

Diterima : 21 April 2025 | Selesai Direvisi : 05 Mei 2025 | Disetujui : 02 Juni 2025 | Dipublikasikan : Juli 2025  
DOI : <https://doi.org/10.24853/jk.16.2.146-156>  
Copyright © 2025 Jurnal Konstruksia  
This is an open access article under the CC BY-NC licence (<http://creativecommons.org/licenses/by-nc/4.0/>)

## Koding QB64 dan Python Sebagai Bahasa Komunikasi Teknik Sipil Dalam Memahami Alur Pikir

Haryo Koco Buwono<sup>1\*</sup>, Andika Setiawan<sup>1</sup>, Deby Puspitaningrum<sup>2</sup>, dan Harwidyo Eko Prasetyo<sup>1</sup>

<sup>1</sup>Prodi Teknik Sipil, Universitas Muhammadiyah Jakarta, Jl. Cempaka Putih Tengah 27, Jakarta, 10510

Email korespondensi: [haryo.koco@umj.ac.id](mailto:haryo.koco@umj.ac.id)

<sup>2</sup>Prodi Ilmu Komunikasi, Universitas Bina Sarana Informatika, Jl. Kramat Raya No. 98, Jakarta, 10450

### ABSTRAK

Dalam era digital, keterampilan pemrograman menjadi elemen penting dalam berbagai disiplin ilmu, termasuk Teknik Sipil dan Ilmu Komunikasi. Penelitian ini membandingkan dua bahasa pemrograman, QB64 dan Python, dalam konteks visualisasi gaya geser dan momen lentur pada balok, guna mengidentifikasi efektivitas keduanya dalam menyampaikan informasi teknis secara visual dan komunikatif. QB64, sebagai bahasa berbasis BASIC, menawarkan sintaks sederhana dan mudah dipahami, sehingga cocok untuk pemula dan pembelajaran konsep dasar. Namun, keterbatasan dalam visualisasi grafis dan skalabilitas program membatasi penerapannya dalam konteks profesional. Sebaliknya, Python menyediakan fleksibilitas tinggi, didukung oleh pustaka ilmiah seperti NumPy dan Matplotlib, yang memungkinkan visualisasi data teknik yang presisi dan komunikatif. Dari sudut pandang komunikasi, Python mampu menerjemahkan data numerik menjadi bentuk visual yang lebih mudah dipahami oleh audiens lintas disiplin, meningkatkan efektivitas penyampaian pesan dalam ranah akademik dan praktis. Melalui pendekatan ini, diperoleh pemahaman bahwa pemilihan bahasa pemrograman bukan hanya soal teknis, tetapi juga berkaitan erat dengan strategi komunikasi visual. Python dinilai lebih unggul dalam menyampaikan informasi kompleks dengan cara yang sistematis, modular, dan interaktif. Oleh karena itu, integrasi antara kompetensi teknis dan kemampuan komunikasi digital menjadi kunci dalam pengembangan program edukatif dan profesional di bidang teknik. Penelitian ini merekomendasikan Python sebagai media utama untuk visualisasi teknik yang informatif dan komunikatif.

**Kata kunci:** Python, QB64, diagram, geser, momen, komunikasi, visual, data, koding

### ABSTRACT

*In the digital era, programming skills have become essential across disciplines, including Civil Engineering and Communication Science. This study compares two programming languages, QB64 and Python, in the context of visualizing shear force and bending moment diagrams for beams. The objective is to assess their effectiveness in conveying technical information through visual and communicative means. QB64, a BASIC-based language, offers simple syntax and is suitable for beginners and conceptual learning. However, it is limited in graphical capabilities and lacks scalability for professional applications. In contrast, Python provides high flexibility, supported by scientific libraries such as NumPy and Matplotlib, enabling precise and interactive visualizations. From communication perspective, Python allows numerical data to be translated into accessible visual formats, enhancing the clarity and reach of technical messages across disciplines. This comparison reveals that programming language selection is not merely*

*a technical decision, but also a strategic communication choice. Python excels in transforming complex data into structured, modular, and visually engaging outputs, making it an effective tool for both educational and professional use. Therefore, integrating technical proficiency with visual communication skills is critical for developing impactful programs in engineering fields. The study concludes by recommending Python as the preferred platform for communicative and informative technical visualization.*

**Keywords:** Python, QB64, diagram, shear\_force, bending\_moment, communication, visualization, data, coding

## 1. LATAR BELAKANG

Teknik Sipil modern menuntut kemampuan analisis data, simulasi struktur, dan otomatisasi proses perencanaan. Bahasa pemrograman seperti Python telah banyak digunakan dalam analisis numerik, pemodelan struktur, dan visualisasi data teknik sipil. Sementara itu, QB64, sebagai penerus QBasic, menawarkan lingkungan pemrograman yang sederhana dan intuitif, cocok untuk pemula dalam memahami logika pemrograman dasar [1][2].

