

**LAPORAN TUGAS BESAR
PEMROGRAMAN BERORIENTASI OBJEK**

“STARBORNE STRIFE”



KELOMPOK 7:

**Silva Oktaria Putri - 122140085
Muklis Mustaqim - 122140115
Aulia Putri Sayyidina - 122140060
Zaky Ahmad Makarim - 122140182**

**PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNOLOGI INDUSTRI
INSTITUT TEKNOLOGI SUMATERA
2024**

DAFTAR ISI

● DESKRIPSI PENUGASAN.....	3
● NAMA GAME.....	4
● KATEGORI GAME.....	4
● ENTITAS.....	5
● DESKRIPSI GAME.....	6
● TAMPILAN GAME.....	6
● CLASS.....	9
● OBJEK.....	12
● ENKAPSULASI.....	13
● PEWARISAN.....	14
● POLIMORFISME.....	15
● ABSTRAKSI.....	17
● PENANGANAN KESALAHAN.....	19
● GRAPHICAL USER INTERFACE.....	19
● DIAGRAM UNIFIED MODELING LANGUAGE.....	20
● KODE.....	20
● LAMPIRAN.....	25

- **DESKRIPSI PENUGASAN**

No	Nama	NIM	Penugasan
1.	Muklis Mustaqim	122140115	<ul style="list-style-type: none"> • Code Maker ; Bertanggung jawab dalam pembuatan kode game • Desain ; Bertanggung jawab atas desain,sound dan visual game • Laporan ; Bertanggung jawab sepenuhnya atas pelaksanaan tugas dan penyusunan seluruh laporan serta dokumentasi yang terkait dengan proyek yang sedang berjalan
2.	Silva Oktaria Putri	122149985	<ul style="list-style-type: none"> • Code Maker; Bertanggung jawab dalam pembuatan kode game • Desain; Bertanggung jawab atas desain,sound dan visual game • Laporan ; Bertanggung jawab sepenuhnya atas pelaksanaan tugas dan penyusunan seluruh laporan serta dokumentasi yang terkait dengan proyek yang sedang berjalan • Pembuatan PPT

3.	Aulia putri Sayyidina	122140060	<ul style="list-style-type: none"> ● Code Check ; Bertanggung jawab dalam pengecekan kode game ● Laporan ; Bertanggung jawab sepenuhnya atas pelaksanaan tugas dan penyusunan seluruh laporan serta dokumentasi yang terkait dengan proyek yang sedang berjalan ● Pembuatan PPT
4.	Zaky Ahmad Makarim	122140182	<ul style="list-style-type: none"> ● Laporan ; Bertanggung jawab sepenuhnya atas pelaksanaan tugas dan penyusunan seluruh laporan serta dokumentasi yang terkait dengan proyek yang sedang berjalan ● Desain: Bertanggung jawab atas desain dan visual game ● Pembuatan UML

- **NAMA GAME**

Game ini diberi nama Starborne Strife. "Starborne" mengacu pada sesuatu yang berasal dari bintang atau terkait dengan bintang, sementara "Strife" mengacu pada konflik atau pertempuran. Jadi, secara keseluruhan, makna "Starborne Strife" adalah pertempuran atau konflik yang terjadi di dunia dengan bintang atau ruang angkasa.

- **KATEGORI GAME**

Kami memilih kategori arcade untuk game Starborne Strife ini karena popularitasnya yang tinggi pada era ini. Arcade memerlukan tingkat konsentrasi yang tinggi untuk meraih kemenangan, dan juga mendorong player untuk refleks

dan keterampilan menantang dalam gameplay ini. Itulah sebabnya kami memilih arcade sebagai kategori game ini.

- **ENTITAS**

1. Pilot - Player

Pilot merupakan player dari game Starborne Strife. Pilot akan menembakkan Blast pada Saucer dan Boss untuk mendapatkan skor. Jika Pilot terkena entitas Saucer dan Boss maka nyawa player akan berkurang.

2. Saucer - Ufo

Saucer merupakan musuh dari Pilot. Saucer akan muncul secara acak di hadapan Pilot jika Saucer terkena entitas Blast dari Pilot maka Saucer akan mati.

3. Energi - Peluru Tambahan

Energi merupakan peluru tambahan yang didapatkan Pilot dengan rentang waktu acak. Jika Pilot berinteraksi dengan entitas Energi maka akan menambahkan jumlah peluru yang dikeluarkan oleh Pilot.

4. Blast - Peluru Pilot

Blast merupakan peluru yang dikeluarkan oleh player yang berfungsi untuk menembak Saucer dan Boss. Jika Saucer dan Boss terkena Blast dari player, maka Saucer dan Boss akan mati.

5. Healthbar - Indikator Nyawa

Healthbar merupakan indikator yang menunjukkan jumlah “nyawa” yang dimiliki oleh player. Biasanya, indikator ini berbentuk bar dengan warna tertentu. Ketika player menerima kerusakan, jumlah nyawa akan berkurang secara bertahap. Batas maksimum nyawa biasanya ditetapkan dalam permainan, dan ketika nyawa mencapai nol, player kehilangan satu nyawa.

6. Boss - Boss Ufo

Boss merupakan boss ufo yang harus ditembak oleh player untuk menaikkan level kesulitan pada game. Karakter musuh yang jauh lebih kuat daripada musuh biasa. Mereka memiliki nyawa yang tinggi dan serangan yang mematikan.

