**Script Presentasi - 2 Orang**

**Pembukaan**

**Orang 1:**

* Selamat pagi/siang/sore semua. Terima kasih sudah meluangkan waktu untuk mendengarkan presentasi kami. Pada kesempatan Kali ini, kami akan membahas tentang **Implementasi Transformasi Objek 2D Pygame**, yang mencakup **scaling**, **translasi**, dan **refleksi**.
* Kami akan menjelaskan bagaimana kita bisa menerapkan tiga jenis transformasi geometris ini dalam grafika komputer, khususnya dalam pembuatan objek 2D menggunakan **Pygame**.
* Untuk pengembangan proyek ini, kami menggunakan **Visual Studio Code** sebagai editor kode. Visual Studio Code adalah editor teks sumber terbuka yang sangat populer dan memiliki banyak fitur seperti debugging, kontrol versi Git, dan berbagai ekstensi yang membantu dalam pengembangan proyek dengan Python, termasuk Pygame. Dengan Visual Studio Code, kami dapat menulis dan menguji kode dengan efisien, mengelola file proyek, serta melakukan debugging yang lebih mudah

**Pendahuluan**

**Orang 2:**

* Sebelum kita masuk ke detail implementasi, mari kita bahas dulu apa itu transformasi geometris.
* **Transformasi geometris** adalah perubahan posisi atau ukuran objek dalam ruang. Pada proyek ini, kami fokus pada transformasi dalam dua dimensi, yaitu **scaling** (perbesaran dan pengecilan), **translasi** (perpindahan), dan **refleksi** (pembalikan).
* Kita menggunakan **Pygame**, sebuah library Python, untuk mengembangkan game atau aplikasi grafis lainnya. Di sini, kita menggunakan tiga jenis transformasi untuk memanipulasi objek gambar dalam game.

**Isi Utama**

**Orang 1:**

**Sebelum Itu Kami Sudah Membuat Projek Nya Sudah Dihilangkan Segala Transformasi Nya Dan Kodingan Nya Seperti Gambar Ini:**

**JEDA MENUNJUKAN KODING**

**Dan Setelah Di Run Nanti Akan Muncul Output Seperti Gambar Disamping:**

**JEDA BENTAR**

**Bisa Kita Lihat Ketiga Objek Disini Burung Sapi Dan Kincir Air Tidak Ada Pergerakan Sama Sekali Yang Berarti Belum Terjadi Transformasi Apapun Hanya Diam Di Tempat Saja**

**Orang 1:**

* **Baiklah Selanjut Nya Kita Akan Menambahkan Transformasi Scalling Pada Objek Sapi.**
* **Scalling** atau **perbesaran dan pengecilan** adalah transformasi yang mengubah ukuran objek. Pada proyek ini, kami menerapkan **scaling** pada objek **kincir angin**, dengan tujuan untuk memperbesar objek tersebut secara bertahap.
* **Nah Kita Akan Menambahkan Kode Ini Pada kodingan Kita :**

|  |  |
| --- | --- |
| Menambah Variabel untuk Kontrol Scaling | # Variabel untuk kontrol gambar yang tumbuh  **growing\_scale\_factor = 1**  # Faktor skala untuk gambar yang tumbuh  **growing\_scale\_speed = 0.01**  # Kecepatan pertumbuhan gambar |
| Menambah Logika Pembesaran Gambar dalam Fungsi draw\_growing\_image | Fungsi draw\_growing\_image sebelumnya mengubah ukuran gambar berdasarkan faktor skala yang dihitung. Kami menghapus proses scaling gambar ini dan menggambarnya dengan ukuran asli. Berikut adalah fungsi yang diubah:  # Fungsi untuk menggambar gambar yang tumbuh  **def draw\_growing\_image(screen, growing\_image, x, y, scale\_factor):**  # Skalakan gambar untuk membuatnya tumbuh  **new\_width = int(growing\_image.get\_width() \* scale\_factor)**  **new\_height = int(growing\_image.get\_height() \* scale\_factor)**  **scaled\_image = pygame.transform.scale(growing\_image, (new\_width, new\_height))**  **screen.blit(scaled\_image, (x - new\_width // 2, y - new\_height // 2))** |
| Menambah Pemanggilan Fungsi draw\_growing\_image | # Menggambar gambar yang tumbuh di kanan bawah  **draw\_growing\_image(screen, growing\_image, SCREEN\_WIDTH - 50, SCREEN\_HEIGHT - 50, growing\_scale\_factor)** |

* Dan Setelah Itu Kita Run Program Kita Dan Akan Muncul Output Seperti Di Video Ini, . JEDA BENTAR LIAT VIDEO
* Bisa Kita Lihat Objek Sapi Nya Melakukan Proses Scalling Yaitu Memperbesar Dan Memperkecil Objek Tersebut..

