**Tugas Besar 3 Pemrograman Berorientasi Objek**

**LAPORAN**

Diajukan untuk memenuhi Tugas Besar 3 mata kuliah IF2210 Pemrograman Berorientasi Objek

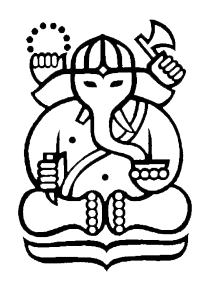
oleh

**ERICK WIJAYA 13515057**

**VEREN ILIANA KURNIADI 13515078**

**KEVIN ISWARA 13515085**

**CATHERINE ALMIRA 13515111**



**PROGRAM STUDI TEKNIK INFORMATIKA**

**SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA**

**INSTITUT TEKNOLOGI BANDUNG**

**2017**

# DESKRIPSI APLIKASI

Aplikasi yang dibuat oleh kelompok kami (*Hot Games*) untuk memenuhi tugas besar 3 ini adalah sebuah permainan yang bertemakan petualangan bernama “You Can’t Escape!”. *Player* yang dikendalikan oleh *user* akan ditempatkan di dalam sebuah rumah tua yang gelap dan dipenuhi oleh berbagai jenis hantu. Tugas utama *user* adalah mencari jalan keluar dari rumah tua tersebut sebelum user mati ketakutan karena kejaran para hantu.

Di dalam rumah tua tersebut terdapat berbagai barang yang dapat membantu ataupun menyulitkan *player* keluar dari rumah tersebut. Barang-barang yang dapat membantu *player* adalah lilin untuk menerangi ruangan, kunci untuk membuka pintu, dan senjata yang dapat digunakan untuk membunuh hantu. Sementara itu, penghambat *player* untuk keluar dari rumah tersebut adalah pintu yang masih terkunci. *Player* dapat mengambil lilin, kunci, maupun senjata yang terdapat dalam permainan. Benda-benda yang telah diambil akan diletakkan di dalam inventori.

Hantu yang terdapat pada permainan dapat bergerak mendekati *player* dengan kecepatan yang berbeda-beda, mulai dari yang berkecepatan lebih rendah dari *player* sampai dengan yang berkecepatan lebih cepat dari *player*.

*User* dapat mengendalikan player dengan menggunakan keyboard untuk menggerakkan player ke kiri, kanan, depan, maupun belakang. Selain itu, user juga dapat melihat barang apa saja yang sudah diambilnya dan terletak di dalam inventori.

Aplikasi dilengkapi dengan grafik yang menarik dan dipadukan dengan musik yang sangat sesuai dengan tema permainan ini.

Aplikasi yang kelompok kami buat terdiri dari dua versi, yaitu versi 1 (tanpa ghost) dan versi 2 (dengan ghost).

