**Nama : Pani Permatasari**

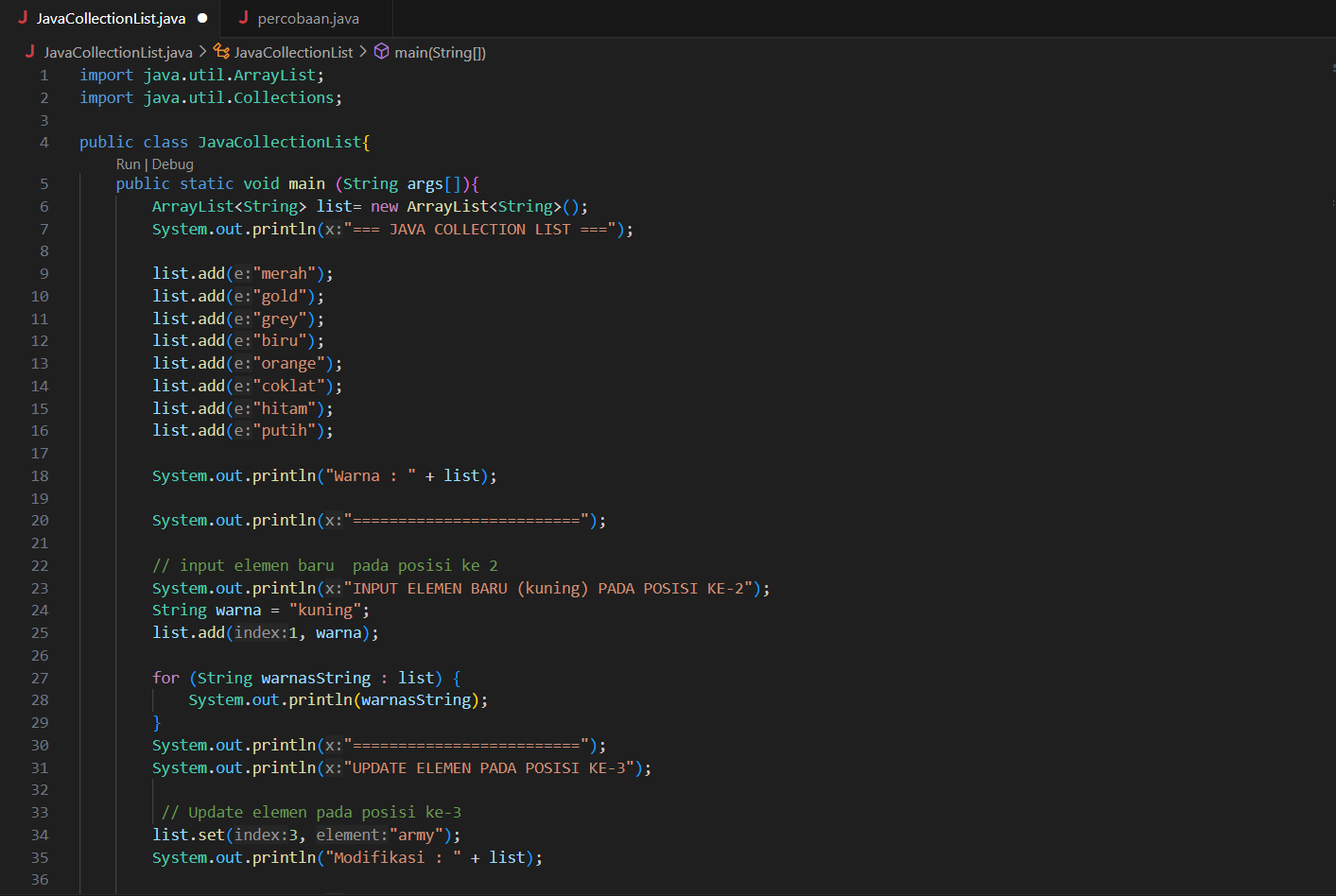
**Nim : 09011182328088**

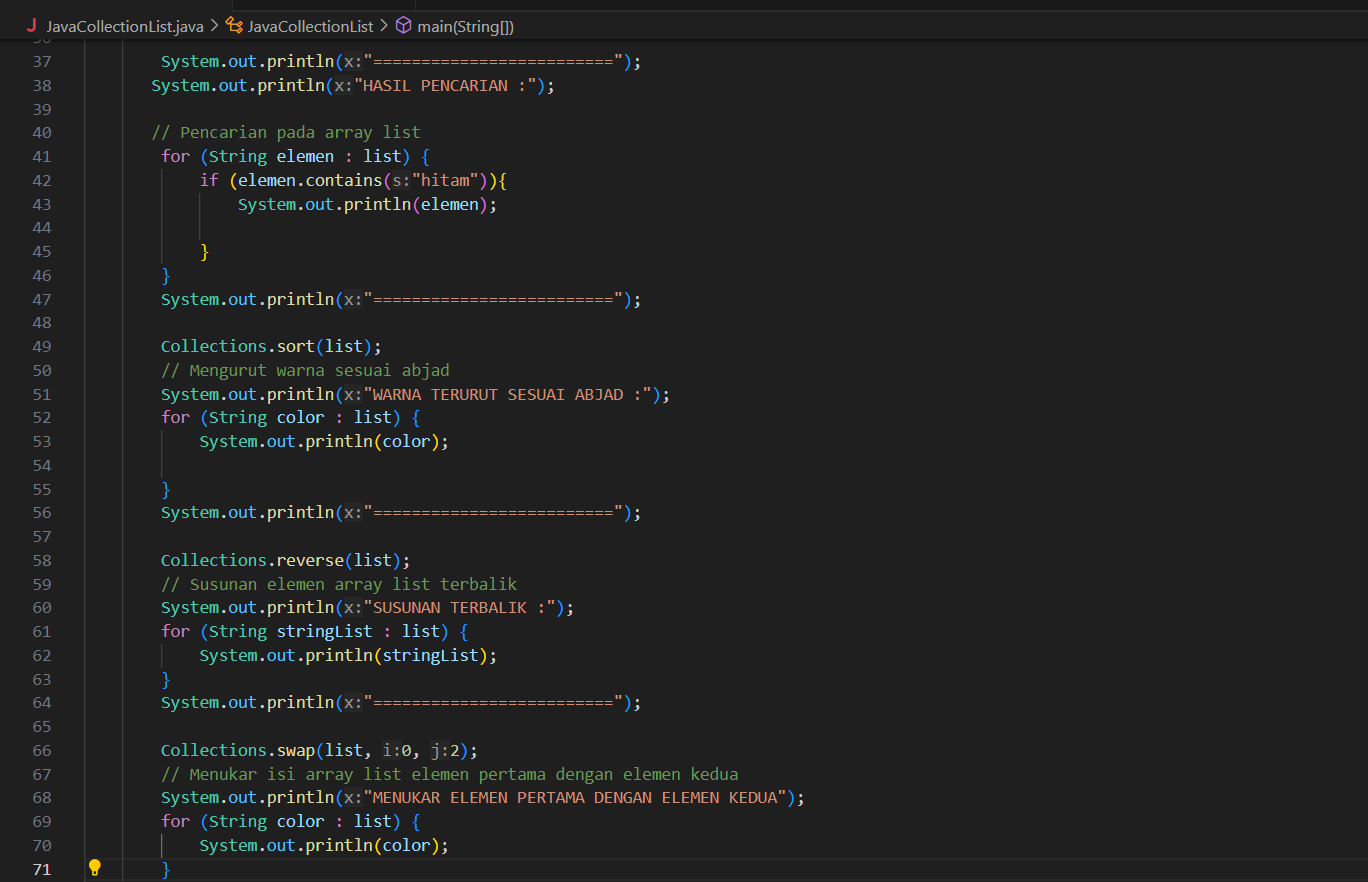
**Kelas : SK1A**

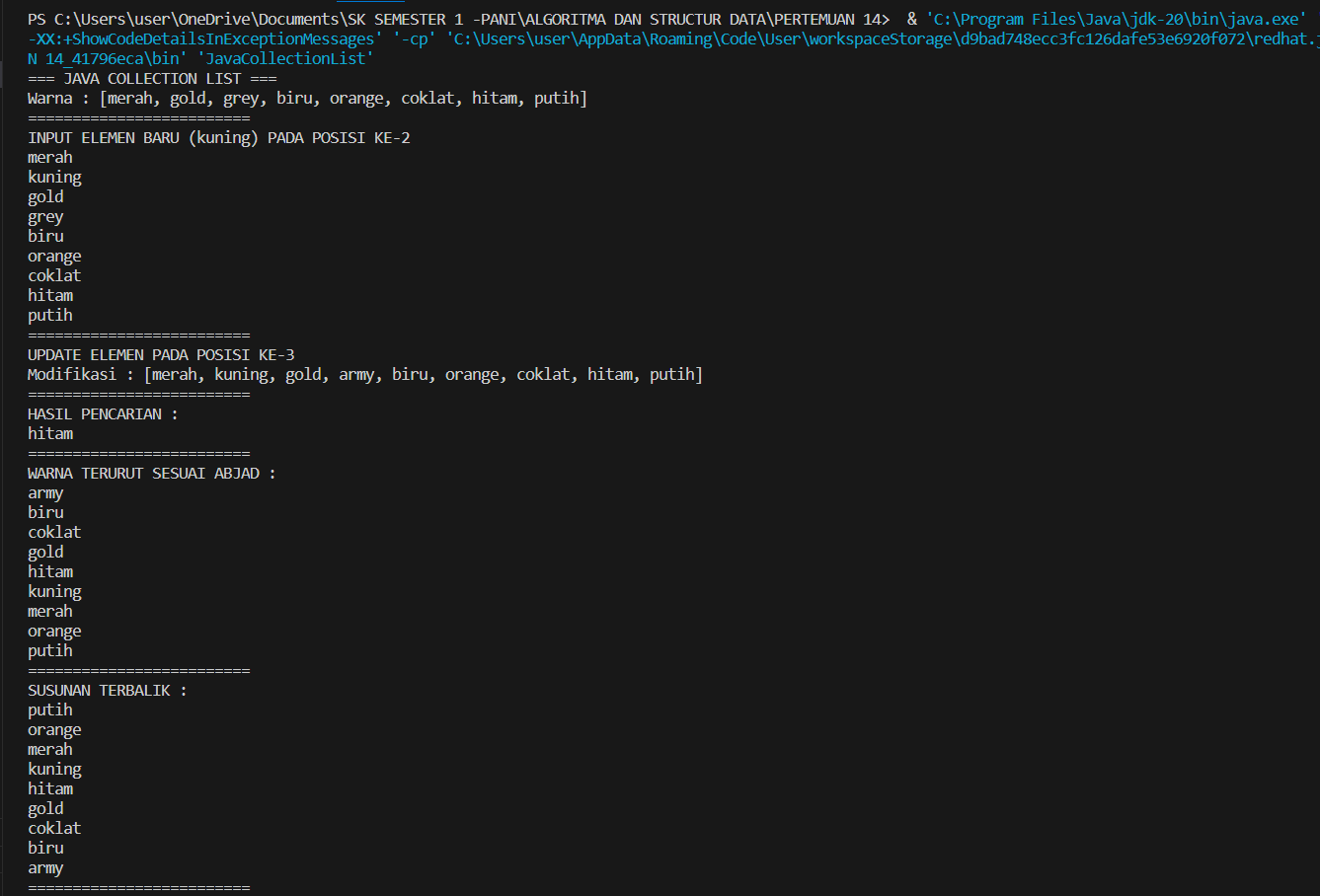
**ALGORITMA DAN STRUKTUR DATA**

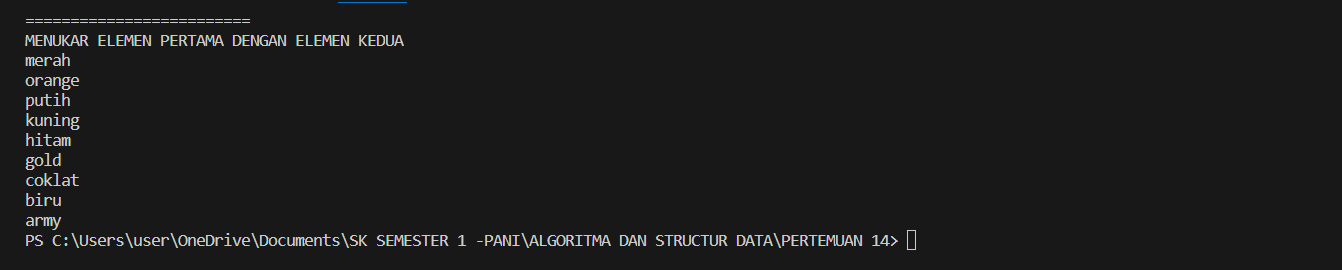
**JAVA COLLECTION**

1. **Latihan ArrayList**

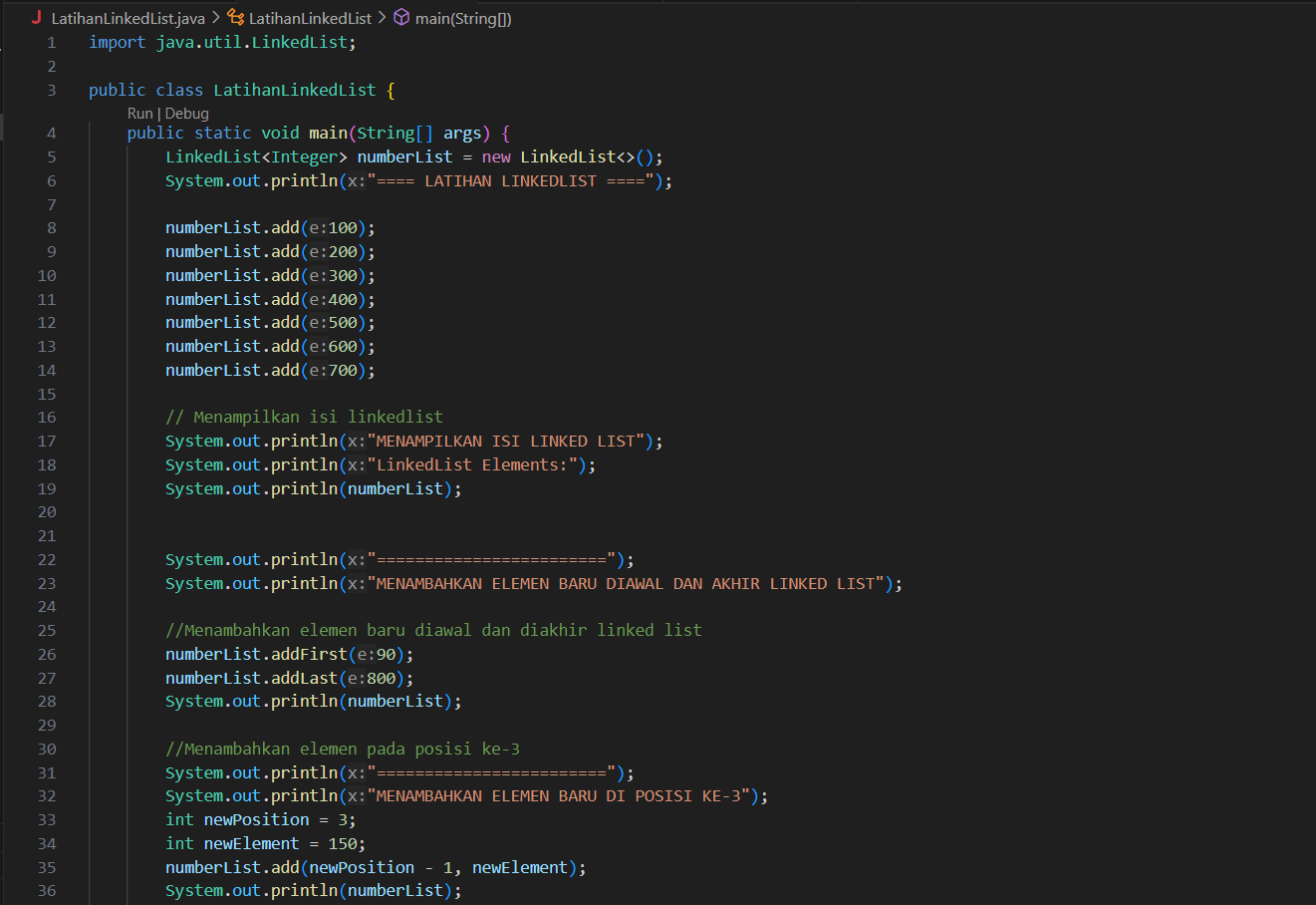
****

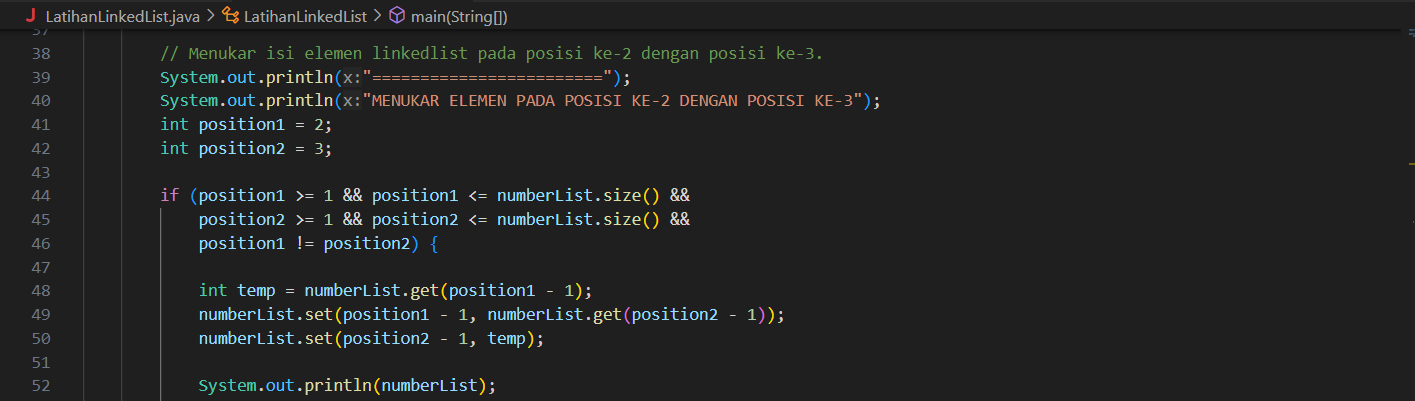
****

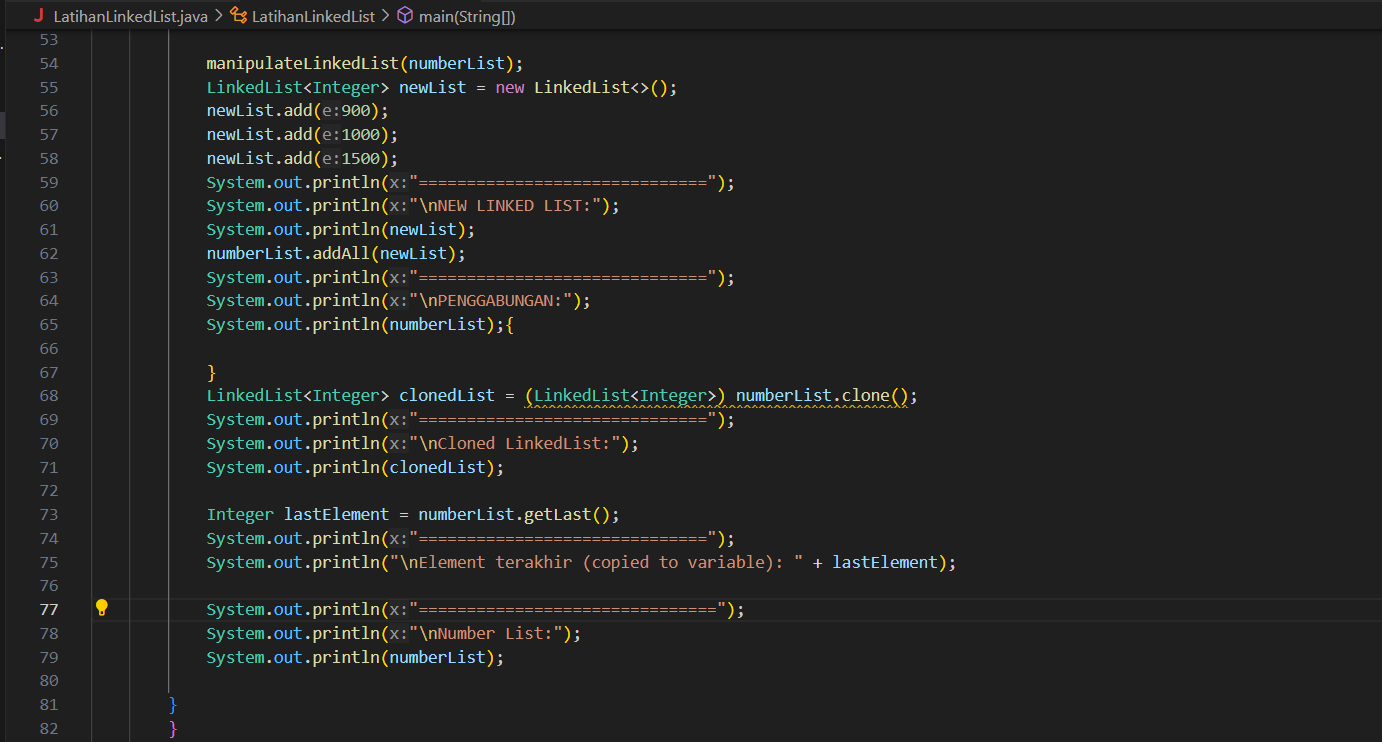
****

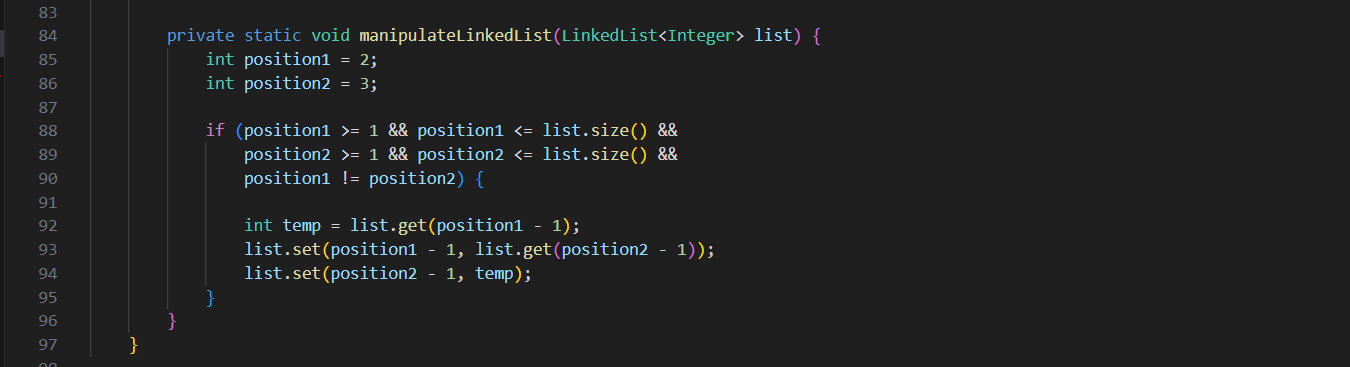
****

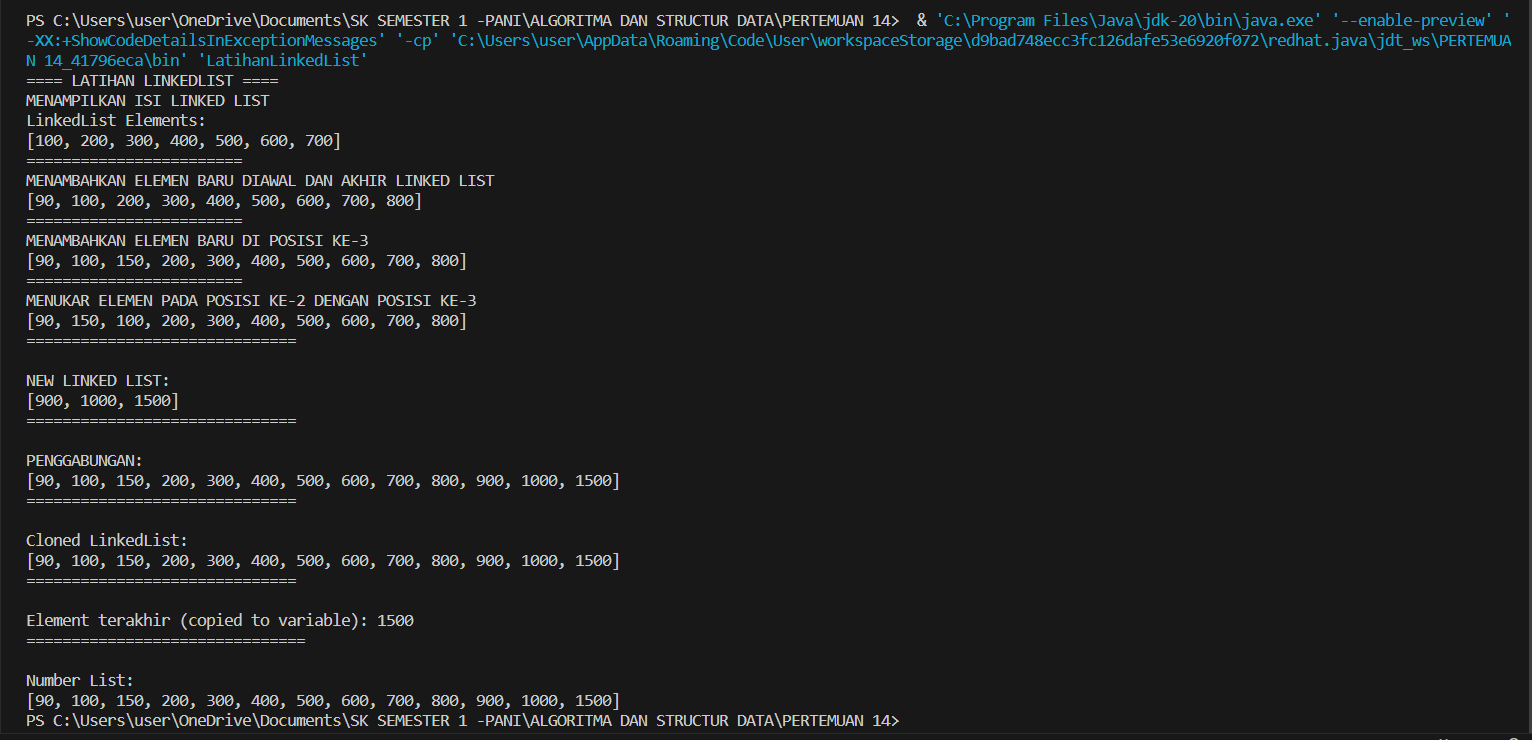
1. **Linkend List**