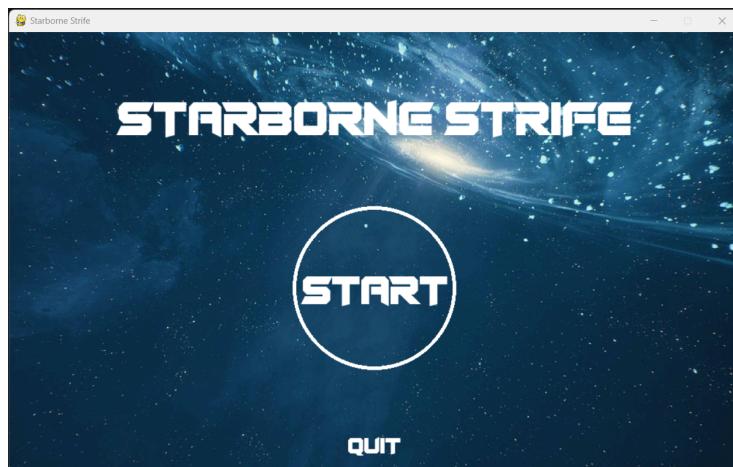
- **DESKRIPSI GAME**

Game Starborne Strife memiliki alur game yang dimana player mengendalikan sebuah Pilot untuk melawan musuh, mengumpulkan poin, dan bertahan selama mungkin dengan level tingkat kesulitan yang semakin tinggi ketika bertempur dengan Boss.

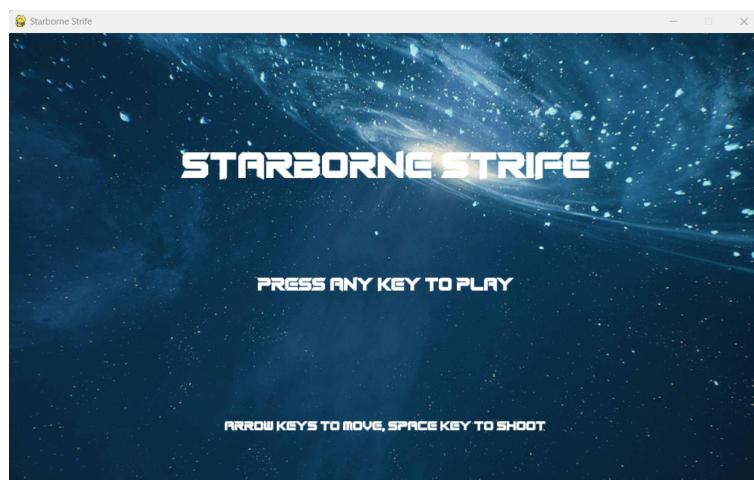
Player mengandalkan kendali fisik (joystick atau tombol) untuk menggerakan Pilot, menembak, dan menghindari serangan. Game ini, player harus berpikir cepat, menghindari tembakan musuh, dan menyerang balik dengan tepat. Kondisi ini mengharuskan layar, suara, dan kontrol harus berfungsi dengan baik agar player dapat menikmati pengalaman bermain yang optimal.