# 

# 

# 

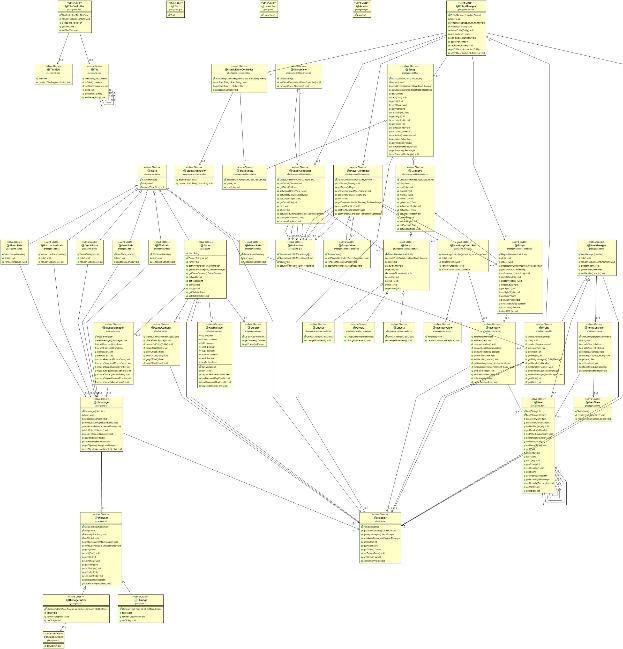
# RANCANGAN VERSI 1

1. Diagram Package

# 

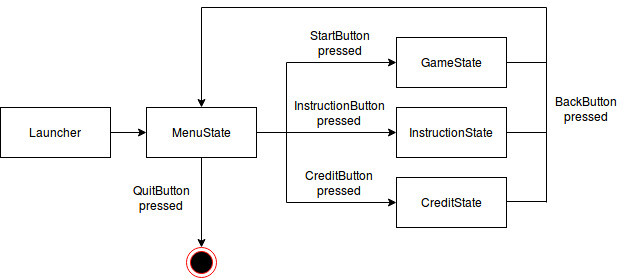
Gambar 1 Diagram Package versi 1

1. Diagram Kelas



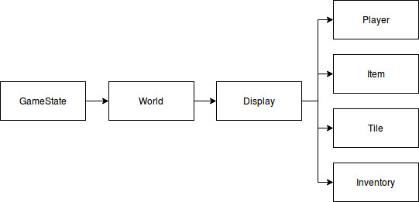
Gambar 2 Diagram Kelas Versi 1

1. Diagram Kolaborasi Antarkelas



Gambar 3 Diagram Kolaborasi AntarKelas Versi 1 (1)

Inisialisasi permainan dimulai dengan menampilkan MenuState. Tampilan akan berubah sesuai dengan tombol yang ditekan oleh user mengikuti state pada gambar 3.



Gambar 4 Diagram Kolaborasi Antarkelas Versi 1 (2)

Inisialisasi GameState adalah menampilkan World yang terdiri dari Ghost, Player, Item, dan Tile. dan Player adalah turunan dari kelas Creature yang adalah turunan dari kelas Entity. Kelas Item merupakan turunan dari StaticEntity yang adalah turunan dari kelas Entity. Inventory akan ditampilkan ketika user menekan tombol ‘E’ pada keyboard.

Kelas lain yang terdapat dalam aplikasi ini merupakan kelas penunjang untuk tampilan aplikasi serta penerima input dari user.

