

LAPORAN PEMROGRAMAN BERORIENTASI OBJEK
“PENANGANAN MASALAH PEMELIHARAAN IKAN LELE DAN PATIN”

Link youtube : <https://youtu.be/IYHAi5fiHms>



Oleh:

Nama	NIM
FIRMAN SULTONI	: 09030582226010
RIANDA SAPUTRA	: 09030582226018
M. FACHRI RAMADHAN	: 09030582226026

FAKULTAS ILMU KOMPUTER JURUSAN TEKNIKKOMPUTER
UNIVERSITAS SRIWIJAYA

Mei 2024

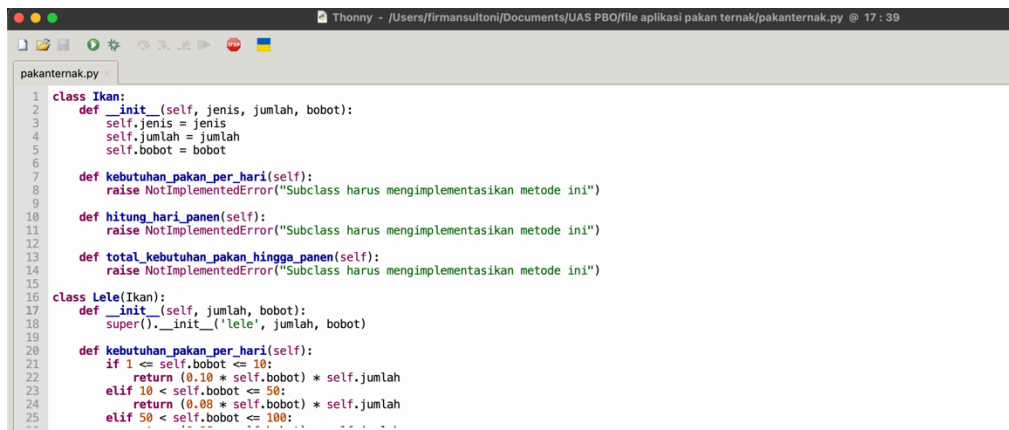
PEMELIHARAAN IKAN

A. Pendahuluan

Program "Pemeliharaan Ikan" merupakan sebuah alat yang digunakan untuk membantu dalam mengelola pemeliharaan ikan. Program ini memungkinkan pengguna untuk menghitung kebutuhan pakan harian, perkiraan hari panen, dan estimasi biaya pakan berdasarkan jenis ikan yang dipilih.

1. Prinsip Class

Potongan Kode:



```
1 class Ikan:
2     def __init__(self, jenis, jumlah, bobot):
3         self.jenis = jenis
4         self.jumlah = jumlah
5         self.bobot = bobot
6
7     def kebutuhan_pakan_per_hari(self):
8         raise NotImplementedError("Subclass harus mengimplementasikan metode ini!")
9
10    def hitung_hari_panen(self):
11        raise NotImplementedError("Subclass harus mengimplementasikan metode ini!")
12
13    def total_kebutuhan_pakan_hingga_panen(self):
14        raise NotImplementedError("Subclass harus mengimplementasikan metode ini!")
15
16 class Lele(Ikan):
17     def __init__(self, jumlah, bobot):
18         super().__init__('lele', jumlah, bobot)
19
20     def kebutuhan_pakan_per_hari(self):
21         if 1 <= self.bobot <= 10:
22             return (0.10 * self.bobot) * self.jumlah
23         elif 10 < self.bobot <= 50:
24             return (0.08 * self.bobot) * self.jumlah
25         elif 50 < self.bobot <= 100:
26             return (0.05 * self.bobot) * self.jumlah
```

Penjelasan: Kelas **Ikan** berfungsi sebagai kerangka dasar untuk semua jenis ikan. Di dalamnya, terdapat informasi-informasi umum seperti jenis ikan, jumlah ikan, dan bobot ikan, serta metode-metode untuk menghitung kebutuhan pakan dan perkiraan hari panen.

