Nama : Bagas Farrelino Harsono Putro  
NIM : 21120122140101  
Kelas : Rekayasa Perangkat Lunak Berbasis Komponen C  
Group : C

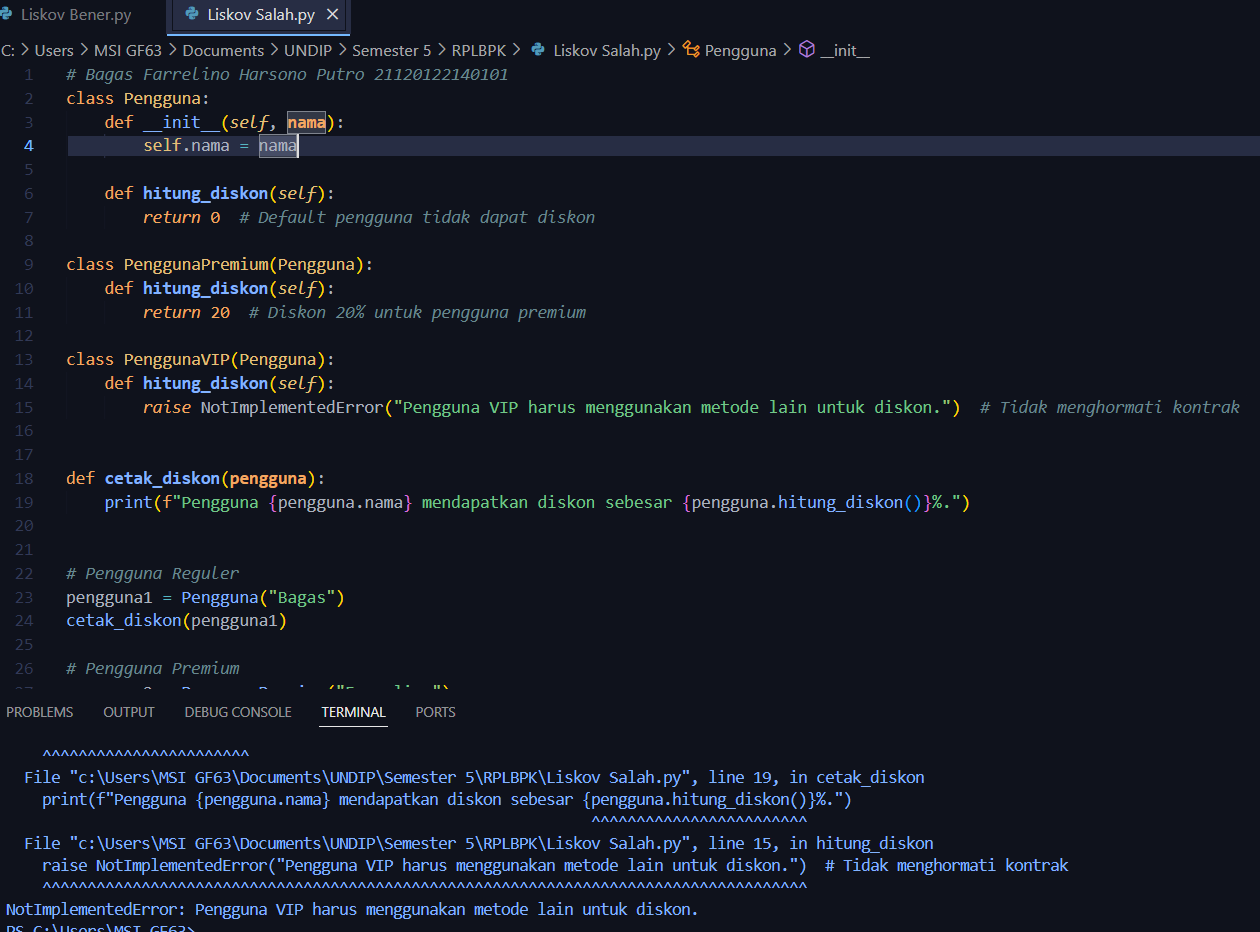
**TUGAS RPLBK- Running Code Example   
Liskov Substitution Principle (LSP)**

**Problem**Misalkan kita memiliki sebuah kelas Pengguna yang menangani berbagai macam jenis pengguna di sistem aplikasi. Kelas ini memiliki sebuah metode untuk menghitung diskon yang bisa diberikan kepada pengguna berdasarkan statusnya, misalnya pengguna reguler, premium, atau VIP. Namun, ketika kita mencoba menggunakan subclass yang menggantikan superclass, ada potensi masalah karena subclass tidak menghormati kontrak yang ditetapkan oleh superclass.

