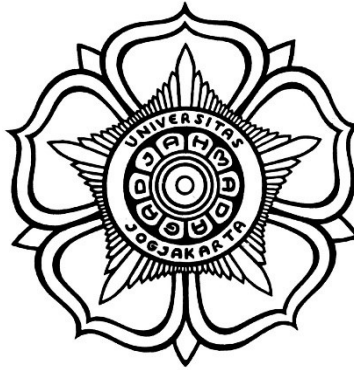


LAPORAN PRAKTIKUM  
PEMOGRAMAN KOMPUTER  
ARRAY TUNGGAL



Disusun Oleh:

Nama	: Noni Cindy Klaudia Matatar
NIM	: 24/545671/SV/25729
Kelas	: RI1B1
Dosen Pegampu	: Yuris Mulya Saputra, S.T.,M.Sc.,Ph.

PROGRAM STUDI D-IV TEKNOLOGI REKAYASA INTERNET  
DEPARTEMEN TEKNIK ELEKTRO DAN INFORMATIKA  
SEKOLAH VOKASI  
UNIVERSITAS GADJAH MADA  
YOGYAKARTA  
2024

## DAFTAR ISI

<b>DAFTAR ISI</b> .....	<b>i</b>
<b>DAFTAR GAMBAR</b> .....	<b>ii</b>
<b>BAB I PENDAHULUAN</b> .....	<b>1</b>
<b>1.1 Latar Belakang</b> .....	<b>1</b>
<b>1.2 Rumusan Masalah</b> .....	<b>1</b>
<b>1.3 Tujuan</b> .....	<b>1</b>
<b>BAB II PEMBAHASAN</b> .....	<b>2</b>
<b>2.1 Source Code 1</b> .....	<b>3</b>
<b>2.2 Source Code 2</b> .....	<b>4</b>
<b>2.3 Source Code 3</b> .....	<b>5</b>
<b>2.4 Sorce Code 4</b> .....	<b>6</b>
<b>2.5 Source Code 5</b> .....	<b>7</b>
<b>2.6 Source Code 6</b> .....	<b>8</b>
<b>BAB III PENUTUP</b> .....	<b>9</b>
<b>3.1 Kesimpulan</b> .....	<b>10</b>
<b>DAFTAR PUSTAKA</b> .....	<b>11</b>

## DAFTAR GAMBAR

<b>GAMBAR 2.1 SOURCE CODE 1 .....</b>	<b>5</b>
<b>GAMBAR 2.2 SOURCE CODE 2 .....</b>	<b>6</b>
<b>GAMBAR 2.3 SOURCE CODE 3 .....</b>	<b>7</b>
<b>GAMBAR 2.4 SOURCE CODE 4 .....</b>	<b>8</b>
<b>GAMBAR 2.5 SOURCE CODE 5 .....</b>	<b>9</b>
<b>GAMBAR 2.6 SOURCE CODE 6 .....</b>	<b>10</b>

# **BAB I**

## **PENDAHULUAN**

### **1.1 Latar Belakang**

Array Tunggal merupakan salah satu struktur dasar yang sering digunakan dalam pemograman. Array berfungsi untuk menyimpan Kumpulan data dengan tipe yang sama dalam satu wadah yang terorganisir secara berurutan. Dengan adanya array, programmer dapat mengelola dan mengakses data secara efisien menggunakan indeks, sehingga tidak perlu membuat banyak variabel untuk setiap nilai yang disimpan. Struktur ini sangat berguna ketika kita ingin menyimpan sejumlah besar data yang berhubungan seperti daftar angka, dan nama.

Pada dasarnya, array tunggal menyediakan cara yang mudah dan terstruktur untuk mengelola data dalam jumlah besar, sehingga proses pencairan, pengurutan, atau modifikasi data dapat dilakukan dengan cepat. Dalam banyak Bahasa pemograman, array digunakan untuk berbagai keperluan seperti menyimpan hasil perhitungan, mengelola daftar objek, atau melakukan literasi data dalam algoritma. Array juga menjadi landasan untuk memahami struktur data yang lebih kompleks seperti array multidimensi, linked list, atau matriks.

