**Laporan Praktikum Algoritma dan Struktur Data**

**Jobsheet 16 - Collection**

**Dosen Pengampu : Triana Fatmawati, S.T., M.T.**



**Nama : Annisa**

**NIM : 2341760032**

**Kelas : SIB 1E**

**Prodi : D-IV Sistem Informasi Bisnis**

**JURUSAN TEKNOLOGI INFORMASI**

**POLITEKNIK NEGERI MALANG**

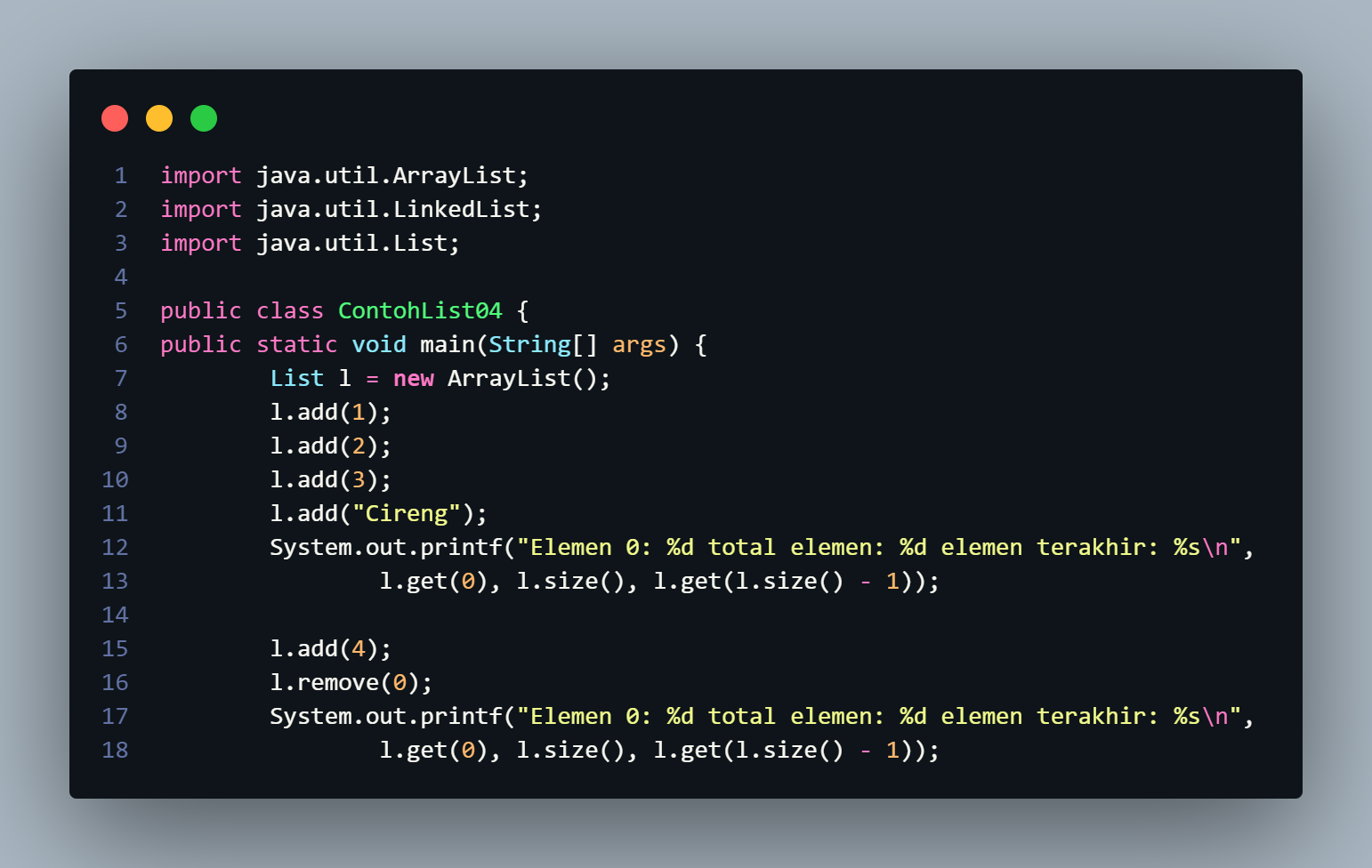
