

LAPORAN PRAKTIKUM

PEMROGRAMAN BERORIENTASI OBJEK LANJUT

2023



Prepared By:

NAMA : RIFKI PRAMAYANDI MAHESA

NIM : 21051156

KELAS : D (Reguler)

PEMROGRAMAN BERORIENTASI OBJECK LANJUTAN

Dosen Matakuliah : Freddy Wicaksono, M. Kom

Tugas Praktikum :

1. Buatlah masing-masing 2 contoh jenis pewarisan di luar dari contoh yang telah diberikan, beri nama :

a. Sintaks :

1) Single-1.py

```
class Komputer:
    def __init__(self, merk, harga):
        self.merk = merk
        self.harga = harga

    def spesifikasi(self):
        print(f"Komputer merk {self.merk} dengan harga {self.harga}.")

class Laptop(Komputer):
    def __init__(self, merk, harga, ukuran_layar):
        super().__init__(merk, harga)
        self.ukuran_layar = ukuran_layar

    def spesifikasi(self):
        print(f"Laptop merk {self.merk} dengan harga {self.harga} dan ukuran layar {self.ukuran_layar} inci.")

laptop_acer = Laptop("Acer", 5000000, 15.6)
laptop_acer.spesifikasi() # output: "Laptop merk Acer dengan harga 5000000 dan ukuran layar 15.6 inci."

print("\n|   RIFKI PRAMAYANDI MAHESA   |")
print("|           210511156           |")
print("|           KELAS D               |\n")
```

2) Single-2.py

```
class UnsurAlam:
    def __init__(self, nama):
        self.nama = nama

    def get_info(self):
        print(f"Ini adalah unsur alam {self.nama}.")

class UnsurLogam(UnsurAlam):
    def __init__(self, nama, sifat):
        super().__init__(nama)
        self.sifat = sifat

    def get_info(self):
        print(f"Ini adalah unsur logam {self.nama} yang memiliki sifat {self.sifat}.")

unsur_1 = UnsurAlam("Oksigen")
unsur_2 = UnsurLogam("Emas", "Mudah ditempa")

unsur_1.get_info() # output: "Ini adalah unsur alam Oksigen."
unsur_2.get_info() # output: "Ini adalah unsur logam Emas yang memiliki sifat Mudah ditempa."

print("\n|  RIFKI PRAMAYANDI MAHESA  |")
print("|          210511156          |")
print("|          KELAS D          |")
print("|          |\n")
```

3) Multiple-1.py

```
class Switchable:
    def switch_on(self):
        self.is_on = True
        print("Switched on")

    def switch_off(self):
        self.is_on = False
        print("Switched off")

class Dimmable:
    def set_brightness(self, brightness):
        self.brightness = brightness
        print("Brightness set to", brightness)

class Lampu(Switchable, Dimmable):
    def __init__(self):
        self.is_on = False
        self.brightness = 0

lampu1 = Lampu()
lampu1.switch_on()
lampu1.set_brightness(50)
print("Brightness:", lampu1.brightness)

lampu1.switch_off()
print("Is on:", lampu1.is_on)

print("\n|  RIFKI PRAMAYANDI MAHESA  |")
print("|          210511156          |")
print("|          KELAS D              |\n")
```

4) Multiple-2.py

```
class WebPage:
    def __init__(self, url):
        self.url = url

    def open(self):
        print("Opening webpage:", self.url)

class OperatingSystem:
    def __init__(self, name):
        self.name = name

    def boot_up(self):
        print(self.name, "is booting up")

class Browser(WebPage, OperatingSystem):
    def __init__(self, url, name):
        WebPage.__init__(self, url)
        OperatingSystem.__init__(self, name)

    def display(self):
        self.boot_up()
        self.open()
        print("Browser is ready to use")

# Membuat objek dari kelas Browser
my_browser = Browser("https://www.google.com", "Windows 10")

# Memanggil metode display() pada objek my_browser
my_browser.display()

print("\n| RIFKI PRAMAYANDI MAHESA |")
print("| 210511156 |")
print("| KELAS D | \n")
```

5) Hierarchy-1.py

```
class OutputDevice:
    def __init__(self):
        self.connected = False

    def connect(self):
        self.connected = True

    def disconnect(self):
        self.connected = False

class Printer(OutputDevice):
    def __init__(self):
        super().__init__()
        self.print_queue = []

    def print(self, document):
        if self.connected:
            self.print_queue.append(document)
            print(f"Printing {document}")
        else:
            print("Printer is not connected.")

    def cancel_print(self, document):
        if document in self.print_queue:
            self.print_queue.remove(document)
            print(f"{document} has been removed from print queue.")
        else:
            print(f"{document} is not in print queue.")

class Scanner(OutputDevice):
    def __init__(self):
        super().__init__()

    def scan(self):
        if self.connected:
            print("Scanning document...")
        else:
            print("Scanner is not connected.")

# Membuat objek printer dan scanner
printer = Printer()
scanner = Scanner()

# Menghubungkan printer dan scanner
printer.connect()
scanner.connect()
```

```
# Mencetak dokumen dengan printer
printer.print("Sample Document")

# Membatalkan pencetakan dokumen dengan printer
printer.cancel_print("Sample Document")

# Melakukan scanning dokumen dengan scanner
scanner.scan()

# Memutuskan koneksi printer dan scanner
printer.disconnect()
scanner.disconnect()

print("\n|  RIFKI PRAMAYANDI MAHESA  |")
print("|          210511156          |")
print("|          KELAS D          | \n")
```

6) Hierarchy-2.py

```
class Hardware:
    def __init__(self):
        self.powered_on = False

    def power_on(self):
        self.powered_on = True
        print("Hardware is now powered on.")

    def power_off(self):
        self.powered_on = False
        print("Hardware is now powered off.")

class InputDevice(Hardware):
    def __init__(self):
        super().__init__()

class OutputDevice(Hardware):
    def __init__(self):
        super().__init__()

class Keyboard(InputDevice):
    def __init__(self):
        super().__init__()

    def type(self, text):
        if self.powered_on:
            print(f"Typed: {text}")
        else:
            print("Keyboard is not powered on.")

class Monitor(OutputDevice):
    def __init__(self):
        super().__init__()

    def display(self, text):
        if self.powered_on:
            print(text)
        else:
            print("Monitor is not powered on.")

# Membuat objek keyboard dan monitor
keyboard = Keyboard()
monitor = Monitor()

# Menghidupkan keyboard dan monitor
keyboard.power_on()
monitor.power_on()
```



```
# Mengetikkan teks dengan keyboard
```

```
keyboard.type("Hello World")
```

```
# Menampilkan teks di monitor
```

```
monitor.display("Welcome to the world of Python")
```

```
# Mematikan keyboard dan monitor
```

```
keyboard.power_off()
```

```
monitor.power_off()
```

```
print("\n|  RIFKI PRAMAYANDI MAHESA  |")
```

```
print("|          210511156          |")
```

```
print("|          KELAS D          | \n")
```