# RANCANGAN VERSI 2

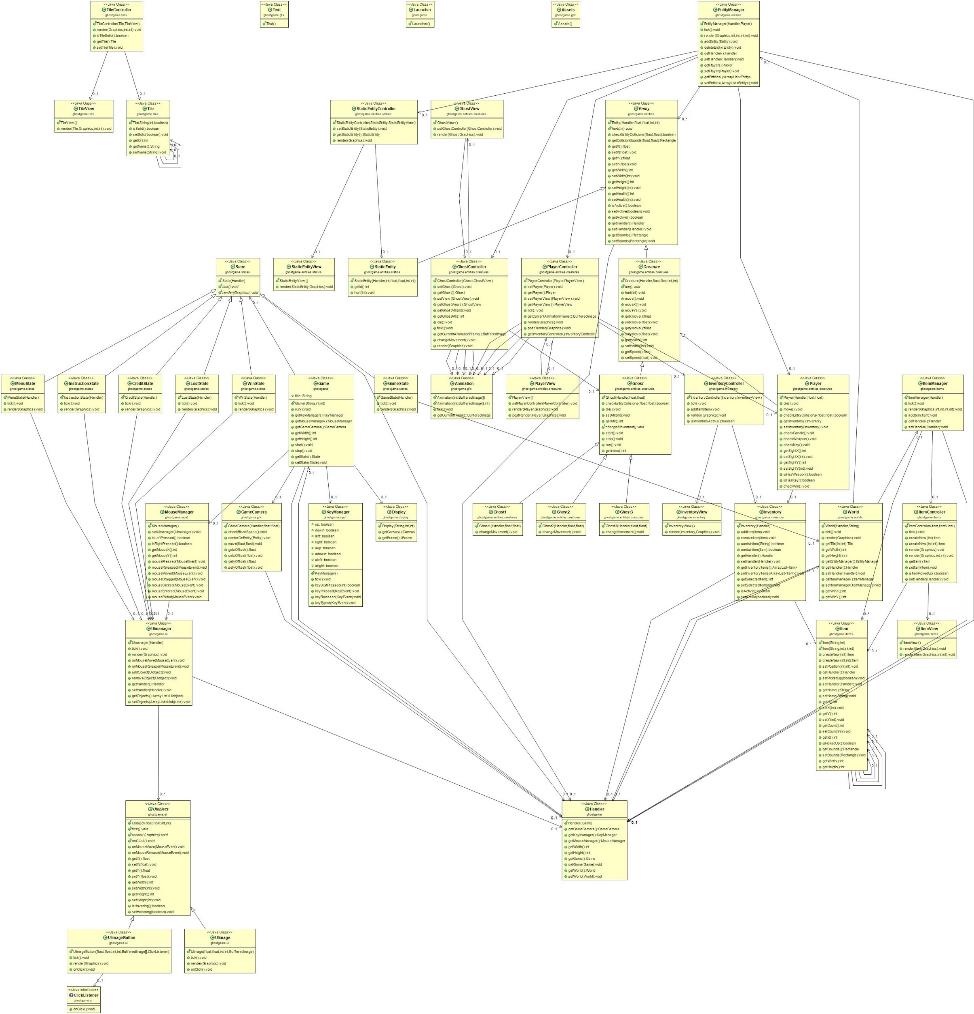
1. Diagram Package

# 

Gambar 5 Diagram Package Versi 2

Rancangan diagram package pada versi 1 dan versi 2 tidak terdapat perbedaan dari jumlah packagenya. Perbedaan terdapat pada jumlah kelas yang terdapat dalam setiap versi. Pada versi 2 terdapat penambahan kelas Ghost, Ghost1, Ghost2, dan Ghost3 yang merupakan bagian dari package creatures.

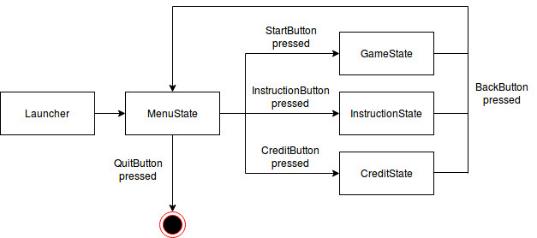
1. Diagram Kelas



Gambar 6 Diagram Kelas

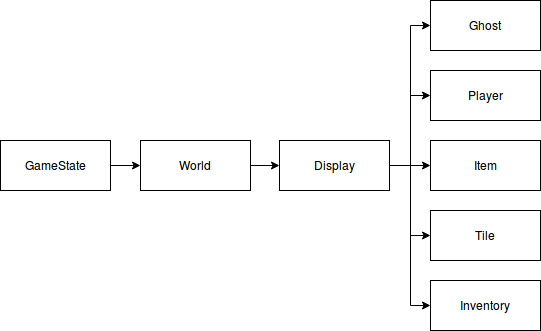
Perubahan yang terjadi pada diagram kelas versi 1 sehingga menghasilkan diagram kelas versi 2 adalah penambahan kelas Ghost yang merupakan turunan dari kelas Creature dan Ghost1, Ghost2, dan Ghost3 yang merupakan turunan dari kelas Ghost.

1. Diagram Kolaborasi Antarkelas



Gambar 7 Diagram Kolaborasi AntarKelas (1)

Inisialisasi permainan dimulai dengan menampilkan MenuState. Tampilan akan berubah sesuai dengan tombol yang ditekan oleh user mengikuti state pada gambar 3.



Gambar 8 Diagram Kolaborasi AntarKelas (2)

Inisialisasi GameState adalah menampilkan World yang terdiri dari Ghost, Player, Item, dan Tile. Ghost dan Player adalah turunan dari kelas Creature yang adalah turunan dari kelas Entity. Kelas Item merupakan turunan dari StaticEntity yang adalah turunan dari kelas Entity. Inventory akan ditampilkan ketika user menekan tombol ‘E’ pada keyboard.

Kelas lain yang terdapat dalam aplikasi ini merupakan kelas penunjang untuk tampilan aplikasi serta penerima input dari user.

# 

# 

# 

# 

# DESKRIPSI PERUBAHAN

.

