

**DOKUMENTASI MATERI NETDEV  
PYTHON KLOTER 3**



**Adaptive Network**  
Laboratory

Disusun oleh :

Ega Fauzan Sani  
M. Ikhsan  
Alfira Triazalma

**ADAPTIVE NETWORK LABORATORY  
FAKULTAS TEKNIK ELEKTRO  
UNIVERSITAS TELKOM BANDUNG  
2023**

# DAFTAR ISI

<b>DAFTAR ISI.....</b>	<b>ii</b>
<b>DASAR TEORI.....</b>	<b>2</b>
1. Definisi .....	2
2. Keunggulan.....	2
3. Tipe data .....	2
4. Fundamental Python .....	3
5. Penamaan Variabel.....	4
6. Fungsi.....	5
7. Object Oriented Programming ( OOP ).....	6
<b>SOFTWARE, TOOLS DAN INSTALASI.....</b>	<b>8</b>
1. Perangkat lunak (Software) dan tools.....	8
2. Melakukan Instalasi dan Membuat Repository di GitHub .....	8
<b>DEMO DAN PERCOBAAN PYTHON .....</b>	<b>12</b>
<b>DAFTAR PUSTAKA .....</b>	<b>19</b>

# DASAR TEORI

## 1. Definisi

Python adalah bahasa pemrograman multifungsi yang ditafsirkan dengan filosofi desain yang berpusat pada keterbacaan kode. Python dikembangkan oleh Guido Van Rossum pada tahun 1990 di CWI, Amsterdam, sebagai kelanjutan dari bahasa pemrograman ABC (Ranking Bahasa Pemrograman RedMonk, Juni 2015) Python diklasifikasikan sebagai bahasa skrip, jadi Python bukan bahasa yang dikompilasi untuk menghasilkan kode yang dapat dibaca mesin, dan kodenya terkait dengan program lain sebagai proses kontrol. Python sebagai rutin kontrol berarti bahwa Python dapat bertindak sebagai penghubung antara program lain. Sebagai contoh, Python sering digunakan sebagai bahasa scripting untuk game, dengan sebagian besar pekerjaan dilakukan oleh modul yang sudah dikompilasi (Jackson, 2009)

## 2. Keunggulan

Salah satu keunggulan bahasa pemrograman ini adalah kepastakaan yang luas. Python memiliki modul-modul yang dapat digunakan untuk berbagai keperluan. Python memiliki lebih dari 140.000 kepastakaan yang dikembangkan dari opensource project. Bahasa pemrograman python juga dapat berorientasi objek serta memiliki modul modul yang dapat dikembangkan menggunakan bahasa C/C++. Selain itu, python juga mudah dipahami sehingga membuat bahasa ini semakin populer untuk digunakan. Program python dapat ditulis pada text editor biasa seperti notepad atau notepad++. Atau dapat juga ditulis menggunakan Integrated Develop Environment (IDE) baik secara online maupun diinstall langsung ke sistem operasi. IDE merupakan suatu software aplikasi, yang biasanya berbasis GUI, yang digunakan sebagai tempat menulis baris-baris kode pemrograman. IDE dinilai lebih layak untuk menulis kode bahasa pemrograman karena menyediakan berbagai fitur seperti text editor, debugger, compiler dsb. Beberapa contoh IDE adalah seperti Visual Studio Code, Atom dan Sublime Text.

## 3. Tipe data

### ➤ Integer

Tipe data integer adalah tipe data numerik yang menampung bilangan bulat. Contohnya bilangan 1,2,3 dan seterusnya. Sehingga setiap variabel yang memiliki nilai bilangan bulat, maka ia akan dikategorikan sebagai integer. Dalam bahasa Python, panjang dari data integer dibatasi oleh besarnya memori yang tersedia. Berbeda dengan tipe data float yang panjangnya mencapai 17 angka dibelakang koma. Berikut ini adalah contoh dari tipe data integer:

```
#tipe data integer
```

```
panjang = 10
```

```
print(panjang)
```

### ➤ Float

Hampir sama dengan tipe data integer, hanya saja tipe data float dipergunakan untuk variabel-variabel yang memiliki nilai pecahan / desimal. Tipe data float juga termasuk ke dalam tipe data numerik karena jenis data ini menyimpan bilangan pecahan atau disebut

juga dengan bilangan real. Pemisah dari bilangan desimal menggunakan tanda titik (.). Berikut ini adalah contoh dari tipe data float:

```
#tipe data float
```

```
lebar = 5.5
```

```
print(lebar)
```

#### ➤ Complex

Sedangkan tipe data numerik yang lainnya adalah tipe data complex, sesuai namanya, ini adalah tipe data yang kompleks. Ia merepresentasikan nilai imajiner. Berikut ini adalah contoh dari tipe data float:

```
#tipe data complex
```

```
a = 5j
```

```
b = 10j
```

```
c = a + b
```

```
print(a, '+', b, '=', c)
```