****

****

****

****

****

**Nama : Pani Permatasari**

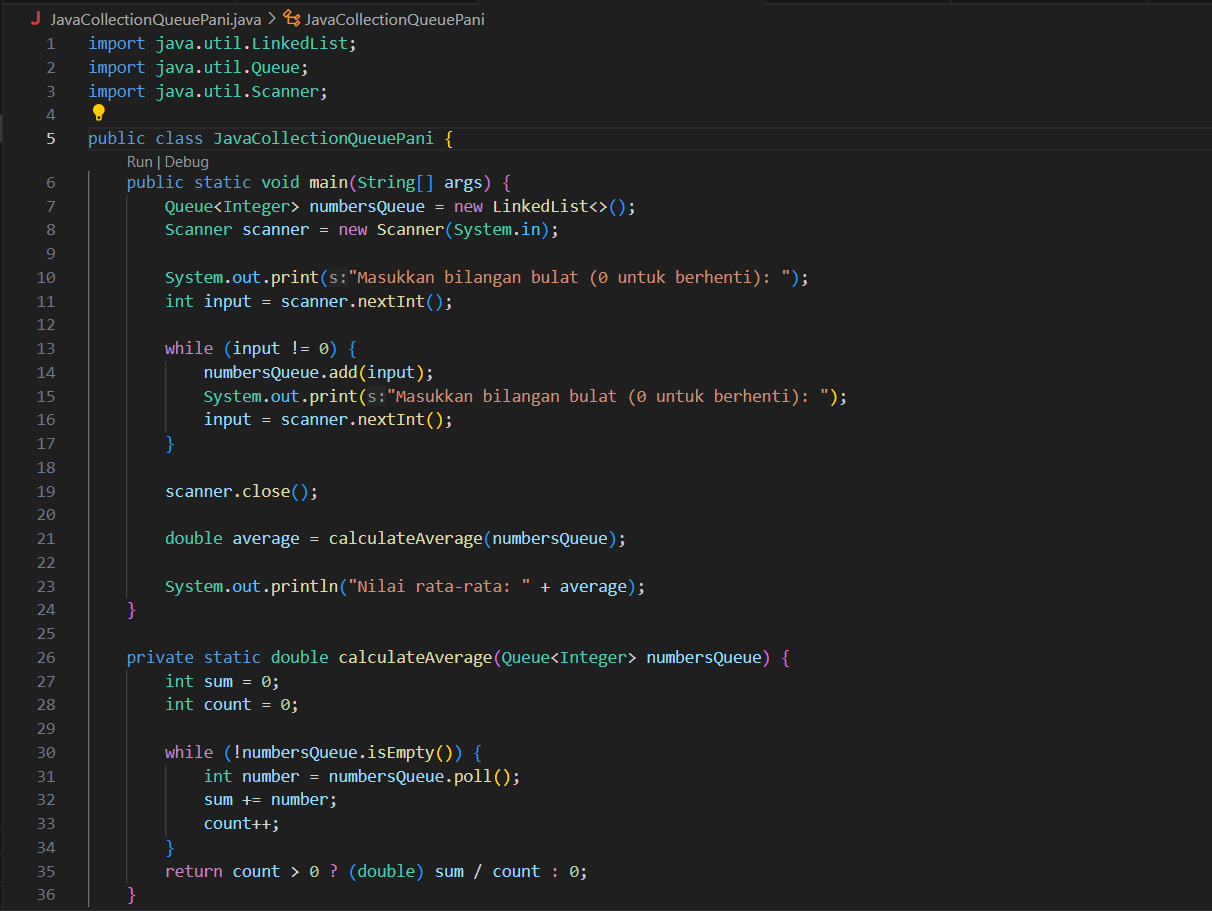
**Nim : 09011182328088**

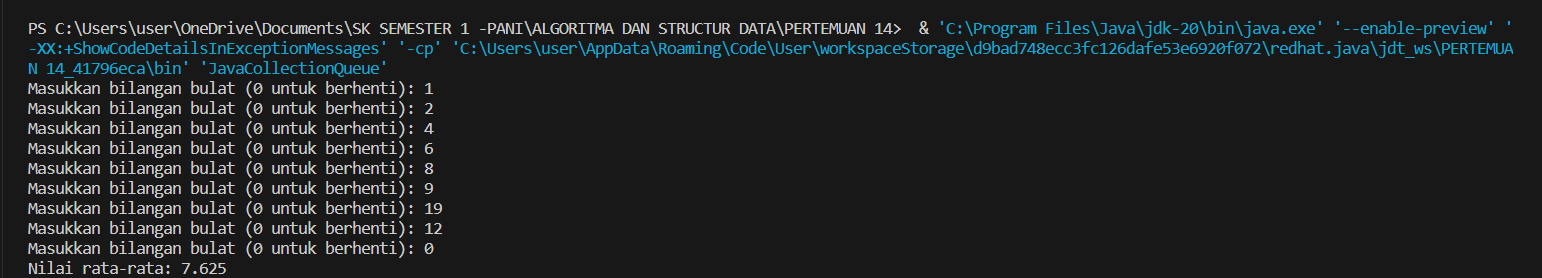
**Kelas : SK1A**

**ALGORITMA DAN STRUKTUR DATA**

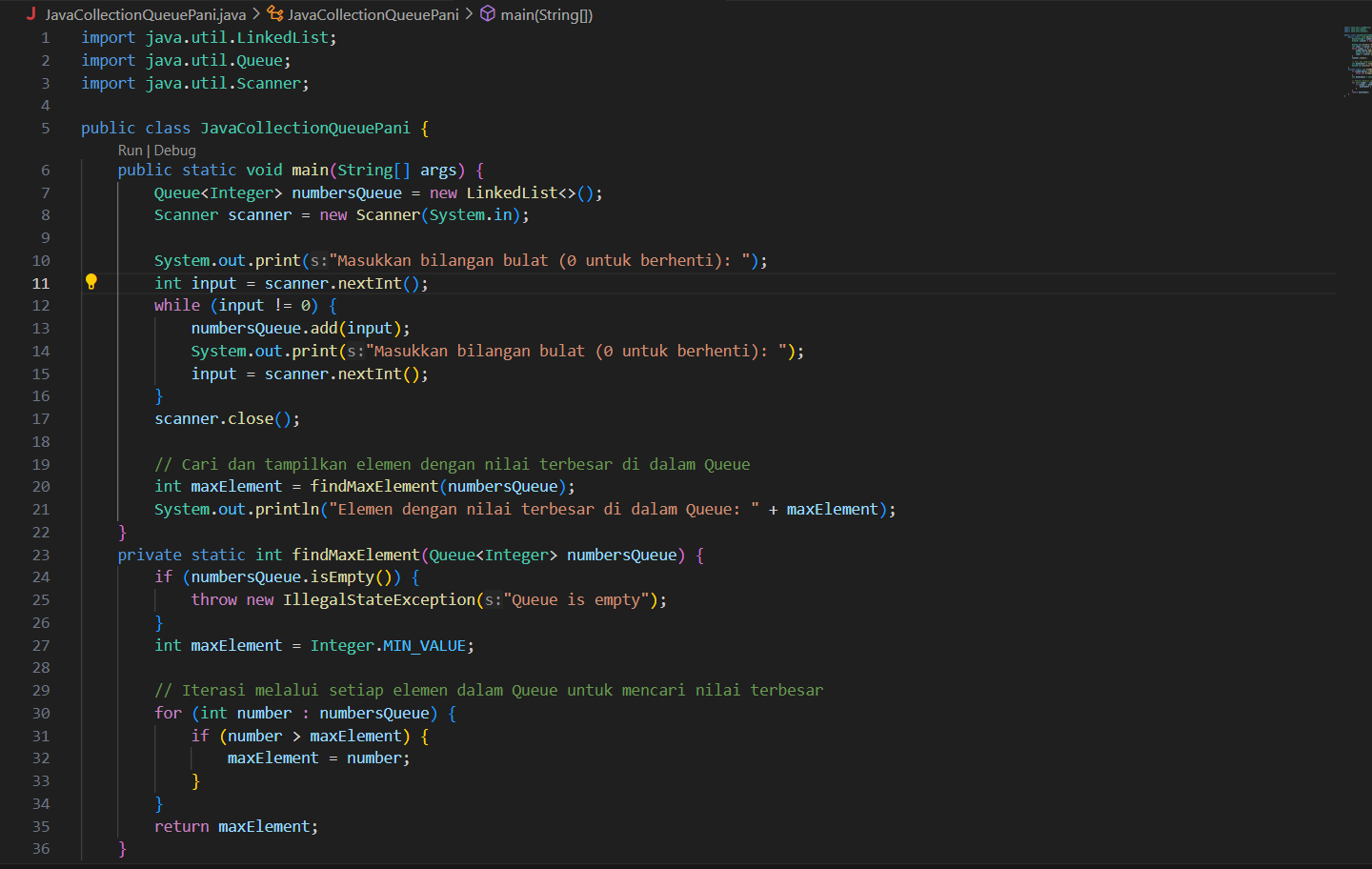
**JAVA COLLECTION QUEUE**

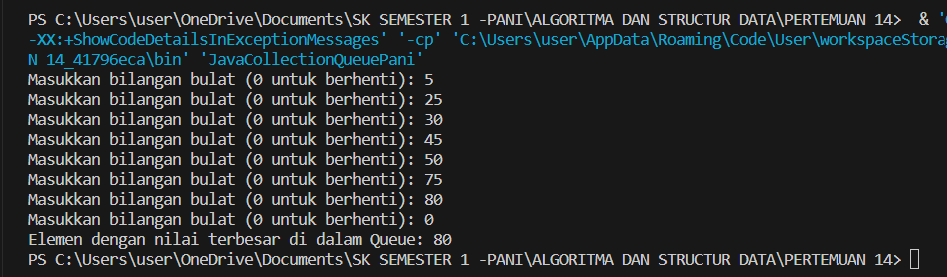
1. menghitung dan menampilkan nilai rata-rata yang tersimpan dalam queue**.**

****

****

1. mencari dan menampilkan elemen dengan nilai terbesar di dalam queue.





**Nama : Pani Permatasari**

**Nim : 09011182328088**

**Kelas : SK1A**

**ALGORITMA DAN STRUKTUR DATA**

**Pengurutan (Sorting)**

1. **Insertion Sort:**