### **1.2 Rumusan Masalah**

1. Bagaimana cara menyimpan sejumlah besar data yang memiliki tipe yang sama dalam satu variabel agar lebih efisien dibandingkan menggunakan banyak variabel secara terpisah?
2. Bagaimana cara mengakses dan memanipulasi elemen-elemen data tersebut dengan cepat menggunakan indeks?

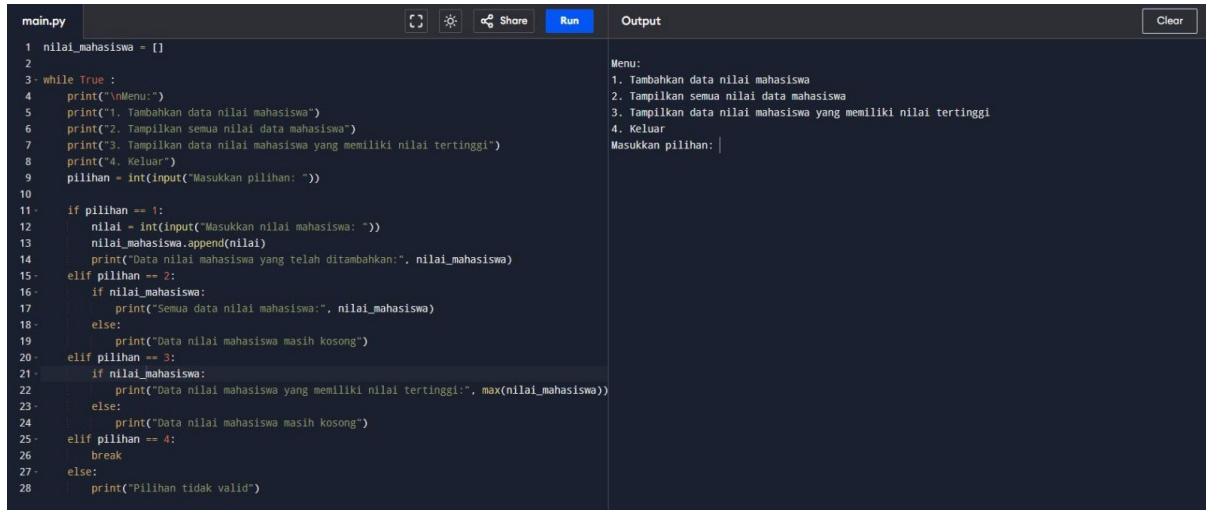
### **1.3 Tujuan**

1. Array Tunggal menyimpan tipe data yang sama dalam satu variabel, memungkinkan akses terorganisir dan efisien menggunakan indeks, sehingga menghindari kebutuhan mendeklarasikan banyak variabel secara terpisah.
2. Elemen dalam Array dapat diakses dan dimanipulasi dengan cepat menggunakan indeks yang menunjukkan posisi elemen. Indeks biasanya dimulai dari 0. Untuk mengakses elemen, cukup gunakan sintaks `array[n]`, `n` A array `[n]` = nilai\_baru.

## BAB II

### PEMBAHASAN

#### 2.1 Source Code 1



```
1 nilai_mahasiswa = []
2
3 while True :
4     print("\nMenu:")
5     print("1. Tambahkan data nilai mahasiswa")
6     print("2. Tampilkan semua nilai data mahasiswa")
7     print("3. Tampilkan data nilai mahasiswa yang memiliki nilai tertinggi")
8     print("4. Keluar")
9     pilihan = int(input("Masukkan pilihan: "))
10
11     if pilihan == 1:
12         nilai = int(input("Masukkan nilai mahasiswa: "))
13         nilai_mahasiswa.append(nilai)
14         print("Data nilai mahasiswa yang telah ditambahkan:", nilai_mahasiswa)
15     elif pilihan == 2:
16         if nilai_mahasiswa:
17             print("Semua data nilai mahasiswa:", nilai_mahasiswa)
18         else:
19             print("Data nilai mahasiswa masih kosong")
20     elif pilihan == 3:
21         if nilai_mahasiswa:
22             print("Data nilai mahasiswa yang memiliki nilai tertinggi:", max(nilai_mahasiswa))
23         else:
24             print("Data nilai mahasiswa masih kosong")
25     elif pilihan == 4:
26         break
27     else:
28         print("Pilihan tidak valid")
```

Output

Menu:

1. Tambahkan data nilai mahasiswa

2. Tampilkan semua nilai data mahasiswa

3. Tampilkan data nilai mahasiswa yang memiliki nilai tertinggi

4. Keluar

Masukkan pilihan: |

Gambar 2.1 Source Code 1

Analisis:

#### 1. Daftar Inisialisasi:

Sebuah daftar kosong bernama `nilai_mahasiswa` dibuat untuk menyimpan nilai-nilai siswa.

