

LAPORAN PRAKTIKUM

PEMROGRAMAN BERORIENTASI OBJEK LANJUT

2023



Prepared By:

Nama : Kiki
Indiani Kelas
: R1
NIM 210511046
Praktikum 2 PB02

Single1

Script:

```
#Nama : Kiki Indiani
#Kelas : R1
#NIM : 210511046

#Nama : Kiki Indiani
#Kelas : R1
#NIM : 210511046

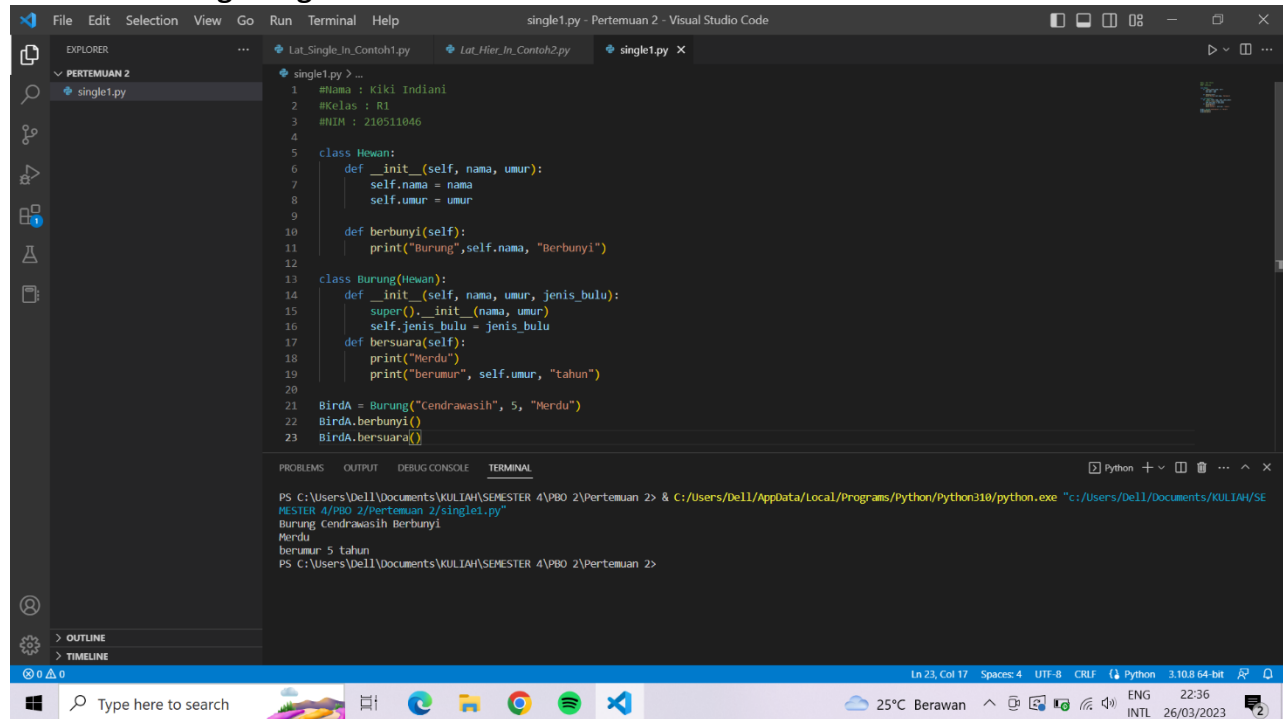
class Hewan:
    def __init__(self, nama, umur):
        self.nama = nama
        self.umur = umur

    def berbunyi(self):
        print("Burung", self.nama, "Berbunyi")

class Burung(Hewan):
    def __init__(self, nama, umur, jenis_bulu):
        super().__init__(nama, umur)
        self.jenis_bulu = jenis_bulu
    def bersuara(self):
        print("Merdu")
        print("berumur", self.umur, "tahun")

BirdA = Burung("Cendrawasih", 5, "Merdu")
BirdA.berbunyi()
BirdA.bersuara()
```

Hasil Running Program:



The screenshot shows a Visual Studio Code window with a Python file named `single1.py`. The code defines a `Burung` class and a `Burung` instance `BirdA`. The terminal output shows the execution of the script, which prints the name, class, and age of the bird.

```
1 #nama : Kiki Indiani
2 #Kelas : R1
3 #NIM : 210511046
4
5 class Burung:
6     def __init__(self, nama, umur):
7         self.nama = nama
8         self.umur = umur
9
10    def berbunyi(self):
11        print("Burung",self.nama, "Berbunyi")
12
13 class Burung(hewan):
14     def __init__(self, nama, umur, jenis_bulu):
15         super().__init__(nama, umur)
16         self.jenis_bulu = jenis_bulu
17     def bersuara(self):
18         print("Merdu")
19         print("berumur", self.umur, "tahun")
20
21 BirdA = Burung("Cendrawasih", 5, "Merdu")
22 BirdA.berbunyi()
23 BirdA.bersuara()
```

Terminal Output:

```
PS C:\Users\Dell\Documents\KULIAH\SEMESTER 4\PRO 2\Pertemuan 2> & c:\Users\Dell\AppData\Local\Programs\Python\Python310\python.exe "c:\Users\Dell\Documents\KULIAH\SEMESTER 4\PRO 2\Pertemuan 2\single1.py"
Burung Cendrawasih Berbunyi
Merdu
berumur 5 tahun
PS C:\Users\Dell\Documents\KULIAH\SEMESTER 4\PRO 2\Pertemuan 2>
```

Single2

Script:

```
#Nama : Kiki Indiani
#Kelas : R1
#NIM : 210511046
```

```
class Manusia:
    def __init__(self, nama, umur):
        self.nama = nama
        self.umur = umur
    def presentasi(self):
        print(f"{self.nama} sedang presentasi.")

class Dosen(Manusia):
    def __init__(self, nama, umur, nim):
        super().__init__(nama, umur)
```

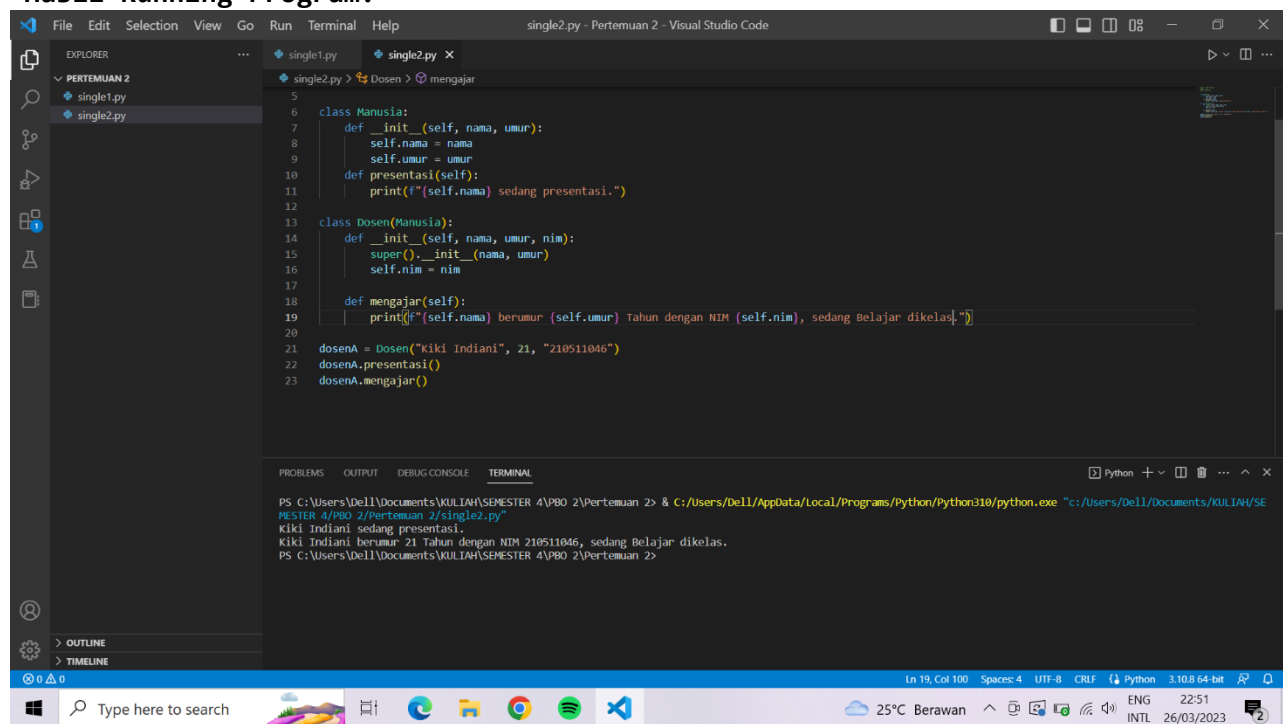
```
self.nim = nim
```

```
def mengajar(self):  
    print(f"{self.nama} berumur {self.umur} Tahun dengan NIM  
{self.nim}, sedang Belajar dikelas.")
```

```
dosenA = Dosen("Kiki Indiani", 21, "210511046")  
dosenA.presentasi()  
dosenA.mengajar()  
print(f"{self.nama} berumur {self.umur} Tahun dengan NIM  
{self.nim}, sedang Belajar.")
```

```
dosenA = Dosen("Kiki Indiani", 21,  
"210511046") dosenA.presentasi()  
dosenA.mengajar()
```