**Implementasi yang melanggar LSP**

|  |
| --- |
| # Bagas Farrelino Harsono Putro 21120122140101  class Pengguna:  def \_\_init\_\_(self, nama):  self.nama = nama  def hitung\_diskon(self):  return 0 # Default pengguna tidak dapat diskon  class PenggunaPremium(Pengguna):  def hitung\_diskon(self):  return 20 # Diskon 20% untuk pengguna premium  class PenggunaVIP(Pengguna):  def hitung\_diskon(self):  raise NotImplementedError("Pengguna VIP harus menggunakan metode lain untuk diskon.") # Tidak menghormati kontrak  def cetak\_diskon(pengguna):  print(f"Pengguna {pengguna.nama} mendapatkan diskon sebesar {pengguna.hitung\_diskon()}%.")  # Pengguna Reguler  pengguna1 = Pengguna("Bagas")  cetak\_diskon(pengguna1)  # Pengguna Premium  pengguna2 = PenggunaPremium("Farrelino")  cetak\_diskon(pengguna2)  # Pengguna VIP, akan menyebabkan error  pengguna3 = PenggunaVIP("Harsono Putro")  cetak\_diskon(pengguna3) # Ini akan menyebabkan NotImplementedError |

Kelas PenggunaVIP tidak memenuhi kontrak dari kelas Pengguna. Kelas induk mengharapkan semua pengguna memiliki diskon yang bisa dihitung, tetapi PenggunaVIP tidak mengimplementasikan metode hitung\_diskon dengan benar. Akibatnya, ketika kita mencoba memanggil metode tersebut, muncul error NotImplementedError.