#### ➤ String

Tipe data string sering juga disebut dengan tipe data teks, tipe data ini digunakan untuk menyimpan sebuah teks. Data yang bertipe string harus diapit oleh tanda kutip, baik tanda kutip satu (") maupun tanda kutip dua (" ") setelah karakter sama dengan (=)

Berikut ini adalah contoh dari tipe data string:

```
#tipe data string
```

```
data = "DQLab 2021"
```

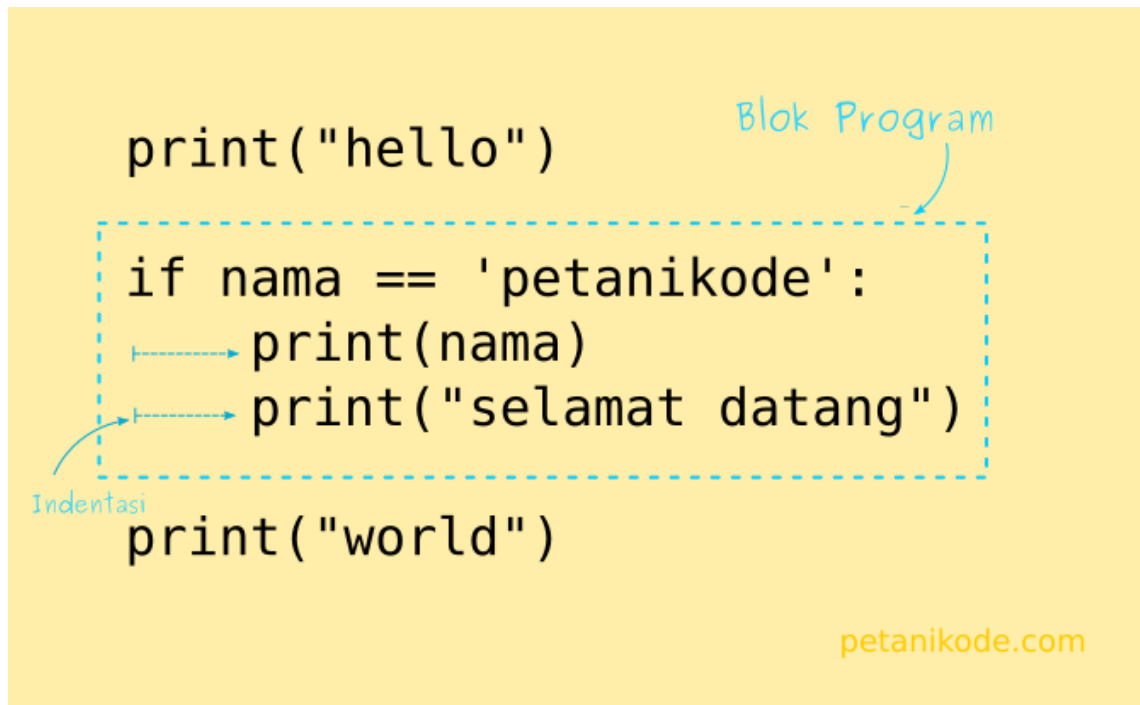
```
print(data)
```

#### ➤ Boolean

Selanjutnya adalah tipe data boolean. Tipe data boolean adalah tipe data yang paling simpel dan mudah. Tipe data boolean hanya menyimpan dua buah nilai, yaitu; True dan False. Nilai True untuk pernyataan bernilai benar, dan False untuk merepresentasikan pernyataan yang bernilai salah. Penulisan True dan False di huruf pertama harus menggunakan huruf besar dan biasanya tipe data ini digunakan ketika melakukan pengecekan oleh suatu kondisi yang menggunakan fungsi IF. Tipe data Boolean sangat penting sekali digunakan untuk membangun program/aplikasi skala besar sekalipun. Ia bisa berfungsi untuk mengontrol laju dan alur dari program yang kita bangun.

## 4. Fundamental Python

Python menggunakan indentasi untuk me-highlight sebuah baris kode. Indentasi yang digunakan python ialah spasi. Semua pernyataan dengan jarak yang sama ke kanan adalah bagian dari blok kode yang sama. Jika sebuah kode bersarang pada sebuah kode maka blok tersebut harus terindensasi lebih dalam.



Pada Python, ada 3 macam bentuk komentar, yaitu single line, inline, dan multiline.