Hasil Running Program:



```
class Manusia:  
    def __init__(self, nama, umur):  
        self.nama = nama  
        self.umur = umur  
    def presentasi(self):  
        print(f"{self.nama} sedang presentasi.")  
  
class Dosen(Manusia):  
    def __init__(self, nama, umur, nim):  
        super().__init__(nama, umur)  
        self.nim = nim  
    def mengajar(self):  
        print(f"{self.nama} berumur {self.umur} Tahun dengan NIM {self.nim}, sedang Belajar dikelas.")  
  
dosenA = Dosen("Kiki Indiani", 21, "210511046")  
dosenA.presentasi()  
dosenA.mengajar()
```

```
PS C:\Users\Dell\Documents\KULIAH\SEMESTER 4\PBO 2\Pertemuan 2> & C:\Users\Dell\AppData\Local\Programs\Python\Python310\python.exe "c:\Users\Dell\Documents\KULIAH\SEMESTER 4\PBO 2\Pertemuan 2\single2.py"  
Kiki Indiani sedang presentasi.  
Kiki Indiani berumur 21 Tahun dengan NIM 210511046, sedang Belajar dikelas.  
PS C:\Users\Dell\Documents\KULIAH\SEMESTER 4\PBO 2\Pertemuan 2>
```

Multiple1

Script:

```
#Nama : Kiki Indiani
#Kelas : R1
#NIM : 210511046

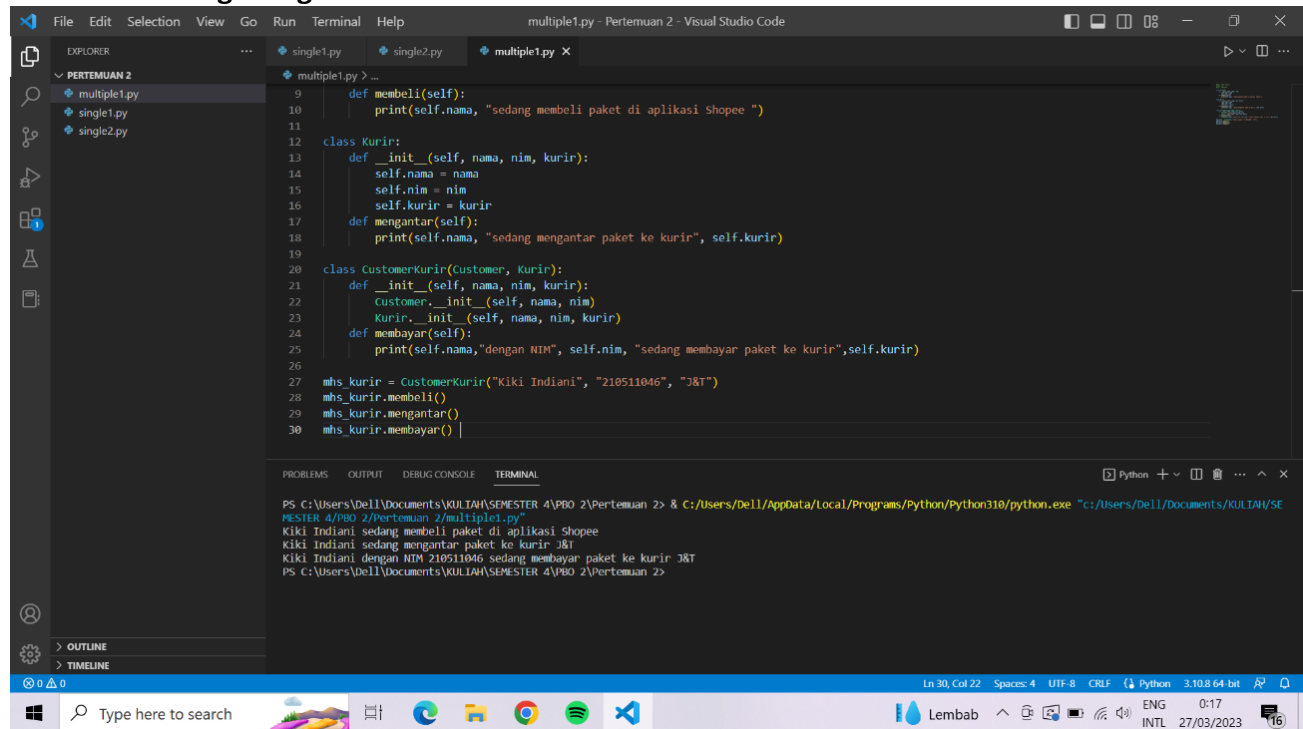
class Customer:
    def __init__(self, nama, nim):
        self.nama = nama
        self.nim = nim
    def membeli(self):
        print(self.nama, "sedang membeli paket di aplikasi Shopee ")

class Kurir:
    def __init__(self, nama, nim, kurir):
        self.nama = nama
        self.nim = nim
        self.kurir = kurir
    def mengantar(self):
        print(self.nama, "sedang mengantar paket ke kurir", self.kurir)

class CustomerKurir(Customer, Kurir):
    def __init__(self, nama, nim, kurir):
        Customer.__init__(self, nama, nim)
        Kurir.__init__(self, nama, nim, kurir)
    def membayar(self):
        print(self.nama,"dengan NIM", self.nim, "sedang membayar paket ke kurir",self.kurir)

mhs_kurir = CustomerKurir("Kiki Indiani", "210511046", "J&T")
mhs_kurir.membeli()
mhs_kurir.mengantar()
mhs_kurir.membayar()
```

Hasil Running Program:



The screenshot displays the Visual Studio Code interface with a Python file named `multiple1.py` open. The code defines a `Kurir` class and a `CustomerKurir` class. The `Kurir` class has methods `membeli` and `mengantar`. The `CustomerKurir` class inherits from `Kurir` and has a `membayar` method. The main execution block creates a `CustomerKurir` object and calls its methods.

```
9 def membeli(self):
10     print(self.nama, "sedang membeli paket di aplikasi Shopee ")
11
12 class Kurir:
13     def __init__(self, nama, nim, kurir):
14         self.nama = nama
15         self.nim = nim
16         self.kurir = kurir
17     def mengantar(self):
18         print(self.nama, "sedang mengantar paket ke kurir", self.kurir)
19
20 class CustomerKurir(Customer, Kurir):
21     def __init__(self, nama, nim, kurir):
22         Customer.__init__(self, nama, nim)
23         Kurir.__init__(self, nama, nim, kurir)
24     def membayar(self):
25         print(self.nama, "dengan NIM", self.nim, "sedang membayar paket ke kurir", self.kurir)
26
27 mhs_kurir = CustomerKurir("Kiki Indiani", "210511046", "3&T")
28 mhs_kurir.membeli()
29 mhs_kurir.mengantar()
30 mhs_kurir.membayar() |
```

The terminal output shows the execution of the program:

```
PS C:\Users\ DELL\Documents\KULIAH\SEMESTER 4\PRO 2\Pertemuan 2> & C:\Users\ DELL\AppData\Local\Programs\Python\Python310\python.exe "c:\Users\ DELL\Documents\KULIAH\SEMESTER 4\PRO 2\Pertemuan 2\multiple1.py"
Kiki Indiani sedang membeli paket di aplikasi Shopee
Kiki Indiani sedang mengantar paket ke kurir 3&T
Kiki Indiani dengan NIM 210511046 sedang membayar paket ke kurir 3&T
PS C:\Users\ DELL\Documents\KULIAH\SEMESTER 4\PRO 2\Pertemuan 2>
```

The status bar at the bottom indicates the file is at line 30, column 22, with 4 spaces, UTF-8 encoding, CRLF line endings, and is running Python 3.10.8 64-bit.