7) MultiLevel-1.py

```
class Food:
    def __init__(self, name):
        self.name = name

class Ingredient(Food):
    def __init__(self, name, quantity):
        super().__init__(name)
        self.quantity = quantity

class Spice(Ingredient):
    def __init__(self, name, quantity, is_hot):
        super().__init__(name, quantity)
        self.is_hot = is_hot

# membuat objek Spice
cumin = Spice("Cumin", 50, True)

# mencetak atribut dari objek Spice
print(cumin.name)
print(cumin.quantity)
print(cumin.is_hot)

print("\n|  RIFKI PRAMAYANDI MAHESA  |")
print("|          210511156          |")
print("|          KELAS D              |\n")
```

8) MultiLevel-2.py

```
class Computer:
    def __init__(self, brand, model):
        self.brand = brand
        self.model = model

class Processor(Computer):
    def __init__(self, brand, model, cores):
        super().__init__(brand, model)
        self.cores = cores

class CPU(Processor):
    def __init__(self, brand, model, cores, frequency):
        super().__init__(brand, model, cores)
        self.frequency = frequency

    def print_specs(self):
        print(f"Brand: {self.brand}\nModel: {self.model}\nCores: {self.cores}\nFrequency: {self.frequency}")

# membuat objek CPU
my_cpu = CPU("Intel", "i7-10700K", 8, "3.8 GHz")

# mencetak spesifikasi dari objek CPU
my_cpu.print_specs()

print("\n| RIFKI PRAMAYANDI MAHESA |")
print("| 210511156 |")
print("| KELAS D |")
```

9) Hybrid-1.py

```
# membuat class Configurate
class Configurate:
    def __init__(self, name, processor):
        self.name = name
        self.processor = processor

    def get_name(self):
        return self.name

    def get_processor(self):
        return self.processor

# membuat class Computing
class Computing:
    def __init__(self, ram, storage):
        self.ram = ram
        self.storage = storage

    def get_ram(self):
        return self.ram

    def get_storage(self):
        return self.storage

# membuat class Configurate Computing dengan menggunakan multiple inheritance
class ConfigurateComputing(Configurate, Computing):
    def __init__(self, name, processor, ram, storage, graphics_card):
        Configurate.__init__(self, name, processor)
        Computing.__init__(self, ram, storage)
        self.graphics_card = graphics_card

    def get_graphics_card(self):
        return self.graphics_card

# membuat objek dari class ConfigurateComputing
pc1 = ConfigurateComputing("Laptop", "Intel Core i5", "8 GB", "256 GB",
"Nvidia GTX 1650")

# memanggil method yang dimiliki oleh class Configurate
print(pc1.get_name())      # Output: Laptop
print(pc1.get_processor()) # Output: Intel Core i5

# memanggil method yang dimiliki oleh class Computing
print(pc1.get_ram())       # Output: 8 GB
print(pc1.get_storage())   # Output: 256 GB
```

```
# memanggil method yang dimiliki oleh class ConfigureComputing  
print(pc1.get_graphics_card()) # Output: Nvidia GTX 1650
```

```
print("\n|  RIFKI PRAMAYANDI MAHESA  |")  
print("|          210511156          |")  
print("|          KELAS D          | \n")
```

10) Hybrid-2.py

```
# membuat class Kernel
class Kernel:
    def __init__(self, name, version):
        self.name = name
        self.version = version

    def get_name(self):
        return self.name

    def get_version(self):
        return self.version

# membuat class User Interface
class UserInterface:
    def __init__(self, display_type, theme):
        self.display_type = display_type
        self.theme = theme

    def get_display_type(self):
        return self.display_type

    def get_theme(self):
        return self.theme

# membuat class Operating System dengan menggunakan multiple inheritance
class OperatingSystem(Kernel, UserInterface):
    def __init__(self, name, version, display_type, theme, default_browser):
        Kernel.__init__(self, name, version)
        UserInterface.__init__(self, display_type, theme)
        self.default_browser = default_browser

    def get_default_browser(self):
        return self.default_browser

# membuat objek dari class OperatingSystem
os1 = OperatingSystem("Windows", "10", "GUI", "Light", "Chrome")

# memanggil method yang dimiliki oleh class Kernel
print("Name OS :", os1.get_name())      # Output: Windows
print("Version :", os1.get_version())    # Output: 10

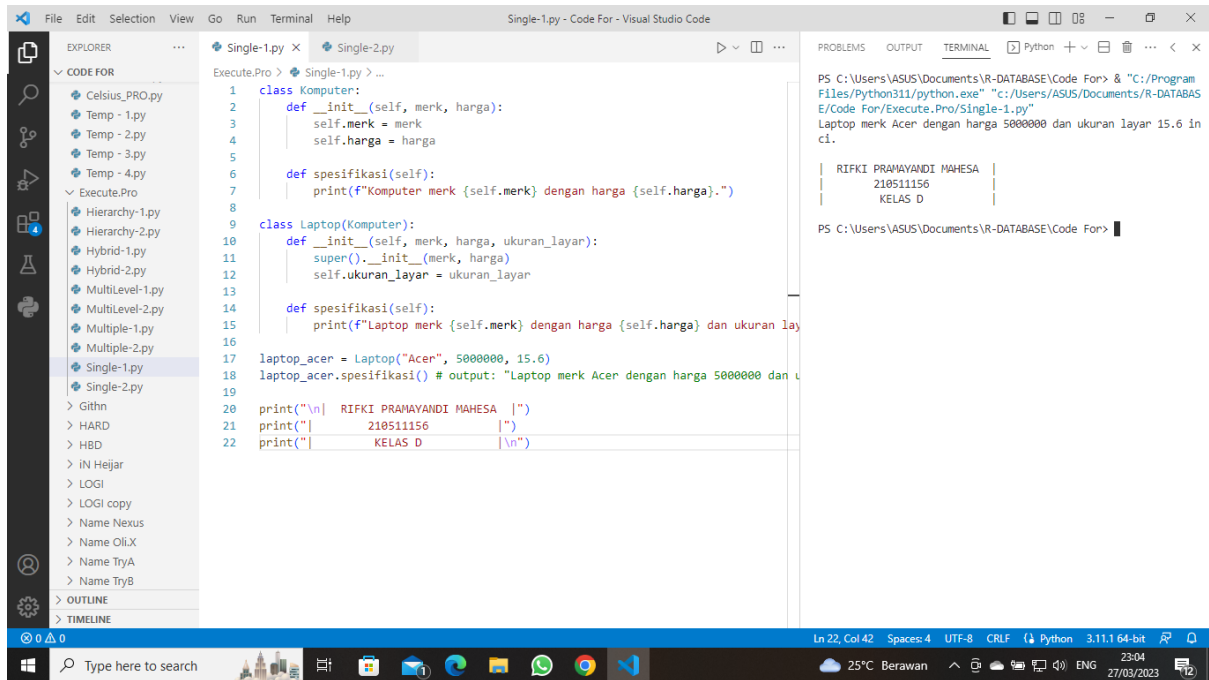
# memanggil method yang dimiliki oleh class UserInterface
print("Display Type :", os1.get_display_type())  # Output: GUI
print("Theme :", os1.get_theme())               # Output: Light

# memanggil method yang dimiliki oleh class OperatingSystem
print("Default Browser :", os1.get_default_browser())  # Output: Chrome
```

```
print("\n|  RIFKI PRAMAYANDI MAHESA  |")
print("|          210511156          |")
print("|          KELAS D          | \n")
```