**2023/2024**

**16.1. Tujuan Praktikum**

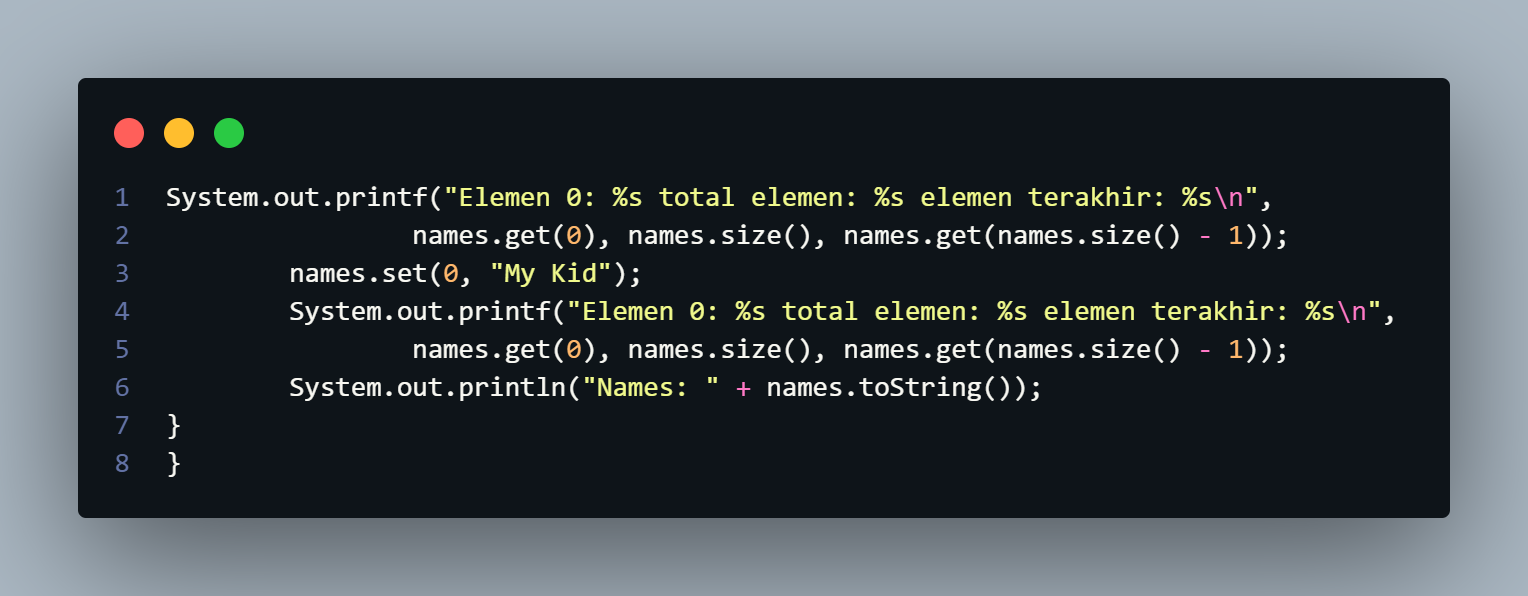
**16.2. Kegiatan Praktikum 1**

**16.2.1. Percobaan 1**

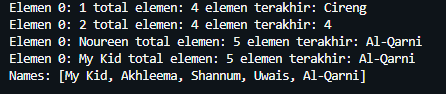
**1.** Buatlah sebuah class ContohList yang main methode berisi kode program seperti di bawah ini