Gambar 1 Bukti Running Code Salah=

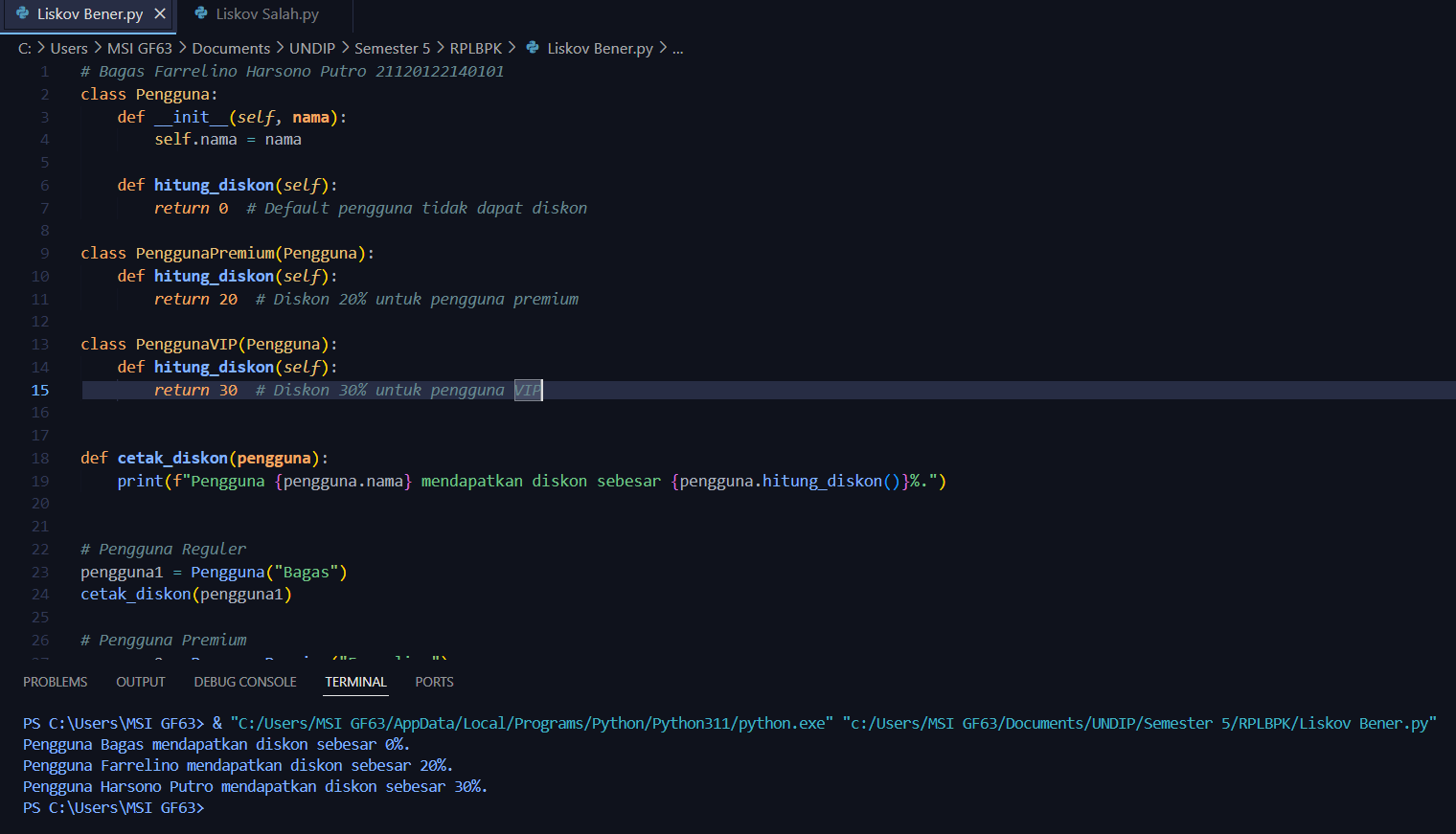
**Solusi yang Memenuhi LSP**Untuk memperbaiki pelanggaran terhadap LSP, kita harus memastikan bahwa setiap subclass dari Pengguna dapat digunakan secara konsisten dengan cara yang sama tanpa mengubah perilaku yang diharapkan.

|  |
| --- |
| class Pengguna:  def \_\_init\_\_(self, nama):  self.nama = nama  def hitung\_diskon(self):  return 0 # Default pengguna tidak dapat diskon  class PenggunaPremium(Pengguna):  def hitung\_diskon(self):  return 20 # Diskon 20% untuk pengguna premium  class PenggunaVIP(Pengguna):  def hitung\_diskon(self):  return 30 # Diskon 30% untuk pengguna VIP  def cetak\_diskon(pengguna):  print(f"Pengguna {pengguna.nama} mendapatkan diskon sebesar {pengguna.hitung\_diskon()}%.")  # Pengguna Reguler  pengguna1 = Pengguna("Bagas")  cetak\_diskon(pengguna1)  # Pengguna Premium  pengguna2 = PenggunaPremium("Farrelino")  cetak\_diskon(pengguna2)  # Pengguna VIP, sekarang tidak menyebabkan error  pengguna3 = PenggunaVIP("Harsono Putro")  cetak\_diskon(pengguna3) |

**Penjelasan**

* **Kelas** Pengguna adalah kelas induk yang memiliki metode hitung\_diskon() default yang mengembalikan 0 (tanpa diskon).
* **Kelas** PenggunaPremium mewarisi kelas Pengguna dan menimpa metode hitung\_diskon() untuk memberikan diskon sebesar 20%.
* **Kelas** PenggunaVIP juga menimpa metode hitung\_diskon() dan memberikan diskon sebesar 30%.

Fungsi cetak\_diskon() dipakai untuk mencetak diskon yang didapatkan oleh setiap pengguna. Fungsi ini akan bekerja dengan semua jenis pengguna (baik Pengguna, PenggunaPremium, maupun PenggunaVIP), dan tidak ada error yang terjadi karena setiap subclass menghormati kontrak yang ditetapkan oleh superclass.



Gambar 2 Bukti Running Code Benar