   * Algoritma ini mengurutkan elemen satu per satu.
   * Dimulai dari elemen kedua, setiap elemen dibandingkan dengan elemen-elemen sebelumnya dan ditempatkan pada posisi yang tepat.
   * Kinerjanya bagus untuk jumlah data kecil atau data yang sudah hampir terurut.
2. **Selection Sort:**
   * Algoritma ini mencari elemen minimum dari array dan menukarnya dengan elemen pertama.
   * Kemudian, mencari elemen minimum dari array yang tersisa dan menukarnya dengan elemen kedua, dan seterusnya.
   * Algoritma ini sederhana tetapi memiliki performa yang kurang baik untuk jumlah data yang besar.
3. **Bubble Sort:**
   * Algoritma ini membandingkan dan menukarkan elemen-elemen berdekatan yang tidak terurut satu per satu.
   * Proses ini diulang hingga tidak ada lagi pertukaran yang diperlukan.
   * Sederhana tetapi tidak efisien untuk jumlah data besar.
4. **Heap Sort:**
   * Algoritma ini menggunakan struktur data heap (biasanya pohon biner) untuk mengurutkan data.
   * Pertama, data diatur sebagai heap. Kemudian, elemen terbesar (atau terkecil, tergantung pada implementasi) diambil dan ditukar dengan elemen terakhir.
   * Proses ini diulang untuk menghasilkan array yang terurut.
5. **Merge Sort:**
   * Algoritma ini menggunakan pendekatan "divide and conquer".
   * Array dibagi menjadi dua bagian, kedua bagian tersebut diurutkan secara terpisah, dan kemudian digabungkan kembali.
   * Proses ini diulang hingga array terurut sepenuhnya.
6. **Quick Sort:**
   * Algoritma ini juga menggunakan pendekatan "divide and conquer".
   * Memilih elemen tertentu sebagai "pivot", kemudian mengatur elemen-elemen yang lebih kecil daripada pivot ke satu sisi dan elemen-elemen yang lebih besar ke sisi lain.
   * Proses ini diulang untuk setiap bagian hingga seluruh array terurut.