Multiple2

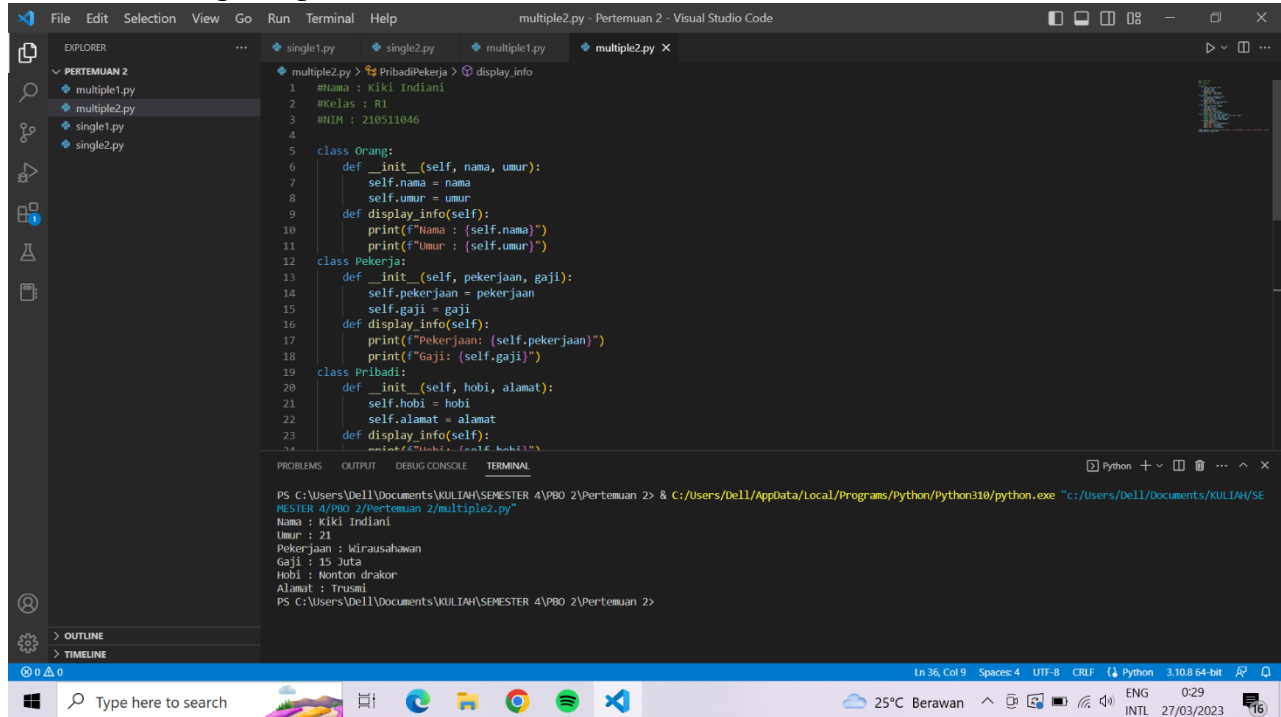
Script:

```
#Nama : Kiki Indiani
#Kelas : R1
#NIM : 210511046

class Orang:
    def __init__(self, nama, umur):
        self.nama = nama
        self.umur = umur
    def display_info(self):
        print(f>Nama : {self.nama}")
        print(f"Umur : {self.umur}")
class Pekerja:
    def __init__(self, pekerjaan, gaji):
        self.pekerjaan = pekerjaan
        self.gaji = gaji
    def display_info(self):
        print(f"Pekerjaan: {self.pekerjaan}")
        print(f"Gaji: {self.gaji}")
class Pribadi:
    def __init__(self, hobi, alamat):
        self.hobi = hobi
        self.alamat = alamat
    def display_info(self):
        print(f"Hobi: {self.hobi}")
        print(f"Alamat: {self.alamat}")
class PribadiPekerja(Orang, Pekerja, Pribadi):
    def __init__(self, nama, umur, pekerjaan, gaji, hobi, alamat):
        Orang.__init__(self, nama, umur)
        Pekerja.__init__(self, pekerjaan, gaji)
        Pribadi.__init__(self, hobi, alamat)
    def display_info(self):
        super().display_info()
        print(f"Pekerjaan : {self.pekerjaan}")
        print(f"Gaji : {self.gaji}")
        print(f"Hobi : {self.hobi}")
        print(f"Alamat : {self.alamat}")

pribadi_pekerjaC = PribadiPekerja("Kiki Indiani", 21, "Wirausahawan", "15
Juta", "Nonton drakor", "Trusmi")
pribadi_pekerjaC.display_info()
```

Hasil Running Program:



The screenshot displays the Visual Studio Code interface with a Python file named `multiple2.py` open. The code defines three classes: `Orang`, `Pekerja`, and `Pribadi`, each with an `__init__` method and a `display_info` method. The `display_info` method prints the attributes of the object. The `main` function creates instances of these classes and calls their `display_info` methods.

```
1 #nama : Kiki Indiani
2 #Kelas : R1
3 #NIM : 210511046
4
5 class Orang:
6     def __init__(self, nama, umur):
7         self.nama = nama
8         self.umur = umur
9     def display_info(self):
10        print(f>Nama : {self.nama}")
11        print(f"Umur : {self.umur}")
12
13 class Pekerja:
14     def __init__(self, pekerjaan, gaji):
15         self.pekerjaan = pekerjaan
16         self.gaji = gaji
17     def display_info(self):
18        print(f"Pekerjaan: {self.pekerjaan}")
19        print(f"Gaji: {self.gaji}")
20
21 class Pribadi:
22     def __init__(self, hobi, alamat):
23         self.hobi = hobi
24         self.alamat = alamat
25     def display_info(self):
26        print(f"Hobi : {self.hobi}")
27        print(f"Alamat : {self.alamat}")
28
29 def main():
30     orang = Orang("Kiki Indiani", 21)
31     pekerja = Pekerja("Wirasahawan", 15000000)
32     pribadi = Pribadi("Nonton drakor", "Trusmi")
33     orang.display_info()
34     pekerja.display_info()
35     pribadi.display_info()
36
37 if __name__ == "__main__":
38     main()
```

The terminal output shows the execution of the program, displaying the information for each instance:

```
PS C:\Users\Dell\Documents\KULIAH\SEMESTER 4\PRO 2\Pertemuan 2> & C:\Users\Dell\AppData\Local\Programs\Python\Python310\python.exe "c:\Users\Dell\Documents\KULIAH\SEMESTER 4\PRO 2\Pertemuan 2\multiple2.py"
Nama : Kiki Indiani
Umur : 21
Pekerjaan : Wirasahawan
Gaji : 15 Juta
Hobi : Nonton drakor
Alamat : Trusmi
PS C:\Users\Dell\Documents\KULIAH\SEMESTER 4\PRO 2\Pertemuan 2>
```