#### 2. Loop Utama:

Program memasuki sebuah loop `while True` yang akan terus berjalan hingga pengguna memilih opsi untuk keluar.

Di dalam loop, program menampilkan menu pilihan kepada pengguna.

#### 3. Pilihan Menu:

Pengguna diminta untuk memilih salah satu opsi dari menu yang tersedia.

Pilihan pengguna kemudian diproses menggunakan struktur `if-elif-else`.

#### 4. Pengolahan Pilihan:

**Pilihan 1:** Menambahkan data nilai siswa baru.

- Pengguna diminta untuk memasukkan nilai siswa baru.
- Nilai baru tersebut kemudian ditambahkan ke dalam daftar `nilai_mahasiswa` menggunakan metode `append()`.

**Pilihan 2:** Menampilkan semua nilai siswa.

Jika daftar nilai\_mahasiswatidak kosong, maka semua nilai akan dicetak. Jika kosong, maka akan ditampilkan pesan bahwa data masih kosong.

**Pilihan 3:** Menampilkan nilai siswa tertinggi.

Jika daftar nilai\_mahasiswatidak kosong, maka nilai tertinggi akan dicari menggunakan fungsi max()dan dicetak. Jika kosong, maka akan ditampilkan pesan bahwa data masih kosong.

**Pilihan 4:** Keluar dari program.

Loop whiledihentikan, sehingga program berakhir.

## 2.2 Source Code 2



```
main.py  Run  Output
1 def hitung_rata_rata(angka):
2     """Fungsi untuk menghitung rata-rata dari sebuah list angka."""
3     jumlah = sum(angka)
4     rata_rata = jumlah / len(angka)
5     return rata_rata
6
7 angka = []
8 for i in range(5):
9     nilai = float(input(f'Masukkan angka ke-{i+1}: '))
10    angka.append(nilai)
11    print(angka)
12
13 pilihan = input("Ingin melihat hasil jumlah atau rata-rata: ")
14
15 if pilihan == "jumlah":
16     jumlah = sum(angka)
17     print(f'Jumlah dari semua angka adalah: {jumlah}')
18 elif pilihan == "rata-rata":
19     rata_rata = hitung_rata_rata(angka)
20     print(f'Rata-rata dari nilai datanya adalah: {rata_rata:.2f}')
21 else:
22     print("Pilihan tidak valid.")

Masukkan angka ke-1: 2.5
[2.5]
Masukkan angka ke-2: 2.4
[2.5, 2.4]
Masukkan angka ke-3: 1
[2.5, 2.4, 1.0]
Masukkan angka ke-4: 4
[2.5, 2.4, 1.0, 4.0]
Masukkan angka ke-5: 5
[2.5, 2.4, 1.0, 4.0, 5.0]
Ingin melihat hasil jumlah atau rata-rata: rata-rata
Rata-rata dari nilai datanya adalah: 2.98

=== Code Execution Successful ===
```

Gambar 2.2 Source Code 2

Analisis:

### 1. Fungsi hitung\_rata\_rata:

- Fungsi ini menerima daftar angka sebagai input.
- Menghitung jumlah semua angka dalam daftar menggunakan fungsi sum().
- Menghitung rata-rata dengan membagi jumlah dengan banyaknya angka.
- Mengembalikan nilai rata-rata.

### 2. Membuat Daftar Kosong:

- List angkadibuat untuk menyimpan angka-angka yang dimasukkan oleh pengguna.