- **TAMPILAN GAME**

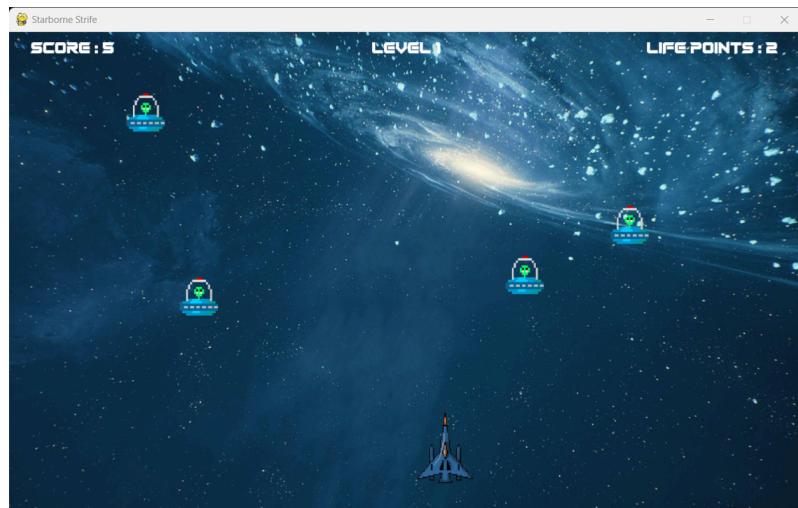
1. Tampilan Awal Game



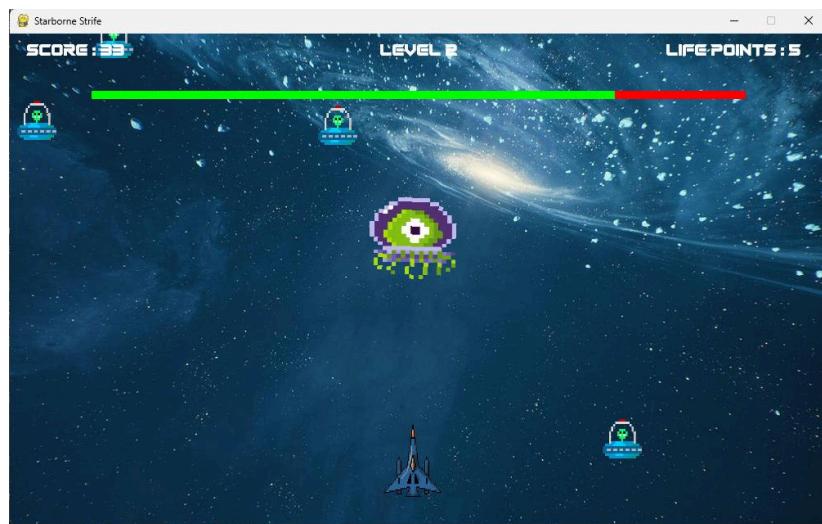
2. Tampilan Start Game



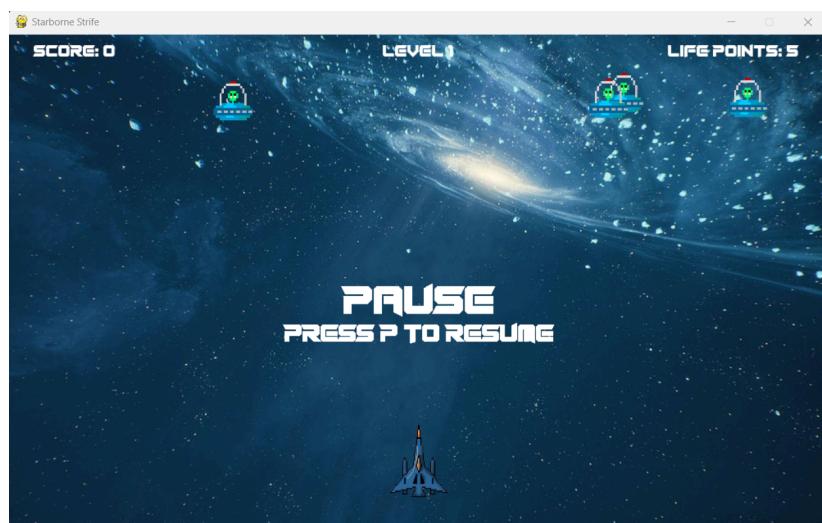
3. Tampilan Dalam Game



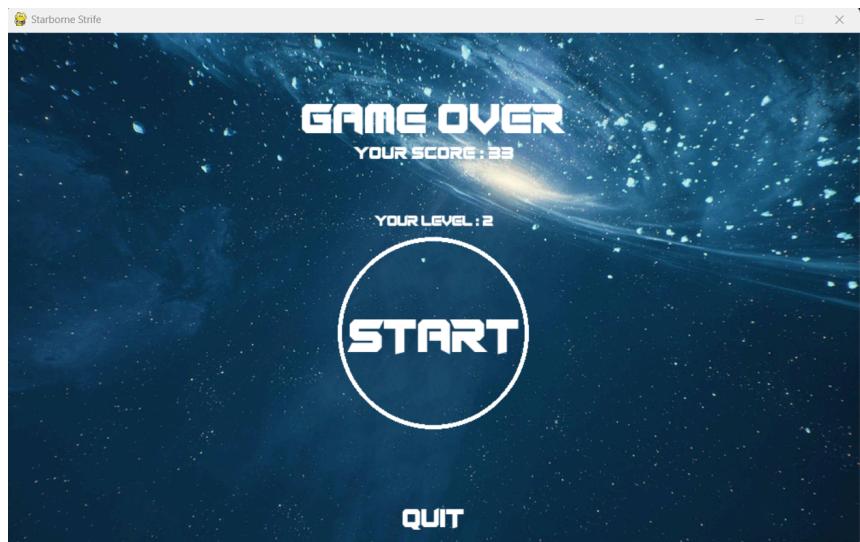
4. Tampilan Melawan GalBoss



5. Tampilan Pause



6. Tampilan Game Over



7. Tampilan aksi player, aksi musuh dan aksi Musuh Boss



Gambaran Pilot sebagai player.



Gambaran Saucer sebagai umpan musuh.



Gambaran Boss sebagai umpan terobosan.



Gambaran Blast sebagai peluru +1.



Gambaran Energi sebagai peluru +2.

- **CLASS**

Dalam game ini terdapat beberapa class yaitu :

1. Pilot: Class ini mewakili karakter player dalam permainan. Ini mengontrol perilaku dan tampilan dari karakter player. Termasuk dalam class ini adalah atribut dan metode yang mengatur gerakan player, penembakan, dan tampilan skor serta poin kehidupan.

```

43     #Class Pilot (Class Child)
44     class Pilot(pygame.sprite.Sprite):
45         def __init__(self):
46             pygame.sprite.Sprite.__init__(self)
47             self.image=pygame.transform.scale(image.pilot,(145,115))
48             self.rect=self.image.get_rect()
49             self.rect.centerx = WIDTH/2
50             self.rect.bottom= layar.get_height() - 20
51             self.speedx = 8
52             self.score_val = 0
53             self.life = 5
54             self.button = 1
55             self.button_time = pygame.time.get_ticks()
56             self.last_shot = pygame.time.get_ticks()
57             self.shoot_delay = 250
58             self.pause = False

```

2. Saucer : Class ini mewakili musuh atau objek yang akan ditembak oleh player.

Mereka memiliki perilaku acak dalam pergerakan dan kemunculan.

```

114     #Class Musuh atau Lawan
115     class Saucer(pygame.sprite.Sprite):
116         def __init__(self):
117             pygame.sprite.Sprite.__init__(self)
118             self.image=pygame.transform.scale(image.saucer,(80,95))
119             self.rect=self.image.get_rect()
120             self.radius=self.rect.width*0.1/2
121             self.rect.x=random.randrange(0,WIDTH-self.rect.width)
122             self.rect.y=random.randrange(-50,-10)
123             self.speedx=random.randrange(-1,2)
124             self.speedy=random.randrange(1,2)

```

3. Energi: Class ini mewakili peluru tambahan yang bisa didapatkan oleh player.

Mereka jatuh dari atas layar dan bisa ditembak oleh player untuk mendapatkan keuntungan tambahan.

```

137     #Class Energi atau Peluru Tambahan
138     class Energi(pygame.sprite.Sprite):
139         def __init__(self):
140             pygame.sprite.Sprite.__init__(self)
141             self.image = pygame.transform.scale(image.energi,(35,55))
142             self.rect = self.image.get_rect()
143             self.radius=self.rect.width*0.1/2
144             self.rect.x=random.randrange(0,WIDTH-self.rect.width)
145             self.speedy = 5

```

4. Blast: Class ini mewakili peluru yang ditembakkan oleh player atau musuh. Mereka memiliki perilaku bergerak secara linear dan akan dihapus dari permainan jika melewati batas layar.

```

154     #Class Blast(Class Child)
155     class Blast(pygame.sprite.Sprite):
156         def __init__(self,position:pygame.Vector2,angle:float=-90):
157             pygame.sprite.Sprite.__init__(self)
158             self.image=pygame.transform.rotate(pygame.transform.scale(image.blast,(25,35)), -angle+180+90)
159             self.rect=self.image.get_rect()
160             self.rect.midbottom=position
161             speedy = 10
162             self.velocity = pygame.math.Vector2(math.cos(math.radians(angle))*speedy,math.sin(math.radians(angle))*speedy)

