

LAPORAN PRAKTIKUM
POSTTEST (2)
ALGORITMA PEMROGRAMAN DASAR

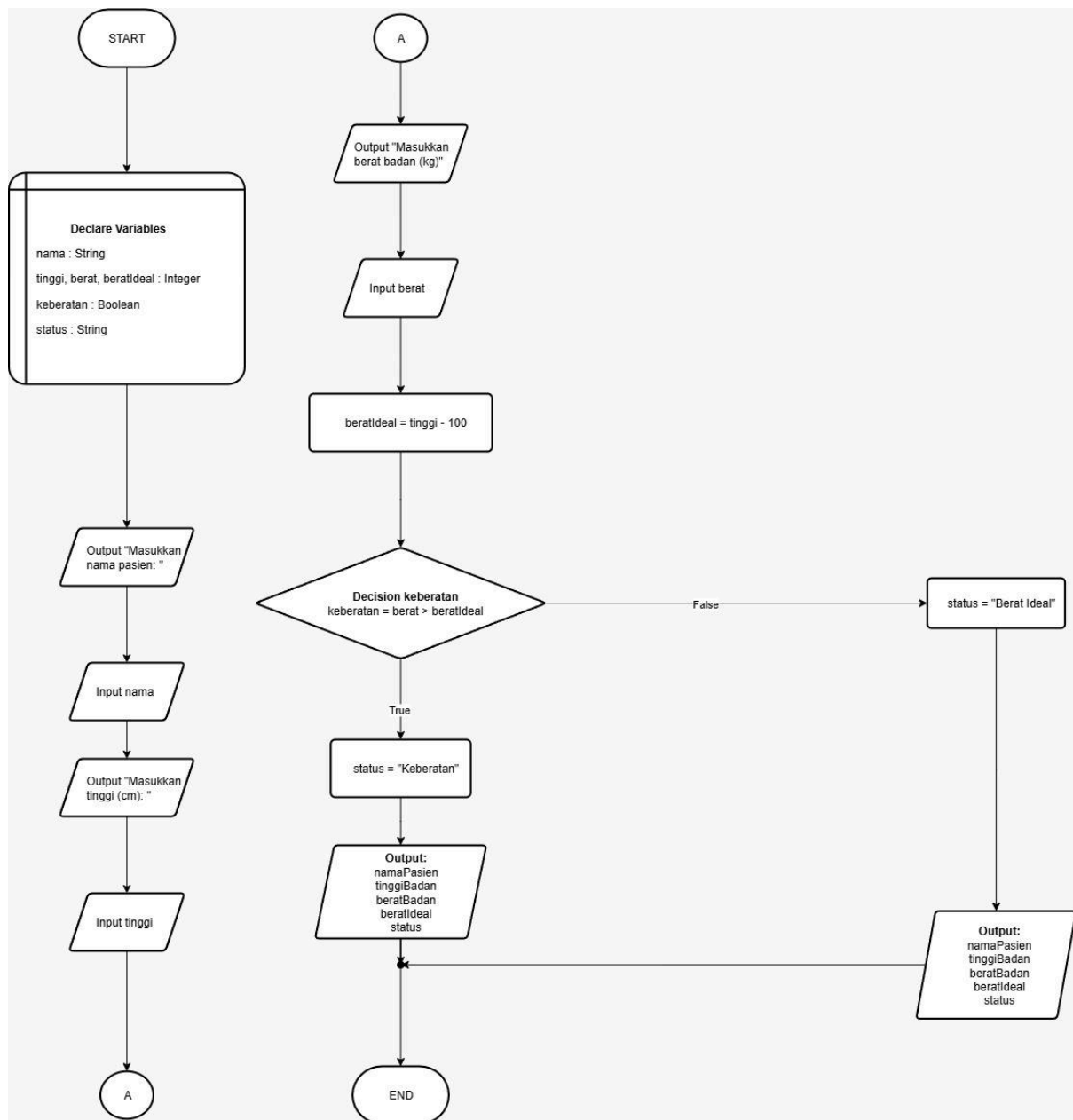


Heiza Rizki Pratama 2509106019

A'25

PROGRAM STUDI INFORMATIKA
UNIVERSITAS MULAWARMAN
SAMARINDA
2025

1. Flowchart



2. Deskripsi Singkat Program

Program dibuat untuk mengecek berat badan dan tinggi badan ideal apa tidak. Dengan begitu orang jadi tau dia kelebihan berat badan atau tidak. Di flowchart menunjukkan pengecekan berat badan dari rumus $\text{Berat badan ideal} = \text{Tinggi badan} - 100$