Pemrograman tidak hanya tentang menulis kode, tetapi juga tentang menyampaikan logika dan proses berpikir secara sistematis. Dalam konteks teknik sipil, kemampuan untuk mengkomunikasikan alur pikir melalui kode sangat penting, terutama dalam kolaborasi tim dan dokumentasi proyek. Melalui komunikasi membantu dalam menyusun kode yang jelas, terstruktur, dan mudah dipahami oleh anggota tim lainnya [3].

Pemerintah Indonesia telah menunjukkan komitmen dalam mengintegrasikan koding dan kecerdasan artifisial (KA/AI) ke dalam kurikulum pendidikan. Mulai tahun ajaran 2025/2026, koding dan KA akan diperkenalkan sebagai mata pelajaran pilihan di sekolah dasar dan menengah, sebagai bagian dari program digitalisasi pendidikan nasional [4][5]. Bahasa pemrograman mengenai pemodelan polinomial kecepatan kendaraan ringan pada bundaran bertujuan untuk mengidentifikasi dan

mengkuantifikasi hubungan fungsional antara kecepatan operasional kendaraan ringan dengan parameter geometrik bundaran yang relevan, serta faktor-faktor lingkungan lainnya [6]. Secara ilmiah, pemodelan polinomial menawarkan fleksibilitas untuk menangkap hubungan non-linear yang kompleks antara variabel kecepatan dan berbagai prediktor. Hal lain mengenai Bahasa pemrograman ialah Studi mengenai pemodelan empiris pemeliharaan pada perkerasan chip seal dan laston berfokus pada pengembangan hubungan matematis yang didasarkan pada data observasi lapangan [7]. Model-model ini bertujuan untuk memprediksi degradasi kinerja perkerasan, mengidentifikasi ambang batas intervensi pemeliharaan yang optimal, dan mengevaluasi dampak berbagai strategi pemeliharaan terhadap umur layan dan biaya siklus hidup.

Selain itu, Kementerian Komunikasi dan Digital (Kemkomdigi) bekerja sama dengan Microsoft meluncurkan program "elevAlte Indonesia" yang bertujuan untuk melatih 1 juta warga Indonesia dalam keterampilan AI. Hal ini menunjukkan upaya serius pemerintah dalam mempersiapkan sumber daya manusia untuk era digital [8]. Dengan kemampuan teknologi terdepan di era revolusi industri 4.0., AI mampu menggabungkan perilaku dan kecerdasan manusia ke dalam sistem [9]. AI mampu membuat berbagai konten digital, mampu membantu dalam kebutuhan akademis atau pendidikan [10].

Sebagai bahasa pemrograman yang digunakan dalam pendidikan Teknik Sipil, dipilihnya QB64 dan Python bukan hanya relevan secara teknis, namun juga selaras dengan kebijakan nasional dalam memajukan literasi digital dan AI. Integrasi Ilmu Komunikasi dalam proses pembelajaran pemrograman memperkuat kemampuan mahasiswa dalam menyusun dan menyampaikan logika teknis secara efektif, yang sangat penting dalam praktik profesional Teknik Sipil.

## 2. BAHASA PEMROGRAMAN

Bahasa pemrograman adalah sistem notasi yang dirancang untuk menuliskan instruksi-instruksi yang dapat dijalankan oleh komputer. Bahasa ini memungkinkan manusia untuk memberikan perintah kepada mesin agar melakukan tugas tertentu, seperti melakukan perhitungan, menyimpan data, atau mengendalikan perangkat keras.

Jenis-jenis Bahasa Pemrograman

1. Bahasa Tingkat Tinggi (High-Level Language):
  - Lebih dekat dengan bahasa manusia. Contoh: Python, Java, C++, JavaScript.
  - Mudah dibaca dan dipahami oleh pemula.
  - Digunakan untuk pengembangan aplikasi, analisis data, kecerdasan buatan, dan lainnya.
2. Bahasa Tingkat Rendah (Low-Level Language):
  - Lebih dekat dengan bahasa mesin (0 dan 1). Contoh: Assembly, bahasa mesin.
  - Memberikan kontrol langsung ke perangkat keras tapi sulit dipahami oleh manusia.
3. Bahasa Pemrograman Visual:
  - Menggunakan elemen grafis (drag and drop) seperti Scratch atau Blockly.

- Umumnya digunakan untuk pendidikan anak dan pemula.