### 3. Input Angka:

- Menggunakan perulangan for untuk meminta pengguna memasukkan 5 angka.
- Setiap angka yang dimasukkan diubah menjadi tipe float dan ditambahkan ke dalam list angka.
- Setelah setiap angka dimasukkan, list angka dicetak untuk memberikan umpan balik kepada pengguna.

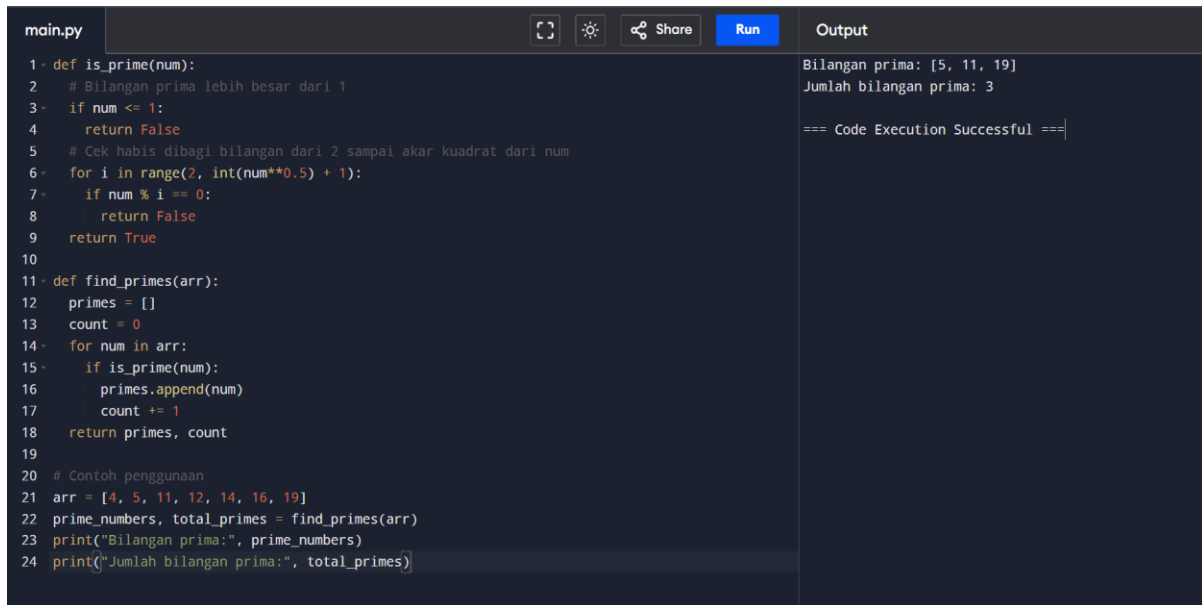
### 4. Pilihan Pengguna:

- Meminta pengguna memilih apakah ingin menghitung jumlah atau rata-rata.
- Menggunakan struktur if-elif-else untuk memproses pilihan pengguna.

### 5. Perhitungan dan Output:

- Jika pengguna memilih "jumlah", maka semua angka dalam daftar dijumlahkan menggunakan fungsi sum() dan hasilnya dicetak.
- Jika pengguna memilih "rata-rata", maka fungsi hitung\_rata\_rata dipanggil untuk menghitung rata-rata dan hasilnya dicetak dengan format dua angka di belakang koma.
- Jika pilihan pengguna tidak valid, maka pesan kesalahan akan ditampilkan.

## 2.3 Source Code 3



```
main.py  [Icons] [Run] [Output]

1 def is_prime(num):
2     # Bilangan prima lebih besar dari 1
3     if num <= 1:
4         return False
5     # Cek habis dibagi bilangan dari 2 sampai akar kuadrat dari num
6     for i in range(2, int(num**0.5) + 1):
7         if num % i == 0:
8             return False
9     return True
10
11 def find_primes(arr):
12     primes = []
13     count = 0
14     for num in arr:
15         if is_prime(num):
16             primes.append(num)
17             count += 1
18     return primes, count
19
20 # Contoh penggunaan
21 arr = [4, 5, 11, 12, 14, 16, 19]
22 prime_numbers, total_primes = find_primes(arr)
23 print("Bilangan prima:", prime_numbers)
24 print("Jumlah bilangan prima:", total_primes)
```