**Bubble Sort Pseudocode:**

function bubbleSort(arr)

n = length(arr)

for i from 0 to n-1

for j from 0 to n-i-1

if arr[j] > arr[j+1]

swap(arr[j], arr[j+1])

return arr

### Quick Sort Pseudocode:

### function quickSort(arr, low, high)

### if low < high

### pivotIndex = partition(arr, low, high)

### quickSort(arr, low, pivotIndex - 1)

### quickSort(arr, pivotIndex + 1, high)

### function partition(arr, low, high)

### pivot = arr[high]

### i = low - 1

### for j from low to high-1

### if arr[j] <= pivot

### i = i + 1

### swap(arr[i], arr[j])

### swap(arr[i+1], arr[high])

### return i + 1

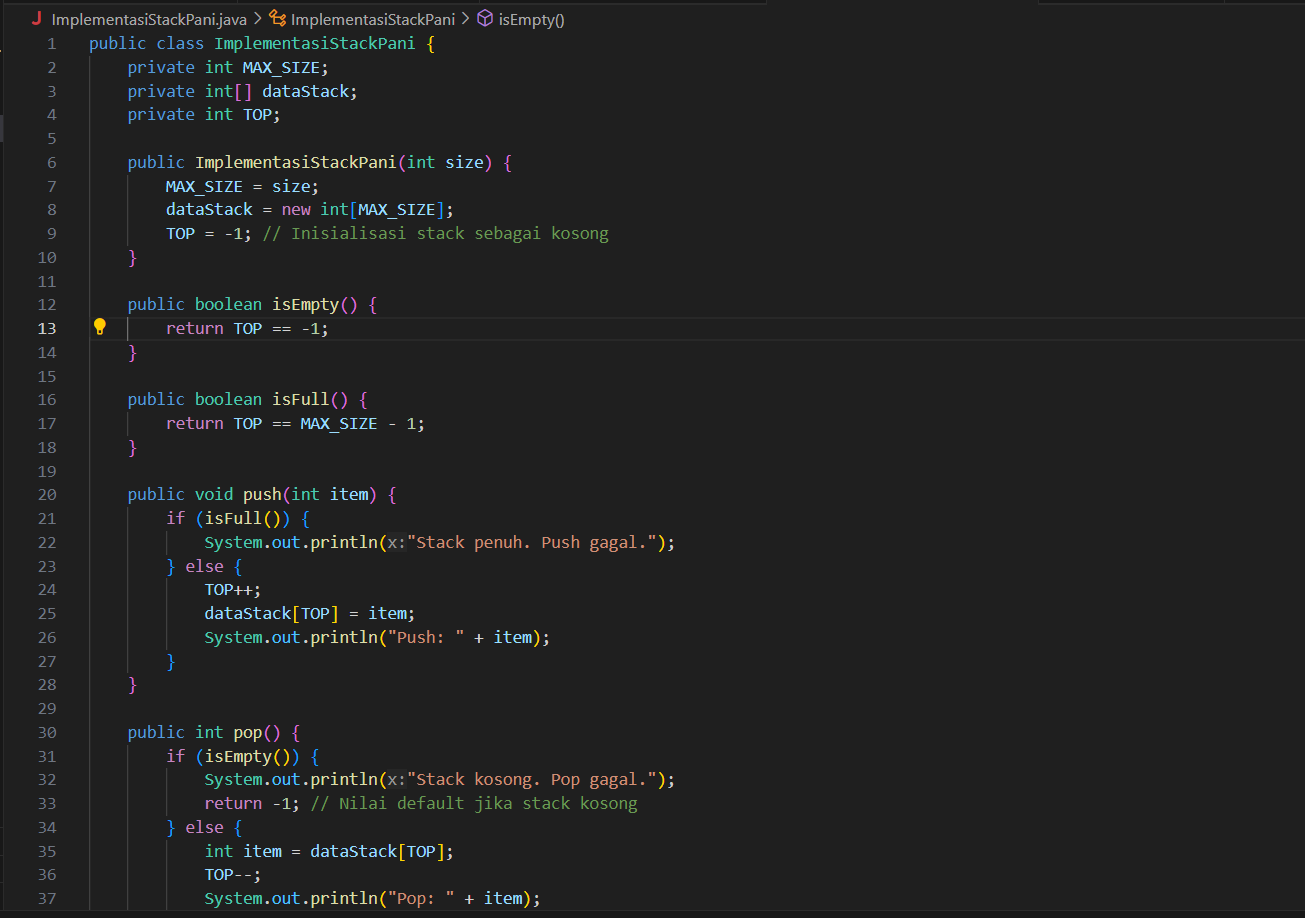
**Nama : Pani Permatasari**

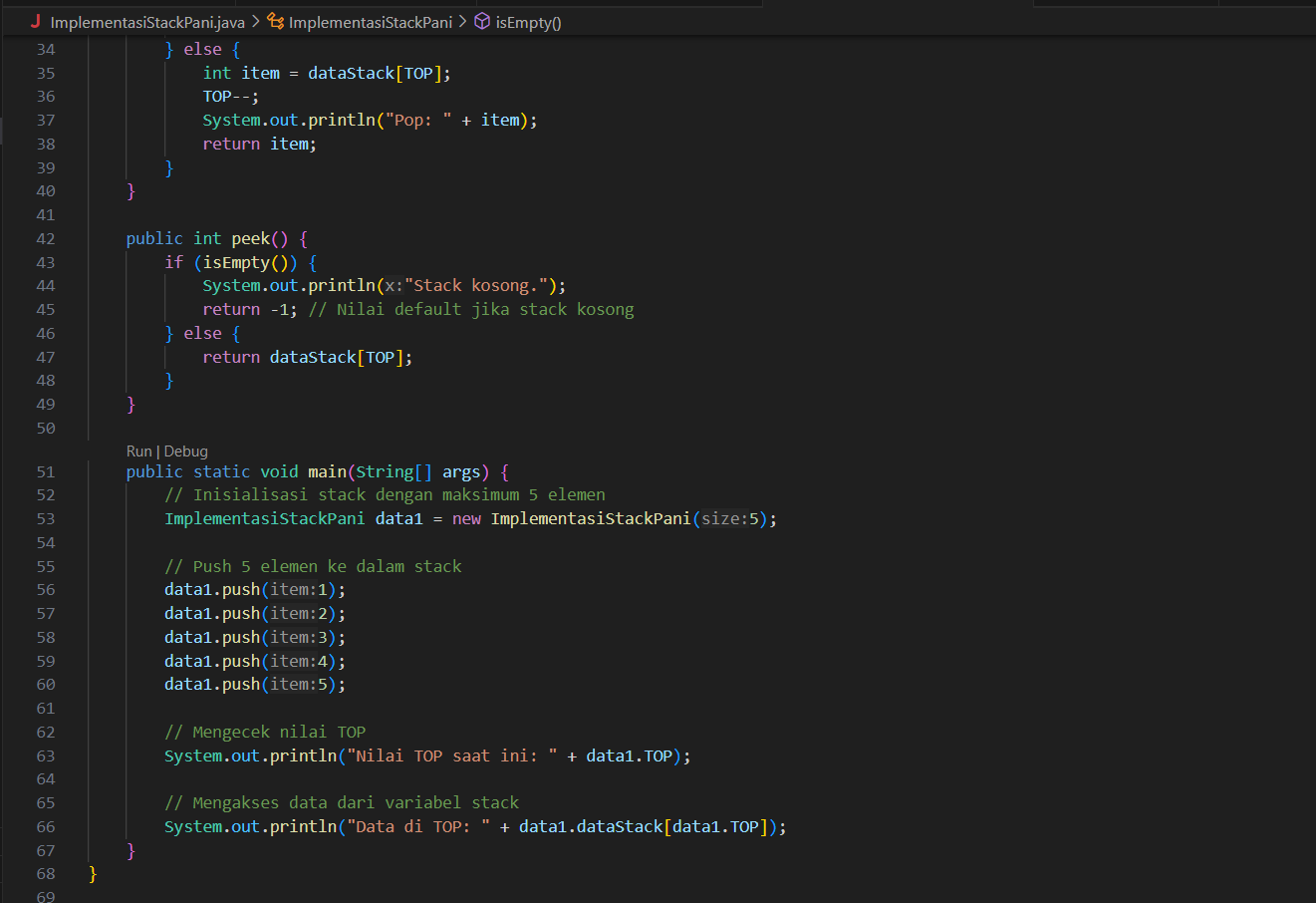