Fungsi Bahasa Pemrograman

- Komunikasi antara manusia dan komputer
- Mengembangkan aplikasi/*software*
- Mengotomatisasi tugas-tugas komputasi
- Membuat sistem atau alat bantu teknis, seperti simulasi di teknik sipil
- Mengembangkan teknologi cerdas seperti AI (*Artificial Intelligence*)

### Bahasa Qbasic (QB64)

Bahasa pemrograman QB64 merupakan evolusi modern dari QuickBASIC, yang dirancang untuk mempertahankan kesederhanaan dan keterbacaan sintaksisnya, sekaligus menambahkan fitur-fitur modern yang mendukung pengembangan aplikasi pada sistem operasi terkini seperti Windows, macOS, dan Linux.

QB64 mempertahankan sintaksis yang mirip dengan bahasa alami, seperti penggunaan perintah PRINT, INPUT, dan IF...THEN, yang memudahkan pemula dalam memahami logika pemrograman dasar tanpa harus mempelajari konsep-konsep kompleks seperti pemrograman berorientasi objek sejak awal.

Berbeda dengan pendahulunya yang terbatas pada lingkungan DOS, QB64 dapat dijalankan pada berbagai sistem operasi modern, memungkinkan mahasiswa teknik sipil untuk mengembangkan dan menjalankan program tanpa perlu khawatir tentang kompatibilitas sistem.

QB64 menyediakan dukungan untuk pemrosesan grafis dan audio, memungkinkan pengguna untuk membuat aplikasi yang lebih interaktif dan visual, yang sangat berguna dalam simulasi dan pemodelan dalam bidang teknik sipil.

Dengan adanya InForm, sebuah alat pengembangan aplikasi berbasis GUI untuk QB64, pengguna dapat merancang antarmuka pengguna secara visual dan

menghasilkan kode yang sesuai, mempercepat proses pengembangan aplikasi.

### Bahasa Python

Python adalah bahasa pemrograman tingkat tinggi yang dirancang dengan filosofi sederhana, mudah dibaca, dan mudah dipelajari. Python dikembangkan oleh Guido van Rossum dan pertama kali dirilis pada tahun 1991. Python saat ini menjadi salah satu bahasa pemrograman paling populer di dunia karena fleksibilitas dan kemampuannya digunakan dalam berbagai bidang.

Dengan sintaksis yang sederhana dan bersih, Python dapat melakukan penulisan kode yang mirip bahasa manusia dan tidak membutuhkan banyak simbol rumit. Bersifat *multi-platform*, membuat Python dapat dijalankan pada berbagai sistem operasi seperti Windows, macOS, dan Linux. Selain itu, didukung Komunitas Python yang besar, memiliki ribuan pustaka (*library*) dan forum komunitas aktif. Kode Python dijalankan baris demi baris (*interpreted*), memudahkan proses *debug* dan eksperimen, sehingga menjadi interpretatif dan interaktif bagi penggunaannya.

Fungsi dan Kegunaan Python yang digunakan dalam berbagai bidang, seperti:

- Ilmu Data & Statistik: menggunakan pustaka seperti NumPy, Pandas, dan Matplotlib.
- Kecerdasan Buatan (AI) & *Machine Learning*: menggunakan TensorFlow, PyTorch, dan Scikit-Learn.
- Pengembangan Web: dengan *framework* seperti Django dan Flask.
- Otomasi Tugas (*Scripting*): untuk pekerjaan rutin seperti mengelola *file* atau *web scraping*.
- Pengolahan Citra & Sinyal: misalnya di teknik sipil untuk interpretasi data dari sensor.
- Pendidikan: karena mudah dipahami, sangat cocok sebagai bahasa pertama untuk belajar koding.

Kelebihan Python adalah mudah dipelajari oleh pemula, sintaksis ringkas dan mudah dibaca, komunitas dan dokumentasi luas serta banyak digunakan dalam dunia industri dan akademik.

### Bahasa Pemrograman dan Bahasa Komunikasi

Keterkaitan antara bahasa pemrograman dan bahasa komunikasi sangat penting, terutama dalam konteks pendidikan dan kolaborasi lintas disiplin, seperti di bidang teknik sipil atau pengembangan perangkat lunak. Meskipun keduanya memiliki struktur dan tujuan yang berbeda, keduanya sama-sama berfungsi sebagai media untuk menyampaikan informasi, logika, dan maksud secara jelas dan efektif.

1. Bahasa pemrograman memiliki *sintaksis* (aturan penulisan) yang ketat seperti halnya bahasa manusia memiliki tata bahasa. Sedangkan untuk menyampaikan instruksi kepada komputer secara benar, struktur sintaksis harus tepat – mendekati dengan bagaimana kalimat dalam komunikasi manusia harus disusun secara logis agar bisa dipahami.