```

5. Healthbar: Class ini mewakili tampilan bar untuk menunjukkan kesehatan (health) dari Boss dalam permainan.

```

149     class Healthbar(pygame.sprite.Sprite):
150         def __init__(self):
151             pygame.sprite.Sprite.__init__(self)
152             self.image = pygame.Surface((WIDTH*4/5, 10))
153             self.image.fill((255, 0, 0))
154             self.rect = self.image.get_rect()
155             self.rect.centerx = WIDTH/2
156             self.rect.bottom = 80

```

6. Boss: Class ini mewakili bos musuh dalam permainan. Mereka memiliki kesehatan (health) yang bisa dikurangi oleh player dan akan menembak balik ke arah player.

```

181     #Class Boss(Class Child)
182     class Boss(pygame.sprite.Sprite):
183         def __init__(self, max_health:int, attack_speed:int = 50):
184             pygame.sprite.Sprite.__init__(self)
185             self.source_image = pygame.transform.rotate(pygame.transform.scale(image.boss,(110,130)),90)
186             self._angle = 180
187             self.image = pygame.transform.rotate(self.source_image, self.angle)
188             self.rect=self.image.get_rect()
189             self.rect.centerx = WIDTH/2
190             self.rect.bottom = 0
191
192             self.max_health = max_health
193             self._health = 0
194             self.healthbar = Healthbar()
195             self.health = self.max_health
196             self.move_in = pygame.Vector2(0,15)
197             all_sprites.add(self.healthbar)
198             self.tick = 0
199             self.alt = False
200             self.attack_speed = attack_speed

```

Class ini mencakup fungsionalitas utama dari permainan "Starborne Strife" yang disediakan dalam kode.

- **OBJEK**

Objek atau instansiasi dalam pemrograman Python merupakan sebuah instance dari suatu class atau type data. Dalam kasus ini, kita memiliki beberapa objek yang merepresentasikan komponen game:

1. Pilot merupakan objek yang ada pada class Pilot(). Objek ini merupakan pesawat luar angkasa yang dikendalikan oleh player.
2. Saucer merupakan objek yang ada pada class Saucer(). Objek ini merupakan musuh yang dikendalikan oleh sistem dan muncul secara acak di hadapan player.
3. Energi merupakan objek yang ada pada class Energi(). Objek ini merupakan peluru tambahan untuk player. Objek akan muncul dengan rentang waktu acak.
4. Blast merupakan objek yang ada pada class Blast(). Objek ini merupakan peluru yang dikeluarkan oleh Pilot untuk menembak Saucer dan Boss.
5. HealthBar merupakan objek yang ada pada class Healthbar(). Objek ini merupakan jumlah nyawa yang dimiliki oleh player.
6. Boss merupakan objek yang ada pada class Boss(). Objek ini merupakan sebuah alien yang menembakan peluru pada player dan memiliki lebih banyak kesehatan dari pada musuh yang lainnya.

1. pilot = **Pilot()**
2. saucer = **Saucer()**
3. energi = **Energi()**
4. blast = **Blast()**
5. healthbar = **Healthbar()**
6. boss = **Boss()**

- **ENKAPSULASI**

Enkapsulasi adalah praktik menjaga bidang dalam class tetap pribadi, kemudian memberikan akses ke bidang tersebut melalui metode publik. Ini adalah penghalang pelindung yang menjaga data dan kode aman dari gangguan eksternal dan penyalahgunaan. Contoh enkapsulasi dalam kode:

1. Class Pilot: Class Pilot merangkum semua properti dan perilaku player dalam game. Ia memiliki atribut seperti *image*, *rect*, *speedx*, *score_val*, *life*, *button*, *button_time*, *last_shot*, *shoot_delay*, dan *pause*, yang bersifat pribadi untuk class. Class menyediakan metode seperti *update*, *buttonup*, *shoot*, *show_lifepoints*, dan *show_score* untuk berinteraksi dengan atribut ini.
2. Class Saucer : Class Saucer ini merangkum semua properti dan perilaku musuh di dalam game. Ia memiliki atribut seperti *image*, *rect*, *radius*, *speedx*, dan *speedy*, yang bersifat pribadi untuk class. Class menyediakan metode *update* untuk berinteraksi dengan atribut ini.
3. Class Energi: Class Energi merangkum semua properti dan perilaku tombol dalam game. Ia memiliki atribut seperti *image*, *rect*, dan *radius*, yang bersifat pribadi untuk class. Class menyediakan metode *update* untuk berinteraksi dengan atribut ini.
4. Class Blast: Class Blast merangkum semua properti dan perilaku peluru dalam game. class. Class menyediakan metode *update* untuk berinteraksi dengan atribut ini.
5. Class Healthbar: Class ini merangkum semua properti dan perilaku healthbar dalam game. Ia memiliki atribut seperti *image*, *rect*, yang bersifat pribadi untuk class. Class menyediakan metode *update* untuk berinteraksi dengan atribut ini.
6. Class Boss: Class GalBoss ini merangkum semua properti dan perilaku bos alien dalam game. Ia memiliki atribut seperti *source_image*, *_angle*, *image*, *rect*,

max_health, _health, healthbar, health, , move_in, tick, alt, attack_speed yang bersifat pribadi untuk class. Class menyediakan metode seperti *on_angle_change, hurt, shoot, update, kill* untuk berinteraksi dengan atribut ini.

Pada semua class ini, atribut dijaga kerahasiaannya dan hanya dapat diakses melalui metode yang disediakan, yang menjamin integritas dan keamanan data.