#### 1. Single line

```
# Single line dimulai dengan tanda pagar
```

#### 2. Inline

```
print("Hello world") # inline dimulai dengan tanda pagar juga
```

#### 3. Multiline

```
"""  
Komentar multiline berada dalam kutip ganda 3 kali  
dan semua yang didalamnya tidak akan dieksekusi  
"""
```

## 5. Penamaan Variabel

Variabel merupakan tempat penyimpanan data yang bersifat mutable, artinya nilainya bisa berubah-ubah. Variabel dalam python memiliki format penulisan nama\_variabel = <nilai>. Variabel dapat berisi teks maupun bilangan. Terdapat beberapa aturan penulisan variabel, yaitu :

1. nama variabel boleh diawali menggunakan huruf atau garis bawah (\_) dan tidak dapat dimulai dengan angka (0-9), contoh: `namasaya`, `_nama`
2. karakter pada variabel bersifat sensitif, artinya huruf kapital dan huruf kecil memiliki arti yang berbeda.

3. nama variabel tidak boleh menggunakan kata kunci yang sudah ada dalam python

Berikut ini adalah contoh penggunaan variabel dalam coding python

```
1 bill1, bil2 = 3, 4
2 salam = "Selamat Pagi", penutup = "Salam Sejahtera"
```

## 6. Fungsi

Fungsi adalah blok kode yang dapat digunakan kembali yang melakukan tugas tertentu. Dalam Python, fungsi didefinisikan dengan kata kunci "def" dan diikuti oleh nama fungsi, argumen, dan kode yang harus dijalankan. Berikut adalah beberapa konsep penting yang terkait dengan fungsi di Python:

1. Mendefinisikan fungsi:

Untuk mendefinisikan fungsi, kita menggunakan kata kunci "def" diikuti oleh nama fungsi, argumen dan kode yang harus dijalankan. Berikut adalah sintaksisnya:

```
def nama_fungsi(argumen):
    # kode fungsi
    return nilai_kembalian
```

2. Argumen:

Argumen adalah nilai yang diberikan ke fungsi sebagai input. Ada dua jenis argumen: argumen posisi dan argumen kata kunci. Contoh:

```
def contoh_fungsi(arg1, arg2, arg3):
    # kode fungsi

# memanggil fungsi dengan argumen posisi
contoh_fungsi(nilai1, nilai2, nilai3)

# memanggil fungsi dengan argumen kata kunci
contoh_fungsi(arg1=nilai1, arg2=nilai2, arg3=nilai3)
```

3. Memanggil Fungsi

Untuk memanggil fungsi di Python, kita perlu mengikuti sintaksis berikut:

1. Tuliskan nama fungsi yang ingin dipanggil.
2. Berikan nilai input (argumen) jika diperlukan, di dalam kurung buka-tutup setelah nama fungsi.

3. Jika fungsi mengembalikan nilai, simpan nilai tersebut ke dalam variabel.

Berikut adalah contoh sederhana untuk memanggil fungsi bernama "greeting" dengan satu argumen "nama" yang akan menampilkan pesan sapaan:

```
def greeting(nama):  
    print("Halo, " + nama + "! Selamat datang.")  
  
# Memanggil fungsi greeting  
greeting("Budi") # Output: "Halo, Budi! Selamat datang."
```

## 7. Object Oriented Programming ( OOP )

OOP (Object-Oriented Programming) atau Pemrograman Berorientasi Objek adalah paradigma pemrograman yang berfokus pada pengorganisasian kode menjadi objek-objek yang dapat berinteraksi satu sama lain. Di Python, OOP dilakukan dengan menggunakan kelas dan objek.

1. Kelas

Kelas adalah blueprint atau cetakan untuk membuat objek. Kelas mendefinisikan atribut (variabel) dan metode (fungsi) yang dapat digunakan oleh objek yang dibuat dari kelas tersebut.

2. Objek

Objek adalah instansi dari sebuah kelas. Setiap objek memiliki atribut dan metode yang sama dengan kelasnya, namun nilai dari atribut objek dapat berbeda-beda

3. Encapsulation

Encapsulation adalah konsep yang memungkinkan atribut dan metode pada sebuah kelas dapat disembunyikan dari akses langsung dari luar kelas. Hal ini dilakukan untuk menjaga keamanan dan konsistensi data. Di Python, encapsulation dilakukan dengan menggunakan konvensi nama `_namaAtribut` dan `__namaMetode`.

4. Inheritance

Inheritance atau pewarisan adalah konsep dimana sebuah kelas dapat menurunkan atribut dan metode dari kelas induknya. Hal ini memungkinkan untuk menghindari duplikasi kode dan mempermudah perawatan kode. Di Python, inheritance dilakukan dengan menggunakan tanda kurung siku pada definisi kelas.

5. Polymorphism:

Polymorphism adalah konsep dimana sebuah objek dapat memiliki beberapa bentuk atau perilaku. Hal ini dapat dilakukan dengan menggunakan method overriding atau method overloading. Di Python, polymorphism dapat dilakukan dengan mudah karena Python adalah bahasa pemrograman yang berorientasi objek secara dinamis.

6. Abstraksi

Abstraksi adalah konsep dimana kita hanya memperhatikan atribut dan metode yang penting dalam sebuah kelas dan mengabaikan yang tidak penting. Hal ini dilakukan untuk mempermudah pemahaman dan pengembangan kode. Di Python, abstraksi dilakukan dengan menggunakan kelas abstrak dan metode abstrak.