Contoh:

*Bahasa pemrograman:*

```
if suhu > 100:
```

```
    print("Air mendidih")
```

*Bahasa komunikasi:*

"Jika suhu melebihi 100 derajat, maka air akan mendidih."

2. Dalam komunikasi manusia, tujuan utama adalah menyampaikan pesan atau ide. Sedangkan dalam pemrograman, tujuannya adalah menyampaikan logika pemrosesan kepada komputer, namun juga kepada sesama *programmer* dalam tim (melalui komentar, dokumentasi, dan struktur kode).
3. Keterbacaan dan dokumentasi dari koding yang baik, menjadikan dapat dibaca dan dipahami oleh manusia, bukan hanya dari sisi komputer. Selain itu, memahami prinsip-prinsip dari ilmu komunikasi (seperti *clarity*,

- coherence*, dan *audience awareness*) diterapkan juga dalam penulisan koding dan dokumentasi agar pengguna lain mudah mengerti dan melanjutkan pekerjaan tersebut.
4. Dalam proyek-proyek Teknik Sipil yang melibatkan simulasi atau otomasi, *programmer* teknik harus memiliki kemampuan: (a) Menerjemahkan ide teknis ke dalam kode, dan (b) Menjelaskan kode kepada rekan satu tim agar dipahami bersama, maka diperlukan kemampuan komunikasi lintas disiplin, hal inilah salah satu alasan pentingnya keterampilan komunikasi bagi insinyur atau *developer*.
  5. Dalam dunia pendidikan, koding adalah sebagai bahasa kedua, dimana kebanyakan ahli pendidikan menyebut pemrograman sebagai bentuk “literasi digital” atau “bahasa kedua”, sebagaimana bahasa asing, koding melatih struktur berpikir logis dan sistematis, yang juga bermanfaat dalam komunikasi tertulis maupun lisan agar menjadi efektif.

### Hubungan Teknik Sipil dan Ilmu Komunikasi

Dalam konteks Teknik Sipil, kemampuan untuk mengkomunikasikan alur pikir melalui kode sangat penting, terutama dalam kolaborasi tim dan dokumentasi proyek. Ilmu Komunikasi membantu dalam menyusun kode yang jelas, terstruktur, dan mudah dipahami oleh anggota tim lainnya. Dengan menggunakan QB64 dan Python, mahasiswa dapat belajar menyusun logika program yang tidak hanya berfungsi dengan baik, tetapi juga mudah dipahami dan dimodifikasi oleh orang lain, meningkatkan efisiensi dan efektivitas dalam kerja tim.

### 3. METODOLOGI

Penelitian ini menggunakan pendekatan kualitatif deskriptif dengan metode studi program. Tujuannya adalah untuk

mengamati, membandingkan, dan menganalisis bentuk kode program dari dua bahasa (QB64 dan Python) dalam penyelesaian permasalahan sederhana dalam Teknik Sipil, serta menelaah bagaimana struktur komunikasi logika dan penyampaian makna melalui kode dilakukan dalam masing-masing bahasa.

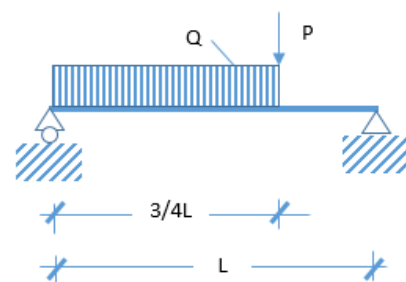
Jenis penelitian ini adalah analisis tekstual kode (*code-oriented textual analysis*), yang fokus pada (1) struktur dan gaya penulisan kode. (2) Pola komunikasi logika dalam program. (3) Kejelasan pesan/instruksi dalam kode yang dapat dibaca dan dipahami oleh manusia (rekayasa komunikasi dalam pemrograman).

Sumber Data dan Teknik Pengumpulan Data diambil dari data primer yaitu berupa dua program yang dibuat oleh peneliti, dimana masing-masing dalam QB64 dan Python, yang menyelesaikan satu kasus Teknik Sipil (mengambil analisis program statika). Sedangkan data sekunder, berupa dokumentasi bahasa pemrograman (manual, referensi resmi, pustaka komunitas). Selanjutnya berupa literatur terkait prinsip komunikasi dalam penulisan kode (komentar, penamaan variabel, pengorganisasian blok kode).

Teknik pengumpulan data berupa: (1) studi dokumen dan pustaka. (2) penulisan dan dokumentasi dua program dengan pendekatan masalah yang sama. (3) catatan-catatan dalam proses penyusunan kode dan informasi logika selama penulisan koding.

### 4. ANALISIS DAN DISKUSI

Berikut ini adalah penulisan koding dari Qb64 dan Python dari persoalan statika.