Pada tugas besar ini kelompok kami tidak menggunakan kelas yang sudah pernah dibuat sebelumnya. Aplikasi permainan yang kelompok kami buat tidak membutuhkan kelas-kelas yang sudah pernah dibuat karena kelas-kelas tersebut tidak relevan dengan tugas yang kami buat.

# 

# DESKRIPSI ALGORITMA

Algoritma hasil modifikasi dari tugas yang pernah dibuat sebelumnya (tugas mata kuliah Strategi Algoritma) adalah algoritma BFS (*Breadth-First Search*). Algoritma ini diterapkan pada Kelas Ghost2 dan Ghost3. Algoritma ini berfungsi untuk mencari jalan menuju pemain. Ghost2 dan Ghost3 akan mencari jalan yang paling baik untuk menuju pemain dengan cara menghitung jarak minimal dari ke-empat sisi yang ada, yaitu sisi atas, bawah, kanan, dan kiri. Dari ke-empat sisi tersebut, akan dipilih yang jarak ke pemain nya dekat. Jika tidak tidak menemukan jalan yang paling baik, maka Ghost2 dan Ghost3 akan tetap diam dan tidak bergerak.

Algoritma BFS adalah algoritma pencarian secara melebar yang mengunjungi suatu simpul, kemudian mengunjungi semua simpul yang bertetangga dengan simpul tersebut terlebih dahulu. Selanjutnya, simpul yang belum dikunjungi dan bertetangga dengan simpul-simpul yang telah dikunjungi, dan seterusnya. Dalam implementasinya pada permainan ini, simpul dinyatakan sebagai posisi hantu. Simpul selanjutnya yang dipilih akan menyatakan gerakan yang akan diambil oleh hantu. Posisi yang akan dipilih adalah posisi yang mendekati posisi *player*. Algoritma ini dapat diterapkan dengan berbagai cara. Ada yang dengan menggunakan rekursif, maupun tidak. Dalam tugas kali ini, kami menggunakan algoritma yang tidak ada rekursifnya. Kami melakukan itu dengan cara mencari nilai (jarak terdekat) terkecil dari ke-empat sisi ghost terhadap player.

# 

# KEUNGGULAN TEKNIS

Keunggulan teknis dari aplikasi “You Can’t Escape!” adalah kelas yang dibuat menggunakan model desain MVC yang memisahkan penanganan objek menjadi tiga, yaitu model, view, dan controller dengan baik sehingga tampilan bisa diubah dengan mudah tanpa memengaruhi kelas model.

Kelas yang dibuat dalam aplikasi “You Can’t Esacape” sudah menerapkan prinsip inheritance sehingga penanganan kelas yang sejenis dapat dilakukan dengan mudah. Prinsip SOLID sudah diterapkan dalam aplikasi ini.

Animasi yang terdapat di dalam aplikasi “You Can’t Escape” sangat menarik. Setiap karakter memiliki animasi masing-masing sehingga saat sedang bermain, animasi akan berubah dalam setiap satuan waktu. Animasi setiap karakter pun dibuat unik untuk pergerakan ke kiri, kanan, atas, dan bawah sehingga untuk ukuran aplikasi sederhana, animasi yang terdapat dalam “You Can’t Escape” sangat memuaskan.

Aplikasi juga dilengkapi dengan audio yang menunjang setiap detail permainan, mulai dari audio ketika membuka pintu, membunuh hantu, sampai dengan ketika player kalah.