b. Hasil Pemrograman :

1) Single-1.py



```
File Edit Selection View Go Run Terminal Help Single-1.py - Code For - Visual Studio Code

EXPLORER
CODE FOR
Celsius_PRO.py
Temp - 1.py
Temp - 2.py
Temp - 3.py
Temp - 4.py
Execute.Pro
Hierarchy-1.py
Hierarchy-2.py
Hybrid-1.py
Hybrid-2.py
MultiLevel-1.py
MultiLevel-2.py
Multiple-1.py
Multiple-2.py
Single-1.py
Single-2.py
Githn
HARD
HBD
iN Hejar
LOGI
LOGI copy
Name Nexus
Name OliX
Name TryA
Name TryB
OUTLINE
TIMELINE

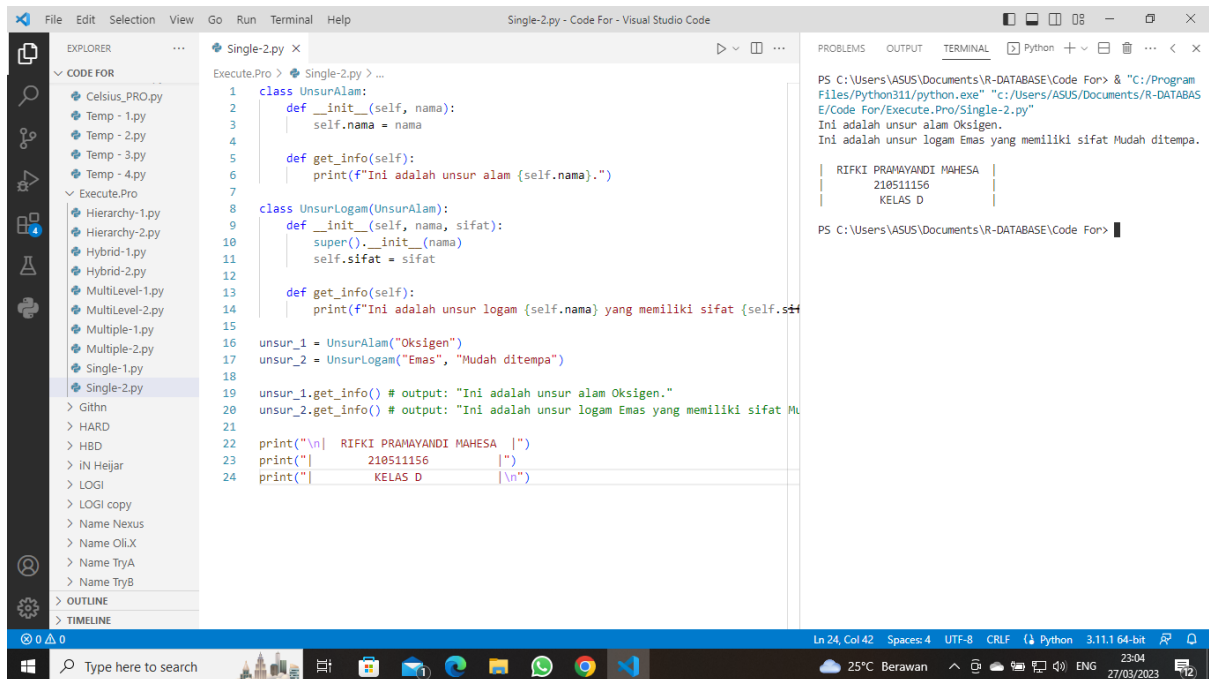
Single-1.py
1 class Komputer:
2     def __init__(self, merk, harga):
3         self.merk = merk
4         self.harga = harga
5
6     def spesifikasi(self):
7         print(f"Komputer merk {self.merk} dengan harga {self.harga}.")
8
9 class Laptop(Komputer):
10     def __init__(self, merk, harga, ukuran_layar):
11         super().__init__(merk, harga)
12         self.ukuran_layar = ukuran_layar
13
14     def spesifikasi(self):
15         print(f"Laptop merk {self.merk} dengan harga {self.harga} dan ukuran lay
16
17 laptop_acer = Laptop("Acer", 5000000, 15.6)
18 laptop_acer.spesifikasi() # output: "Laptop merk Acer dengan harga 5000000 dan u
19
20 print("\n| RIFKI PRAMAYANDI MAHESA |")
21 print("| 210511156 |")
22 print("| KELAS D |")

PROBLEMS OUTPUT TERMINAL Python
PS C:\Users\ASUS\Documents\R-DATABASE\Code For> & "C:/Program Files/Python311/python.exe" "c:/Users/ASUS/Documents/R-DATABAS E/Code For/Execute.Pro/Single-1.py"
Laptop merk Acer dengan harga 5000000 dan ukuran layar 15.6 in
ci.

| RIFKI PRAMAYANDI MAHESA |
| 210511156 |
| KELAS D |

PS C:\Users\ASUS\Documents\R-DATABASE\Code For>
```