Gambar 1. Soal Statika

Dari soal tersebut diminta mendapatkan reaksi tumpuan, serta gambar gaya lintang dan momennya.

### 1. Qb64

SCREEN 12

CLS

```
' === DEKLARASI VARIABEL ===
DIM L AS SINGLE
DIM P AS SINGLE
DIM Q AS SINGLE
DIM RA AS SINGLE, RB AS SINGLE
DIM w AS SINGLE, x AS SINGLE
DIM V(100) AS SINGLE, M(100) AS SINGLE
DIM i AS INTEGER
DIM position AS SINGLE ' Changed from
pos to position
DIM xPlot AS SINGLE, yPlotV AS SINGLE,
yPlotM AS SINGLE
DIM skalaX AS SINGLE, skalaV AS SINGLE,
skalaM AS SINGLE
DIM x0 AS INTEGER, y0 AS INTEGER

' === INPUT DATA ===
INPUT "Masukkan panjang balok (L): ", L
INPUT "Masukkan gaya terpusat (P): ", P
INPUT "Masukkan gaya merata (Q): ", Q

' === HITUNG REAKSI TUMPUAN ===
w = Q * (3 / 4) * L
x = (3 / 8) * L
RB = (P * (3 / 4) * L + w * x) / L
RA = P + w - RB

PRINT "Reaksi di tumpuan kiri (RA): "; RA
PRINT "Reaksi di tumpuan kanan (RB): ";
RB

' === SKALA GAMBAR ===
skalaX = 500 / L
skalaV = 2
skalaM = 0.2
x0 = 50
```

y0 = 250

```
' === HITUNG DAN GAMBAR DIAGRAM
===
FOR i = 0 TO 100
    ' Use the renamed variable 'position' to
    calculate the position
    position = (i * L) / 100.0 ' Using 100.0 to
    ensure SINGLE division

    ' GAYA GESER
    IF position <= (3 / 4) * L THEN
        V(i) = RA - Q * position
    ELSE
        V(i) = RA - Q * (3 / 4) * L - P
    END IF

    ' MOMEN LENTUR
    IF position <= (3 / 4) * L THEN
        M(i) = RA * position - (Q * position *
        position) / 2
    ELSE
        M(i) = RA * position - Q * (3 / 4) * L *
        ((3 / 8) * L) - P * (position - (3 / 4) * L)
    END IF

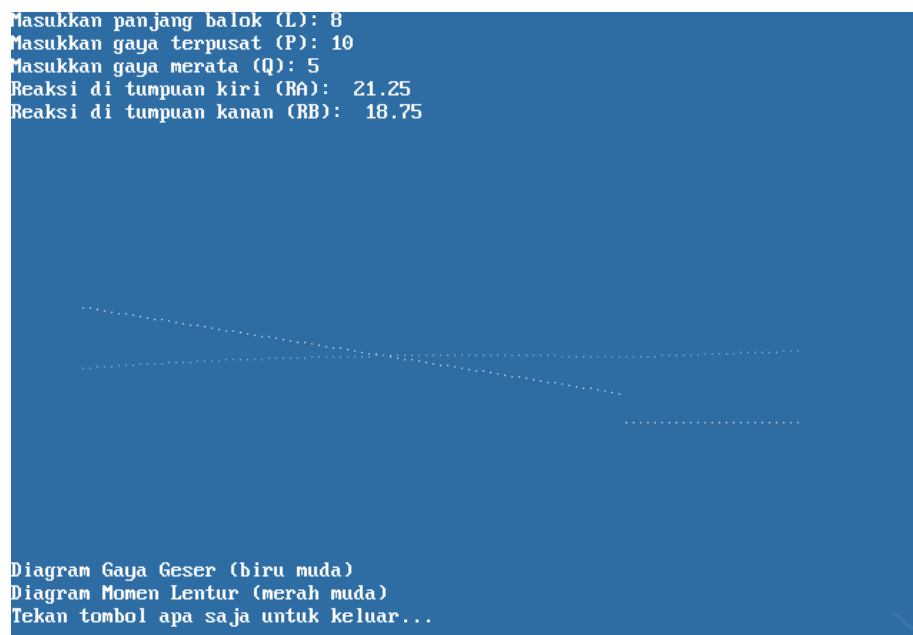
    ' GRAFIK GAYA GESER
    xPlot = x0 + position * skalaX
    yPlotV = y0 - V(i) * skalaV
    PSET (xPlot, yPlotV), 10

    ' GRAFIK MOMEN
    yPlotM = y0 - M(i) * skalaM
    PSET (xPlot, yPlotM), 12
NEXT

' === KETERANGAN ===
COLOR 15
LOCATE 25, 1: PRINT "Diagram Gaya Geser
(biru muda)"
LOCATE 26, 1: PRINT "Diagram Momen
Lentur (merah muda)"
```

LOCATE 27, 1: PRINT "Tekan tombol apa  
saja untuk keluar..."