- **PEWARISAN**

Dalam kode yang diberikan, dapat melihat bahwa warisan digunakan di class berikut:

1. Class Pilot mewarisi dari pygame.sprite.Sprite
2. Class Saucer mewarisi dari pygame.sprite.Sprite
3. Class Energi mewarisi dari pygame.sprite.Sprite
4. Class Blast mewarisi dari pygame.sprite.Sprite
5. Class Healthbar mewarisi dari pygame.sprite.Sprite
6. Class Boss mewarisi dari pygame.sprite.Sprite

Alasan menggunakan pewarisan yang digunakan dalam kode ini untuk:

1. Reuse Code: Dengan mewarisi dari pygame.sprite.Sprite, masing-masing class ini dapat menggunakan kembali atribut dan metode umum yang ditentukan dalam Class Sprite, seperti *image, rect, dan update()*.
2. Establish a Hierarchy: Pewarisan membantu menetapkan hierarki class, di mana Child Classes (Pilot, Saucer, Energi, Blast, Healthbar, dan Boss) adalah versi khusus dari Parent Class (Sprite).
3. Improve Code Organization: Pewarisan membantu mengatur code dengan cara yang lebih terstruktur, sehingga lebih mudah untuk dipahami dan dipelihara.
4. Reduce Code Duplication: Dengan mewarisi atribut dan metode umum, kode menghindari duplikasi dan mempermudah pembaruan atau modifikasi perilaku beberapa class sekaligus.

Misalnya, Class Pilot mewarisi Sprite dan menambahkan atribut dan metode spesifiknya sendiri, seperti *score_val, life, dan shoot()*. Hal ini memungkinkan Class Pilot untuk menggunakan kembali atribut dan metode umum Sprite, sekaligus

menambahkan perilaku uniknya sendiri. Secara keseluruhan, penggunaan warisan dalam kode ini membantu menciptakan basis kode yang lebih terorganisir, mudah dikelola, dan efisien.

- **POLIMORFISME**

Polimorfisme adalah konsep pemrograman berorientasi objek yang kuat yang memungkinkan objek dari Class yang berbeda diperlakukan sebagai objek dari SuperClass yang sama. Dalam code yang ada, polimorfisme digunakan di Class Healthbar dan Class Boss.

1. Class Healthbar

Class Healthbar adalah Child Class `pygame.sprite.Sprite` dan digunakan untuk mewakili bar kesehatan objek permainan. Ini memiliki satu metode yaitu `update` yang digunakan untuk memperbarui posisi bilah kesehatan di layar.

Polimorfisme digunakan di Class Healthbar melalui konstruktornya. Konstruktor tidak mengambil argumen, tetapi menginisialisasi variabel instan `image` dengan objek `pygame.Surface`. Ini adalah contoh polimorfisme karena `image` variabel dapat digunakan untuk mewakili bilah kesehatan objek permainan apa pun, apa pun class objeknya.

```
class Healthbar(pygame.sprite.Sprite):
    def __init__(self):
        pygame.sprite.Sprite.__init__(self)
        # Polimorfisme: Metode pembaruan Healthbar diganti untuk memperbarui
        # gambar healthbar
        self.image = pygame.Surface((WIDTH*4/5, 10))
        self.image.fill((255, 0, 0))
        self.rect = self.image.get_rect()
        ...
    
```

2. Class Boss

Class Boss adalah Child Class `pygame.sprite.Sprite` dan digunakan untuk mewakili Boss dalam permainan. Ini memiliki beberapa variabel instan, termasuk `max_health`, `health`, `healthbar`, `move_in`, `tick`, `alt`, dan `attack_speed`.

Polimorfisme digunakan di Class Boss melalui konstruktor dan propertinya health. Konstruktor mengambil dua argumen, *max_health* dan *attack_speed*, dan menginisialisasi Boss objek dengan nilai-nilai. Properti health adalah properti baca-tulis yang mendapatkan dan menetapkan nilai *_health* variabel instan.

Properti ini health merupakan contoh polimorfisme karena memungkinkan Boss kesehatan objek diakses dan dimodifikasi menggunakan sintaksis yang sama dengan Healthbar kesehatan objek. Hal ini dimungkinkan karena health properti diimplementasikan sebagai properti, yang memungkinkannya diperlakukan sebagai atribut objek Boss.

Class Boss(pygame.sprite.Sprite):

```
def __init__(self, max_health:int, attack_speed:int = 50):
    pygame.sprite.Sprite.__init__(self)
    ...

    @property
    def health(self):
        return self._health

    @health.setter
    def health(self, value):
        self._health = value
        # Polimorfisme: Panggil metode pembaruan Healthbar untuk memperbarui
        # gambar healthbar
        self.healthbar.image.fill((255,0,0))
        self.healthbar.image.fill((0, 255, 0), (0, 0,
        self.healthbar.image.get_width()*self.health/self.max_health,
        self.healthbar.image.get_height())))

```

Singkatnya, polimorfisme digunakan di class Healthbar dan Boss untuk memungkinkan objek dari class yang berbeda diperlakukan sebagai objek dari superclass yang sama. Hal ini mempermudah penulisan kode yang dapat digunakan kembali untuk objek permainan yang berbeda, dan membuat kode lebih mudah dibaca..

- **ABSTRAKSI**

Terdapat beberapa abstraksi yang membantu mengatur dan menyusun permainan.