****

**2.** Tambahkan kode program untuk menggunakan collection dengan aturan penulisan kode program seperti berikut

****

**16.2.2. Verifikasi Hasil Percobaan**

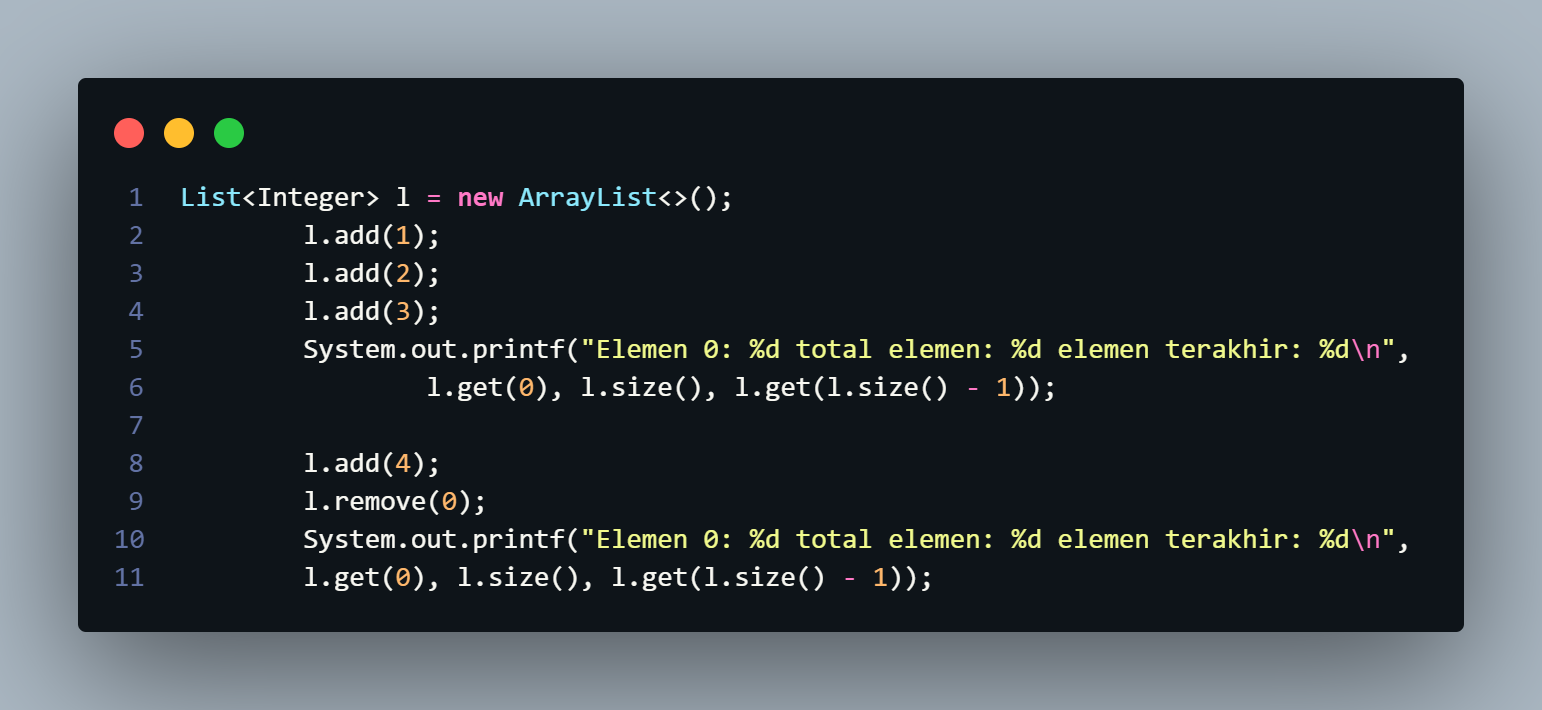
****

**16.2.3. Pertanyaan Percobaan**

**1. Perhatikan baris kode 25-36, mengapa semua jenis data bisa ditampung ke dalam sebuah Arraylist?**

Pada baris kode tersebut, semua jenis data bisa ditampung ke dalam sebuah ArrayList karena tidak ada pembatasan tipe yang ditetapkan pada List, Dimana pendeklarasikan List dilakukan tanpa parameter generik, seperti List l = new ArrayList();, dan menggunakan versi "mentah" dari List, yang secara implisit memperbolehkan objek dari tipe apa pun.