# SOFTWARE, TOOLS DAN INSTALASI

## 1. Perangkat lunak (Software) dan tools

Perangkat lunak dan beberapa tools yang digunakan dalam percobaan kami sebagai berikut:

- a. Python 3.10 atau di atasnya,
- b. Text Editor (Visual Studio Code)
- c. Git ( opsional )
- d. Pygame

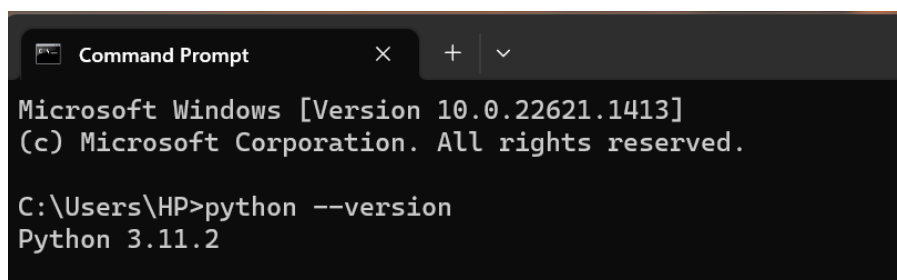
## 2. Melakukan Instalasi dan Membuat Repository di GitHub

- Melakukan Instalasi Python Versi 3.10 atau diatasnya ( python.org )



- Langkah Selanjutnya yang dilakukan adalah memastikan Python sudah terinstall pada komputer kita dengan perintah

```
python --version
```



- Selanjutnya Kita Install Pygame di Command Promp dengan command :

```
python -m pip install pygame
```

```
Command Prompt
Microsoft Windows [Version 10.0.22621.1265]
(c) Microsoft Corporation. All rights reserved.

C:\Users\HP>py --version
Python 3.11.2

C:\Users\HP>python -m pip install pygame
Collecting pygame
  Downloading pygame-2.3.0-cp311-cp311-win_amd64.whl (10.5 MB)
    10.5/10.5 MB 178.3 kB/s eta 0:00:00
Installing collected packages: pygame
Successfully installed pygame-2.3.0

[notice] A new release of pip available: 22.3.1 -> 23.0.1
[notice] To update, run: python.exe -m pip install --upgrade pip
```

```
python.exe -m pip install --upgrade pip
```

```
Command Prompt

File "C:\Users\HP\AppData\Local\Programs\Python\Python311\Lib\site-packages\pip\_vendor\urllib3\response.py", line 62
    data = self.read(amt=amt, decode_content=decode_content)
    ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
File "C:\Users\HP\AppData\Local\Programs\Python\Python311\Lib\site-packages\pip\_vendor\urllib3\response.py", line 55
    with self._error_catcher():
    ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
File "C:\Users\HP\AppData\Local\Programs\Python\Python311\Lib\contextlib.py", line 155, in __exit__
    self.gen.throw(typ, value, traceback)
    ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
File "C:\Users\HP\AppData\Local\Programs\Python\Python311\Lib\site-packages\pip\_vendor\urllib3\response.py", line 44
    atcher
    ^^^^^
    raise ReadTimeoutError(self._pool, None, "Read timed out.")
pip._vendor.urllib3.exceptions.ReadTimeoutError: HTTPSConnectionPool(host='files.pythonhosted.org', port=443): Read ti

[notice] A new release of pip available: 22.3.1 -> 23.0.1
[notice] To update, run: python.exe -m pip install --upgrade pip

C:\Users\HP>python.exe -m pip install --upgrade pip
Requirement already satisfied: pip in c:\users\hp\appdata\local\programs\python\python311\lib\site-packages (22.3.1)
Collecting pip
  Downloading pip-23.0.1-py3-none-any.whl (2.1 MB)
    2.1/2.1 MB 333.1 kB/s eta 0:00:00
Installing collected packages: pip
  Attempting uninstall: pip
    Found existing installation: pip 22.3.1
    Uninstalling pip-22.3.1:
      Successfully uninstalled pip-22.3.1
  Successfully installed pip-23.0.1

C:\Users\HP>
```

*\*Successfully installed pip-23.0.1*

## - Selanjutnya Membuat Repository baru dalam GitHub

### Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Owner \* ikhsuns / Repository name \* pythonNetdev23\_pythonkloter3palingkeren ✓

Great repository names are short and memorable. Need inspiration? How about [animated-octo-train](#)?

Description (optional)

☒ **Public**  
Anyone on the internet can see this repository. You choose who can commit.

☐ **Private**  
You choose who can see and commit to this repository.

Initialize this repository with:  
Skip this step if you're importing an existing repository.