# SKENARIO TEST VERSI 1

|  |  |  |
| --- | --- | --- |
| **Skenario Test** | **Reaksi Pengguna** | **Reaksi Sistem** |
| **General** | | |
| Inisiasi kemudian quit | **Skenario Normal** | |
| 1. Menjalankan aplikasi. | 2. Menampilkan daftar menu yang terdiri dari start, instructions, credit, dan quit. |
| 3. Menekan tombol quit. | 4. Keluar dari aplikasi. |
| **Skenario Alternatif** | |
| 1. Menjalankan aplikasi. | 2. Menampilkan daftar menu yang terdiri dari start, instructions, credit, dan quit. |
| 3. Menekan bagian selain tombol start, instructions, credits, dan quit. | 4. Tidak terjadi apa-apa. |
| **Display Instructions** | | |
| Instruksi permainan ditampilkan ke layar | 1. Menekan tombol “Instructions” | 2. Menampilkan instruksi permainan ke layar. |
| **Display Credits** | | |
| Credits ditampilkan ke layar | 1. Menekan tombol “Credits” | 2. Menampilkan credits ke layar. |
| **Display Map** | | |
| Menampilkan peta permainan, player, dan item. | 1. Menekan tombol start pada menu utama. | 2. Menampilkan peta beserta player dan item yang terletak dalam jarak pandang peta. |
| **Player Movement** | | |
| Player bergerak sesuai dengan kontrol dari user dengan tombol pada keyboard (W, A, S, D). | **Skenario Normal** | |
| 1. Menekan tombol start pada menu utama. | 2. Menampilkan peta beserta player dan item ataupun ghost yang terletak dalam jarak pandang peta. |
| 3. Menekan tombol W. | 4. Player bergerak ke atas. |
| 5. Menekan tombol A. | 6. Player bergerak ke kiri. |
| 7. Menekan tombol S. | 8. Player bergerak ke bawah. |
| 9. Menekan tombol D. | 10. Player bergerak ke kanan. |
| **Skenario Alternatif** | |
| 1. Menekan tombol start pada menu utama. | 2. Menampilkan peta beserta player dan item ataupun ghost yang terletak dalam jarak pandang peta. |
| 3. Menekan tombol W ketika di depan player terdapat dinding. | 4. Player diam di tempat. |
| **Display Inventory** | | |
| Menampilkan inventory yang berisi item yang telah diambil oleh | **Skenario Normal** | |
| 1. Player bergerak menuju item (lilin, senjata, atau kunci). | 2. Item menghilang dari tampilan di layar. |
| 3. Menekan tombol E. | 4. Menampilkan isi inventory (item yang telah diambil). |
| **Skenario Alternatif** | |
| 1. Menekan tombol E. | 2. Menampilkan isi inventory (inventory kosong). |
| **Unlock Door** | | |
| Membuka pintu yang terkunci. | **Skenario Normal** | |
| 1. Player bergerak menuju ke kunci. | 2. Kunci masuk ke dalam inventory player. |
| 3. Player bergerak ke pintu dan menekan tombol panah sesuai dengan lokasi pintu berada (tombol kanan apabila pintu berada di kanan player). | 4. Pintu terbuka. |
| **Skenario Alternatif** | |
| 1. Player bergerak ke pintu dan menekan tombol panah sesuai dengan lokasi pintu berada (tombol kanan apabila pintu berada di kanan player). | 2. Pintu tetap terkunci. |
| **Lighten Map** | | |
| Ukuran peta yang terang (dapat terlihat) lebih besar setelah player mengambil lilin. | 1. Player bergerak menuju lilin. | 2. Bagian peta yang berwarna menjadi lebih besar. |
| **Lose State** | | |
| Player akan mati ketika mengenai ghost sebanyak dua kali. | 1. Player bergerak menabrak ghost. | 2. Ghost menghilang dari layar. |
|  | 3. Player bergerak menabrak ghost. | 4. Menampilkan status kekalahan ke layar. |
| **Win State** | | |
| Player menang ketika mencapai jalan keluar. | 1. Player bergerak menuju jalan keluar. | 2. Menampilkan status kemenangan ke layar. |