SLEEP  
END



Gambar 2. Output luaran QB64

## 2. Python

```
import matplotlib.pyplot as plt
import numpy as np
```

```
# === Parameter input simulasi ===
```

```
L = 8.0 # meter
```

```
P = 10.0 # kN
```

```
Q = 5.0 # kN/m
```

```
# === Hitung Reaksi Tumpuan ===
```

```
a = (3/4)*L
```

```
w = Q * (3 / 4) * L
```

```
x = (3 / 8) * L
```

```
RB = (P * (3 / 4) * L + w * x) / L
```

```
RA = P + w - RB
```

```
print("\n=== ANALISIS BALOK  
SEDERHANA ===")
```

```
print(f"Panjang balok (L): {L} m")
```

```
print(f"Gaya terpusat (P): {P} kN")
```

```
print(f"Beban merata (Q): {Q} kN/m")
```

```
print(f"\nReaksi tumpuan kiri (RA):  
{RA:.2f} kN")
```

```
print(f"Reaksi tumpuan kanan (RB):  
{RB:.2f} kN")
```

```
# === Perhitungan dan Plotting ===
```

```
x_vals = np.linspace(0, L, 500)
```

```
V = []
```

```
M = []
```

```
for x in x_vals:
```

```
    if x <= a:
```

```
        V_now = RA - Q * x
```

```
        M_now = RA * x - (Q * x ** 2) / 2
```

```
    else:
```

```
        V_now = RA - Q * a - P
```

```
        M_now = RA * x - Q * a * (x - a / 2) - P *  
(x - a)
```

```
        V.append(V_now)
```

```
        M.append(M_now)
```

```
# === Plot Diagram ===
```

```
plt.figure(figsize=(14, 7))
```

```
# Gaya Geser
```

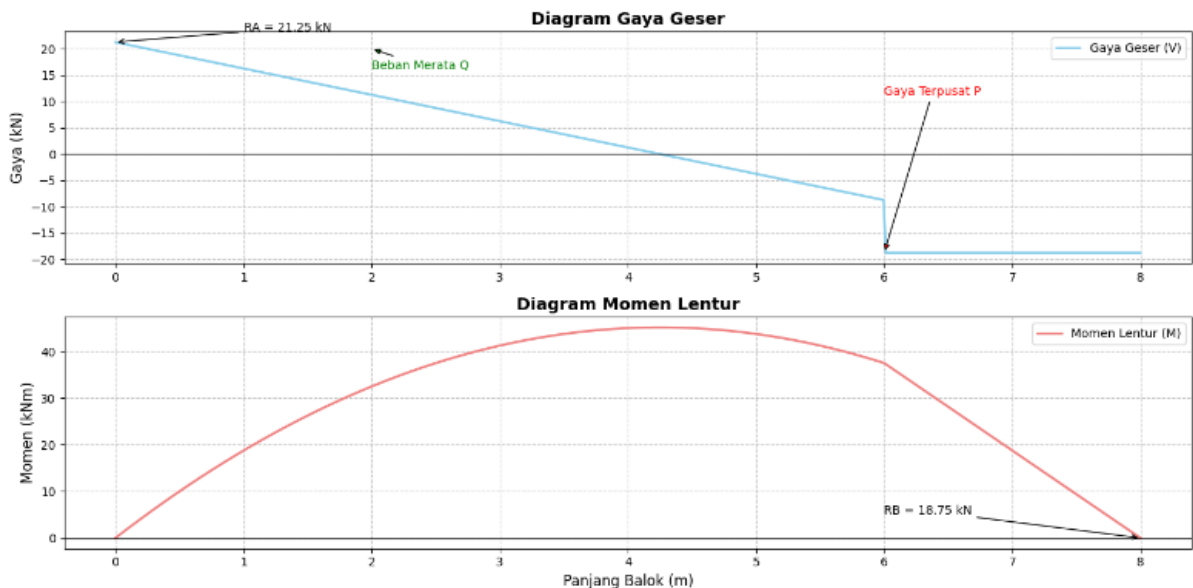
```
plt.subplot(2, 1, 1)
```

```
plt.plot(x_vals, V, color='skyblue',
label='Gaya Geser (V)', linewidth=2)
plt.axhline(0, color='black', linewidth=0.8)
plt.title("Diagram Gaya Geser",
fontsize=14, weight='bold')
plt.ylabel("Gaya (kN)", fontsize=12)
plt.grid(True, linestyle='--', alpha=0.7)
plt.legend()
plt.annotate(f"RA = {RA:.2f} kN", xy=(0,
RA), xytext=(1, RA + 2),
arrowprops=dict(facecolor='black',
arrowstyle='->'))
plt.annotate("Beban Merata Q",
xy=(L*0.25, RA - Q*0.25), xytext=(L*0.25,
RA - 5),
arrowprops=dict(facecolor='green',
arrowstyle='->'), color='green')
plt.annotate("Gaya Terpusat P", xy=(a, RA
- Q*a - P), xytext=(a, RA - 10),
arrowprops=dict(facecolor='red',
arrowstyle='->'), color='red')
```

```
plt.subplot(2, 1, 2)
plt.plot(x_vals, M, color='lightcoral',
label='Momen Lentur (M)', linewidth=2)
plt.axhline(0, color='black', linewidth=0.8)
plt.title("Diagram Momen Lentur",
fontsize=14, weight='bold')
plt.xlabel("Panjang Balok (m)",
fontsize=12)
plt.ylabel("Momen (kNm)", fontsize=12)
plt.grid(True, linestyle='--', alpha=0.7)
plt.legend()
plt.annotate(f"RB = {RB:.2f} kN", xy=(L, 0),
xytext=(L - 2, 5),
arrowprops=dict(facecolor='black',
arrowstyle='->'))

plt.tight_layout()
output_path =
"/mnt/data/diagram_balok.png"
plt.savefig(output_path)
output_path
```