**Nim : 09011182328088**

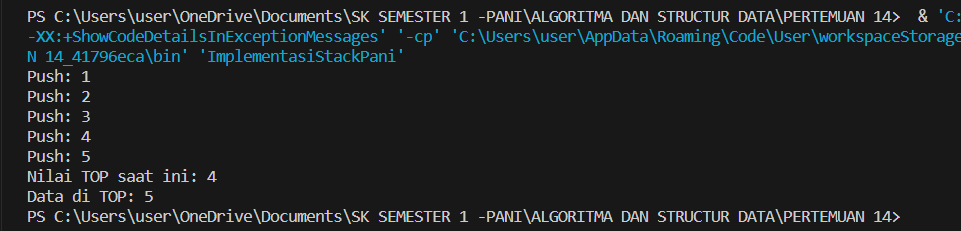
**Kelas : SK1A**

**ALGORITMA DAN STRUKTUR DATA**

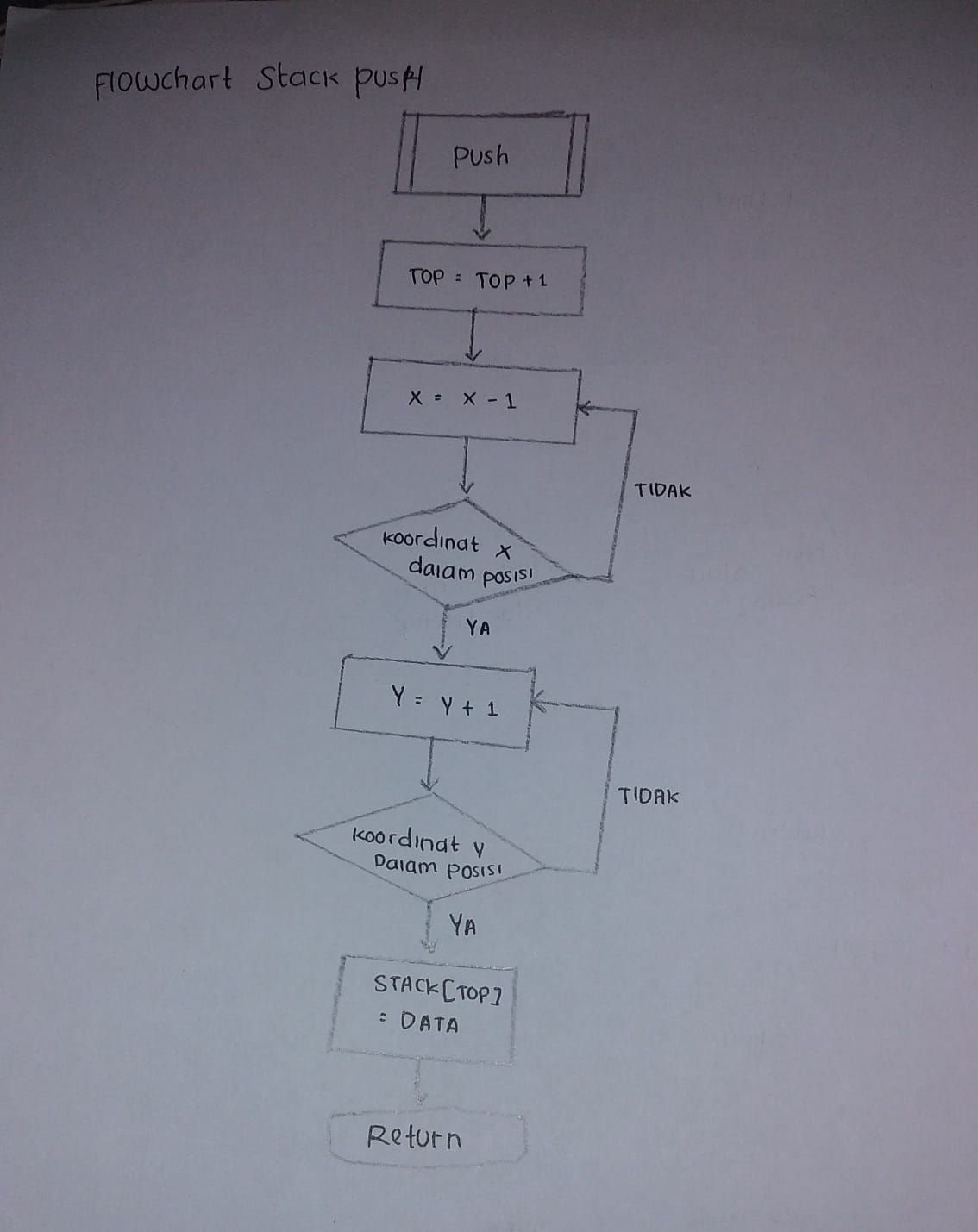
**IMPLEMENTASI STACK**

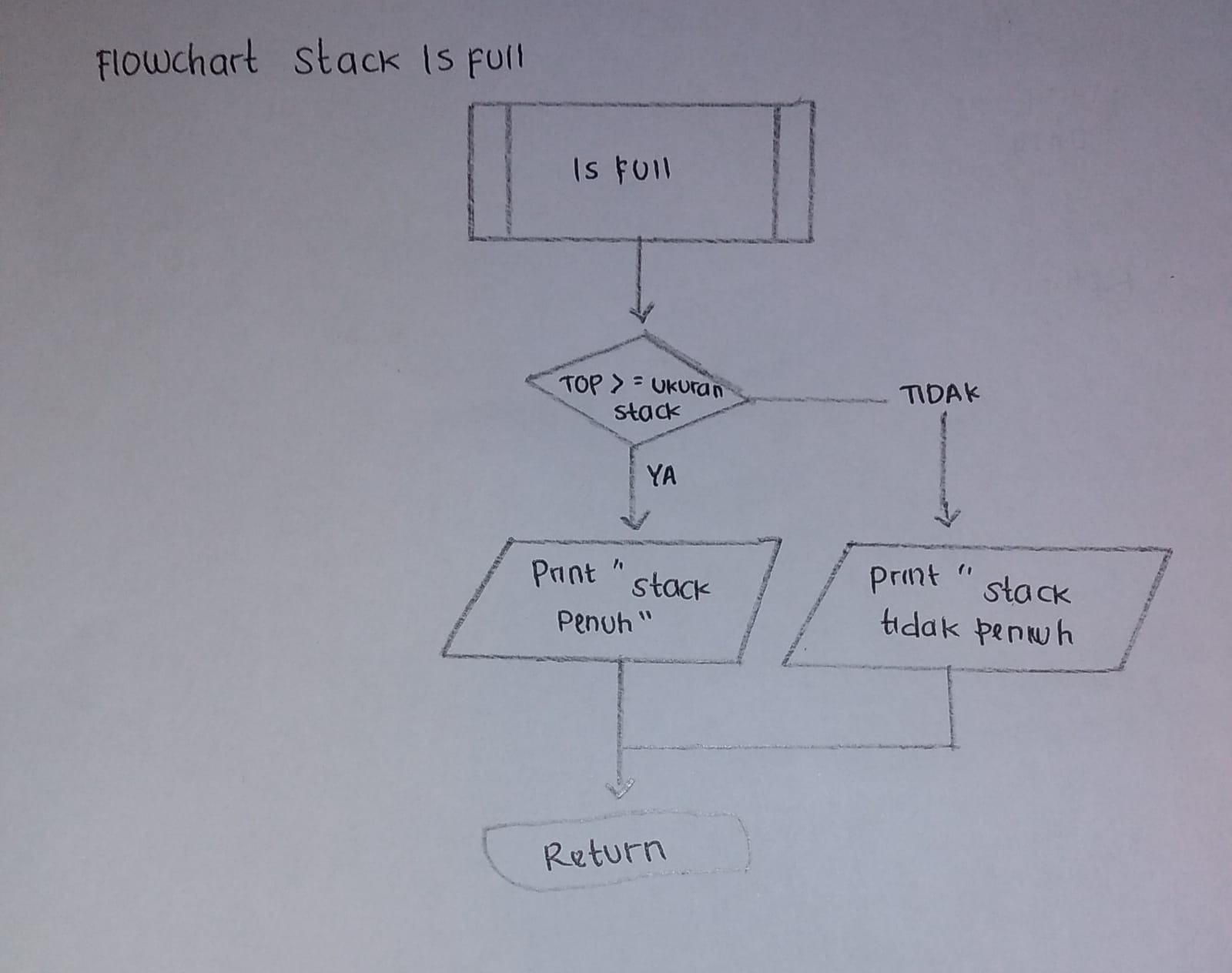
****

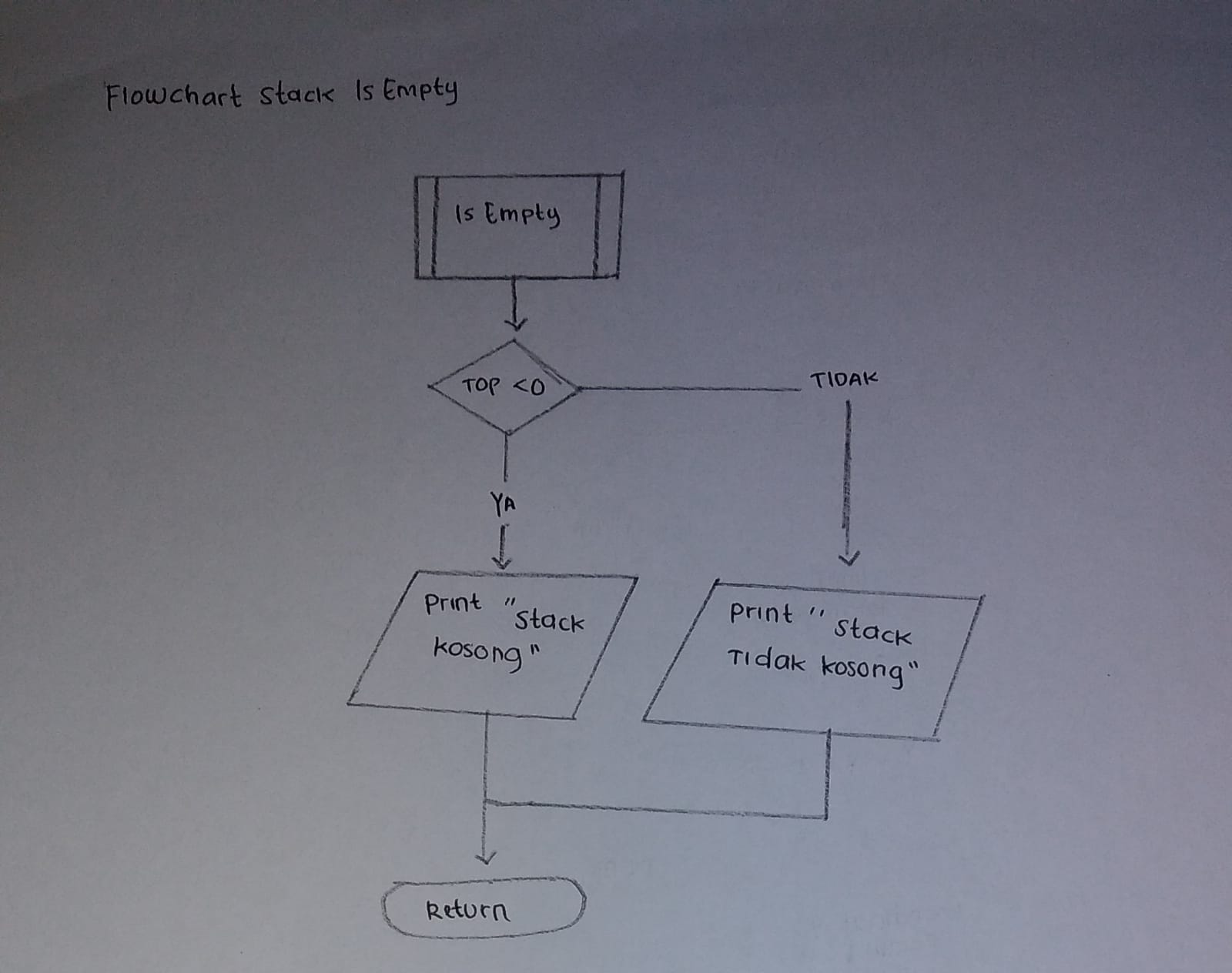
****

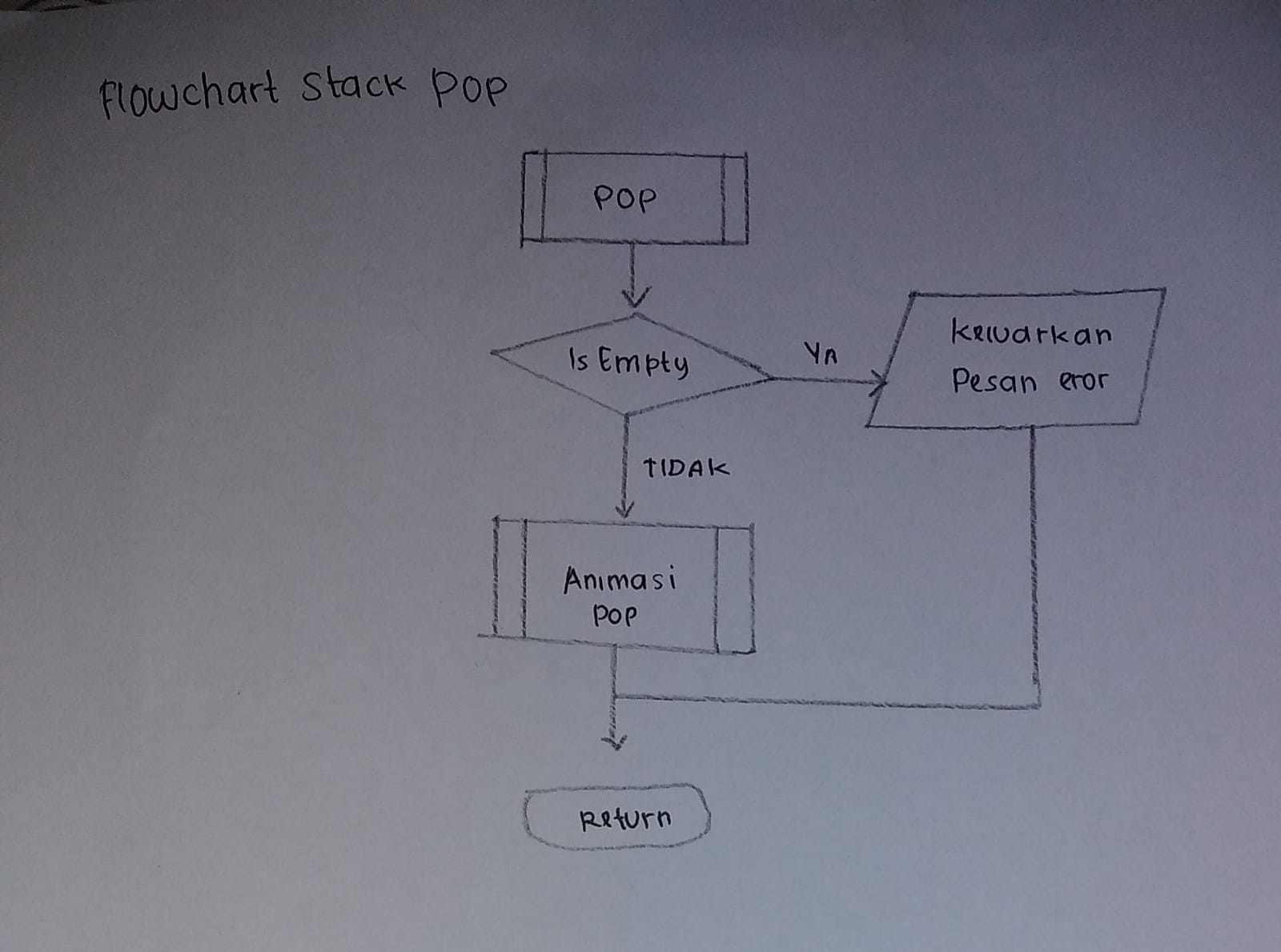
****

**Flowchart fungsi fungsi pengelolaan stack**

****

****

****

****