**2. Modifikasi baris kode 25-36 seingga data yang ditampung hanya satu jenis atau spesifik tipe tertentu!**

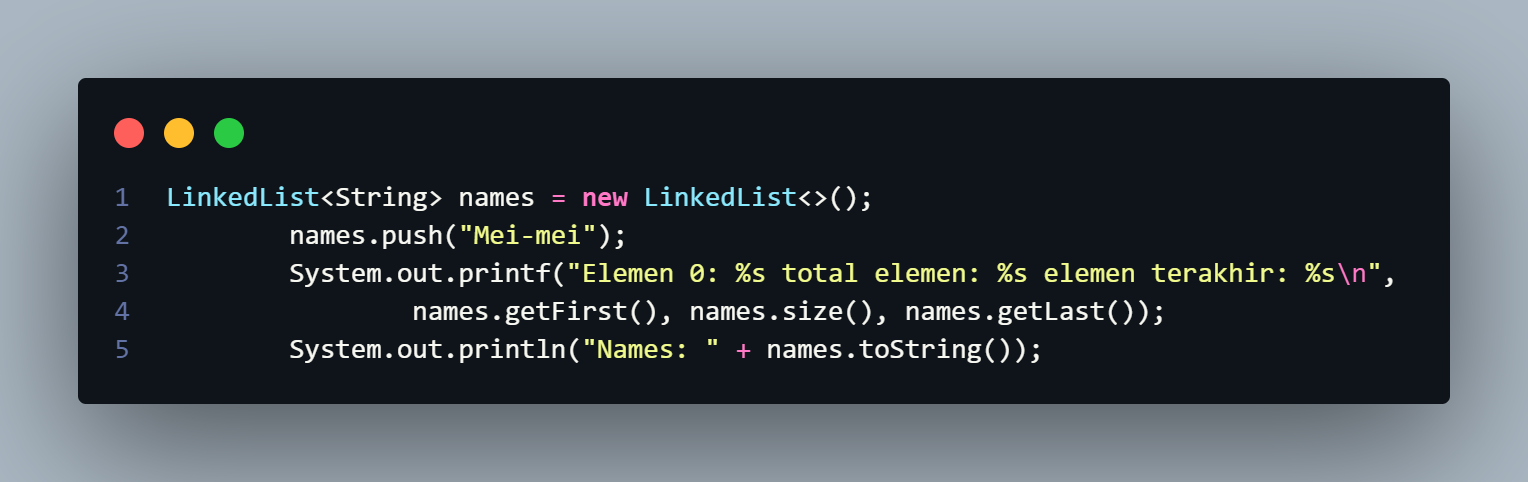
****

Kode yang dimodifikasi yaitu deklarasi daftaryang awalnya List l = new ArrayList(); menjadi List<Integer> l = new ArrayList<>();, memastikan bahwa daftar hanya dapat menampung elemen bertipe Integer. Baris kode l.add("Cireng"); dihapus karena elemen bertipe String tidak sesuai dengan tipe Integer. Penggunaan format specifier pada printf juga diubah dari %s menjadi %d untuk mencocokkan tipe Integer. Perubahan ini dilakukan untuk memastikan keamanan tipe data, sehingga hanya Integer yang dapat ditambahkan ke daftar, menghindari potensi kesalahan runtime dan meningkatkan konsistensi serta keterbacaan kode. Dengan demikian, kesalahan tipe dapat dideteksi saat kompilasi, menjadikan kode lebih aman dan mudah dikelola.