2. Pewarisan (Inheritance)

Potongan Kode:



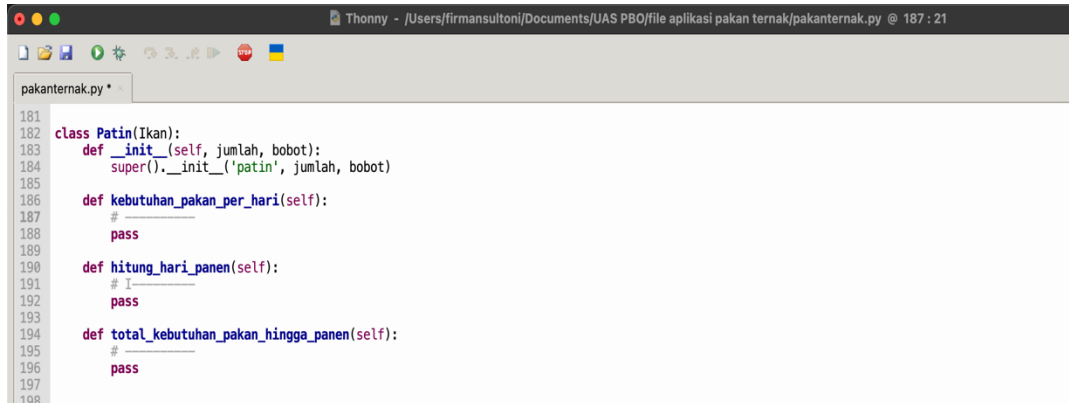
```
183 class Lele(Ikan):
184     def __init__(self, jumlah, bobot):
185         super().__init__('lele', jumlah, bobot)
186
187     def kebutuhan_pakan_per_hari(self):
188         # -----
189         pass
190
191     def hitung_hari_panen(self):
192         # -----
193         pass
194
195     def total_kebutuhan_pakan_hingga_panen(self):
196         # -----
197         pass
198
199
```

Penjelasan: Pewarisan memungkinkan kelas **Lele** untuk mewarisi atribut dan metode dari kelas **Ikan**. Hal ini memungkinkan penggunaan kembali kode yang telah ada, sehingga mengurangi duplikasi dan membuat kode lebih terorganisir.

3. Polymorphism

Potongan Kode:

python

A screenshot of a Thonny Python IDE window. The title bar shows the file path: "/Users/firmansulton/ Documents/ UAS PBO/ file aplikasi pakan ternak/ pakanternak.py @ 187 : 21". The code editor displays the following Python code:

```
181
182 class Patin(Ikan):
183     def __init__(self, jumlah, bobot):
184         super().__init__('patin', jumlah, bobot)
185
186     def kebutuhan_pakan_per_hari(self):
187         #
188         pass
189
190     def hitung_hari_panen(self):
191         #
192         pass
193
194     def total_kebutuhan_pakan_hingga_panen(self):
195         #
196         pass
197
198
```

Penjelasan: Polymorphism memungkinkan penggunaan metode yang sama dengan implementasi yang berbeda tergantung pada jenis ikan yang dipilih. Misalnya, saat memanggil metode **kebutuhan_pakan_per_hari** pada objek **Lele**, itu akan mengembalikan hasil sesuai dengan karakteristik khusus **Lele**, begitu juga dengan objek **Patin**.

dentiknya potongan kode yang menunjukkan prinsip pewarisan dengan potongan kode yang menunjukkan prinsip Polymorphism mungkin membuatnya sulit dibedakan. Namun, perbedaannya terletak pada bagaimana kelas-kelas tersebut digunakan dan diimplementasikan.

Dalam prinsip pewarisan, kelas anak (subclass) mewarisi atribut dan metode dari kelas induk (parent class), dan seringkali menambahkan metode atau atribut tambahan yang khusus untuk kelas anak tersebut. Ini memungkinkan penggunaan kembali kode dan memungkinkan kelas anak untuk memiliki perilaku yang sama atau mirip dengan kelas induknya.

Sementara dalam Polymorphism, kelas-kelas yang berbeda dapat menggunakan metode dengan nama yang sama, tetapi dapat memiliki implementasi yang berbeda. Misalnya, dalam kasus pewarisan yang telah disediakan, metode **kebutuhan_pakan_per_hari**, **hitung_hari_panen**, dan **total_kebutuhan_pakan_hingga_panen** mungkin memiliki implementasi yang berbeda antara kelas **Lele** dan **Patin**, tetapi keduanya memiliki nama yang sama.