Output

```
Bilangan prima: [5, 11, 19]
Jumlah bilangan prima: 3

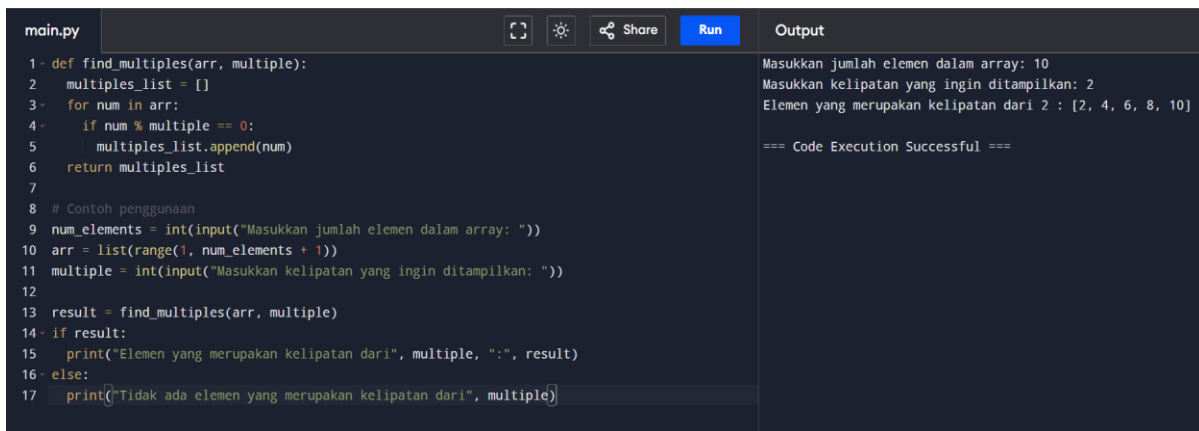
=== Code Execution Successful ===
```

Gambar 2.3 Source Code 3

Analisis:

1. **Meminta input pengguna:** Kita akan meminta pengguna untuk memasukkan jumlah elemen dan kelipatan yang diinginkan.
2. **Membuat array:** Kita akan membuat array dengan elemen dari 1 hingga jumlah yang dimasukkan.
3. **kelipatan:** Kita akan mengiterasi setiap elemen dalam array dan memeriksa apakah elemen tersebut merupakan kelipatan dari bilangan yang ditentukan.

## 2.4 Source Code 4



The screenshot shows a code editor with a file named 'main.py'. The code defines a function 'find\_multiples' that takes an array and a multiple, then iterates through the array to find multiples. It also includes a main block that prompts the user for the number of elements and the multiple, then calls the function and prints the result. The output panel shows the user inputting 10 for the number of elements and 2 for the multiple, resulting in the array [2, 4, 6, 8, 10].

```
main.py
1- def find_multiples(arr, multiple):
2     multiples_list = []
3     for num in arr:
4         if num % multiple == 0:
5             multiples_list.append(num)
6     return multiples_list
7
8 # Contoh penggunaan
9 num_elements = int(input("Masukkan jumlah elemen dalam array: "))
10 arr = list(range(1, num_elements + 1))
11 multiple = int(input("Masukkan kelipatan yang ingin ditampilkan: "))
12
13 result = find_multiples(arr, multiple)
14 if result:
15     print("Elemen yang merupakan kelipatan dari", multiple, ":", result)
16 else:
17     print("Tidak ada elemen yang merupakan kelipatan dari", multiple)
```

Output

Masukkan jumlah elemen dalam array: 10  
Masukkan kelipatan yang ingin ditampilkan: 2  
Elemen yang merupakan kelipatan dari 2 : [2, 4, 6, 8, 10]  
=== Code Execution Successful ===