☐ **Add a README file**  
This is where you can write a long description for your project. [Learn more.](#)

**\*Nama repository ( *pythonNetdev23\_pythonkloter3palingkeren* )**

- Membuat Projects di dalam GitHub dengan Menambahkan Anggota Kelompok dan membuat *branch* sesuai jumlah tugas dan kerjakan tugas pada *branch* yang terpisah. Misal *branch* dengan nama “Tugas 1” khusus untuk mengerjakan tugas codingan tugas 1.

**Tugas Python Kloter 3 NetDev**

View 1 View 2 + New view

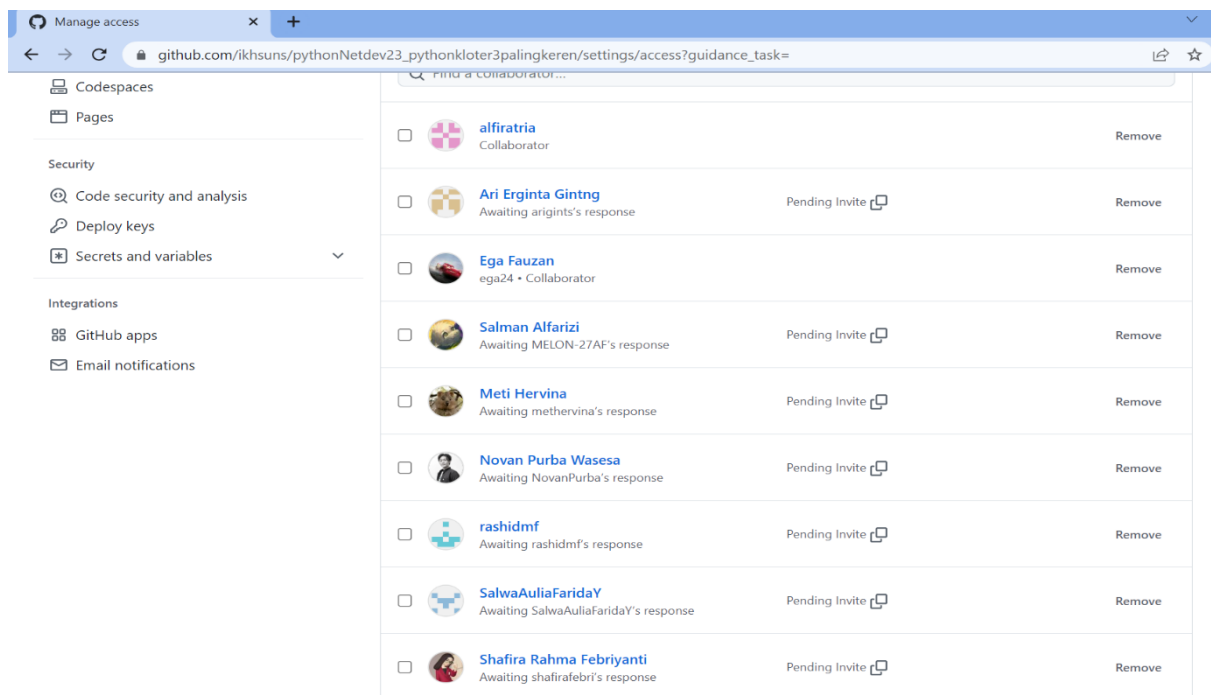
Filter by keyword or by field

<b>TASK 1</b> 2	<b>TASK 2</b> 1	<b>In Progress</b> 0	<b>Done</b> 0
<b>TUGAS ATM OOP</b>	<b>TUGAS GJB</b>	This is actively being worked on	TASK 1 & 2 COMPLETED
<div>pythonNetdev23_pythonkloter3palingkeren #2 Ega (Anggota Kelompok)</div> <div>pythonNetdev23_pythonkloter3palingkeren #3 Alfira (Anggota Kelompok)</div>	<div>pythonNetdev23_pythonkloter3palingkeren #1 Ikhsan (Anggota Kelompok)</div>		
+ Add item	+ Add item	+ Add item	+ Add item

- Menambahkan Collaborator Adaptive Network Lab Members di dalam Projects

Undang seluruh anggota kelompok dan asisten menjadi *collaborator*. Berikut *email* asisten riset yang harus diundang:

- [metihervina@student.telkomuniversity.ac.id](mailto:metihervina@student.telkomuniversity.ac.id)
- [shafira.r.febriyanti@gmail.com](mailto:shafira.r.febriyanti@gmail.com)
- [novanpurba411@gmail.com](mailto:novanpurba411@gmail.com)
- [arigints@student.telkomuniversity.ac.id](mailto:arigints@student.telkomuniversity.ac.id)
- [melonna271202@gmail.com](mailto:melonna271202@gmail.com)
- [salwa\\_aulia\\_27rpl@student.smktelkom-mlg.sch.id](mailto:salwa_aulia_27rpl@student.smktelkom-mlg.sch.id)
- [rashidfajri@gmail.com](mailto:rashidfajri@gmail.com)



# DEMO DAN PERCOBAAN PYTHON

## Software yang Diperlukan :

1. Text Editor ( Visual Studio Code )
2. Command Promp
3. Windows PowerShell

## A. TASK 1 ( ATM OOP )

### Deskripsi Tugas :

1. Menggunakan konsep OOP.
2. Anda diminta untuk membuat sebuah program sederhana yang dapat melakukan transaksi pada sebuah rekening bank.
3. Selanjutnya, Anda akan membuat sebuah objek Bank dengan nama "Robin" dan melakukan beberapa transaksi.