**Orang 2:**

* Selanjutnya, kita punya **Translasi**, yang artinya memindahkan objek dari satu tempat ke tempat lain. Pada proyek ini, kami menggunakan translasi untuk memindahkan objek **burung** (kapal terbang) ke kiri dan ke kanan secara otomatis.
* Burung yang bergerak mengikuti kontrol dari pengguna, menggunakan perhitungan posisi objek berdasarkan waktu dan interaksi dengan keyboard. Fungsi utama yang digunakan adalah pygame.transform.translate() untuk memindahkan posisi objek
* Lebih Lengkap Nya Kita Akan Menambah Kan Kode Ini Untuk Memberikan Translasi Pada Objek Burung:

|  |  |
| --- | --- |
| Penghapusan Variabel untuk Kontrol Pergerakan | Variabel dy dan dx yang digunakan untuk mengontrol pergerakan pesawat dihapus, karena pesawat tidak bergerak lagi.  # Kecepatan pergerakan kapal terbang  **move\_speed = 3**  # Variabel untuk kontrol pergerakan  # Pergerakan vertikal (naik)  **dy = -move\_speed**  # Pergerakan horizontal (ke kiri/kanan nanti)  **dx = 0** |
| Penghapusan Fungsi dan Bagian Pergerakan Pesawat: | * Fungsi move\_plane() yang digunakan untuk mengubah posisi pesawat dihapus, karena pesawat tidak bergerak lagi.   Bagian kode yang memanggil   * fungsi move\_plane() untuk mengubah posisi pesawat dihapus.   # Fungsi untuk pergerakan kapal **terbang**  **def move\_plane(x, y, dx, dy):**  **x += dx**  **y += dy**    **# Membatasi pergerakan agar objek tidak keluar dari layar**  **if x < 0: # Jika pesawat melewati tepi kiri**  **x = 0**  **if x > SCREEN\_WIDTH: # Jika pesawat melewati tepi kanan**  **x = SCREEN\_WIDTH**  **if y < 0: # Jika pesawat melewati tepi atas**  **y = 0**  **if y > SCREEN\_HEIGHT: # Jika pesawat melewati tepi bawah**  **y = SCREEN\_HEIGHT**    **return x, y** |
| Bagian Pengaturan Pergerakan Pesawat (Loop Utama): | * + Bagian kode yang mengubah posisi pesawat secara otomatis (naik, kiri, kanan) dihapus.   + Variabel moving\_left dan logika pergerakan pesawat ke kiri atau kanan dihapus.   # Pergerakan otomatis pesawat  # Pertama pesawat naik  **if plane\_rect.centery > 100: # Pesawat bergerak naik**  **plane\_rect.centerx, plane\_rect.centery = move\_plane(plane\_rect.centerx, plane\_rect.centery, dx, dy)**  **else:**  # Setelah mencapai bagian atas, pesawat mulai bergerak ke kiri dan kanan  **if not moving\_left:**  **dx = move\_speed # Gerak ke kanan**  **else:**  **dx = -move\_speed # Gerak ke kiri**  # Menggerakkan pesawat ke kiri/kanan  **plane\_rect.centerx, plane\_rect.centery = move\_plane(plane\_rect.centerx, plane\_rect.centery, dx, 0)**  # Setelah pesawat mencapai sisi kiri atau kanan, ubah arah gerakan  **if plane\_rect.centerx >= SCREEN\_WIDTH - 50:**  **moving\_left = True**  # Mulai bergerak ke kiri  **elif plane\_rect.centerx <= 50:** # Deteksi jika pesawat mencapai ujung kiri  **moving\_left = False**  # Mulai bergerak ke kanan |
| Variabel moving\_left dan game\_over: | * + Karena pesawat sekarang tetap diam, variabel moving\_left dan game\_over menjadi tidak digunakan dan bisa dihapus.   Setelah perubahan ini, pesawat tetap berada di tengah layar tanpa bergerak, dan tidak ada kontrol pergerakan yang dijalankan dalam loop utama. |

* Dan Bisa Kita Lihat Di Sini, Objek Burung Nya Bergerak Secara Vertikal Dan Horizontal Mengikuti Perintah Kita Di Kodingan Nya. Dan Itulah Output Yang Terjadi Ketika Kita Memasukan Translasi Pada Objek Burung.