1. Class sebagai Abstraksi: Class digunakan untuk mewakili berbagai objek dalam game, seperti player, musuh, peluru, dan bar kesehatan. Setiap Class merangkum properti dan perilaku objek masing-masing
 - Player: Class ini mewakili karakter player. Ia memiliki properti seperti image, rect, speedx, score_val, life, dan button. Class juga memiliki metode seperti update(), shoot(), show_score(), dan show_lifepoints(). Metode-metode ini mengabstraksi perilaku spesifik player, seperti gerakan, menembak, dan menampilkan skor dan poin kehidupan.
 - Saucer: Class ini mewakili musuh. Ia memiliki properti seperti image, rect, radius, speedx, dan speedy. Class tersebut memiliki metode yang disebut update(), yang mengabstraksikan perilaku pergerakan musuh di sekitar layar.
 - Energi: Class ini mewakili peluru tambahan. Ia memiliki properti seperti image, rect, dan radius. Class tersebut memiliki metode yang disebut update(), yang mengabstraksikan perilaku untuk mengarahkan class agar vertikal di layar dan tidak keluar dari layar.
 - Blast: Class ini mewakili peluru yang ditembakkan oleh player. Ia memiliki properti seperti image, rect, dan velocity. Class tersebut memiliki metode yang disebut update(), yang mengabstraksi perilaku mengarahkan peluru ke arah tertentu.
 - Boss: Class ini mewakili bos musuh. Ia memiliki properti seperti max_health, health, healthbar, move_in, tick, alt, dan attack_speed. Class memiliki metode seperti hurt(), shoot(), dan update(). Metode-metode ini mengabstraksi perilaku menerima kerusakan, menembakkan peluru, dan menggerakkan bos musuh di sekitar layar.
 - Healthbar: Class ini mewakili bar kesehatan. Ia memiliki properti yang disebut image, yang digunakan untuk menampilkan bar kesehatan. Class tersebut memiliki metode yang disebut update(), yang mengabstraksikan perilaku memperbarui tampilan bar kesehatan berdasarkan kesehatan musuh saat ini.

2. Metode sebagai Abstraksi: Metode digunakan untuk mengabstraksi perilaku spesifik dari objek yang diwakili oleh class.
 - update(): Metode ini digunakan untuk memperbarui keadaan objek. Misalnya, update() metode di Class Pilot digunakan untuk menggerakkan player di sekitar layar, sedangkan update() metode di Class Saucer digunakan untuk menggerakkan musuh di sekitar layar.
 - shoot(): Metode ini digunakan untuk mengabstraksi perilaku menembakkan peluru. Misalnya, shoot() metode di Class Pilot digunakan untuk membuat objek peluru baru dan menambahkannya ke dalam game.
 - hurt(): Metode ini digunakan untuk mengabstraksi perilaku menerima kerusakan. Misalnya saja hurt() cara di Class Pilot yang digunakan untuk mengurangi health Boss musuh ketika terkena peluru.
 - kill(): Metode ini digunakan untuk mengabstraksi perilaku menghilangkan objek dari permainan. Misalnya, kill() metode di Class Boss digunakan untuk menghilangkan bos musuh dari permainan ketika kesehatannya mencapai nol.
 - buttonup(): Metode ini digunakan untuk menangani input pengguna dan merespons penekanan tombol dengan cara yang fleksibel dan dapat disesuaikan.
 - show_score() dan show_lifepoints(): Metode ini digunakan untuk mengabstraksi perilaku menampilkan skor dan poin nyawa player.
3. Fungsi sebagai Abstraksi: Fungsi digunakan untuk mengabstraksi logika permainan tertentu.
 - waiting_screen(): Fungsi ini digunakan untuk mengabstraksikan perilaku menampilkan layar tunggu sebelum permainan dimulai.
 - menu(): Fungsi ini digunakan untuk mengabstraksi perilaku menampilkan layar menu.
 - menuGameOver(): Fungsi ini digunakan untuk mengabstraksikan perilaku menampilkan layar game over ketika player kalah.

Abstraksi ini membantu mengatur kode, membuatnya lebih modular, dan lebih mudah untuk dipelihara dan diperluas. Dengan merangkum properti dan perilaku berbagai objek dalam game, kode menjadi lebih mudah dibaca dan dipahami. Selain itu, dengan mengabstraksikan perilaku tertentu dan logika permainan.