## B. TASK 2 ( GJB PYGAME )

### Deskripsi Tugas :

Prosedur Pengerjaan Tugas GJB

1. Buatlah visualisasi gerak jatuh bebas dengan hasil akhir visual python yang bergerak. Simulasi jatuh bebas dengan kecepatan beragam (minimal 8 kecepatan). Diketahui ketinggian awal 88 m. Tampilkan perhitungan waktu simulasi dalam visualisasi tersebut.

## C. TASK 3 ( FLOWCHART )

1. Buat *flowchart* cara kerja dari masing-masing kodingan!

## A. DEMO TASK 1 ( ATM OOP ) :

```
C: > Users > HP > Documents > Adaptive NetDev > TUGAS PYTHON KLOTER 3.py > ...
1  ## MEMBUAT PROGRAM BANK ATM OOP##
2
3  class Bank:
4
5      def __init__(self, nama):
6          self.nama = nama
7          self.saldo = 1000000
8
9      def cekSaldo(self):
10         print("Saldo Nasabah Sebesar Rp" + str(self.saldo))
11
12     def kurangSaldo(self, tunai):
13         self.saldo -= tunai
14
15     def tambahSaldo(self, tunai):
16         self.saldo += tunai
17
18     def transferSaldo(self, tunai):
19         if (self.saldo >= tunai):
20             self.kurangSaldo(tunai)
21         else :
22             print("Maaf Saldo " + self.nama + " Tidak Mencukupi")
23
24     def setorSaldo(self, tunai):
25         self.tambahSaldo(tunai)
26         print("Saldo " + self.nama + " Telah Bertambah")
27         self.cekSaldo()
28
29
```

```
29
30     orang1 = Bank("Robin")
31
32     print("Nama Nasabah adalah " + orang1.nama)
33     Bank.cekSaldo(orang1)
34
35     print("Robin Melakukan Transaksi Transfer Rp700.000 ke Rekening Lain")
36     Bank.transferSaldo(orang1, 700000)
37     Bank.cekSaldo(orang1)
38
39     print("Robin Melakukan Transaksi Setor Tunai Rp100.000 ke Rekening Lain")
40     Bank.setorSaldo(orang1, 100000)
41     Bank.cekSaldo(orang1)
42
43     print("Robin Melakukan Transaksi Transfer Rp500.000 ke Rekening Lain")
44     Bank.transferSaldo(orang1, 500000)
45     Bank.cekSaldo(orang1)
46
```

RUN PYTHON CODE / FILE :