2) Single-2.py



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

EXPLORER
CODE FOR
Celsius_PRO.py
Temp - 1.py
Temp - 2.py
Temp - 3.py
Temp - 4.py
Execute.Pro
Hierarchy-1.py
Hierarchy-2.py
Hybrid-1.py
Hybrid-2.py
MultiLevel-1.py
MultiLevel-2.py
Multiple-1.py
Multiple-2.py
Single-1.py
Single-2.py
Githn
HARD
HBD
iN Hejar
LOGI
LOGI copy
Name Nexus
Name OliX
Name TryA
Name TryB
OUTLINE
TIMELINE

Single-2.py
1 class UnsurAlam:
2     def __init__(self, nama):
3         self.nama = nama
4
5     def get_info(self):
6         print(f"Ini adalah unsur alam {self.nama}.")
7
8 class UnsurLogam(UnsurAlam):
9     def __init__(self, nama, sifat):
10         super().__init__(nama)
11         self.sifat = sifat
12
13     def get_info(self):
14         print(f"Ini adalah unsur logam {self.nama} yang memiliki sifat {self.sifat}."
15
16 unsur_1 = UnsurAlam("Oksigen")
17 unsur_2 = UnsurLogam("Emas", "Mudah ditempa")
18
19 unsur_1.get_info() # output: "Ini adalah unsur alam Oksigen."
20 unsur_2.get_info() # output: "Ini adalah unsur logam Emas yang memiliki sifat Mu
21
22 print("\n| RIFKI PRAMAYANDI MAHESA |")
23 print("| 210511156 |")
24 print("| KELAS D |")

PROBLEMS OUTPUT TERMINAL Python
PS C:\Users\ASUS\Documents\R-DATABASE\Code For> & "C:/Program Files/Python311/python.exe" "c:/Users/ASUS/Documents/R-DATABAS E/Code For/Execute.Pro/Single-2.py"
Ini adalah unsur alam Oksigen.
Ini adalah unsur logam Emas yang memiliki sifat Mudah ditempa.

| RIFKI PRAMAYANDI MAHESA |
| 210511156 |
| KELAS D |

PS C:\Users\ASUS\Documents\R-DATABASE\Code For>
```

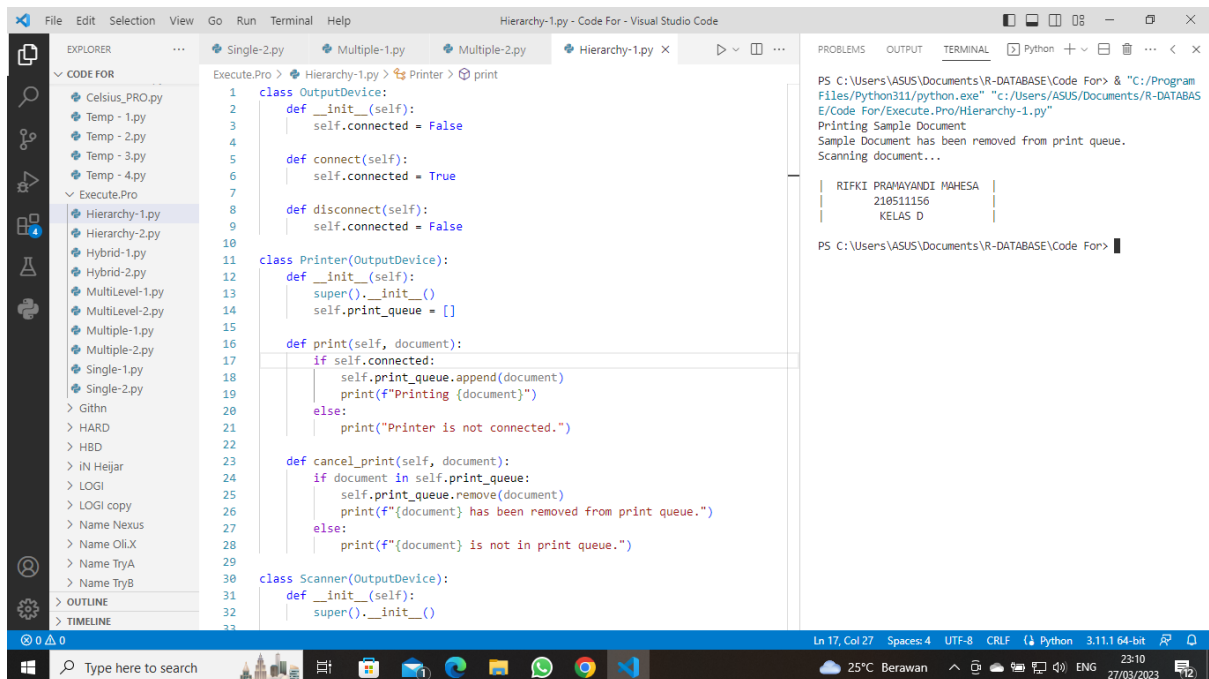

3) Multiple-1.py

```
1 class Switchable:
2     def switch_on(self):
3         self.is_on = True
4         print("Switched on")
5
6     def switch_off(self):
7         self.is_on = False
8         print("Switched off")
9
10
11 class Dimmable:
12     def set_brightness(self, brightness):
13         self.brightness = brightness
14         print("Brightness set to", brightness)
15
16
17 class Lampu(Switchable, Dimmable):
18     def __init__(self):
19         self.is_on = False
20         self.brightness = 0
21
22     lampu1 = Lampu()
23     lampu1.switch_on()
24     lampu1.set_brightness(50)
25     print("Brightness:", lampu1.brightness)
26
27     lampu1.switch_off()
28     print("Is on:", lampu1.is_on)
29
30     print("\n | RIFKI PRAMAYANDI MAHESA |")
31     print(" | 210511156 |")
32     print(" | KELAS D |")
```

4) Multiple-2.py