# 

# 

# SKENARIO TEST VERSI 2

|  |  |  |
| --- | --- | --- |
| **Skenario Test** | **Reaksi Pengguna** | **Reaksi Sistem** |
| **General** | | |
| Inisiasi kemudian quit | **Skenario Normal** | |
| 1. Menjalankan aplikasi. | 2. Menampilkan daftar menu yang terdiri dari start, instructions, credit, dan quit. |
| 3. Menekan tombol quit. | 4. Keluar dari aplikasi. |
| **Skenario Alternatif** | |
| 1. Menjalankan aplikasi. | 2. Menampilkan daftar menu yang terdiri dari start, instructions, credit, dan quit. |
| 3. Menekan bagian selain tombol start, instructions, credits, dan quit. | 4. Tidak terjadi apa-apa. |
| **Display Instructions** | | |
| Instruksi permainan ditampilkan ke layar | 1. Menekan tombol “Instructions” | 2. Menampilkan instruksi permainan ke layar. |
| **Display Credits** | | |
| Credits ditampilkan ke layar | 1. Menekan tombol “Credits” | 2. Menampilkan credits ke layar. |
| **Display Map** | | |
| Menampilkan peta permainan, player, ghost, dan item. | 1. Menekan tombol start pada menu utama. | 2. Menampilkan peta beserta player dan item ataupun ghost yang terletak dalam jarak pandang peta. |
| **Player Movement** | | |
| Player bergerak sesuai dengan kontrol dari user dengan tombol pada keyboard (W, A, S, D). | **Skenario Normal** | |
| 1. Menekan tombol start pada menu utama. | 2. Menampilkan peta beserta player dan item ataupun ghost yang terletak dalam jarak pandang peta. |
| 3. Menekan tombol W. | 4. Player bergerak ke atas. |
| 5. Menekan tombol A. | 6. Player bergerak ke kiri. |
| 7. Menekan tombol S. | 8. Player bergerak ke bawah. |
| 9. Menekan tombol D. | 10. Player bergerak ke kanan. |
| **Skenario Alternatif** | |
| 1. Menekan tombol start pada menu utama. | 2. Menampilkan peta beserta player dan item ataupun ghost yang terletak dalam jarak pandang peta. |
| 3. Menekan tombol W ketika di depan player terdapat dinding. | 4. Player diam di tempat. |
| **Ghost Movement** | | |
| Ghost2 dan Ghost3 bergerak mendekati player. | 1. Player bergerak ke daerah pada peta yang terdapat ghost2 atau ghost3. | 2. Ghost2 atau ghost3 bergerak mendekati player. |
| **Display Inventory** | | |
| Menampilkan inventory yang berisi item yang telah diambil oleh | **Skenario Normal** | |
| 1. Player bergerak menuju item (lilin, senjata, atau kunci). | 2. Item menghilang dari tampilan di layar. |
| 3. Menekan tombol E. | 4. Menampilkan isi inventory (item yang telah diambil). |
| **Skenario Alternatif** | |
| 1. Menekan tombol E. | 2. Menampilkan isi inventory (inventory kosong). |
| **Unlock Door** | | |
| Membuka pintu yang terkunci. | **Skenario Normal** | |
| 1. Player bergerak menuju ke kunci. | 2. Kunci masuk ke dalam inventory player. |
| 3. Player bergerak ke pintu dan menekan tombol panah sesuai dengan lokasi pintu berada (tombol kanan apabila pintu berada di kanan player). | 4. Pintu terbuka. |
| **Skenario Alternatif** | |
| 1. Player bergerak ke pintu dan menekan tombol panah sesuai dengan lokasi pintu berada (tombol kanan apabila pintu berada di kanan player). | 2. Pintu tetap terkunci. |
| **Lighten Map** | | |
| Ukuran peta yang terang (dapat terlihat) lebih besar setelah player mengambil lilin. | 1. Player bergerak menuju lilin. | 2. Bagian peta yang berwarna menjadi lebih besar. |
| **Attack the Ghost** | | |
| Menyerang hantu yang terdapat dalam permainan. | **Skenario Normal** | |
| 1. Player bergerak menuju senjata. | 2. Senjata masuk ke dalam inventory player. |
| 3. Player bergerak menuju ghost (berbeda 1 petak) dan menekan tombol panah sesuai dengan lokasi ghost berada (tombol kanan apabila ghost berada di kanan player). | 4. Ghost mati dan hilang dari layar permainan. |
| **Skenario Alternatif** | |
| 1. Player bergerak menuju ghost (berbeda 1 petak) dan menekan tombol panah sesuai dengan lokasi ghost berada (tombol kanan apabila ghost berada di kanan player). | 2. Tidak terjadi apa-apa. |
| **Lose State** | | |
| Player akan mati ketika mengenai ghost sebanyak dua kali. | 1. Player bergerak menabrak ghost. | 2. Ghost menghilang dari layar. |
|  | 3. Player bergerak menabrak ghost. | 4. Menampilkan status kekalahan ke layar. |
| **Win State** | | |
| Player menang ketika mencapai jalan keluar. | 1. Player bergerak menuju jalan keluar. | 2. Menampilkan status kemenangan ke layar. |