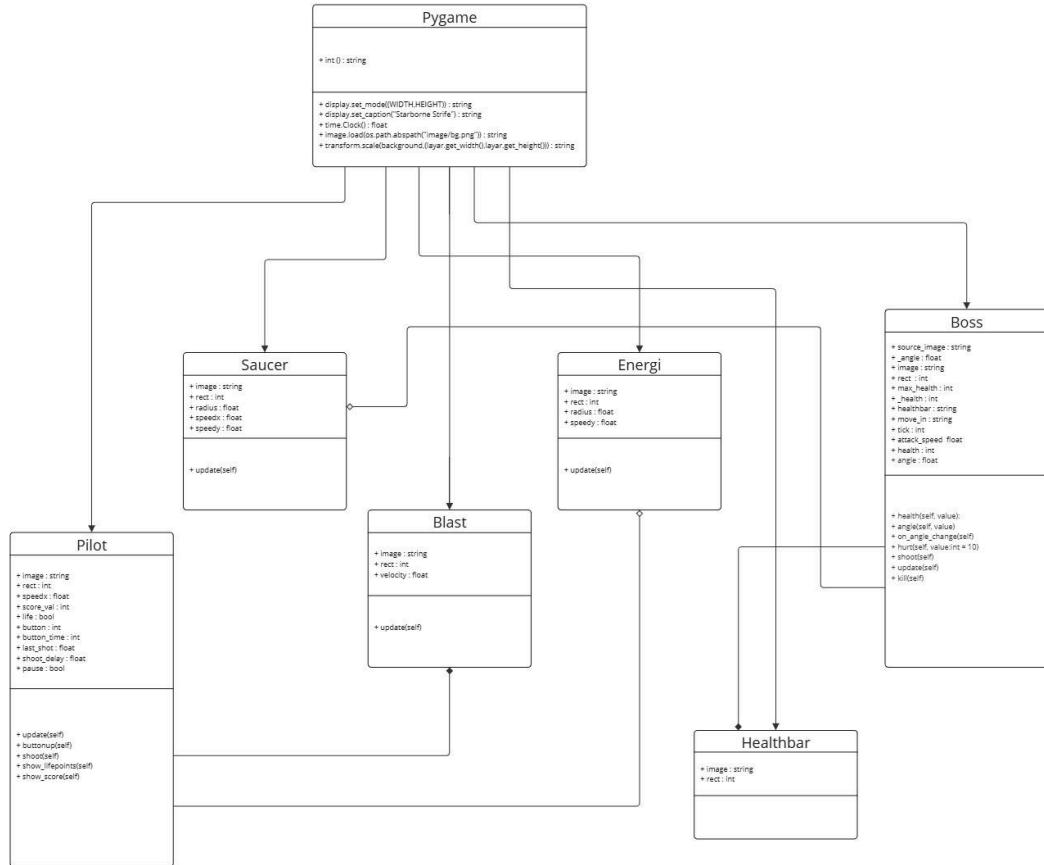
- **PENANGANAN KESALAHAN**

1. Kesalahan pertama terletak pada game yang awalnya game tidak dapat di pause dan Penanganan yang kita lakukan adalah membuat sistem Pause pada dalam game. Kemudian, ketika menekan tombol pause, game memang berhenti akan tetapi game tetap berjalan karena update dari display nya belum diberhentikan, maka dari itu penanganannya adalah dengan memberhentikan running displaynya menekan tombol pause sehingga semuanya berhenti.
2. Kesalahan kedua dimana kita membuat Kesulitan boss menjadi lebih sulit ketika naik level , dan itu membuat player susah mencapai tingkatan yang lebih tinggi, penanganan atau solusi yang kita berikan , kita memberikan sistem cheat untuk menambahkan life point agar player tidak cepat meninggalkan permainan.

- **GRAPHICAL USER INTERFACE**

GUI atau *Graphical User Interface* adalah jenis komponen yang menggunakan elemen visual untuk memungkinkan pengguna berinteraksi dengan suatu perangkat lunak. GUI yang digunakan pada game yang dibuat adalah pygame. Alasan menggunakan GUI pygame karena mudah digunakan dan dipahami, serta memiliki fitur yang cukup lengkap untuk mengembangkan game yang dibuat.

- **DIAGRAM UNIFIED MODELING LANGUAGE**



Class diagram atau diagram kelas adalah salah satu jenis diagram struktur pada UML yang menggambarkan dengan jelas struktur serta deskripsi class, atribut, metode, dan hubungan dari setiap objek.

- **KODE**

```

10     #fitur display/pembangun game
11     FPS=60
12     WIDTH=1000
13     HEIGHT=600
14     RED = (255, 255, 255, 255)
15
16     #class pygame (Class Utama/Parent)
17     pygame.init()
18     layar = pygame.display.set_mode((WIDTH,HEIGHT))
19     pygame.display.set_caption("Starborne Strife")
20     fps = pygame.time.Clock()
21     background = pygame.image.load(os.path.abspath("image/bg.png"))
22     background = pygame.transform.scale(background,(layar.get_width(),layar.get_height()))
23     pause = False
24
25     def pause_screen():
26         draw_text(layar, "Pause", 50, WIDTH / 2, HEIGHT / 2)
27         draw_text(layar, "Press P to Resume", 30, WIDTH / 2, HEIGHT / 2 + 50)
28         # pygame.display.flip()
  
```

Kode fitur display bertujuan untuk mengatur layar game dan fps yang diinginkan. Dalam Pygame, seluruh elemen yang terdapat pada layar, termasuk objek, grafik, dan animasi, dikelola oleh beberapa kelas yang ada di dalamnya. Class Pygame menjadi "parent" atau "class utama" dari semua kelas lain yang terkait dengan game. Pygame mengandung kelas yang menangani semua aspek dari pengembangan game, seperti gambar, animasi, suara, input pengguna, dan masih banyak lagi. Fungsi pause_screen() menggambar layar yang menandakan bahwa game sedang di-pause. Untuk menjalankannya harus memanggilnya dalam perulangan utama game atau saat game di-pause.

```
30  def pause_game(self):
31      self.game_pause = True
32
33      while self.game.is_pause:
34          PauseMenu()
35          for event in pygame.event.get():
36              if event.type == pygame.KEYDOWN:
37                  if event.key == pygame.K_p:
38                      self.game.is_pause = False
39
39          if event.type == pygame.QUIT:
40              self.game.is_pause = False
41              self.run()
```

Kode ini menunda permainan dan menampilkan menu pause. Menetapkan game_pause ke True dan memasuki loop yang terus berlanjut hingga permainan dinyalakan kembali. Dalam loop, dia menampilkan menu pause dan mengecek acara. Jika tombol 'p' ditekan, dia menyala kembali permainan. Jika permainan ditutup, dia menyala kembali permainan dan mengulangnya.

```

43     #Class Pilot (Class Child)
44     class Pilot(pygame.sprite.Sprite):
45         def __init__(self):
46             pygame.sprite.Sprite.__init__(self)
47             self.image=pygame.transform.scale(image.pilot,(145,115))
48             self.rect=self.image.get_rect()
49             self.rect.centerx = WIDTH/2
50             self.rect.bottom= layar.get_height() - 20
51             self.speedx = 8
52             self.score_val = 0
53             self.life = 5
54             self.button = 1
55             self.button_time = pygame.time.get_ticks()
56             self.last_shot = pygame.time.get_ticks()
57             self.shoot_delay = 250
58             self.pause = False

```

Kode diatas mendefinisikan Class Pilot yang merupakan karakter player dalam permainan Starborne Strife untuk menginisialisasi tampilan visual, posisi, kecepatan gerakan, skor, jumlah nyawa, dan beberapa atribut yang terkait dengan penekanan tombol dan mekanisme menembak.

```

14     #Class Musuh atau Lawan
15     class Saucer(pygame.sprite.Sprite):
16         def __init__(self):
17             pygame.sprite.Sprite.__init__(self)
18             self.image=pygame.transform.scale(image.saucer,(80,95))
19             self.rect=self.image.get_rect()
20             self.radius=self.rect.width*0.1/2
21             self.rect.x=random.randrange(0,WIDTH-self.rect.width)
22             self.rect.y=random.randrange(-50,-10)
23             self.speedx=random.randrange(-1,2)
24             self.speedy=random.randrange(1,2)