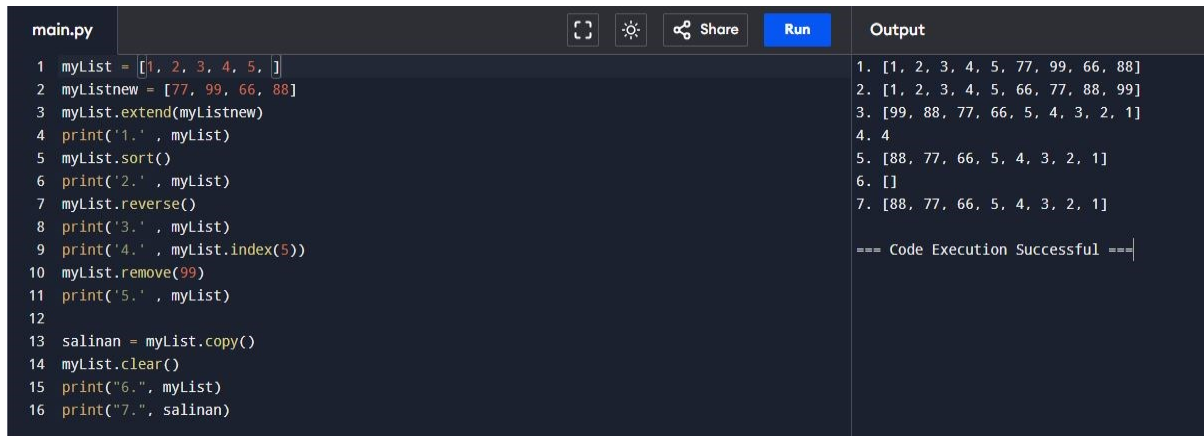
Gambar 2.4 Source Code 4

Analisis:

1. **Fungsi:** Penggunaan fungsi membuat kode lebih modular dan mudah dibaca.
2. **Efisiensi:** Fungsi ini menggunakan optimasi dengan hanya memeriksa pembagi hingga akar kuadrat dari bilangan tersebut.
3. **Fleksibilitas:** Kode pada soal nomor 4 dapat digunakan untuk berbagai kasus dengan mengubah nilai `num_elements` dan `multiple`.
4. **Kemudahan dipahami:** Kode ditulis dengan jelas dan menggunakan komentar untuk menjelaskan setiap bagian.



## 2.5 Source Code 5



The screenshot shows a code editor with a dark theme. The left pane is titled 'main.py' and contains 16 lines of Python code. The right pane is titled 'Output' and shows the execution results of the code. The code performs several list operations: initialization, extension, sorting, reversing, removing an element, creating a copy, clearing, and printing. The output shows the state of the list at each step, with the final state being an empty list after the clear operation.

```
main.py
1 myList = [1, 2, 3, 4, 5, ]
2 myListnew = [77, 99, 66, 88]
3 myList.extend(myListnew)
4 print('1.', myList)
5 myList.sort()
6 print('2.', myList)
7 myList.reverse()
8 print('3.', myList)
9 print('4.', myList.index(5))
10 myList.remove(99)
11 print('5.', myList)
12
13 salinan = myList.copy()
14 myList.clear()
15 print("6.", myList)
16 print("7.", salinan)
```