Jadi, meskipun potongan kode yang menunjukkan prinsip pewarisan dan Polymorphism mungkin terlihat serupa, penting untuk memperhatikan bagaimana kelas-kelas tersebut digunakan dan diimplementasikan secara lebih spesifik untuk memahami perbedaan antara keduanya.

Daftar poin yang menunjukkan penerapan prinsip-prinsip Class, pewarisan, dan Polymorphism berdasarkan output yang diberikan:

```
Shell
>>> %Run pakanternak.py

Pilih jenis ikan (lele/patin): lele
Masukkan jumlah ikan : 1000
Masukkan bobot ikan hari ini (gram): 45
Masukkan stok pakan pelet (kg): 2000
Masukkan harga pakan per kg (dalam rupiah): 12000

Detail Pemeliharaan Ikan
Jenis Ikan: Lele
Jumlah Ikan: satu ribu ekor
Total Berat Ikan Hari ini: 45.0 gram
Kebutuhan Pakan / Hari hari ini: 3.60 kg
Perkiraan Hari Panen sampai 500 gram: 55 hari
Total Kebutuhan Pakan hingga Panen: 466.62 kg
Total Biaya Pakan: Rp 5,599,430.03
Sisa Pakan Hari ini : Rp 1996.4

>>>
```

1. Prinsip Class:

- Kelas Ikan digunakan sebagai kerangka dasar untuk semua jenis ikan.
- Kelas Lele adalah contoh dari subclass yang mewarisi atribut dan metode dari kelas Ikan, seperti yang terlihat dalam output dengan menampilkan informasi tentang jenis ikan, jumlah ikan, dan bobot ikan.

2. Pewarisan (Inheritance):

- Kelas Lele adalah subclass dari kelas Ikan, yang terlihat dari implementasi Lele(Ikan) saat objek Lele dibuat.
- Dalam output, kita melihat bahwa informasi tentang jenis ikan, jumlah ikan, dan bobot ikan diambil dari kelas Ikan, yang menunjukkan pewarisan atribut.

3. Polymorphism:

- Meskipun metode `kebutuhan_pakan_per_hari`, `hitung_hari_panen`, dan `total_kebutuhan_pakan_hingga_panen` dipanggil dengan nama yang sama, mereka memiliki implementasi yang berbeda tergantung pada jenis ikan yang dipilih. Ini mencerminkan Polymorphism, karena perilaku metode tersebut bervariasi tergantung pada konteksnya.

KESIMPULAN

Dengan menggunakan prinsip-prinsip Class, pewarisan, dan Polymorphism, program "Pemeliharaan Ikan" menjadi lebih terstruktur, mudah dimengerti, dan mudah diperluas. Prinsip Class memungkinkan kita untuk mengelompokkan data dan fungsi terkait ke dalam satu entitas yang lebih besar, membuat kode menjadi lebih terorganisir dan mudah dipelajari. Pewarisan memungkinkan kelas anak untuk mewarisi atribut dan metode dari kelas induk, mengurangi duplikasi kode dan meningkatkan efisiensi pengembangan perangkat lunak. Sementara itu, Polymorphism memungkinkan kita untuk menggunakan metode yang sama dengan implementasi yang berbeda tergantung pada konteks atau objek yang sedang digunakan, sehingga meningkatkan fleksibilitas dan modularitas kode.

Dengan struktur yang terorganisir dan konsep yang jelas, pengguna dapat dengan mudah mengelola informasi tentang ikan dan membuat keputusan yang tepat terkait dengan pemeliharaannya. Program ini memberikan pemahaman yang lebih baik tentang kebutuhan pakan harian, perkiraan hari panen, dan biaya pakan yang diperlukan untuk memelihara ikan tertentu. Dengan demikian, pemeliharaan ikan menjadi lebih efisien dan efektif, baik bagi para pemula maupun ahli dalam bidang ini.

Dengan demikian, penerapan prinsip-prinsip ini tidak hanya membuat program "Pemeliharaan Ikan" menjadi lebih baik dari segi teknis, tetapi juga meningkatkan pengalaman pengguna dan memberikan manfaat nyata dalam mengelola usaha pemeliharaan ikan secara keseluruhan.