Hierarchic al1

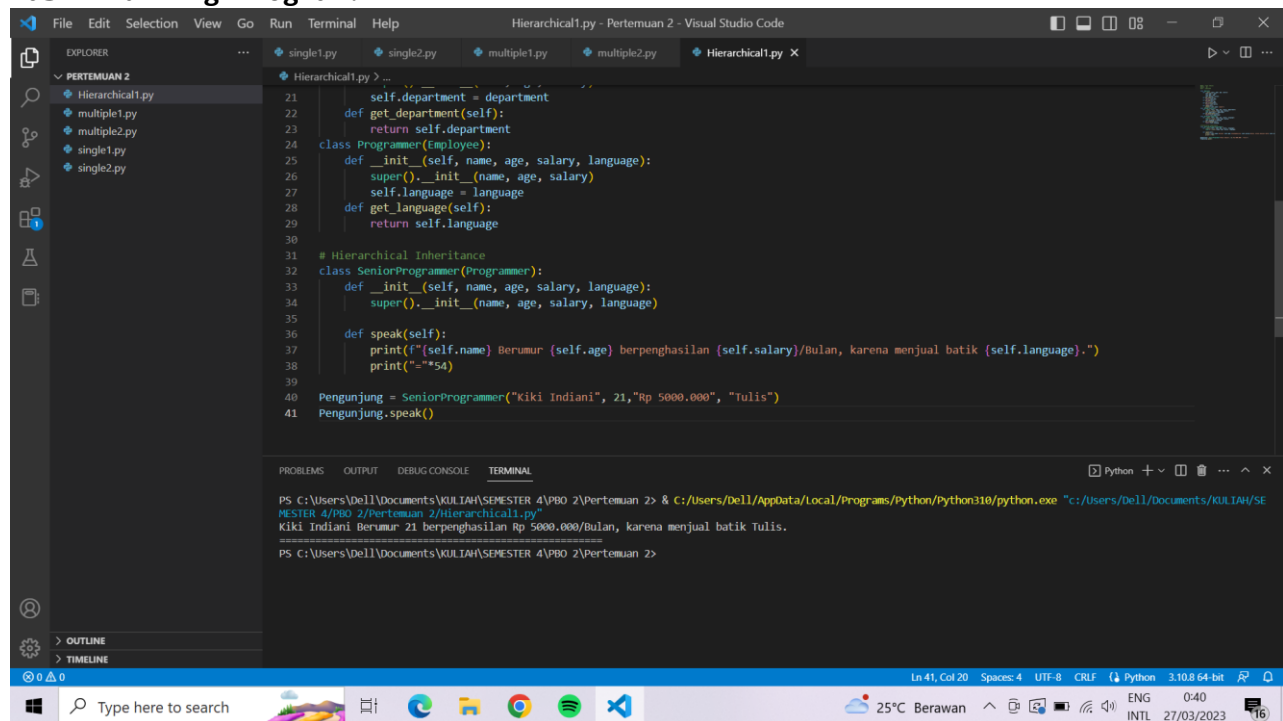
Script:

```
#Nama : Kiki Indiani  
#Kelas : R1  
#NIM : 210511046
```

```
class Employee:  
    def __init__(self, name, age, salary):  
        self.name = name  
        self.age = age  
        self.salary = salary  
    def get_name(self):  
        return self.name  
    def get_age(self):  
        return self.age  
    def get_salary(self):  
        return self.salary  
    def speak(self):  
        print(f"{self.name} speaks")  
class Manager(Employee):  
    def __init__(self, name, age, salary, department):  
        super().__init__(name, age, salary)  
        self.department = department  
    def get_department(self):  
        return self.department  
class Programmer(Employee):  
    def __init__(self, name, age, salary, language):  
        super().__init__(name, age, salary)  
        self.language = language  
    def get_language(self):  
        return self.language  
  
# Hierarchical Inheritance  
class SeniorProgrammer(Programmer):  
    def __init__(self, name, age, salary, language):  
        super().__init__(name, age, salary, language)  
  
    def speak(self):  
        print(f"{self.name} Berumur {self.age} berpenghasilan  
{self.salary}/Bulan, karena menjual batik {self.language}.")  
        print("="*54)
```

```
Pengunjung = SeniorProgrammer("Kiki Indiani", 21,"Rp 5000.000", "Tulis")
```

Hasil Running Program:



The screenshot displays the Visual Studio Code interface with a Python file named `Hierarchical1.py` open. The code defines a `Programmer` class and a `SeniorProgrammer` class that inherits from it. The `SeniorProgrammer` class has a `speak` method that prints the programmer's name, age, salary, and language. The main part of the code creates an instance of `SeniorProgrammer` named `Pengunjung` with the parameters `("Kiki Indiani", 21, "Rp 5000.000", "Tulis")` and calls the `speak` method.

```
21     self.department = department
22     def get_department(self):
23         return self.department
24     class Programmer(Employee):
25         def __init__(self, name, age, salary, language):
26             super().__init__(name, age, salary)
27             self.language = language
28         def get_language(self):
29             return self.language
30
31     # Hierarchical Inheritance
32     class SeniorProgrammer(Programmer):
33         def __init__(self, name, age, salary, language):
34             super().__init__(name, age, salary, language)
35
36         def speak(self):
37             print(f"{self.name} Berumur {self.age} berpenghasilan {self.salary}/Bulan, karena menjual batik {self.language}.")
38             print("-"*54)
39
40     Pengunjung = SeniorProgrammer("Kiki Indiani", 21,"Rp 5000.000", "Tulis")
41     Pengunjung.speak()
```

The terminal output shows the command to run the script and the resulting printed output:

```
PS C:\Users\Dell\Documents\KULIAH\SEMESTER 4\PRO 2\Pertemuan 2> & C:\Users\Dell\AppData\Local\Programs\Python\Python310\python.exe "c:\Users\Dell\Documents\KULIAH\SEMESTER 4\PRO 2\Pertemuan 2\Hierarchical1.py"
Kiki Indiani Berumur 21 berpenghasilan Rp 5000,000/Bulan, karena menjual batik Tulis.
-----
PS C:\Users\Dell\Documents\KULIAH\SEMESTER 4\PRO 2\Pertemuan 2>
```

Hierarchical2

Script:

#Nama : Kiki Indiani

#Kelas : R1

#NIM : 210511046

```
class Ekspedisi:
```

```
def __init__(self, nama, umur, gaji):
```

```
self.nama = nama
```

```
self.umur = umur
```

```
self.gaji = gaji
```

```
def get_nama(self):
```

```
return self.nama
```

```
def get_umur(self):
```

```
return self.umur
```

```
def get_gaji(self):
```

```
return self.gaji
```

```
def speak(self):
```

```
print(f"{self.nama} speaks")
```

```
class Kurir(Ekspedisi):
```

```
def __init__(self, nama, umur, gaji, department):
```

```
super().__init__(nama, umur, gaji)
```

```
self.department = department
```

```
def get_department(self):
```

```

    return self.department

```

```
class Datakurir(Ekspedisi):
```

```
def __init__(self, nama, umur, gaji, alamat):
```

```
super().init(nama, umur, gaji)
```

```
self.alamat = alamat
```

```
def get alamat(self):
```

```
return self.alamat
```

Hierarchical Inheritance

```
class SeniorDatakurir(Datakurir):
```

```
def __init__(self, nama, umur, gaji, alamat, ekspedisi):
```

```
super().init(nama, umur, gaji, alamat)
```

```
self.ekspedisi = ekspedisi
```

```
def get ekspedisi(self):
```

```

    return self.ekspedisi

```

```
def data(self):
```

```
print(f"{self.nama} Berumur {self.umur} tahun berpenghasilan
```