Output

```
1. [1, 2, 3, 4, 5, 77, 99, 66, 88]
2. [1, 2, 3, 4, 5, 66, 77, 88, 99]
3. [99, 88, 77, 66, 5, 4, 3, 2, 1]
4. 4
5. [88, 77, 66, 5, 4, 3, 2, 1]
6. []
7. [88, 77, 66, 5, 4, 3, 2, 1]

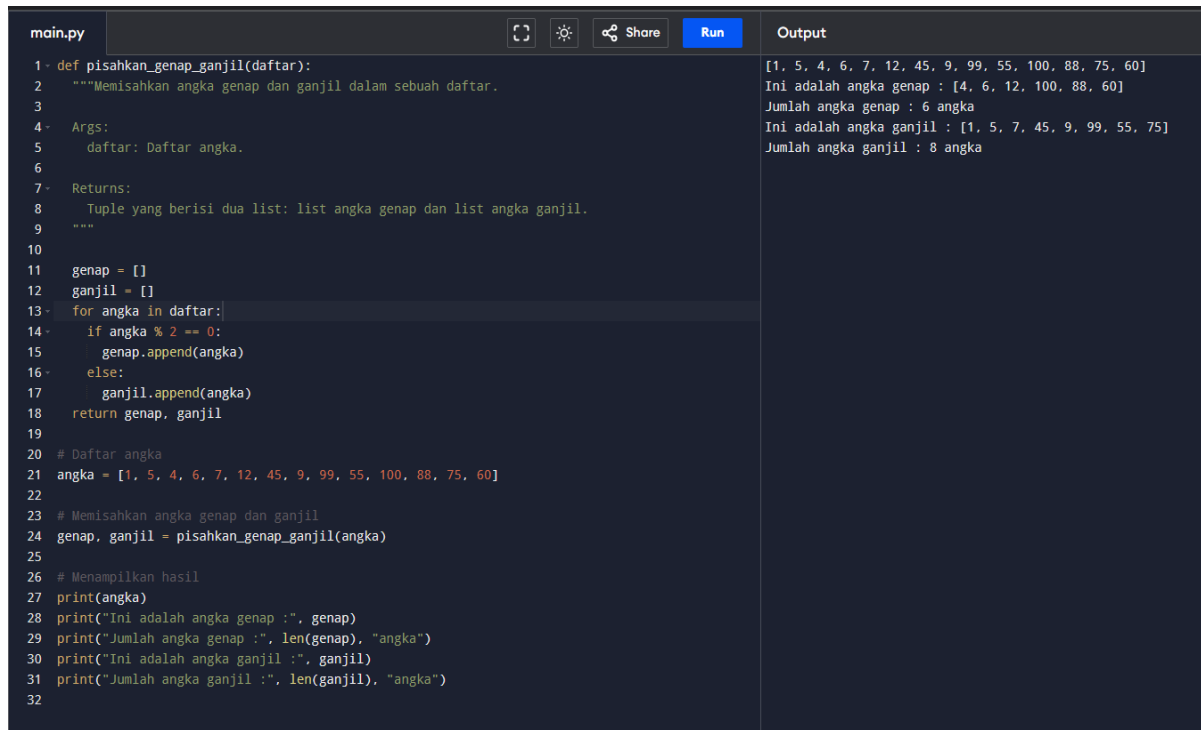
=== Code Execution Successful ===
```

*Gambar 2.5 Source Code*

Analisis:

1. **Pembuatan Daftar:** Dibuat daftar doa, myList dan myListnew.
2. **Penggabungan Daftar:** Daftar myListnew digabungkan ke dalam myList menggunakan metode extend().
3. **Pengurutan:** Daftar myList diurutkan secara ascending (dari terkecil ke terbesar) menggunakan metode sort().
4. **Pembalikan Urutan:** Urutan elemen dalam myList dibalik menggunakan metode reverse().
5. **Penghapusan Elemen:** Elemen dengan indeks 5 dihapus dari myList menggunakan metode remove().
6. **Pembuatan Salinan:** Dibuat dari salinan myList dan disimpan dalam variabel salinan.
7. **Penghapusan Semua Elemen:** Semua elemen dalam myList dihapus menggunakan metode clear().

## 2.6 Source Code 6



The screenshot shows a code editor with a dark theme. The left pane displays the source code for a Python script named `main.py`. The code defines a function `pisahkan_genap_ganjil` that takes a list of numbers and returns a tuple of two lists: even numbers and odd numbers. It then uses this function to process a specific list of numbers and prints the results. The right pane shows the output of the script, which displays the original list, the separated even and odd lists, and their respective counts.

```
main.py
1- def pisahkan_genap_ganjil(daftar):
2-     """Memisahkan angka genap dan ganjil dalam sebuah daftar.
3-
4-     Args:
5-         daftar: Daftar angka.
6-
7-     Returns:
8-         Tuple yang berisi dua list: list angka genap dan list angka ganjil.
9-     """
10-
11-     genap = []
12-     ganjil = []
13-     for angka in daftar:
14-         if angka % 2 == 0:
15-             genap.append(angka)
16-         else:
17-             ganjil.append(angka)
18-     return genap, ganjil
19-
20- # Daftar angka
21- angka = [1, 5, 4, 6, 7, 12, 45, 9, 99, 55, 100, 88, 75, 60]
22-
23- # Memisahkan angka genap dan ganjil
24- genap, ganjil = pisahkan_genap_ganjil(angka)
25-
26- # Menampilkan hasil
27- print(angka)
28- print("Ini adalah angka genap :", genap)
29- print("Jumlah angka genap :", len(genap), "angka")
30- print("Ini adalah angka ganjil :", ganjil)
31- print("Jumlah angka ganjil :", len(ganjil), "angka")
32-
```