## # Momen Lentur



Gambar 3. Hasil Output Python Setelah Proses *Running*

Rekapitulasi mendasar dari perbedaan Qb64 dan Python dinyatakan dalam Tabel 1 berikut ini.



Tabel 1. Konsep Perbedaan Mendasar dari Perintah Qb64 dan Python

| <b>Konsep</b> | <b>QB64</b>     | <b>Python</b>                        |
|---------------|-----------------|--------------------------------------|
| Deklarasi     | DIM x AS SINGLE | Tidak perlu                          |
| Input         | INPUT           | input() + konversi tipe              |
| Array         | DIM V(100)      | np.zeros(101)                        |
| Looping       | FOR ... NEXT    | for i in range(...) atau enumerate() |
| Grafik        | PSET (x, y)     | matplotlib.pyplot.plot()             |

Tabel 2. Perbedaan Qb64 Dan Python Beserta Analisis Kualitatif

| <b>Aspek</b>               | <b>QB64</b>                    | <b>Python</b>                            | <b>Analisis Kualitatif</b>                      |
|----------------------------|--------------------------------|------------------------------------------|-------------------------------------------------|
| Bahasa                     | <i>Basic-style, prosedural</i> | <i>High-level, multi-paradigm</i>        | Python lebih modern dan ekspresif               |
| Deklarasi variabel         | DIM x AS SINGLE                | Tidak perlu deklarasi; langsung x = 10.0 | Python lebih fleksibel, QB64 lebih ketat        |
| Input dari pengguna        | INPUT "Text"; x                | x = float(input("Text"))                 | Python memerlukan konversi tipe                 |
| Perhitungan matematis      | x = (3 / 8) * L                | x = (3 / 8) * L                          | Hampir sama di kedua bahasa                     |
| Array                      | DIM V(100) AS SINGLE           | V = np.zeros(101)                        | Python membutuhkan <i>import numpy</i>          |
| Perulangan ( <i>loop</i> ) | FOR i = 0 TO 100NEXT           | for i in range(101):                     | Python lebih ringkas                            |
| Kondisional (IF)           | IF pos <= x<br>THENELSEEND IF  | if pos <= x:else:                        | Python lebih bersih, QB64 lebih <i>verbose</i>  |
| Penggambaran grafik        | PSET (x, y), warna             | plt.plot(x, y)plt.show()                 | Python jauh lebih kuat dengan <i>matplotlib</i> |
| Komentar                   | ' komentar                     | # komentar                               | Perintah berbeda komentar,                      |

|                        |                                        |                                                       |                                       |
|------------------------|----------------------------------------|-------------------------------------------------------|---------------------------------------|
| Skala dan koordinat    | Manual (x0, y0, skalaX, skalaY)        | Otomatis dengan <i>library plotting</i>               | Python lebih otomatis                 |
| Output teks ke layar   | PRINT "Text"                           | print("Text")                                         | Hampir sama                           |
| Tampilan hasil         | Tampilan langsung ke layar grafis QB64 | Grafik jendela terpisah dengan <i>matplotlib</i>      | Python lebih modern dan bisa disimpan |
| Kesulitan pemula       | Mudah untuk mulai, tetapi terbatas     | Agak curam di awal (karena library), tapi sangat kuat | Python unggul dalam jangka panjang    |
| Ekosistem dan dukungan | Terbatas pada komunitas QB64           | Luas, banyak dokumentasi dan contoh                   | Python menang telak                   |
| Visualisasi lanjutan   | Sangat terbatas (hanya garis, titik)   | Bisa grafik interaktif, 3D, GUI, dll                  | Python fleksibel dan modern           |

