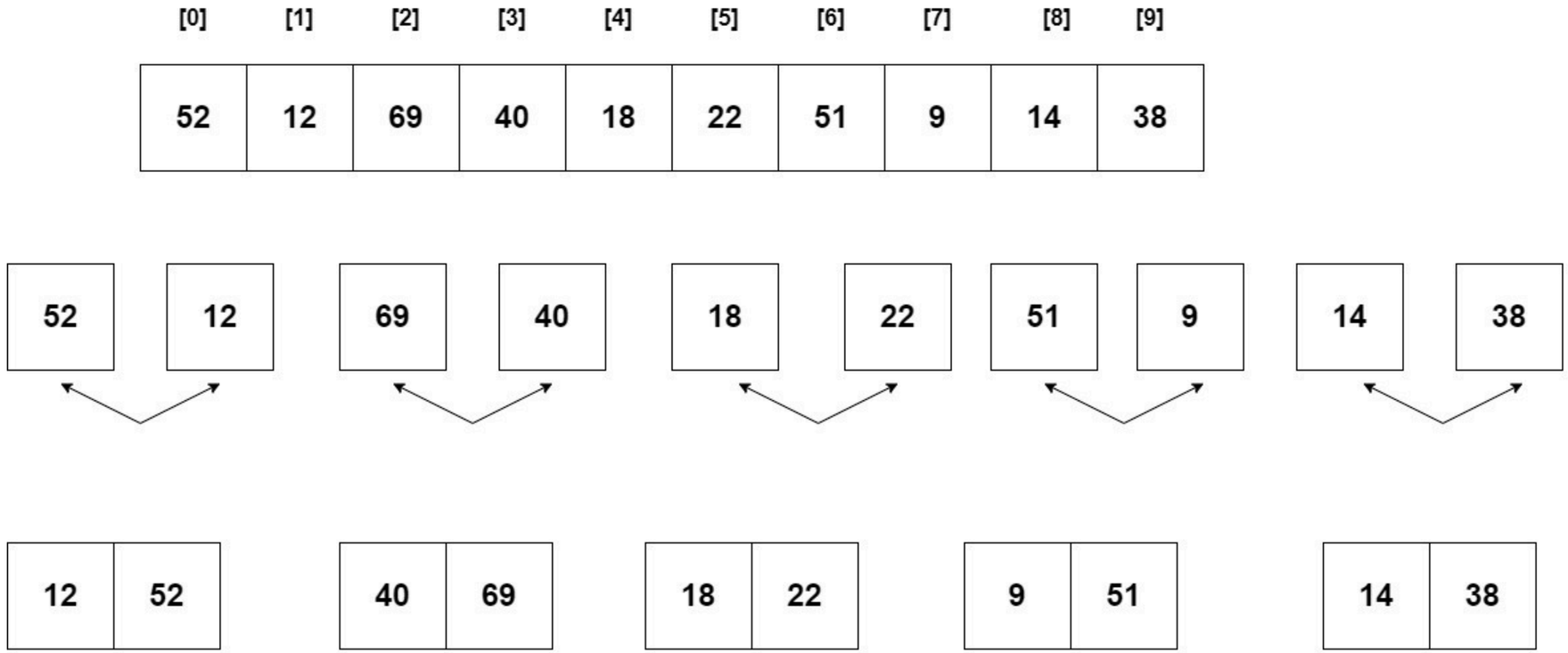
Pada tahap awal, semua data dibagi menjadi bagain/ kelompok data terkecil.

Kemudian tiap dua data digabung menjadi 1 kelompok data gabungan.

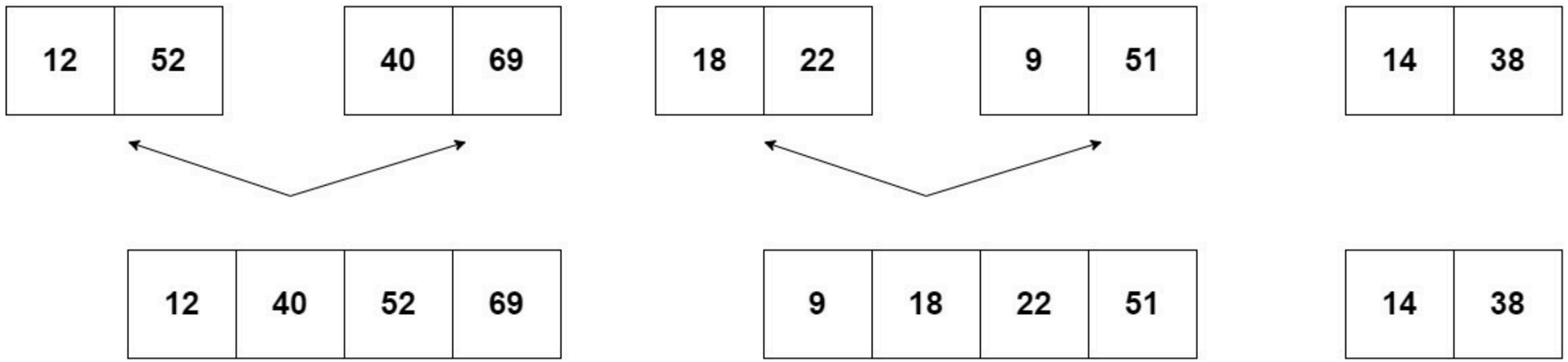
Penggabungan data diteruskan sampai semua data terintegrasi menjadi satu