Output

```
[1, 5, 4, 6, 7, 12, 45, 9, 99, 55, 100, 88, 75, 60]
Ini adalah angka genap : [4, 6, 12, 100, 88, 60]
Jumlah angka genap : 6 angka
Ini adalah angka ganjil : [1, 5, 7, 45, 9, 99, 55, 75]
Jumlah angka ganjil : 8 angka
```

Gambar 2.6 Source Code

Analisis:

### 1. Fungsi `pisahkan_genap_ganjil`:

- Fungsi ini menerima daftar angka sebagai input.
- Membuat daftar dua kosong, genap dan ganjil.
- Melakukan iterasi pada setiap angka dalam daftar.
- Jika angka habis dibagi 2 (genap), angka tersebut ditambahkan ke daftar genap.
- Jika angka tidak habis dibagi 2 (ganjil), angka tersebut ditambahkan ke daftar ganjil.
- Mengembalikan tuple yang berisi kedua list tersebut.

### 2. Fungsi:

- Daftar angka yang diberikan dimasukkan sebagai argumen ke fungsi `pisahkan_genap_ganjil`.
- Hasil pengembalian fungsi disimpan dalam variabel `genap` dan `ganjil`.

### 3. Menampilkan Hasil:

- Mencetak daftar angka asli, daftar angka genap, jumlah angka genap, daftar angka ganjil, dan jumlah angka ganjil.

## **BAB III**

### **PENUTUP**

#### **3.1 Kesimpulan**

Array tunggal adalah struktur data dasar dalam pemrograman yang digunakan untuk menyimpan kumpulan data sejenis dalam satu variabel. Bayangkan seperti sebuah rak buku, di mana setiap buku (data) memiliki nomor urut (indeks) yang unik. Array memungkinkan kita mengakses, mengubah, atau menambahkan data dengan cepat berdasarkan indeksnya. Keuntungan utama menggunakan array adalah efisiensi dalam mengelola data yang berkaitan, seperti nilai ujian siswa, daftar nama, atau harga produk. Namun, perlu diingat bahwa ukuran array umumnya tetap setelah dideklarasikan. Array tunggal juga alat yang sangat berguna dalam pemrograman untuk mengorganisasi dan mengelola data yang memiliki tipe yang sama. Pemahaman yang baik tentang array akan sangat membantu dalam menyelesaikan berbagai masalah pemrograman. Array tunggal adalah struktur data linier yang digunakan untuk menyimpan kumpulan data dengan tipe yang sama secara berurutan dalam satu variabel. Setiap elemen dalam array memiliki indeks unik yang menunjukkan posisinya. Hal ini memungkinkan kita untuk mengakses, memodifikasi, atau menghapus elemen secara efisien berdasarkan indeksnya. Array sangat berguna untuk menyimpan data yang saling berkaitan, seperti nilai ujian siswa, daftar nama, atau harga barang. Namun, perlu diingat bahwa ukuran array biasanya tetap setelah dideklarasikan, sehingga kita perlu menentukan ukurannya dengan cermat sebelum digunakan. Keuntungan utama menggunakan array adalah kemampuannya untuk mengakses data secara cepat dan efisien, serta kemudahan dalam melakukan operasi seperti pencarian, pengurutan, dan perhitungan statistik.

## DAFTAR PUSTAKA

Trivusi. (2023 Januari 08) *Pengertian, Karakteristik, dan Kegunaannya*.

<https://www.trivusi.web.id/2022/07/struktur-data-array.html#:~:text=Array%20satu%20dimensi%2C%20yaitu%20array,yang%20menyimpan%20list%20elemen%20tunggal.>

Putri C Adisty (2024 Mei 23) *Apa Itu Array? Pahami Pengertian, Fungsi, dan Contohnya*. <https://www.domainesia.com/berita/apa-itu-array/>

Fazry (2024 Mei 25) *Array: Konsep, Implementasi, dan Penggunaan*.

<https://rumahcoding.co.id/array-konsep-implementasi-dan-penggunaan/>