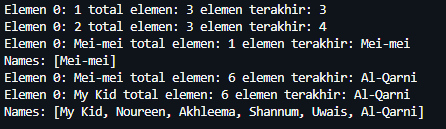
**3. Ubah kode pada baris kode 38 menjadi seperti ini**

**4. Tambahkan juga baris berikut ini, untuk memberikan perbedaan dari tampilan yang sebelumnya**

Berikut merupakan modifikasi dan penambahan kode program pada soal nomor 3 & 4

****

**5. Dari penambahan kode tersebut, silakan dijalankan dan apakah yang dapat Anda jelaskan!**

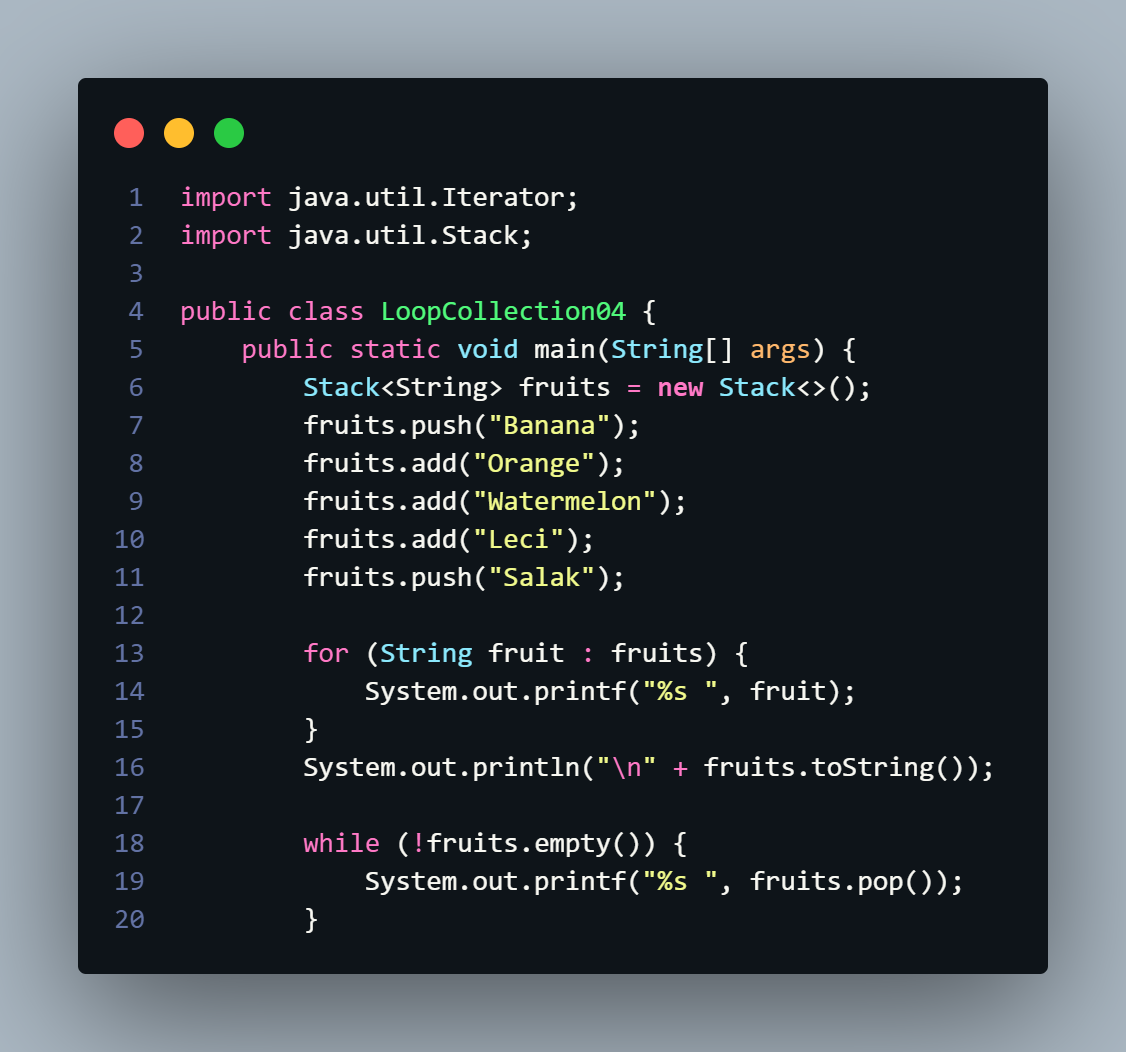
****

Dengan mengganti deklarasi names menjadi LinkedList<String> names = new LinkedList<>(); dan menambahkan baris names.push("Mei-mei");, secara eksplisit memilih LinkedList sebagai jenis implementasi untuk daftar. Penggunaan metode push() menambahkan elemen ke awal daftar, yang berbeda dengan add() yang menambahkan elemen di akhir daftar. Meskipun demikian, penggunaan metode getFirst() dan getLast() dalam perintah printf tetap sama, mengambil elemen pertama dan terakhir dari daftar, secara berurutan. Oleh karena itu, output masih akan menampilkan elemen pertama, jumlah total elemen, elemen terakhir, dan seluruh isi daftar sesuai dengan implementasi LinkedList. Dengan demikian, perubahan ini mempengaruhi perilaku dan cara kerja daftar, tetapi tidak mengubah cara output hasil yang ditampilkan.

**16.3. Kegiatan Praktikum 2**

**16.3.1. Tahapan Percobaan**

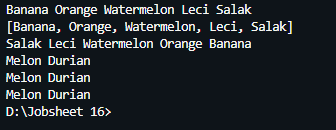
1. Buatlah class dengan nama LoopCollection serta tambahkan method main yang isinya adalah sebagai berikut.

****

2. Tambahkan potongan kode berikut ini dari yang sebelumnya agar proses menampilkan elemen pada sebuah stack bervariasi.

****

**16.3.2. Verifikasi Hasil Percobaan**

****

**16.3.3. Pertanyaan Percobaan**

**1. Apakah perbedaan fungsi push() dan add() pada objek fruits?