```
1 class WebPage:
2     def __init__(self, url):
3         self.url = url
4
5     def open(self):
6         print("Opening webpage:", self.url)
7
8
9 class OperatingSystem:
10     def __init__(self, name):
11         self.name = name
12
13     def boot_up(self):
14         print(self.name, "is booting up")
15
16
17 class Browser(WebPage, OperatingSystem):
18     def __init__(self, url, name):
19         WebPage.__init__(self, url)
20         OperatingSystem.__init__(self, name)
21
22     def display(self):
23         self.boot_up()
24         self.open()
25         print("Browser is ready to use")
26
27
28 # Membuat objek dari kelas Browser
29 my_browser = Browser("https://www.google.com", "Windows 10")
30
31 # Mengganti metode display() pada objek my_browser
32 my_browser.display()
```

5) Hierarchy-1.py



The screenshot shows the Visual Studio Code editor with the file 'Hierarchy-1.py' open. The file contains the following Python code:

```
1 class OutputDevice:
2     def __init__(self):
3         self.connected = False
4
5     def connect(self):
6         self.connected = True
7
8     def disconnect(self):
9         self.connected = False
10
11 class Printer(OutputDevice):
12     def __init__(self):
13         super().__init__()
14         self.print_queue = []
15
16     def print(self, document):
17         if self.connected:
18             self.print_queue.append(document)
19             print(f"Printing {document}")
20         else:
21             print("Printer is not connected.")
22
23     def cancel_print(self, document):
24         if document in self.print_queue:
25             self.print_queue.remove(document)
26             print(f"{document} has been removed from print queue.")
27         else:
28             print(f"{document} is not in print queue.")
29
30 class Scanner(OutputDevice):
31     def __init__(self):
32         super().__init__()
```

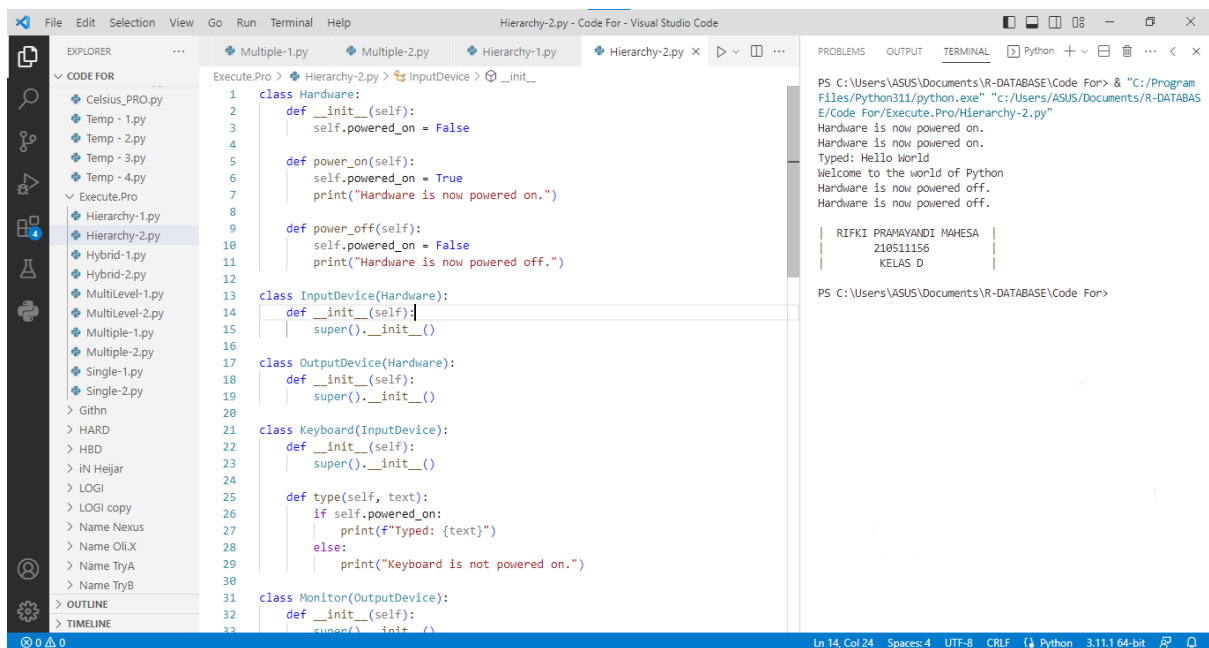
The terminal output shows the execution of the code:

```
PS C:\Users\ASUS\Documents\R-DATABASE\Code For> & "C:/Program Files/Python311/python.exe" "c:/Users/ASUS/Documents/R-DATABASE E/Code For/Execute.Pro/Hierarchy-1.py"
Printing Sample Document
Sample Document has been removed from print queue.
Scanning document...

| RIFKI PRAMAYANDI MAHESA |
| 210511156               |
| KELAS D                 |

PS C:\Users\ASUS\Documents\R-DATABASE\Code For>
```

6) Hierarchy-2.py



The screenshot shows the Visual Studio Code editor with the file 'Hierarchy-2.py' open. The file contains the following Python code:

```
1 class Hardware:
2     def __init__(self):
3         self.powered_on = False
4
5     def power_on(self):
6         self.powered_on = True
7         print("Hardware is now powered on.")
8
9     def power_off(self):
10        self.powered_on = False
11        print("Hardware is now powered off.")
12
13 class InputDevice(Hardware):
14     def __init__(self):
15         super().__init__()
16
17 class OutputDevice(Hardware):
18     def __init__(self):
19         super().__init__()
20
21 class Keyboard(InputDevice):
22     def __init__(self):
23         super().__init__()
24
25     def type(self, text):
26         if self.powered_on:
27             print(f"Typed: {text}")
28         else:
29             print("Keyboard is not powered on.")
30
31 class Monitor(OutputDevice):
32     def __init__(self):
33         super().__init__()
```

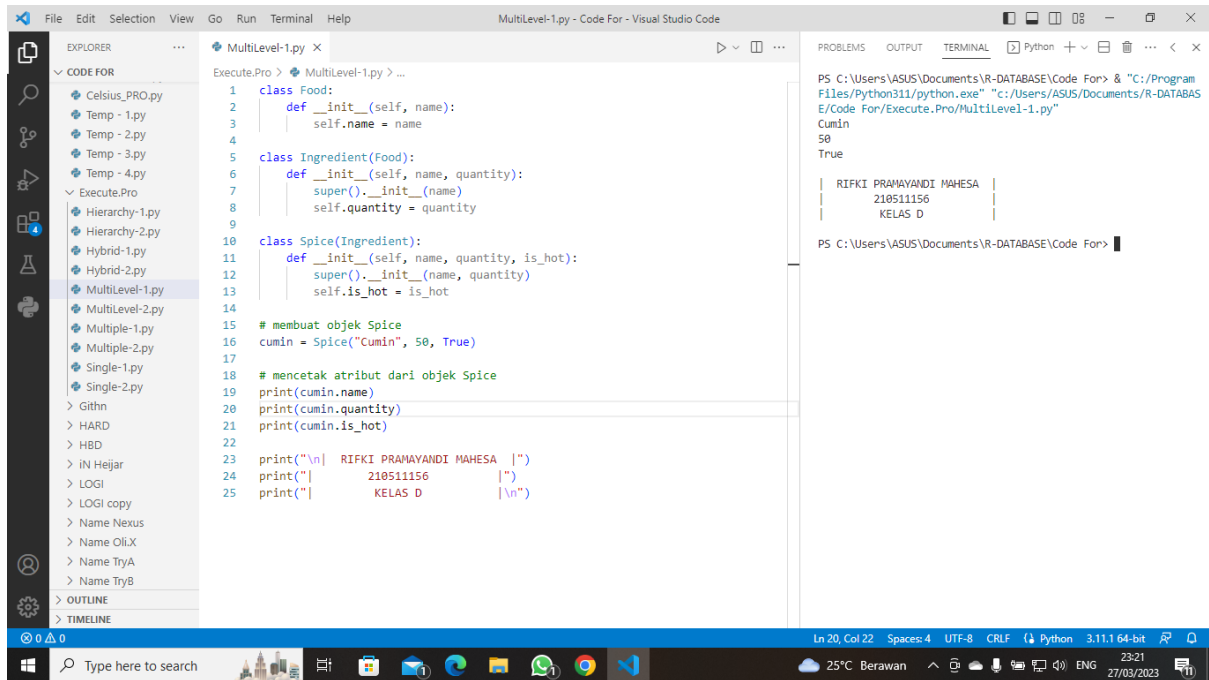
The terminal output shows the execution of the code:

```
PS C:\Users\ASUS\Documents\R-DATABASE\Code For> & "C:/Program Files/Python311/python.exe" "c:/Users/ASUS/Documents/R-DATABASE E/Code For/Execute.Pro/Hierarchy-2.py"
Hardware is now powered on.
Hardware is now powered on.
Typed: Hello World
Welcome to the world of Python
Hardware is now powered off.
Hardware is now powered off.

| RIFKI PRAMAYANDI MAHESA |
| 210511156               |
| KELAS D                 |

PS C:\Users\ASUS\Documents\R-DATABASE\Code For>
```

7) MultiLevel-1.py



```
File Edit Selection View Go Run Terminal Help
MultiLevel-1.py - Code For - Visual Studio Code

EXPLORER
CODE FOR
  Celsius_PRO.py
  Temp - 1.py
  Temp - 2.py
  Temp - 3.py
  Temp - 4.py
  Execute.Pro
    Hierarchy-1.py
    Hierarchy-2.py
    Hybrid-1.py
    Hybrid-2.py
    MultiLevel-1.py
    MultiLevel-2.py
    Multiple-1.py
    Multiple-2.py
    Single-1.py
    Single-2.py
  > Github
  > HARD
  > HBD
  > iN Hejar
  > LOGI
  > LOGI copy
  > Name Nexus
  > Name OliX
  > Name TryA
  > Name TryB
  > OUTLINE
  > TIMELINE

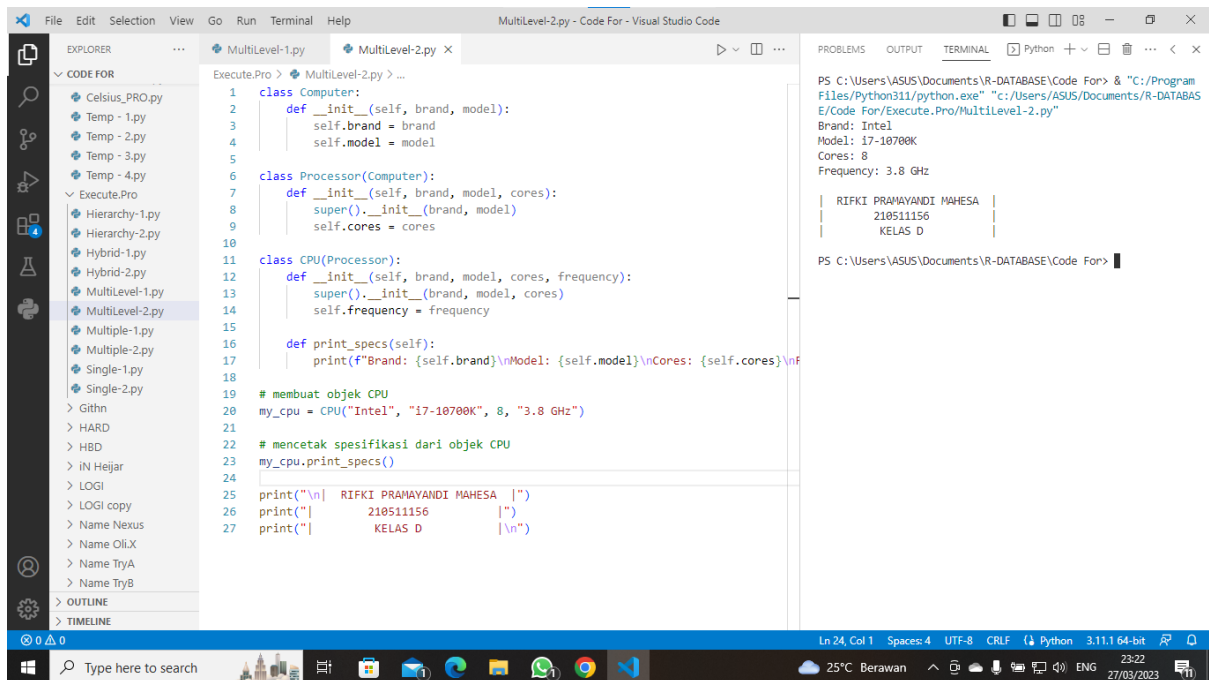
MultiLevel-1.py
1 class Food:
2     def __init__(self, name):
3         self.name = name
4
5 class Ingredient(Food):
6     def __init__(self, name, quantity):
7         super().__init__(name)
8         self.quantity = quantity
9
10 class Spice(Ingredient):
11     def __init__(self, name, quantity, is_hot):
12         super().__init__(name, quantity)
13         self.is_hot = is_hot
14
15 # membuat objek Spice
16 cumin = Spice("Cumin", 50, True)
17
18 # mencetak atribut dari objek Spice
19 print(cumin.name)
20 print(cumin.quantity)
21 print(cumin.is_hot)
22
23 print("\n  RIFKI PRAMAYANDI MAHESA  |")
24 print("|          210511156             |")
25 print("|          KELAS D                 |")