3. Source Code

```
nama = input("Masukkan Nama Pasien: ")
tinggi = float(input("Masukkan Tinggi Badan (cm): "))
beratBadan = float(input("Masukkan Berat Badan (kg): "))

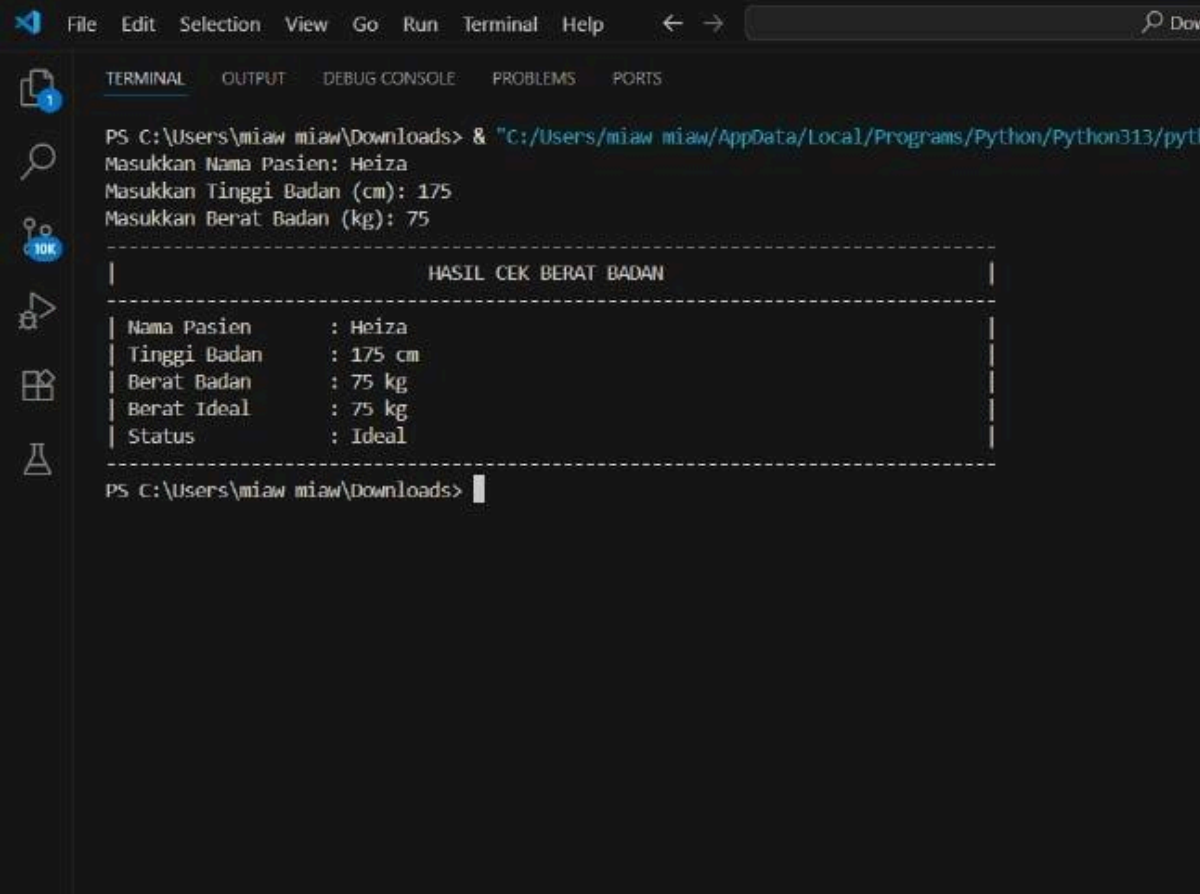
beratIdeal = (tinggi - 100)
isKelebihan = beratBadan > beratIdeal

statusList = ["Ideal", "Keberatan"]
status = statusList[int(isKelebihan)]

def row_formatting(label, value):
    return f"| {label:<18}: {value:<57}|"

print("-" * 80)
print(f"|{'HASIL CEK BERAT BADAN':^78}|")
print("-" * 80)
print(row_formatting("Nama Pasien", nama))
print(row_formatting("Tinggi Badan", f"{tinggi:.0f} cm"))
print(row_formatting("Berat Badan", f"{beratBadan:.0f} kg"))
print(row_formatting("Berat Ideal", f"{beratIdeal:.0f} kg"))
print(row_formatting("Status", status))
print("-" * 80)
```