**

Perbedaan antara push() dan add() pada objek fruits terletak pada fungsionalitas masing-masing dalam konteks sebuah Stack. Saat menggunakan push(), elemen yang ditambahkan akan diletakkan di atas tumpukan (stack), sehingga menjadi elemen teratas. Hal ini khusus untuk Stack, dan sering digunakan Ketika menambahkan elemen baru ke dalam tumpukan, sehingga elemen tersebut menjadi yang paling atas. Di sisi lain, add() adalah metode umum yang ada dalam antarmuka List, yang juga diterapkan oleh Stack. Saat menggunakan add(), elemen akan ditambahkan di akhir tumpukan, yang dalam kasus ini sama dengan menambahkan elemen baru di atas tumpukan karena Stack adalah tipe koleksi terurut. Jadi, penggunaan push() lebih khusus untuk menambahkan elemen ke tumpukan, sementara add() digunakan untuk menambahkan elemen ke koleksi di akhir.

**2. Silakan hilangkan baris 43 dan 44, apakah yang akan terjadi? Mengapa bisa demikian?**

****

Jika baris program fruits.push("Melon"); dan fruits.push("Durian"); dihilangkan, maka tidak akan ada penambahan elemen "Melon" dan "Durian" ke dalam Stack namun elemen lainnya yang telah ditambah sebelumnya ("Banana", "Orange", "Watermelon", "Leci", dan "Salak") masih tetap berada di dalam Stack. Karena LIFO dari Stack, elemen-elemen ini akan tetap diproses dan dicetak terlebih dahulu, sesuai dengan urutan mereka dalam Stack.

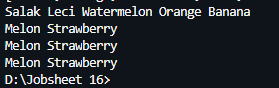
**3. Jelaskan fungsi dari baris 46-49?**

Baris kode tersebut melakukan iterasi melalui setiap elemen dalam objek fruits yang merupakan sebuah Stack dan bertipe String. Untuk melakukan iterasi, kode menggunakan iterator yang disediakan oleh objek fruits. Pada setiap iterasi, kode memeriksa apakah masih ada elemen yang tersisa menggunakan it.hasNext(). Jika masih ada, kode mengambil elemen berikutnya dari fruits menggunakan it.next() dan menyimpannya dalam variabel fruit. Kemudian, kode mencetak nilai fruit ke konsol dengan diakhiri spasi menggunakan System.out.printf(). Proses ini dilakukan untuk setiap elemen dalam fruits, sehingga semua elemen dicetak secara berurutan. Dengan demikian, baris kode tersebut bertanggung jawab untuk mencetak semua elemen dalam fruits ke konsol dengan menggunakan iterator.