TERMINAL
PS C:\Users\ASUS\Documents\R-DATABASE\Code For> & "C:/Program Files/Python311/python.exe" "c:/Users/ASUS/Documents/R-DATABASE/Code For/Execute.Pro/MultiLevel-1.py"
Cumin
50
True

RIFKI PRAMAYANDI MAHESA |
210511156               |
KELAS D                 |

PS C:\Users\ASUS\Documents\R-DATABASE\Code For>
```

8) MultiLevel-2.py



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

EXPLORER
CODE FOR
  Celsius_PRO.py
  Temp - 1.py
  Temp - 2.py
  Temp - 3.py
  Temp - 4.py
  Execute.Pro
    Hierarchy-1.py
    Hierarchy-2.py
    Hybrid-1.py
    Hybrid-2.py
    MultiLevel-1.py
    MultiLevel-2.py
    Multiple-1.py
    Multiple-2.py
    Single-1.py
    Single-2.py
  > Github
  > HARD
  > HBD
  > iN Hejar
  > LOGI
  > LOGI copy
  > Name Nexus
  > Name OliX
  > Name TryA
  > Name TryB
  > OUTLINE
  > TIMELINE

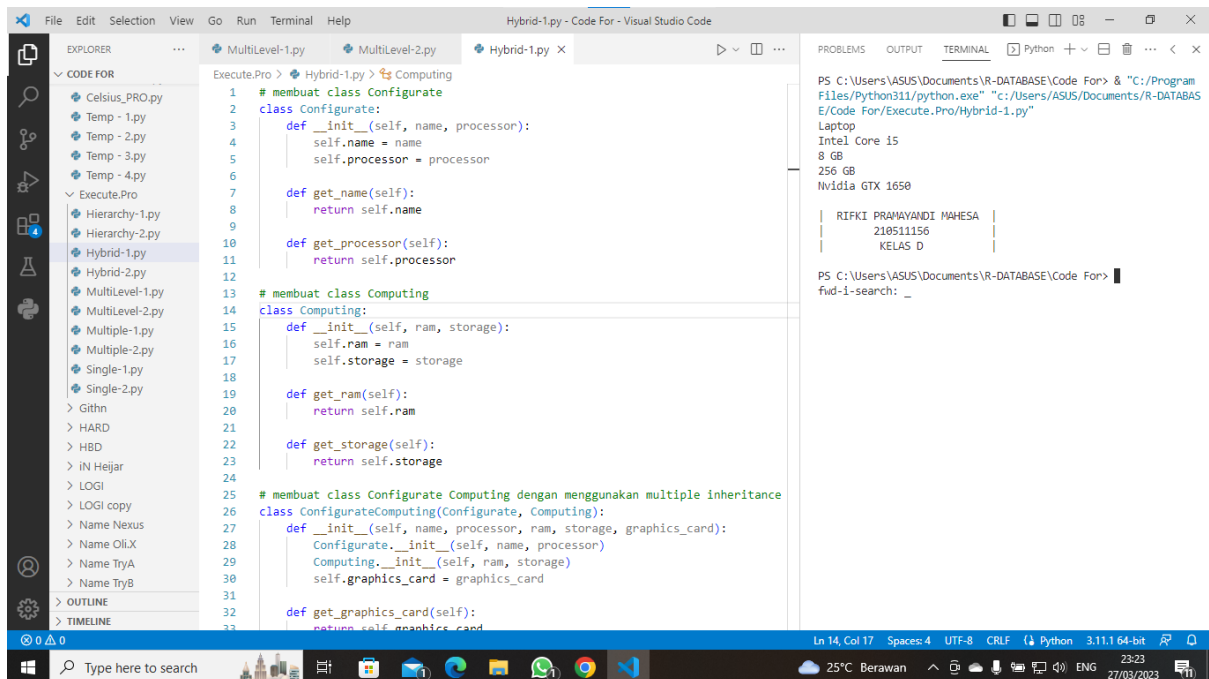
MultiLevel-2.py
1 class Computer:
2     def __init__(self, brand, model):
3         self.brand = brand
4         self.model = model
5
6 class Processor(Computer):
7     def __init__(self, brand, model, cores):
8         super().__init__(brand, model)
9         self.cores = cores
10
11 class CPU(Processor):
12     def __init__(self, brand, model, cores, frequency):
13         super().__init__(brand, model, cores)
14         self.frequency = frequency
15
16     def print_specs(self):
17         print(f"Brand: {self.brand}\nModel: {self.model}\nCores: {self.cores}\nFrequency: {self.frequency}")
18
19 # membuat objek CPU
20 my_cpu = CPU("Intel", "i7-10700K", 8, "3.8 GHz")
21
22 # mencetak spesifikasi dari objek CPU
23 my_cpu.print_specs()
24
25 print("\n  RIFKI PRAMAYANDI MAHESA  |")
26 print("|          210511156             |")
27 print("|          KELAS D                 |")

TERMINAL
PS C:\Users\ASUS\Documents\R-DATABASE\Code For> & "C:/Program Files/Python311/python.exe" "c:/Users/ASUS/Documents/R-DATABASE/Code For/Execute.Pro/MultiLevel-2.py"
Brand: Intel
Model: i7-10700K
Cores: 8
Frequency: 3.8 GHz

RIFKI PRAMAYANDI MAHESA |
210511156               |
KELAS D                 |

PS C:\Users\ASUS\Documents\R-DATABASE\Code For>
```

