**GameMode (game.py)**

GameMode adalah kelas dasar yang mendefinisikan struktur dasar untuk berbagai mode permainan. Kelas ini berisi metode `handle\_event`, `update`, dan `draw` yang dapat diimplementasikan oleh subkelas untuk mengelola event, memperbarui status, dan menggambar konten pada layar masing-masing.

**Menu (game.py)**

Menu adalah subkelas dari GameMode yang menampilkan menu utama permainan. Kelas ini menangani event untuk navigasi menu menggunakan tombol (misalnya, '1' untuk mode solo, '2' untuk mode dual, 's' untuk pengaturan), dan menggambar elemen menu pada layar seperti judul dan opsi menu.

**SettingsScreen (game.py)**

SettingsScreen adalah subkelas dari GameMode yang menyediakan layar pengaturan permainan. Kelas ini menangani event untuk mengubah pengaturan seperti mengaktifkan/mematikan musik (tombol 'm') dan mengganti tema (tombol 't'), serta menggambar elemen pengaturan pada layar.

**SoloTetrisGame (game.py)**

SoloTetrisGame adalah subkelas dari GameMode yang mengelola permainan Tetris untuk satu pemain. Kelas ini menangani event untuk mengontrol potongan Tetris (misalnya, memindahkan atau memutar potongan), memperbarui status permainan, dan menggambar elemen permainan seperti grid dan potongan Tetris pada layar.

**DualTetrisGame (game.py)**

DualTetrisGame adalah subkelas dari GameMode yang mengelola permainan Tetris untuk dua pemain. Kelas ini menangani event untuk mengontrol dua set potongan Tetris secara terpisah, memperbarui status permainan untuk kedua pemain, dan menggambar elemen permainan untuk kedua pemain pada layar.

**GameOverScreen (game.py)**

GameOverScreen adalah subkelas dari GameMode yang menampilkan layar game over. Kelas ini menangani event untuk kembali ke menu utama (tombol 'r') dan menggambar teks "Game Over" serta informasi lain yang relevan pada layar.

**PlayerWinScreen (game.py)**

PlayerWinScreen adalah subkelas dari GameMode yang menampilkan layar kemenangan untuk salah satu pemain. Kelas ini menangani event untuk kembali ke menu utama (tombol 'r') dan menggambar pesan kemenangan serta informasi lain yang relevan pada layar.

**Tetris (tetris.py)**

Tetris adalah kelas utama yang mengelola logika permainan Tetris. Kelas ini mencakup metode untuk menginisialisasi permainan, menangani kontrol pemain, memutar dan memindahkan potongan, mendeteksi dan mengunci potongan, membersihkan garis, memperbarui status permainan, dan menggambar elemen permainan pada layar.

**Settings (settings.py)**

Settings adalah kelas yang mengelola pengaturan permainan seperti tema dan musik. Kelas ini menyediakan metode untuk mengatur tema, mengaktifkan/mematikan musik, dan mendapatkan detail tema saat ini.

**themes (themes.py)**

Themes adalah kamus yang mendefinisikan dua tema visual untuk permainan: 'default' dan 'light'. Setiap tema mencakup pengaturan warna latar belakang, warna teks, warna blok, serta jalur gambar untuk latar belakang dan blok.

**utils.py**

utils.py berisi fungsi-fungsi utilitas yang mendukung berbagai operasi dalam permainan, seperti memuat dan memutar musik, menggambar teks pada layar, dan memuat gambar. Kelas ini juga mendefinisikan konstanta FONT yang menunjuk ke jalur file font yang digunakan dalam permainan. **Polimorfisme (Polymorphism)**

Polimorfisme memungkinkan objek dari berbagai kelas untuk diperlakukan sebagai objek dari kelas dasar yang sama. Ini mempromosikan fleksibilitas dan kemampuan untuk memperluas kode.

game\_modes menyimpan objek dari berbagai kelas (Menu, SettingsScreen, SoloTetrisGame, dll) yang semuanya merupakan turunan dari GameMode. Polimorfisme memungkinkan main.py untuk memanggil metode handle\_event, update, dan draw pada objek ini tanpa mengetahui kelas spesifik dari objek tersebut.

**Enkapsulasi (Encapsulation)**

Enkapsulasi mengacu pada pengemasan data dan metode yang beroperasi pada data tersebut ke dalam satu unit, seperti kelas. Ini juga melibatkan penyembunyian detail implementasi dari pengguna.

Contoh : Kelas Settings mengenkapsulasi properti dan metode terkait pengaturan permainan. Detail implementasi pengaturan tema dan musik disembunyikan dari pengguna kelas.

Contoh: Pada kelas Tetris, data seperti grid, current\_piece, dan metode untuk memanipulasi mereka (seperti move\_piece, rotate\_piece, dll) semuanya didefinisikan di dalam kelas tersebut. Ini adalah contoh enkapsulasi karena data dan metode terkait dikumpulkan dalam satu unit.

**Abstraksi (Abstraction)**

Abstraksi melibatkan penyembunyian detail implementasi dan hanya menampilkan fungsionalitas penting kepada pengguna. Hal ini dicapai melalui kelas abstrak dan metode abstrak.

Contoh: Kelas GameMode adalah kelas abstrak yang menggunakan dekorator @abstractmethod untuk metode handle\_event dan draw. Ini memaksa semua subclass untuk mengimplementasikan metode-metode ini, menyembunyikan detail implementasi dari pengguna.

**Pewarisan (Inheritence)**

Pewarisan memungkinkan satu kelas untuk mewarisi properti dan metode dari kelas lain. Ini mempromosikan penggunaan kembali kode.

Kelas Menu mewarisi dari GameMode, sehingga Menu memiliki semua properti dan metode dari GameMode, serta dapat menambahkan atau mengganti metode tersebut.