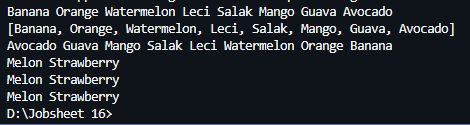
**4. Silakan ganti baris kode 25, Stack<String> menjadi List<String> dan apakah yang terjadi? Mengapa bisa demikian?**

Terdapat kesalahan kompilasi karena perbedaan tipe data. Ketika fruits dideklarasikan sebagai List, kompiler menganggapnya sebagai objek yang mengikuti antarmuka List, bukan sebagai Stack. Karena itu, hanya metode yang didefinisikan dalam antarmuka List yang dapat diakses langsung pada fruits. Namun, metode khusus push(), pop(), dan peek() tidak ada dalam antarmuka List, sehingga kompiler menghasilkan kesalahan kompilasi karena tidak dapat menemukan metode tersebut dalam konteks List.

**5. Ganti elemen terakhir dari dari objek fruits menjadi “Strawberry”!**

**6. Tambahkan 3 buah seperti “Mango”,”guava”, dan “avocado” kemudian dilakukan sorting!**

** **

**16.4. Kegiatan Praktikum 3**

**16.4.1. Tahapan Percobaan**

1. Buatlah sebuah class Mahasiswa dengan attribute, kontruktor, dan fungsi sebagai berikut.



2. Selanjutnya, buatlah sebuah class ListMahasiswa yang memiliki attribute seperti di bawah ini

3. Method tambah(), hapus(), update(), dan tampil()secara berurut dibuat agar bisa melakukan operasi-operasi seperti yang telah disebutkan.

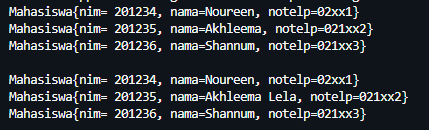
4. Untuk proses hapus, update membutuhkan fungsi pencarian terlebih dahulu yang potongan kode programnya adalah sebagai Berikut



5. Pada class yang sama, tambahkan main method seperti potongan program berikut dan amati hasilnya!



**16.4.2. Verifikasi Hasil Percobaan**

****

**16.4.3. Pertanyaan Percobaan**

**1. Pada fungsi tambah() yang menggunakan unlimited argument itu menggunakan konsep apa? Dan kelebihannya apa?**

Pada fungsi tambah() menggunakan konsep atau method addAll() dari collections Dimana konsep ini Memiliki kelebihan yaitu dapat menambahkan element tanpa ada batas jumlah elemen yang ditambahkan.

**2. Pada fungsi linearSearch() di atas, silakan diganti dengan fungsi binarySearch() dari collection!**

****

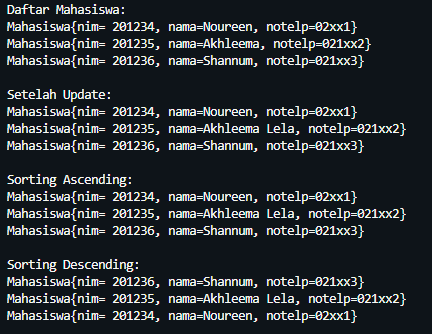
Pada perubahan tersebut tidak mempengaruhi hasil output yang dihasilkan oleh kode lainnya, seperti penambahan, penghapusan, pembaruan mahasiswa, serta tampilan daftar mahasiswa. Jadi, output dari kode akan tetap sama seperti sebelumnya, hanya dengan perubahan cara pencarian data mahasiswa tertentu.

