Menggunakan Struktur Data

Array

- Dapat berisi lebih dari sebuah data
- Jika menggunakan Type ‘Array’ dapat diisi dengan nilai Type apapun
- Kotlin memiliki Primitive Type Array yang hanya dapat diisi dengan nilai Primitive Type tertentu (Int, Short, Long, ...)
- Jumlah data Array bersifat tetap -> struktur data statis

Fungsi-fungsi Type Array

- Get dan set untuk memanggil dan mengisi sebuah item Array -> simbol [index]
- Size untuk memperoleh jumlah item
- Find untuk memperoleh sebuah item pertama yang memenuhi persyaratan predicate (lambda expression) yang diberikan
- IndexOf untuk memperoleh nomor index dari item yang dicari
- Sort dan sortDecending untuk mengurutkan item
- Shuffle untuk mengubah urutan item secara acak
- Random untuk memanggil sebuah item secara acak

Fungsi-fungsi Type Array

- Average untuk memperoleh nilai rata-rata dari item IntArray
- Sum untuk memperoleh nilai total dari item IntArray

Latihan Array

- Buat project baru bernama LatihanArray
- Buat sebuah variabel dengan Type Array
- Gunakan fungsi-fungsi yang dimiliki oleh Type Array

Collection

- Berisi beberapa item yang memiliki Type sama
- List -> collection dengan item yang terurut berdasarkan index berupa angka. Item di dalam list dapat bernilai sama.
- Set -> collection dengan item yang unik. Item di dalam set tidak ada yang bernilai sama.
- Map -> collection berupa pasangan key-value. Key pada map bersifat unik, tidak dapat bernilai sama. Sedangkan value pada map dapat bernilai sama.
- Bersifat dinamis -> dapat diubah jumlah item

Collection

- Immutable Collection -> tidak dapat diubah jumlah itemnya
- Mutable Collection -> dapat diubah jumlah itemnya

List

- Struktur data terurut, item pada list dapat bernilai sama
- Implementasinya adalah ArrayList
- List yang sama adalah list yang memiliki jumlah item yang sama dan memiliki struktur masing-masing item yang sama
- Nomor index list diawali dengan angka 0 dan diakhiri dengan angka lastIndex (list.size - 1)
- List memiliki kemiripan dengan array, bedanya list tidak membutuhkan inisialisasi jumlah item pada saat deklarasi dan jumlah dapat diubah menggunakan fungsi penambahan, perubahan atau penghapusan item
- MutbleList hanya dapat menambahkan item pada index selanjutnya

Set

- Menyimpan item dengan nilai unik, tidak ada item yang bernilai sama
- Implementasinya adalah LinkedHashSet (dapat memanggil item first() dan item last()) atau HashSet
- Hanya boleh memiliki sebuah item null, karena null merupakan nilai unik
- Set yang sama adalah set yang memiliki jumlah item yang sama dan memiliki semua nilai item yang sama tanpa melihat urutan dari item tersebut
- Item set diperoleh dengan memanggil fungsi find, filter, elementAt
- Item set tidak dapat diubah nilai

Map

- Menyimpan item dalam bentuk pasangan key-value
- Implementasinya adalah LinkedHashMap atau HashMap
- Key bersifat unik -> tidak boleh ada yang sama
- Value dapat bernilai sama
- Pasangan key-value dibuat menggunakan kata kunci to, nilai sebelah kiri to adalah key dan nilai sebelah kanan to adalah value
- Map yang sama adalah map yang memiliki jumlah item yang sama dan memiliki pasangan key-value yang sama sebagai item-itemnya
- Pemanggilan item dapat dilakukan dengan simbol [] yang diisi dengan nilai key
- Penambahan item pada MutableMap dapat dilakukan dengan fungsi put atau operasi assignment dengan simbol []

Mendeklarasikan Collection

- Menggunakan listOf, setOf, mapOf, mutableListOf, mutableSetOf, dan mutableMapOf
- Deklarasi collection dapat dilakukan dengna langsung inisialisasi item-itemnya
- Jika hanya ingin mendeklarasikan collection kosong (empty collection), harus berupa mutable collection
- Collection List memiliki initializer function untuk membuat collection berdasarkan jumlah index dan lambda expression
- Copy item collection dilakukan dengan function to{NamaCollection}()

Iterator

- Menggunakan item satu per satu
- Digunakan pada control flow for loop -> membutuhkan variabel baru untuk menyimpan item hasil iterasi
- Digunakan pada function .forEach() -> tidak membutuhkan variabel baru untuk menyimpan item hasil iterasi, melainkan menggunakan kata kunci ‘it’
- Iterator hanya dapat memanggil item selanjutnya (next)
- ListIterator dapat memanggil item selanjutnya (next) maupun item sebelumnya (previous)
- Menerapkan MutableIterator terhadapt mutable collection

Range & Progression

- Sebagai range nilai -> dapat dibuat deklarasi list dengan initializer function (List)
- Range -> kisaran nilai yang terus bertambah nilainya, jarak perubahannya pasti 1 nilai
- Progression -> kisaran nilai yang dapat bertambah atau berkurang, dan jarak perubahannya dapat lebih dari 1 nilai