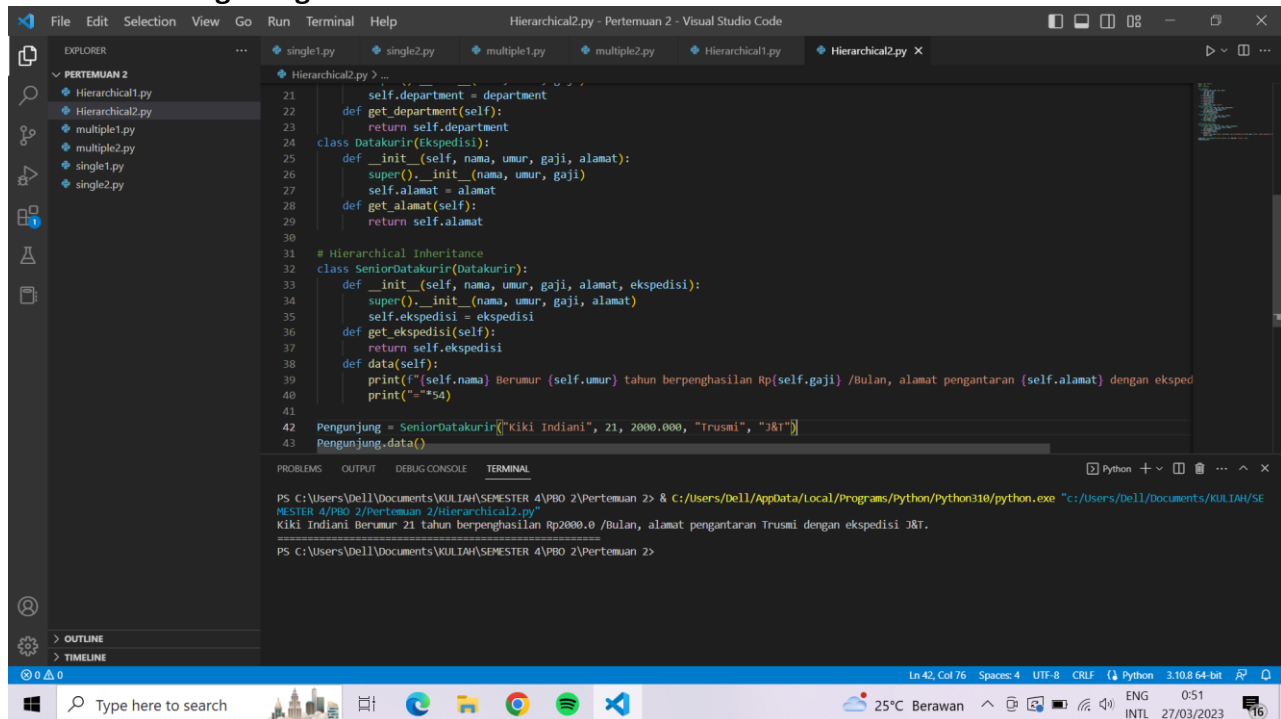
```
Rp{self.gaji} /Bulan, alamat pengantaran {self.alamat} dengan ekspedisi
{self.ekspedisi}.")
```

```
print("="*54)
```

```
Pengunjung = SeniorDatakurir("Kiki Indiani", 21, 2000.000, "Trusmi",  
"J&T")
```

```
Pengunjung.data()
```

Hasil Running Program:



The screenshot displays the Visual Studio Code interface with a Python file named `Hierarchical2.py` open. The code defines a base class `Datakurir` and a derived class `SeniorDatakurir` demonstrating hierarchical inheritance. The `SeniorDatakurir` class inherits from `Datakurir` and adds an `ekspedisi` attribute and a `data` method. The `data` method prints the object's attributes in a formatted string. An instance of `SeniorDatakurir` is created and its `data` method is called.

```
21     self.department = department
22     def get_department(self):
23         return self.department
24     class Datakurir(ekspedisi):
25     def __init__(self, nama, umur, gaji, alamat):
26         super().__init__(nama, umur, gaji)
27         self.alamat = alamat
28     def get_alamat(self):
29         return self.alamat
30
31     # Hierarchical Inheritance
32     class SeniorDatakurir(Datakurir):
33     def __init__(self, nama, umur, gaji, alamat, ekspedisi):
34         super().__init__(nama, umur, gaji, alamat)
35         self.ekspedisi = ekspedisi
36     def get_ekspedisi(self):
37         return self.ekspedisi
38     def data(self):
39         print(f"{self.nama} Berumur {self.umur} tahun berpenghasilan Rp{self.gaji} /Bulan, alamat pengantaran {self.alamat} dengan eksped
40         print("==s4")
41
42     Pengunjung = SeniorDatakurir("Kiki Indiani", 21, 2000.000, "Trusmi", "&T")
43     Pengunjung.data()
```

The terminal output shows the execution of the program, displaying the formatted string output of the `data` method:

```
PS C:\Users\Dell\Documents\KULIAH\SEMESTER 4\PRO 2\Pertemuan 2> & C:\Users\Dell\AppData\Local\Programs\Python\Python310\python.exe "c:/Users/Dell/Documents/KULIAH/SE
MESTER 4/PRO 2/Pertemuan 2/Hierarchical2.py"
Kiki Indiani Berumur 21 tahun berpenghasilan Rp2000.0 /Bulan, alamat pengantaran Trusmi dengan ekspedisi &T.
PS C:\Users\Dell\Documents\KULIAH\SEMESTER 4\PRO 2\Pertemuan 2>
```

Multilevel

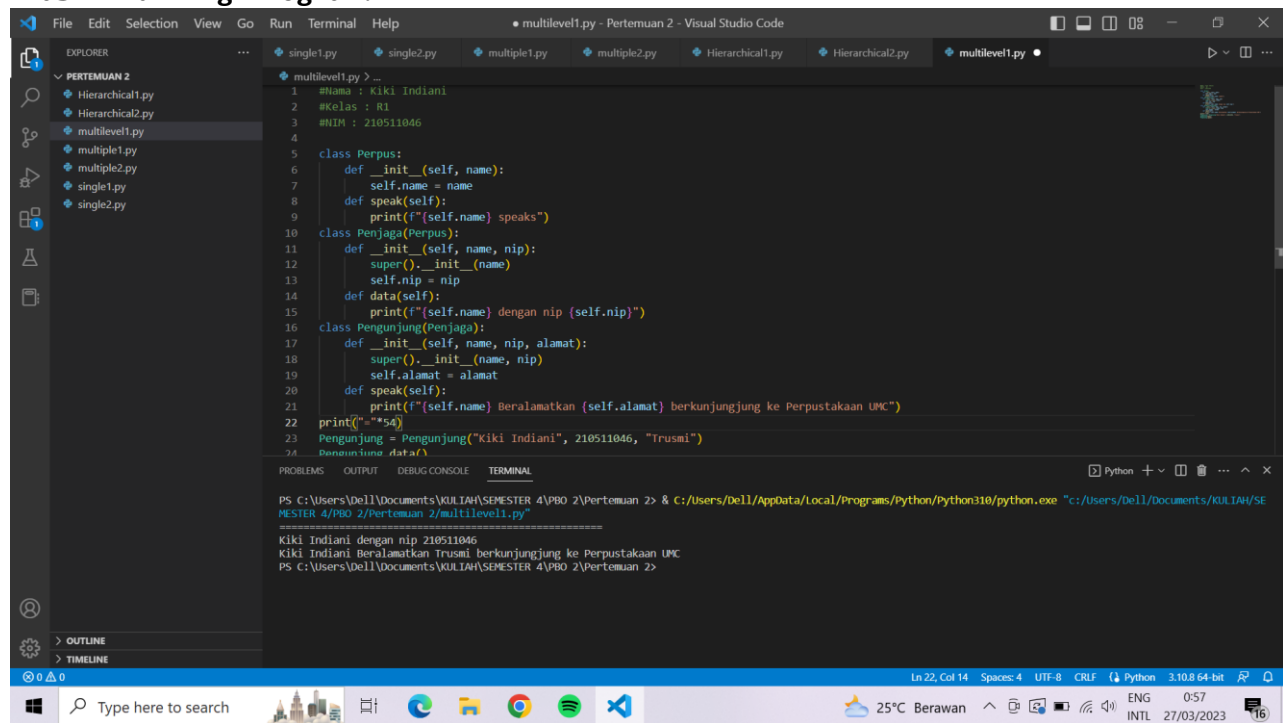
1

Script:

```
#Nama : Kiki Indiani
#Kelas : R1
#NIM : 210511046

class Perpus:
    def __init__(self, name):
        self.name = name
    def speak(self):
        print(f"{self.name} speaks")
class Penjaga(Perpus):
    def __init__(self, name, nip):
        super().__init__(name)
        self.nip = nip
    def data(self):
        print(f"{self.name} dengan nip {self.nip}")
class Pengunjung(Penjaga):
    def __init__(self, name, nip, alamat):
        super().__init__(name, nip)
        self.alamat = alamat
    def speak(self):
        print(f"{self.name} Beralamatkan {self.alamat} berkunjung ke
Perpustakaan UMC")
print("="*54)
Pengunjung = Pengunjung("Kiki Indiani", 210511046, "Trusmi")
Pengunjung.data()
Pengunjung.speak()
```