4. Hasil Output



```
PS C:\Users\miaw miaw\Downloads> & "C:/Users/miaw miaw/AppData/Local/Programs/Python/Python313/python.exe" .\program.py
Masukkan Nama Pasien: Heiza
Masukkan Tinggi Badan (cm): 175
Masukkan Berat Badan (kg): 75
-----
|                                     HASIL CEK BERAT BADAN                                     |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| Nama Pasien      : Heiza                                                    |
| Tinggi Badan    : 175 cm                                                    |
| Berat Badan     : 75 kg                                                      |
| Berat Ideal     : 75 kg                                                      |
| Status          : Ideal                                                       |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
PS C:\Users\miaw miaw\Downloads>
```

5. Langkah-langkah GIT

```
PS C:\Users\Lucius\Documents\New folder> git init
Reinitialized existing Git repository in C:/Users/Lucius/Documents/New folder/.git/
PS C:\Users\Lucius\Documents\New folder> git add .
PS C:\Users\Lucius\Documents\New folder> git commit -m "ini commit"
[main 94354ce] ini commit
1 file changed, 1 insertion(+), 1 deletion(-)
PS C:\Users\Lucius\Documents\New folder> git push origin main
Enumerating objects: 11, done.
Counting objects: 100% (11/11), done.
Delta compression using up to 12 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (6/6), 465 bytes | 155.00 KiB/s, done.
Total 6 (delta 2), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To https://github.com/01001100-KMS/praktikum-apd.git
23547ec..94354ce main -> main
```

5.1 GIT Init

```
PS C:\Users\Lucius\Documents\New folder> git init
Reinitialized existing Git repository in C:/Users/Lucius/Documents/New folder/.git/
```

Fungsinya: Membuat repository Git baru di folder yang sedang aktif.

Setelah perintah ini dijalankan, Git akan membuat folder tersembunyi bernama `.git` yang berisi semua informasi untuk melacak versi file.

Contoh: "*Git init*"

→ Folder langsung jadi repository Git.

5.2 GIT Add

```
PS C:\Users\Lucius\Documents\New folder> git add .
```

Fungsinya: Memilih file yang sudah diubah untuk masuk ke **staging area** (daftar siap commit). Tanpa `git add`, perubahan tidak akan ikut tersimpan saat `git commit`.

Contoh:

```
git add index.html
```

→ hanya file `index.html` yang siap di-commit.

```
git add .
```

→ semua file yang berubah akan masuk *staging area*.

5.3 GIT Commit

```
PS C:\Users\Lucius\Documents\New folder> git commit -m "ini commit"
[main 94354ce] ini commit
1 file changed, 1 insertion(+), 1 deletion(-)
```

Fungsinya: Menyimpan perubahan yang sudah dipilih (**staging area**) ke dalam riwayat repository. Commit ini ibarat **checkpoint** atau **simpan versi** dari proyek. Biasanya commit disertai pesan (`-m`) agar jelas maksud perubahannya. Contoh:

```
git commit -m "Menambahkan fitur login"
```

5.4 GIT Remote

```
PS C:\Users\Lucius\Documents\New folder> git remote add origin https://github.com/01001100-KMS/praktikum-apd.git
```

```
PS C:\Users\Lucius\Documents\New folder> git remote -v
origin https://github.com/01001100-KMS/praktikum-apd.git (fetch)
origin https://github.com/01001100-KMS/praktikum-apd.git (push)
```

Fungsinya: Menghubungkan repository lokal dengan repository **remote** (misalnya di GitHub, GitLab, Bitbucket). Biasanya dikasih nama `origin` untuk mempermudah akses.

Contoh:

```
git remote add origin https://github.com/username/nama-repo.git
```

→ Menghubungkan repo lokal ke GitHub dengan nama origin.

Untuk mengecek remote yang sudah terhubung:

```
git remote -v
```

5.5 GIT Push

```
PS C:\Users\Lucius\Documents\New folder> git push origin main
Enumerating objects: 11, done.
Counting objects: 100% (11/11), done.
Delta compression using up to 12 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (6/6), 465 bytes | 155.00 KiB/s, done.
Total 6 (delta 2), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To https://github.com/01001100-KMS/praktikum-apd.git
 23547ec..94354ce main -> main
```

Fungsinya: Mengirim perubahan (commit) yang ada di repository lokal ke repository remote (misalnya GitHub, GitLab, Bitbucket). Supaya bisa push, biasanya harus sudah git remote add origin <url> terlebih dahulu.

Contoh:

```
git push origin main
```

→ Mengirim commit lokal ke branch main di repository remote bernama origin.