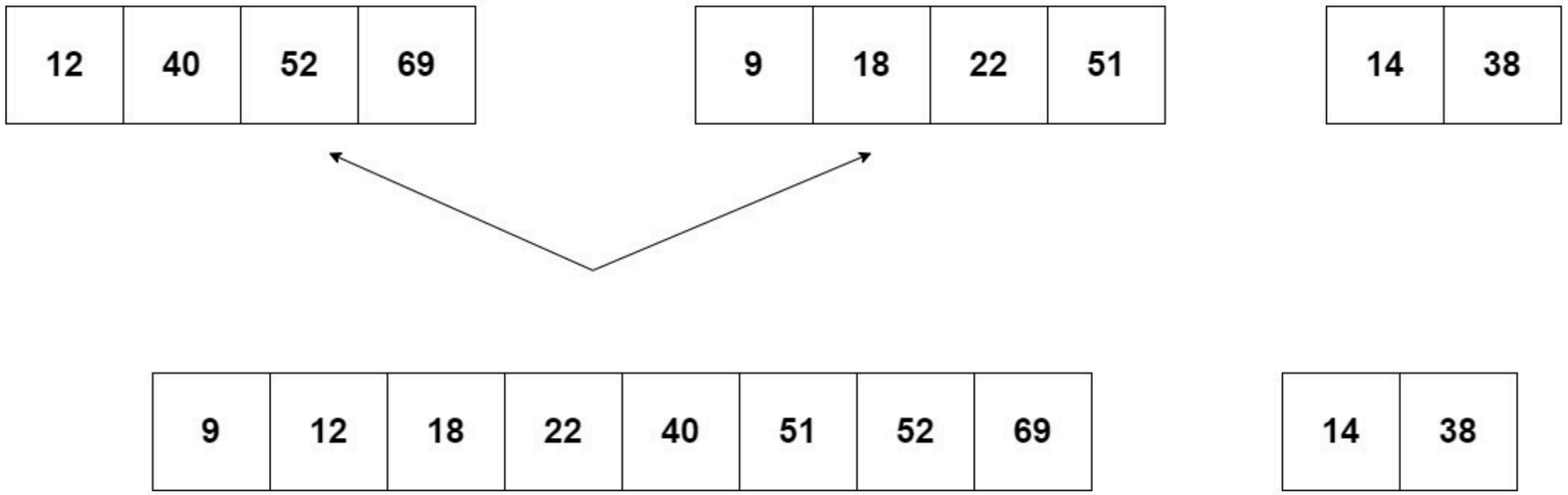
Merge sort


[0] [1] [2] [3] [4] [5] [6] [7] [8] [9]
52 12 69 40 18 22 51 9 14 38
52 12 69 40 18 22 51 9 14 38
12 52 40 69 18 22 9 51 14 38

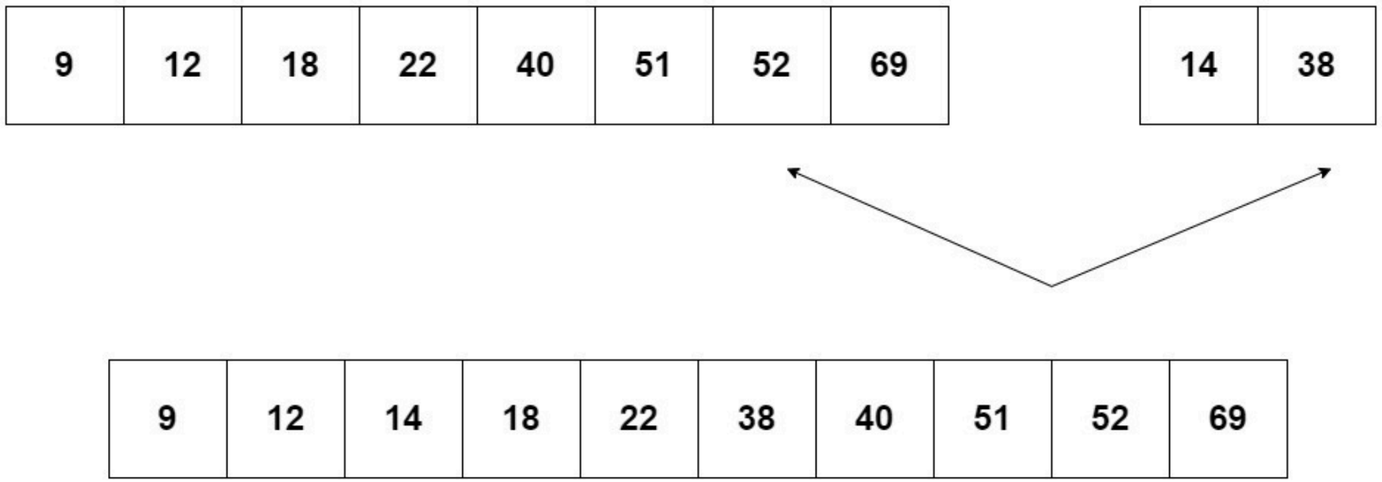
Merge sort



Merge sort



Merge sort



Thank you!