Hasil Running Program:



The screenshot displays the Visual Studio Code interface with a Python file named `multilevel1.py` open. The Explorer sidebar on the left shows a project structure for `PERTEMUAN 2` containing several Python files. The main editor area shows the code for `multilevel1.py`, which defines three classes: `Perpus`, `Penjaga`, and `Pengunjung`. The `Perpus` class has an `__init__` method and a `speak` method. The `Penjaga` class inherits from `Perpus` and adds a `data` method. The `Pengunjung` class inherits from `Penjaga` and adds a `speak` method. The code concludes with an instantiation of the `Pengunjung` class and a call to its `speak` method.

```
1 #nama : Kiki Indiani
2 #Kelas : R1
3 #NIM : 210511046
4
5 class Perpus:
6     def __init__(self, name):
7         self.name = name
8     def speak(self):
9         print(f"{self.name} speaks")
10 class Penjaga(Perpus):
11     def __init__(self, name, nip):
12         super().__init__(name)
13         self.nip = nip
14     def data(self):
15         print(f"{self.name} dengan nip {self.nip}")
16 class Pengunjung(Penjaga):
17     def __init__(self, name, nip, alamat):
18         super().__init__(name, nip)
19         self.alamat = alamat
20     def speak(self):
21         print(f"{self.name} Beralamatkan {self.alamat} berkunjung ke Perpustakaan UMC")
22 print("\n==54")
23 Pengunjung = Pengunjung("Kiki Indiani", 210511046, "Trusmi")
24 Pengunjung.speak()
```

The terminal at the bottom shows the command used to run the program and the resulting output:

```
PS C:\Users\Dell\Documents\KULIAH\SEMESTER 4\PRO 2\Pertemuan 2> & C:\Users\Dell\AppData\Local\Programs\Python\Python310\python.exe "c:/Users/Dell/Documents/KULIAH/SEMESTER 4/PRO 2/Pertemuan 2/multilevel1.py"
Kiki Indiani dengan nip 210511046
Kiki Indiani Beralamatkan Trusmi berkunjung ke Perpustakaan UMC
PS C:\Users\Dell\Documents\KULIAH\SEMESTER 4\PRO 2\Pertemuan 2>
```

Multilevel

2

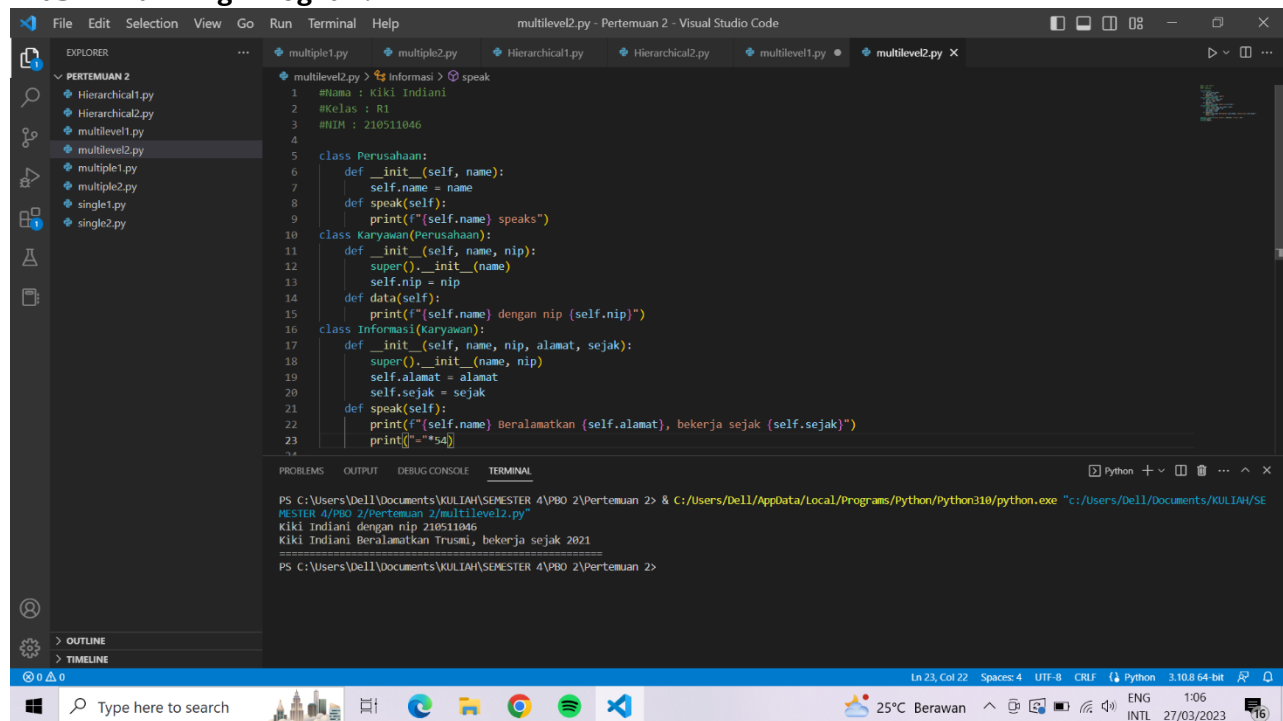
Script:

```
#Nama : Kiki Indiani
#Kelas : R1
#NIM : 210511046

class Perusahaan:
    def __init__(self, name):
        self.name = name
    def speak(self):
        print(f"{self.name} speaks")
class Karyawan(Perusahaan):
    def __init__(self, name, nip):
        super().__init__(name)
        self.nip = nip
    def data(self):
        print(f"{self.name} dengan nip {self.nip}")
class Informasi(Karyawan):
    def __init__(self, name, nip, alamat, sejak):
        super().__init__(name, nip)
        self.alamat = alamat
        self.sejak = sejak
    def speak(self):
        print(f"{self.name} Beralamatkan {self.alamat}, bekerja sejak {self.sejak}")
        print("=="*54)

Informasi = Informasi("Kiki Indiani", 210511046, "Trusmi", 2021)
Informasi.data()
Informasi.speak()
```

Hasil Running Program:



The screenshot shows the Visual Studio Code editor with a file named `multilevel2.py` open. The code defines three classes: `Perusahaan`, `Karyawan` (which inherits from `Perusahaan`), and `Informasi` (which inherits from `Karyawan`). The `Informasi` class has an `__init__` method that takes `name`, `nip`, `alamat`, and `sejak` as arguments. The `main` function creates an instance of `Informasi` with the values `Kiki Indiani`, `210511046`, `R1`, and `2021`, and calls the `speak` method. The terminal output shows the execution of the program, displaying the name, NIP, address, and start year of the employee.

```
1 #nama : Kiki Indiani
2 #Kelas : R1
3 #NIM : 210511046
4
5 class Perusahaan:
6     def __init__(self, name):
7         self.name = name
8     def speak(self):
9         print(f"{self.name} speaks")
10
11 class Karyawan(Perusahaan):
12     def __init__(self, name, nip):
13         super().__init__(name)
14         self.nip = nip
15     def data(self):
16         print(f"{self.name} dengan nip {self.nip}")
17
18 class Informasi(Karyawan):
19     def __init__(self, name, nip, alamat, sejak):
20         super().__init__(name, nip)
21         self.alamat = alamat
22         self.sejak = sejak
23     def speak(self):
24         print(f"{self.name} Beralamatkan {self.alamat}, bekerja sejak {self.sejak}")
25         print(f"{'*' * 54}")
26
27 if __name__ == '__main__':
28     info = Informasi("Kiki Indiani", "210511046", "R1", "2021")
29     info.speak()
30     info.data()
31
32 PS C:\Users\Dell\Documents\KULIAH\SEMESTER 4\PRO 2\Pertemuan 2> & C:\Users\Dell\AppData\Local\Programs\Python\Python310\python.exe "C:\Users\Dell\Documents\KULIAH\SEMESTER 4\PRO 2\Pertemuan 2\multilevel2.py"
Kiki Indiani dengan nip 210511046
Kiki Indiani Beralamatkan Trusmi, bekerja sejak 2021
*****
PS C:\Users\Dell\Documents\KULIAH\SEMESTER 4\PRO 2\Pertemuan 2>
```

Hybrid1

Script:

```
#Nama : Kiki Indiani
#Kelas : R1
#NIM : 210511046

class Seseorang:
    def __init__(self, name, age, address):
        self.name = name
        self.age = age
        self.address = address
    def get_info(self):
        print("Nama:", self.name)
        print("Umur:", self.age)
        print("Alamat:", self.address)

# Single Inheritance
class Mahasiswa(Seseorang):
    def __init__(self, name, age, address):
        super().__init__(name, age, address)

    def get_info(self):
        super().get_info()

# Single Inheritance
class Employee(Seseorang):
    def __init__(self, name, age, address, employee_id, salary):
        super().__init__(name, age, address)
        self.employee_id = employee_id
        self.salary = salary
    def get_info(self):
```



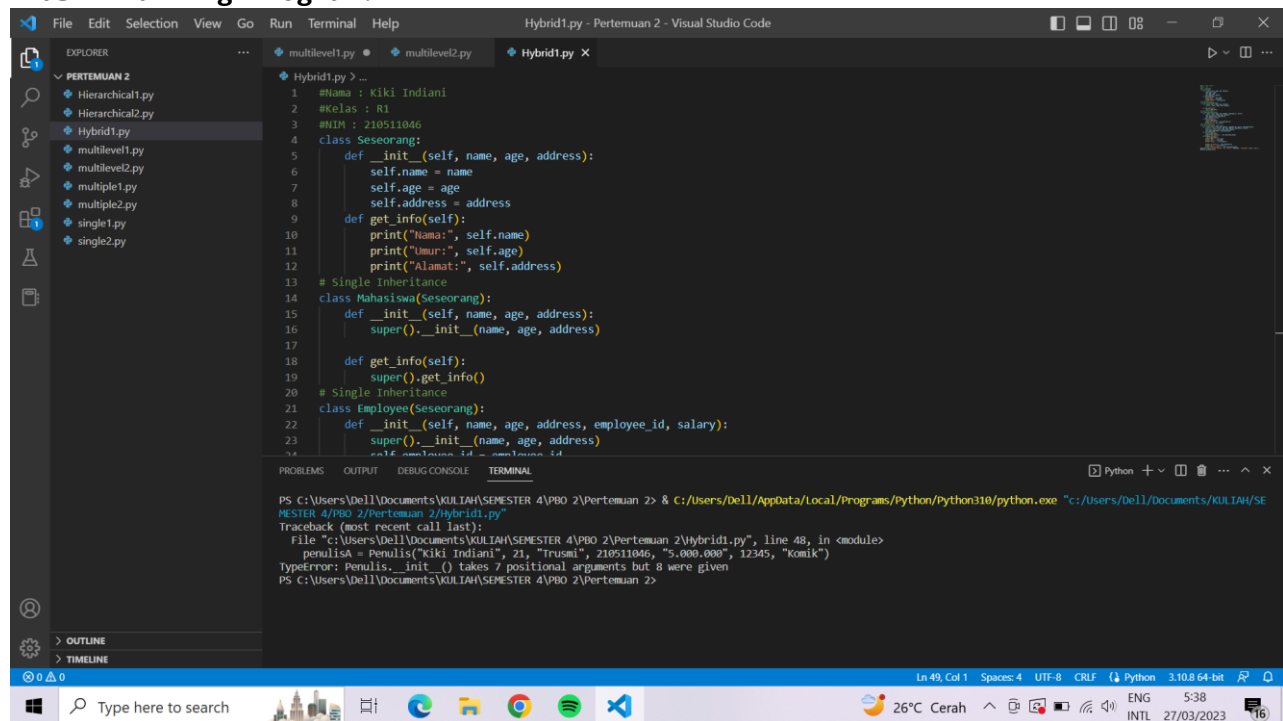
```

        super().get_info()
        print("ID Pekerja:", self.employee_id)
        print("Gaji:", self.salary)
# Multiple Inheritance
class Penulis(Employee, Mahasiswa):
    def __init__(self, name, age, address, employee_id, salary, published_books):
        Employee.__init__(self, name, age, address, employee_id, salary)
        Mahasiswa.__init__(self, name, age, address)
        self.published_books = published_books
    def get_info(self):
        super().get_info()
        print("Buku Publikasi:", self.published_books)
    def display_info(self):
        super().get_info()
        print(f>Nama: ", self.name)
        print(f"Umur: ", self.age)
        print(f"Alamat: ", self.address)

        print(f"ID Pekerja: ", self.employee_id)
        print(f"Gaji: ", self.salary)
        print(f"Buku Publikasi: ", self.published_books)
penulisA = Penulis("Kiki Indiani", 21, "Trusmi", 210511046, "5.000.000", 12345,
"Komik")
penulisA.display_info()

```

Hasil Running Program:



The screenshot shows the Visual Studio Code interface with a file named 'Hybrid1.py' open. The code defines a base class 'Seseorang' and two subclasses, 'Mahasiswa' and 'Employee'. The 'Mahasiswa' class has an additional attribute 'student_id' and a 'get_info' method that prints it. The 'Employee' class has additional attributes 'employee_id' and 'salary'. The terminal output shows the command to run the program and a traceback of a 'TypeError' in the 'penulis' object's '.__init__' method, indicating that 8 positional arguments were given but only 7 were expected.

```
File Edit Selection View Go Run Terminal Help
Hybrid1.py - Pertemuan 2 - Visual Studio Code

EXPLORER
PERTEMUAN 2
  Hierarchical1.py
  Hierarchical2.py
  Hybrid1.py
  multilevel1.py
  multilevel2.py
  multiple1.py
  multiple2.py
  single1.py
  single2.py

Hybrid1.py
1 #Nama : Kiki Indiani
2 #Kelas : R1
3 #NIM : 210511046
4 class Seseorang:
5     def __init__(self, name, age, address):
6         self.name = name
7         self.age = age
8         self.address = address
9     def get_info(self):
10        print("Nama:", self.name)
11        print("Umur:", self.age)
12        print("Alamat:", self.address)
13 # Single Inheritance
14 class Mahasiswa(Seseorang):
15     def __init__(self, name, age, address):
16         super().__init__(name, age, address)
17
18     def get_info(self):
19         super().get_info()
20 # Single Inheritance
21 class Employee(Seseorang):
22     def __init__(self, name, age, address, employee_id, salary):
23         super().__init__(name, age, address)
24         self.employee_id = employee_id
25         self.salary = salary

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
Python + - [ ] ... ^ x

PS C:\Users\De11\Documents\KULIAH\SEMESTER 4\PRO 2\Pertemuan 2> & c:\Users\De11\AppData\Local\Programs\Python\Python310\python.exe "c:\Users\De11\Documents\KULIAH\SEMESTER 4\PRO 2\Pertemuan 2\Hybrid1.py"
Traceback (most recent call last):
  File "c:\Users\De11\Documents\KULIAH\SEMESTER 4\PRO 2\Pertemuan 2\Hybrid1.py", line 48, in <module>
    penulis = Penulis("Kiki Indiani", 21, "Trusmi", 210511046, "5.000.000", 12345, "komik")
TypeError: Penulis.__init__() takes 7 positional arguments but 8 were given
PS C:\Users\De11\Documents\KULIAH\SEMESTER 4\PRO 2\Pertemuan 2>
```