9) Hybrid-1.py



The screenshot shows the Visual Studio Code editor with the file 'Hybrid-1.py' open. The code defines three classes: `Configure`, `Computing`, and `ConfigureComputing` (which inherits from both `Configure` and `Computing`). The `Configure` class has attributes `name` and `processor`. The `Computing` class has attributes `ram` and `storage`. The `ConfigureComputing` class has attributes `name`, `processor`, `ram`, `storage`, and `graphics_card`. The terminal output shows the execution of the script, displaying system information and the results of the `find-1-search` command.

```
1 # membuat class Configure
2 class Configure:
3     def __init__(self, name, processor):
4         self.name = name
5         self.processor = processor
6
7     def get_name(self):
8         return self.name
9
10    def get_processor(self):
11        return self.processor
12
13 # membuat class Computing
14 class Computing:
15     def __init__(self, ram, storage):
16         self.ram = ram
17         self.storage = storage
18
19     def get_ram(self):
20         return self.ram
21
22     def get_storage(self):
23         return self.storage
24
25 # membuat class Configure Computing dengan menggunakan multiple inheritance
26 class ConfigureComputing(Configure, Computing):
27     def __init__(self, name, processor, ram, storage, graphics_card):
28         Configure.__init__(self, name, processor)
29         Computing.__init__(self, ram, storage)
30         self.graphics_card = graphics_card
31
32     def get_graphics_card(self):
33         return self.graphics_card
```

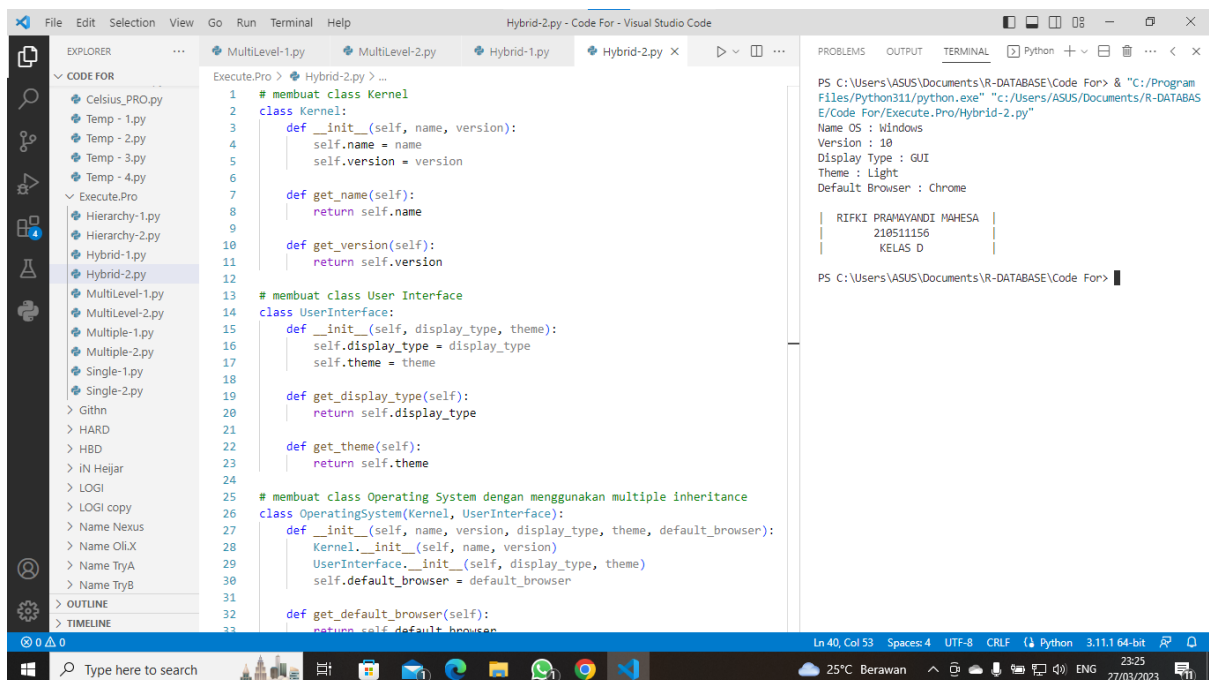
PS C:\Users\ASUS\Documents\R-DATABASE\Code For> & "C:/Program Files/Python311/python.exe" "c:/Users/ASUS/Documents/R-DATABASE/Code For/Execute.Pro/Hybrid-1.py"

Laptop
Intel Core i5
8 GB
256 GB
Nvidia GTX 1650

| RIFKI PRAMAYANDI MAHESA |
| 210511156 |
| KELAS D |

PS C:\Users\ASUS\Documents\R-DATABASE\Code For> find-1-search: _

10) Hybrid-2.py



The screenshot shows the Visual Studio Code editor with the file 'Hybrid-2.py' open. The code defines three classes: `Kernel`, `UserInterface`, and `OperatingSystem` (which inherits from both `Kernel` and `UserInterface`). The `Kernel` class has attributes `name` and `version`. The `UserInterface` class has attributes `display_type` and `theme`. The `OperatingSystem` class has attributes `name`, `version`, `display_type`, `theme`, and `default_browser`. The terminal output shows the execution of the script, displaying system information and the results of the `find-1-search` command.

```
1 # membuat class Kernel
2 class Kernel:
3     def __init__(self, name, version):
4         self.name = name
5         self.version = version
6
7     def get_name(self):
8         return self.name
9
10    def get_version(self):
11        return self.version
12
13 # membuat class User Interface
14 class UserInterface:
15     def __init__(self, display_type, theme):
16         self.display_type = display_type
17         self.theme = theme
18
19     def get_display_type(self):
20         return self.display_type
21
22     def get_theme(self):
23         return self.theme
24
25 # membuat class Operating System dengan menggunakan multiple inheritance
26 class OperatingSystem(Kernel, UserInterface):
27     def __init__(self, name, version, display_type, theme, default_browser):
28         Kernel.__init__(self, name, version)
29         UserInterface.__init__(self, display_type, theme)
30         self.default_browser = default_browser
31
32     def get_default_browser(self):
33         return self.default_browser
```

PS C:\Users\ASUS\Documents\R-DATABASE\Code For> & "C:/Program Files/Python311/python.exe" "c:/Users/ASUS/Documents/R-DATABASE/Code For/Execute.Pro/Hybrid-2.py"

Name OS : Windows
Version : 10
Display Type : GUI
Theme : Light
Default Browser : Chrome

| RIFKI PRAMAYANDI MAHESA |
| 210511156 |
| KELAS D |

PS C:\Users\ASUS\Documents\R-DATABASE\Code For>