```

Kode diatas mewakili objek grafis pada musuh yang dapat dipindahkan di layar dengan sudut dan kecepatan tertentu. Gambar objek diatur ke versi skala dan diputar dari gambar yang telah ditentukan, dan posisi awal dan kecepatannya dihitung berdasarkan argumen dan diteruskan ke konstruktor.

```

60      #atributed movement
61  ↘  def update(self):
62      if self.button >= 2 and pygame.time.get_ticks() - self.button_time > 5000:
63          self.button -= 1
64          self.button_time = pygame.time.get_ticks()
65
66      key_pressed = pygame.key.get_pressed()
67      if not updated:
68          if key_pressed[pygame.K_RIGHT]:
69              self.rect.x += self.speedx
70          if key_pressed[pygame.K_LEFT]:
71              self.rect.x -= self.speedx
72          if key_pressed[pygame.K_UP]:
73              self.rect.y -= self.speedx
74          if key_pressed[pygame.K_DOWN]:
75              self.rect.y += self.speedx
76
77          if self.rect.right > WIDTH:
78              self.rect.right = WIDTH
79          if self.rect.left < 0:
80              self.rect.left = 0
81          if self.rect.bottom > layar.get_height():
82              self.rect.bottom = layar.get_height()
83          if self.rect.top < 0:
84              self.rect.top = 0

```

Bagian ini merupakan metode update yang bertujuan untuk membantu memproses input pada tiap masing masing class seperti Class Pilot, Saucer, Energi, Blast, dan Boss untuk bergerak dalam game, mengatur frame, mengatur kecepatan, mengatur posisi agar tidak melewati batas layar yang sudah ditetapkan.

```

295      waiting = True
296      while waiting:
297          fps.tick(FPS)
298          for event in pygame.event.get():
299              if event.type == pygame.QUIT:
300                  pygame.quit()
301                  sys.exit()
302              if event.type == pygame.MOUSEBUTTONDOWN:
303                  xpos, ypos = pygame.mouse.get_pos()
304                  cek = math.sqrt((xvar - xpos)**2 + (yvar - ypos)**2)
305                  if cek <= 70:
306                      waiting = False
307                      waiting_screen()
308                      return False
309              if event.type == pygame.MOUSEBUTTONDOWN:
310                  xpos, ypos = pygame.mouse.get_pos()
311                  cek = math.sqrt((xvar - xpos)**2 + (550 - ypos)**2)
312                  if cek <= 25:
313                      pygame.quit()
314                      sys.exit()
315                  pygame.display.update()

```

Bagian ini berfungsi untuk interaksi player dengan tombol-tombol GUI yang ada di menu utama. Ketika player mengklik tombol Start, maka akan menuju ke waiting screen. Jika player mengklik tombol Quit, maka player akan keluar dari permainan.

```

267  def waiting_screen():
268      layar.blit(pygame.transform.scale(image.background,(layar.get_width(),layar.get_height()),(0,0))
269      draw_text(layar, "STARBORNE STRIFE", 50, WIDTH/2, HEIGHT/4)
270      draw_text(layar, "Press any key to play", 25, WIDTH/2, HEIGHT/2+20)
271      draw_text(layar, "Arrow keys to move, Space key to Shoot", 18, WIDTH/2, HEIGHT*3/4+60)
272
273      pygame.display.flip()
274      waiting = True
275      while waiting:
276          fps.tick(FPS)
277          for event in pygame.event.get():
278              if event.type == pygame.QUIT:
279                  pygame.quit()
280                  sys.exit()
281              if event.type == pygame.KEYUP:
282                  waiting = False

```

Kode diatas merupakan waiting screen yang berfungsi untuk menunggu kesiapan player untuk memulai permainan. Untuk memulai permainan, player harus mengklik tombol apa saja

```

317  #Tampilan ketika GameOver
318  def menuGameOver():
319      layar.blit(pygame.transform.scale(image.background,(layar.get_width(),layar.get_height()),(0,0))
320      draw_text(layar, "Game Over", 50, WIDTH/2, HEIGHT/8)
321
322      yvar=350
323      xvar=500
324
325      draw_text(layar, "START", 55, WIDTH/2, yvar-25)
326      draw_text(layar, f"Your score : {pilot.score_val}", 20, WIDTH/2, 130)
327      draw_text(layar, f"Your level : {level}", 17, WIDTH/2, 210)
328      draw_text(layar, "QUIT",30, WIDTH/2, 550)
329      pygame.draw.circle(layar, (RED), (xvar,yvar), 112,5)
330
331      waiting = True
332      while waiting:
333          fps.tick(FPS)
334          for event in pygame.event.get():
335              if event.type == pygame.QUIT:
336                  pygame.quit()
337                  sys.exit()
338              if event.type == pygame.MOUSEBUTTONDOWN:
339                  xpos, ypos = pygame.mouse.get_pos()
340                  cek = math.sqrt((xvar - xpos)**2 + (yvar - ypos)**2)
341                  if cek <= 70:
342                      pilot.score_val = 0
343                      waiting = False
344                  if event.type == pygame.MOUSEBUTTONDOWN:
345                      xpos, ypos = pygame.mouse.get_pos()
346                      cek = math.sqrt((xvar - xpos)**2 + (550 - ypos)**2)
347                      if cek <= 25:
348                          pygame.quit()
349                          sys.exit()
350
351      pygame.display.update()

```

Kode diatas merupakan tampilan saat GameOver dengan kondisi player tidak memiliki nyawa yang tersisa. Tampilan ini berisi skor yang diperoleh, level yang dicapai, tombol memulai kembali, dan tombol untuk keluar dari permainan.

- **LAMPIRAN**

[*Game-PBO-KELOMPOK-7*](#)