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

PS C:\Users\HP> & C:/Users/HP/AppData/Local/Programs/Python/Python311/python.exe "c:/Users/HP/Documents/Adaptive NetDev/TUGAS PYTHON KLOTER 3.py"
Nama Nasabah adalah Robin
Saldo Nasabah Sebesar Rp1000000
Robin Melakukan Transaksi Transfer Rp700.000 ke Rekening Lain
Saldo Nasabah Sebesar Rp300000
Robin Melakukan Transaksi Setor Tunai Rp100.000 ke Rekening Lain
Saldo Robin Telah Bertambah
Saldo Nasabah Sebesar Rp400000
Saldo Nasabah Sebesar Rp400000
Robin Melakukan Transaksi Transfer Rp500.000 ke Rekening Lain
Maaf Saldo Robin Tidak Mencukupi
Saldo Nasabah Sebesar Rp400000
PS C:\Users\HP>
```

**Hasil :**

- Menampilkan saldo awal pada rekening "Robin".
- Melakukan transfer sebesar 700.000 dari rekening "Robin" ke rekening lain.
- Melakukan setoran tunai sebesar 100.000 ke rekening "Robin".
- Melakukan transfer sebesar 500.000 dari rekening "Robin" ke rekening lain.

## B. DEMO TASK 2 ( GJB PYGAME ) :

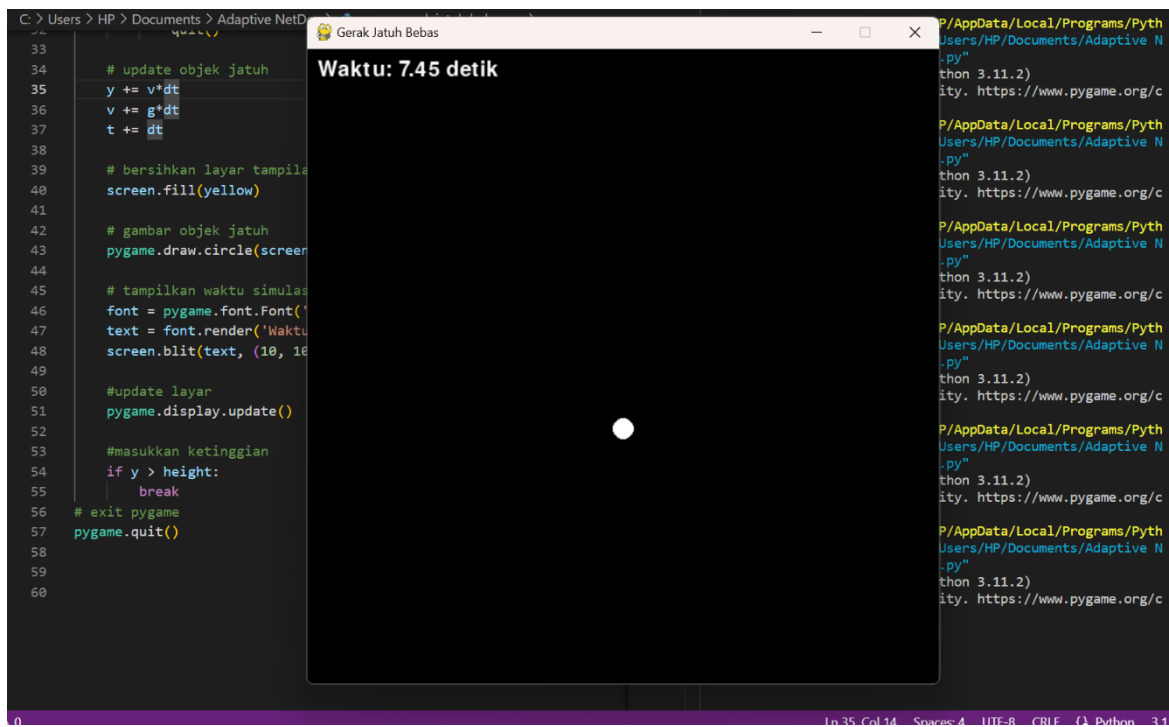
C: > Users > HP > Documents > Adaptive NetDev > game gerak jatuh bebas.py > ...

```
1  #TUGAS GJB#
2  #import pygame dan math
3  import pygame
4  import math
5
6  #inisialisasi pygame nya dulu, jika belum install pygame, pergi ke cmd python -m pip install pygame
7  pygame.init()
8
9
10 #konfigurasi tampilan layar game
11 width, height = 600, 600
12 screen = pygame.display.set_mode((width, height))
13 pygame.display.set_caption("Gerak Jatuh Bebas")
14
15 #masukkan format warna
16 pink = (255, 255, 255)
17 yellow = (0, 0, 0)
18
19 #konfigurasi objek gerak jatuh bebas
20 x, y = 300, 88 # ketinggian awal
21 v = 0 # kecepatan awal
22 g = 9.8 # gravitasi
23 t = 0 # waktu awal
24 dt = 0.01 # delta waktu
25
```