Hybrid2

Script:

```
#Nama : Kiki Indiani
#Kelas : R1
#NIM : 210511046

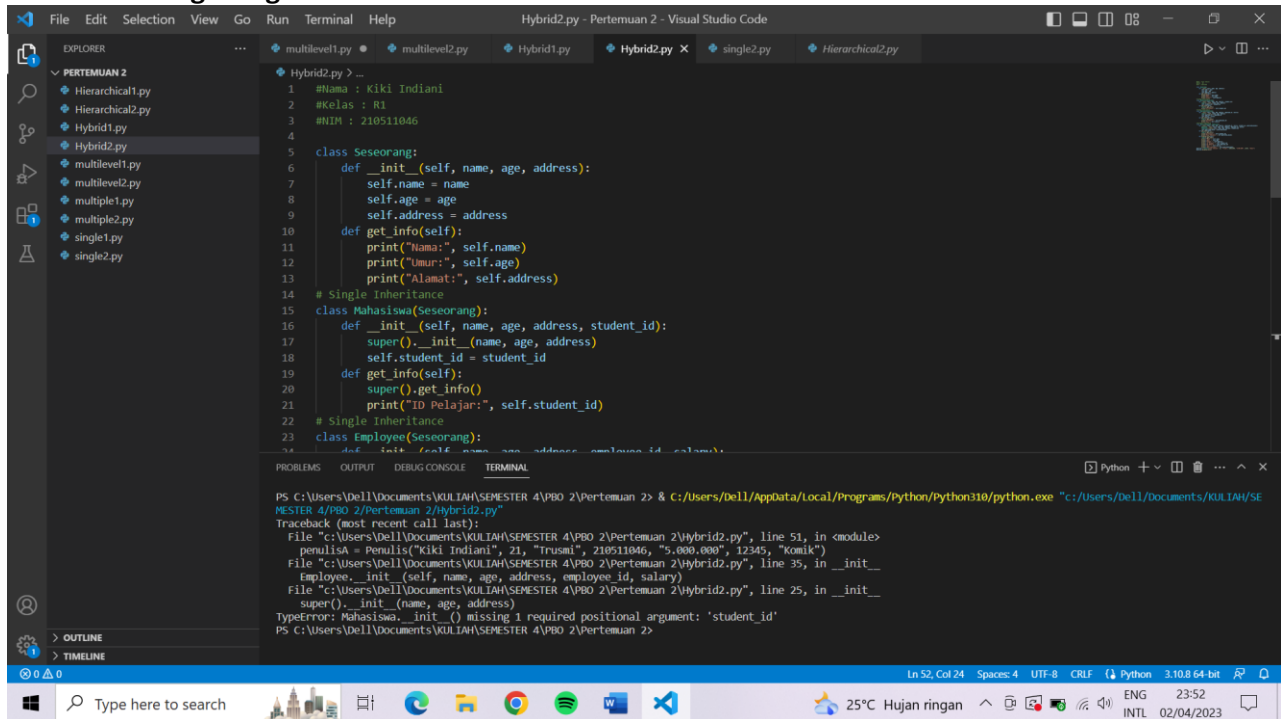
class Seseorang:
    def __init__(self, name, age, address):
        self.name = name
        self.age = age
        self.address = address
    def get_info(self):
        print("Nama:", self.name)
        print("Umur:", self.age)
        print("Alamat:", self.address)
# Single Inheritance
class Mahasiswa(Seseorang):
    def __init__(self, name, age, address, student_id):
        super().__init__(name, age, address)
        self.student_id = student_id
    def get_info(self):
        super().get_info()
        print("ID Pelajar:", self.student_id)
# Single Inheritance
class Employee(Seseorang):
    def __init__(self, name, age, address, employee_id, salary):
        super().__init__(name, age, address)
        self.employee_id = employee_id
        self.salary = salary
```

```

    def get_info(self):
        super().get_info()
        print("ID Pekerja:", self.employee_id)
        print("Gaji:", self.salary)
# Multiple Inheritance
class Penulis(Employee, Mahasiswa):
    def __init__(self, name, age, address, employee_id, salary, student_id,
published_books):
        Employee.__init__(self, name, age, address, employee_id, salary)
        Mahasiswa.__init__(self, name, age, address, student_id)
        self.published_books = published_books
    def get_info(self):
        super().get_info()
        print("ID Pelajar:", self.student_id)
        print("Buku Publikasi:", self.published_books)
    def display_info(self):
        super().get_info()
        print(f>Nama: ", self.name)
        print(f"Umur: ", self.age)
        print(f"Alamat: ", self.address)
        print(f"ID Pelajar: ", self.student_id)
        print(f"ID Pekerja: ", self.employee_id)
        print(f"Gaji: ", self.salary)
        print(f"Buku Publikasi: ", self.published_books)
penulisA = Penulis("Kiki Indiani", 21, "Trusmi", 210511046, "5.000.000", 12345,
"Komik")
penulisA.display_info()

```

Hasil Running Program:



The screenshot shows the Visual Studio Code interface with a Python file named `Hybrid2.py` open. The code defines a base class `Seseorang` and two subclasses, `Mahasiswa` and `Employee`. The `Mahasiswa` class has an `__init__` method that calls `super().__init__` with four arguments, but the `Seseorang` class's `__init__` method only accepts three arguments. This causes a `TypeError` when a `Mahasiswa` object is created.

```
1 #nama : Kiki Indiani
2 #kelas : R1
3 #NIM : 210511046
4
5 class Seseorang:
6     def __init__(self, name, age, address):
7         self.name = name
8         self.age = age
9         self.address = address
10    def get_info(self):
11        print("Nama:", self.name)
12        print("Umur:", self.age)
13        print("Alamat:", self.address)
14    # Single Inheritance
15    class Mahasiswa(Seseorang):
16        def __init__(self, name, age, address, student_id):
17            super().__init__(name, age, address)
18            self.student_id = student_id
19        def get_info(self):
20            super().get_info()
21            print("ID Pelajar:", self.student_id)
22    # Single Inheritance
23    class Employee(Seseorang):
24        def __init__(self, name, age, address, employee_id, salary):
25            super().__init__(name, age, address)
```

The terminal output shows the command to run the program and the resulting error:

```
PS C:\Users\Dell\Documents\KULIAH\SEMESTER 4\PRO 2\Pertemuan 2> & C:\Users\Dell\AppData\Local\Programs\Python\Python310\python.exe "c:/Users/Dell/Documents/KULIAH/SEMESTER 4/PRO 2/Pertemuan 2/Hybrid2.py"
Traceback (most recent call last):
  File "c:/Users/Dell/Documents/KULIAH/SEMESTER 4/PRO 2/Pertemuan 2/Hybrid2.py", line 51, in <module>
    penulisA = Penulis("Kiki Indiani", 21, "Trusmi", 210511046, "5.000.000", 12345, "Komik")
  File "c:/Users/Dell/Documents/KULIAH/SEMESTER 4/PRO 2/Pertemuan 2/Hybrid2.py", line 35, in __init__
    Employee.__init__(self, name, age, address, employee_id, salary)
  File "c:/Users/Dell/Documents/KULIAH/SEMESTER 4/PRO 2/Pertemuan 2/Hybrid2.py", line 25, in __init__
    super().__init__(name, age, address)
TypeError: Mahasiswa.__init__() missing 1 required positional argument: 'student_id'
PS C:\Users\Dell\Documents\KULIAH\SEMESTER 4\PRO 2\Pertemuan 2>
```

The status bar at the bottom indicates the file is at line 52, column 24, using UTF-8 encoding and CRLF line endings. The system tray shows the date and time as 02/04/2023, 23:52.