Tabel 3. Rekomendasi Dari Hasil Analisis Kualitatif

| Kriteria                       | QB64  | Python | Rekomendasi                 |
|--------------------------------|-------|--------|-----------------------------|
| Kemudahan awal belajar         | ★★★★★ | ★★★★   | QB64 cocok untuk pengenalan |
| Visualisasi teknik             | ★★    | ★★★★★  | Python unggul besar         |
| Skalabilitas program           | ★     | ★★★★★  | Python lebih fleksibel      |
| Dukungan perhitungan teknik    | ★     | ★★★★★  | Python jauh lebih unggul    |
| Portabilitas dan dukungan alat | ★★    | ★★★★★  | Python unggul               |
| Komunitas dan dokumentasi      | ★★    | ★★★★★  | Python punya ekosistem luas |
| Cocok untuk profesional teknik | ★     | ★★★★★  | Python sangat cocok         |

Berdasarkan hasil pembuatan koding hingga analisis keduanya dapat dinyatakan bahwa Python jauh lebih baik dibandingkan Qb64. Jika dikaitkan dengan Ilmu Komunikasi, maka teknis penyampaian informasi bahasa tingkat

tinggi menjadi lebih memudahkan dalam komunikasi antar disiplin ilmu.

## 5. KESIMPULAN

1. QB64 unggul untuk pemula, tetapi Python unggul dalam keterbacaan dan pemeliharaan program jangka panjang.
2. Python jauh lebih unggul untuk visualisasi teknik. Hasil visual Python bisa digunakan langsung dalam laporan teknik atau akademik.
3. QB64 tidak memiliki pustaka *built-in* untuk kalkulus, matriks, metode numerik, atau analisis struktur.
4. Gunakan QB64 untuk pembelajaran logika dasar dan eksplorasi visualisasi manual.
5. Gunakan Python untuk penerapan teknik yang lebih serius, kompleks, dan profesional.
6. Python lebih *scalable* dan fleksibel, cocok untuk proyek teknik berkelanjutan atau tim.
7. Bahasa tingkat tinggi Python lebih memudahkan komunikasi antar disiplin ilmu.

## DAFTAR PUSTAKA

- [1] Qb64, "https://qb64.com/."
- [2] M. S. Tazakka, "Basic Python Programming Untuk Teknik Sipil," no. November, 2024, doi: 10.6084/m9.figshare.22223707.
- [3] M. Arief et al., "Program Python Dan Pengembangan Produk Media Online Terakota . Id," vol. 7, no. 02, pp. 222–236, 2024.
- [4] Kemendikbud, "https://www.kemdikbud.go.id/main/blog/2024/12/integrasi-coding-dan-ai-di-dunia-pendidikan-langkah-indonesia-menuju-ekosistem-digital-global."
- [5] Et. all Asiva Noor Rachmayani, "Naskah Akademik Koding dan Kecerdasan Artifisial," 2015.
- [6] H. K. Buwono, A. Setiawan, and O. Damarwulan, "Pemodelan Polinomial Kecepatan Kendaraan Ringan Pada Bundaran," Agregat, vol. 7, no. 1, pp. 642–648, 2022, doi: 10.30651/ag.v7i1.13297.
- [7] H. K. Buwono, H. Khoeri, A. Setiawan, B. Badaruddin, D. Sofiana, and D. Puspitaningrum, "Pemodelan Empiris Pemeliharaan Pada Perkerasan Chip Seal Dan Laston," Konstruksia, vol. 15, no. 1, p. 179, 2023, doi: 10.24853/jk.15.1.179-191.
- [8] Kemkomdigi, "Kemkomdigi Program Elevaite."
- [9] I. H. Sarker, "AI-Based Modeling: Techniques, Applications and Research Issues Towards Automation, Intelligent and Smart Systems," SN Computer Science, vol. 3, no. 158, pp. 1–20, 2022, doi: https://doi.org/10.1007/s42979-022-01043-x.
- [10] I. Marzuki, Widiati, U., Rusdin, D., Darwin, & Indrawati, "The Impact of AI Writing Tools on the Content and Organization of Students' Writing: EFL Teachers' Perspective," Cogent Education, vol. 10, no. 2, 2023, doi: . https://doi.org/10.1080/2331186X.2023.2236469.