**3. Tambahkan fungsi sorting baik secara ascending ataupun descending pada class tersebut!**

Berikut kode program yang telah dimodifikasi dengan penambahan fungsi sorting ascending & descending

****

Berikut merupakan output dari kode program di atas



**16.5. Tugas Praktikum**

**1. Buatlah implementasi program daftar nilai mahasiswa semester, minimal memiliki 3 class yaitu Mahasiswa, Nilai, dan Mata Kuliah. Data Mahasiswa dan Mata Kuliah perlu melalui penginputan data terlebih dahulu.**

**2. Tambahkan prosedur hapus data mahasiswa melalui implementasi Queue pada collections Tugas nomor 1!**

**Jawab**

Berikut merupakan kode program dari tugas nomor 1 dan 2, disini menggunakan 5 class yaitu Mahasiswa, Nilai, MataKuliah, ListNilai, dan MahasiswaMain

Class Mahasiswa

****

Class Nilai



Class MataKuliah



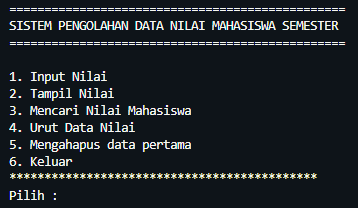
Class ListNilai

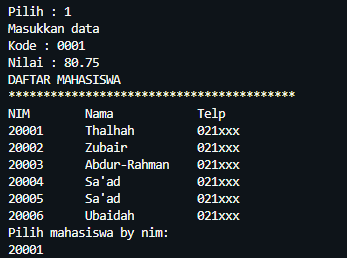


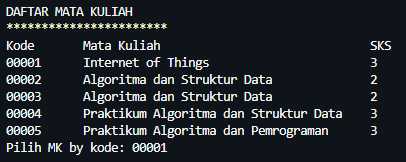
Class MahasiswaMain



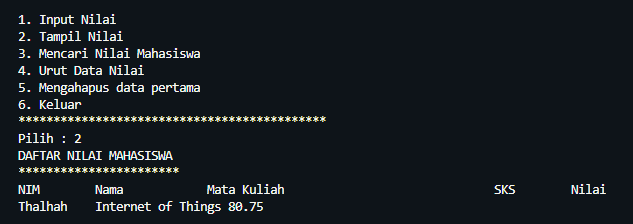
Berikut merupakan output dari kode program



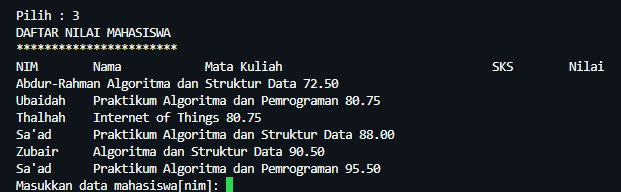




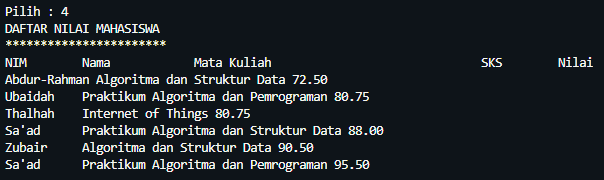
Tampil Nilai



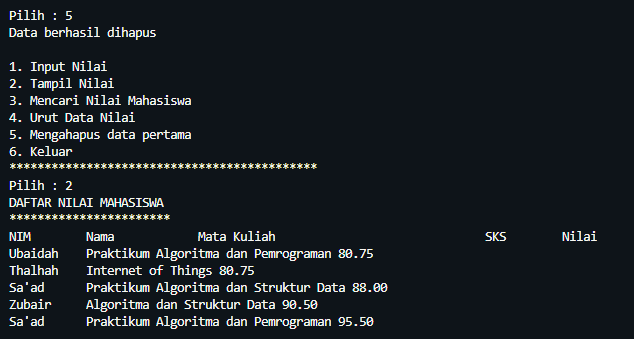
Pencarian Data Mahasiswa



Pengurutan Data Nilai



Penghapusan Data Pertama



**NB: mohon maaf bu atas keterlambatan pengumpulan dikarenakan sudah selesai tapi hilang jadi buat dari awal lagi**