**Orang 1:**

* Selanjutnya Kita Akan Menambah Transformasi Refleksi Pada Objek Kincir Air Di Projek Kita
* **Refleksi** adalah transformasi di mana objek dibalik secara simetris, baik secara horizontal atau vertikal. Pada proyek ini, kami mengimplementasikan refleksi untuk objek **kincir angin**. Objek tersebut dapat dibalik secara horizontal, memberi efek visual yang menarik.
* **Contoh**: Bayangkan objek yang seharusnya menghadap ke kanan, dibalik sehingga menghadap ke kiri. Kami menggunakan pygame.transform.flip() untuk melakukan transformasi ini
* **Lebih Lengkap Nya Kami Menambahkan Kode Ini Pada Kodingan Nya**:

|  |  |
| --- | --- |
| Penambahan Variabel untuk Status Refleksi | Menambah Variabel **reflected** digunakan untuk mengontrol apakah gambar kincir harus dibalik (*flipped*) atau tidak.  **# Variabel** untuk kontrol refleksi kincir angina |
| Modifikasi Fungsi draw\_windmill | Fungsi ini diperbarui agar dapat menggambar gambar kincir dengan atau tanpa refleksi berdasarkan nilai variabel reflected.  # Fungsi untuk menggambar kincir angin dengan refleksi  **def draw\_windmill(screen, windmill\_image, x, y, reflected=False):**  **if reflected:**  # Membalik gambar secara horizontal  **flipped\_image = pygame.transform.flip(windmill\_image, True, False)**  **screen.blit(flipped\_image, (x - flipped\_image.get\_width() // 2, y - flipped\_image.get\_height() // 2))**  **else**  # Gambar tanpa pembalikan  **screen.blit(windmill\_image, (x - windmill\_image.get\_width() // 2, y - windmill\_image.get\_height() // 2))** |
| Logika untuk Mengubah Status Refleksi Secara Berkala | Logika ini menggunakan waktu untuk mengubah status reflected setiap 3 detik.  # Perubahan refleksi kincir angin setiap 3 detik  **if pygame.time.get\_ticks() % 3000 <50:**  # Modul waktu untuk bergantian refleksi  **reflected = not reflected** |
| Pemanggilan Fungsi draw\_windmill dengan Parameter Refleksi | Fungsi ini dipanggil dalam *loop* utama dengan parameter reflected.  # Menggambar kincir angin dengan status refleksi  **draw\_windmill(screen, windmill\_image, windmill\_rect.centerx, windmill\_rect.centery, reflected)** |

* Dan Setelah Kita Run Code Nya Bisa Kita Lihat Di Sini

JEDA BENTAR

Objek Kincir Air Nya Melakukan Refleksi Atau Pencerminan Dimana Kincir Air Nya Bergerak Berlawanan Arah Dengan Bentuk AsliNya

**Orang 2:**

* Dan Terakhir Setelah Kita Menambahkan Semua Transformasi Nya YaituTranslasi, Scalling Dan Refleksi Maka Kodingan Kita Telah Selesai Dan Ini Lah Hasil Final Nya Setelah Menambahkan Semua Transformasi Nya:
* JEDA BENTAR LIAT KODING AKHIRNYA
* Dan Jika Kita Run Kodinganya Maka Hasilnya Akan Seperti Ini:
* JEDA LIAT VIDEO

Bisa Kita Lihat Disana Sudah Terjadi 3 Transformasi Nya Atau Sudah Lengkap Transformasi Nya Yaitu Translasi Pada Objek Burung, Scalling Pada Objek Sapi Dan Refleksi Pada Objek Kincir Air Nya.

**Kesimpulan**

**Orang 1:**

* Jadi, melalui proyek ini, kami berhasil menerapkan dan menguji tiga transformasi geometris yang sangat penting dalam pembuatan game atau aplikasi grafis: **scaling**, **translasi**, dan **refleksi**. Kami menggunakan Pygame untuk mengubah ukuran, posisi, dan orientasi objek secara dinamis.
* Dengan menguasai transformasi ini, kita bisa menciptakan pengalaman visual yang menarik dan interaktif dalam game.

**Orang 2:**

* Untuk referensi lebih lanjut, kami menggunakan dokumentasi resmi Pygame dan beberapa buku tentang pengembangan game serta matematika dalam grafika komputer.
* Mungkin Hanya Ini Saja Yang Bisa Kami Sampai kan
* Terima kasih atas perhatiandan Mohon Maaf Jika Ada Salah Terima Kasih