```
26 #loop 1
27 while True:
28     #handle event
29     for event in pygame.event.get():
30         if event.type == pygame.QUIT:
31             pygame.quit()
32             quit()
33
34     # update objek jatuh
35     y += v*dt
36     v += g*dt
37     t += dt
38
39     # bersihkan layar tampilan
40     screen.fill(yellow)
41
42     # gambar objek jatuh
43     pygame.draw.circle(screen, pink, (int(x), int(y)), 10)
44
45     # tampilkan waktu simulasi
46     font = pygame.font.Font('freesansbold.ttf', 20)
47     text = font.render('Waktu: {:.2f} detik'.format(t), True, pink)
48     screen.blit(text, (10, 10))
49
50     #update layar
51     pygame.display.update()
52
53     #masukkan ketinggian
54     if y > height:
55         break
56 # exit pygame
57 pygame.quit()
```



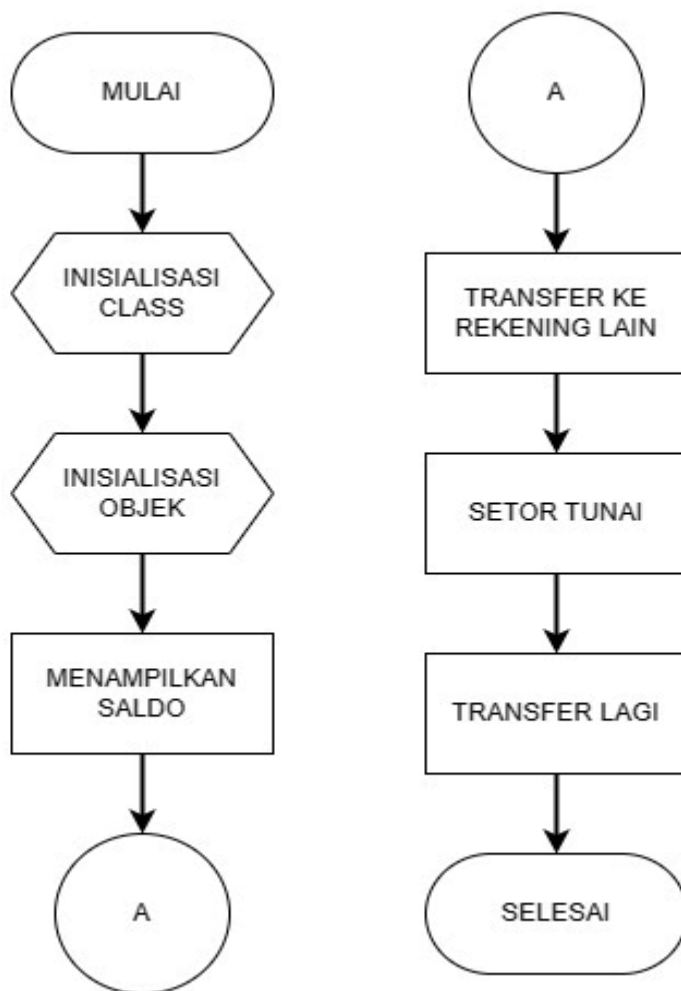
## RUN PYTHON CODE / FILE :

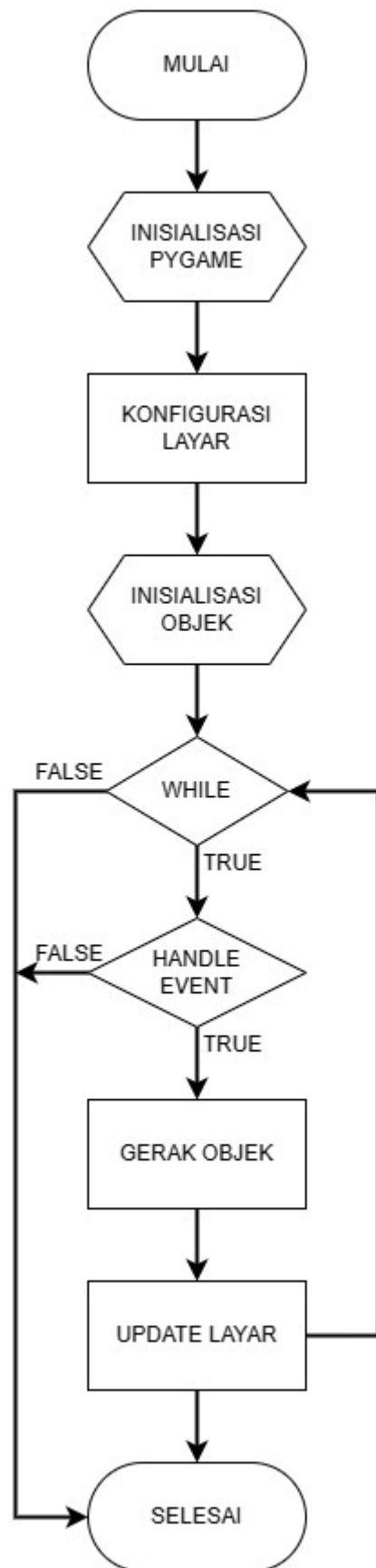


```
33
34 # update objek jatuh
35 y += v*dt
36 v += g*dt
37 t += dt
38
39 # bersihkan layar tampilan
40 screen.fill(yellow)
41
42 # gambar objek jatuh
43 pygame.draw.circle(screen, white, (500, 300), 10)
44
45 # tampilkan waktu simulasi
46 font = pygame.font.Font('freesansbold.ttf', 32)
47 text = font.render('Waktu: 7.45 detik', True, black)
48 screen.blit(text, (10, 10))
49
50 #update layar
51 pygame.display.update()
52
53 #masukkan ketinggian
54 if y > height:
55     break
56
57 # exit pygame
58 pygame.quit()
59
60
```

**Hasil :** Didapat Simulasi jatuh bebas dengan kecepatan yang berubah-ubah dengan Diketahui ketinggian awal 88 m. dan Menampilkan perhitungan waktu simulasi dalam visualisasi Pygame

**C. DEMO TASK 3 ( FLOWCHART ) :**





## **DAFTAR PUSTAKA**

- [1] <https://www.pygame.org/docs/ref/display.html>
- [2] <https://www.youtube.com/watch?v=AY9MnQ4x3zk> ( The Ultimate Introduction to PyGame )

*Halaman ini sengaja dikosongkan*