Sequence

- Sequence bersifat lazy -> menunggu proses terhadap sebuah item selesai dulu baru melanjutkan proses terhadap item selanjutnya
- Iterator bersifat eager -> mengeksekusi proses terhadap semua item secara bersamaan
- Sequence dapat dibuat dengan perintah sequenceOf() atau konversi Type dari Type Iterable termasuk Type Collection (List dan Set)
- Jika ingin menerapkan Sequence pada Collection Map, lakukan konversi terhadap value Map ke List atau key Map ke Set, kemudain hasil konversi tersebut dikonversi ke dalam Sequence
- Operasi Stateless -> memproses item secara mandiri
- Operasi Statefull -> memproses item secara bersama
- Operasi Intermediate -> menghasilkan object Sequence
- Operasi Terminal -> menghasilkan object selain Sequence (.toList() menghasilkan object List)

Operasi terhadap Collection

- Common Operation -> operasi yang dapat diterapkan untuk semua jenis Collection (List, Set, Map)
- Write Operation -> operasi yang hanya dapat diterapkan pada mutable Collection
- Specific Operation -> operasi yang hanya dapat diterapkan pada salah satu jenis Collection
- Umumnya dipanggil melalui operator .
- Dapat juga diterapkan dengan operator lain seperti [], +, -, +=, -=

Common Operation

- Transformation
- Filtering
- Plus & Minus Operation
- Grouping
- Mengambil bagian collection
- Mengambil sebuah item
- Ordering
- Agregate Operation

Transformation

- Menghasilkan collection yang baru berdasarkan collection dasarnya
- Map -> menghasilkan object collection baru berdasarkan proses pernyataan suatu
- Zip -> menghasilkan item berpasangan yang menggabungkan item di posisi yang sama pada dua collection
- Associate -> menghasilkan object collection baru berupa Collection Map yang diasosiasikan dengan hasil operasi terhadap masing-masing item collection tersebut
- Flatten -> menghasilkan object collection baru berdasarkan item-item berupa collection bersarang
- String Representation -> menghasilkan object berdasarkan collection ke dalam bentuk format yang mudah dibaca

Map

- map() -> operasi terhadap nilai item
- mapIndexed() -> operasi terhadapt index dan nilai item
- mapNotNull() -> operasi terhadap nilai item yang tidak menghasilkan nilai null
- mapIndexedNotNull -> operasi terhadap index dan nilai item yang tidak menghasilkan nilai null
- Mapping terhadap Collection Map dilakukan terhadap key-nya saja atau value-nya saja
- mapKeys() -> operasi untuk mengubah key collection map
- mapValues() -> operasi untuk mengubah value collection map
- Variabel it pada mapKeys() dan mapValues() dapat mengakses key dan value item

Zip

- zip() atau infix zip -> menghasilkan collection List yang berisi object pair
- Dilakukan terhadap 2 collection
- Jika collection memiliki jumlah item yang berbeda, akan menghasilkan collection dengan jumlah item terkecil
- Transformasi terhadap nilai item setelah operasi zip, dapat diterapkan dengan menambahkan lambda expression -> memiliki 2 parameter, parameter 1 sama dengan item dari collection 1, parameter 2 sama dengan item dari collection 2
- Bila mempunyai objek List yang berisi object Pair, dapat melakukan operasi yang berlawanan, yaitu operasi Unzip -> menghasilkan object Pair yang berisi masing-masing List dasarnya

Association

- associateWith {} -> menghasilkan map berdasarkan nilai item sebagai key dan hasil operasi sebagai value
- Jika terdapat item collection yang bernilai sama, akan digabungkan menjadi satu pada key collection map yang dihasilkan
- associateBy {} -> menghasilkan map berdasarkan hasil operasi sebagai key dan nilai item sebagai value
- Dapat mengisi kedua key maupun value dari hasil operasi dengan memasukkan argumen keySelector untuk mengisi key pada map, dan argumen valueTransform untuk mengisi value pada map pada perintah associateBy()
- associate{} -> menghasilkan map berdasarkan object Pair yang dihasilkan dari suatu operasi

Flatten

- flatten() -> menggabungkan item-item yang berupa collection bersarang menjadi sebuah collection list
- flatMao() -> melakukan flattening dengan melakukan operasi layaknya function map()

String Representation

- joinToString() -> menghasilkan object string
- joinTo() -> menghasilkan object Appendable
- Collection jika dipanggil di dalam println -> toString() dengan hasil item-item yang dikelilingi dengan []
- Bila menggunakan joinToString() dapat menghilangkan []
- Bila joinTo() dapat menghilangkan [] dan menambahkannya dengan object Appendable
- Separator -> string untuk pemisah antar item
- Prefix -> string yang ditambahkan di depan joinToString()/joinTo()
- Postfix -> string yang ditambahkan di belakang joinToString()/joinTo()
- Limit -> jumlah item yang ditampilkan
- Truncated -> simbol untuk menandakan masih terdapat item lainnya
- Dapat menambahkan operasi lambda expression untuk menerapkan transformasi nilai setiap itemnya