# RINGKASAN METRIKS VERSI 1

--------------------------------------------------

- Summary:

--------------------------------------------------

Name, Class Count, Abstract Class Count, Ca, Ce, A, I, D, V:

ghostgame,3,0,12,10,0,0.45,0.55,1

ghostgame.display,1,0,1,3,0,0.75,0.25,1

ghostgame.entities,3,1,5,7,0.33,0.58,0.08,1

ghostgame.entities.creatures,10,2,4,15,0.2,0.79,0.01,1

ghostgame.entities.statics,3,0,4,6,0,0.6,0.4,1

ghostgame.gfx,4,0,8,9,0,0.53,0.47,1

ghostgame.input,2,0,4,3,0,0.43,0.57,1

ghostgame.inventory,3,0,3,8,0,0.73,0.27,1

ghostgame.items,4,0,5,10,0,0.67,0.33,1

ghostgame.states,15,1,2,8,0.07,0.8,0.13,1

ghostgame.tiles,3,0,2,4,0,0.67,0.33,1

ghostgame.ui,5,2,3,6,0.4,0.67,0.07,1

ghostgame.worlds,1,0,6,11,0,0.65,0.35,1

ghostgametest.entities.creatures,6,0,0,5,0,1,0,1

ghostgametest.entities.statics,2,0,0,4,0,1,0,1

ghostgametest.inventory,1,0,0,5,0,1,0,1

ghostgametest.items,1,0,0,3,0,1,0,1

ghostgametest.ui,2,0,0,5,0,1,0,1

java.awt,0,0,12,0,0,0,1,1

java.awt.event,0,0,2,0,0,0,1,1

java.awt.image,0,0,9,0,0,0,1,1

java.io,0,0,5,0,0,0,1,1

java.lang,0,0,18,0,0,0,1,1

java.util,0,0,6,0,0,0,1,1

javax.imageio,0,0,1,0,0,0,1,1

javax.sound.sampled,0,0,3,0,0,0,1,1

javax.swing,0,0,2,0,0,0,1,1

org.junit,0,0,5,0,0,0,1,1

# 

# RINGKASAN METRIKS VERSI 2

# --------------------------------------------------

# - Summary:

# --------------------------------------------------

# 

# Name, Class Count, Abstract Class Count, Ca, Ce, A, I, D, V:

# 

# ghostgame,3,0,12,10,0,0.45,0.55,1

# ghostgame.display,1,0,1,3,0,0.75,0.25,1

# ghostgame.entities,3,1,5,7,0.33,0.58,0.08,1

# ghostgame.entities.creatures,10,2,4,15,0.2,0.79,0.01,1

# ghostgame.entities.statics,3,0,4,6,0,0.6,0.4,1

# ghostgame.gfx,4,0,8,9,0,0.53,0.47,1

# ghostgame.input,2,0,4,3,0,0.43,0.57,1

# ghostgame.inventory,3,0,3,8,0,0.73,0.27,1

# ghostgame.items,4,0,5,10,0,0.67,0.33,1

# ghostgame.states,15,1,2,8,0.07,0.8,0.13,1

# ghostgame.tiles,3,0,2,4,0,0.67,0.33,1

# ghostgame.ui,5,2,3,6,0.4,0.67,0.07,1

# ghostgame.worlds,1,0,6,11,0,0.65,0.35,1

# ghostgametest.entities.creatures,6,0,0,5,0,1,0,1

# ghostgametest.entities.statics,2,0,0,4,0,1,0,1

# ghostgametest.inventory,1,0,0,5,0,1,0,1

# ghostgametest.items,1,0,0,3,0,1,0,1

# ghostgametest.ui,2,0,0,5,0,1,0,1

# java.awt,0,0,12,0,0,0,1,1

# java.awt.event,0,0,2,0,0,0,1,1

# java.awt.image,0,0,9,0,0,0,1,1

# java.io,0,0,5,0,0,0,1,1

# java.lang,0,0,18,0,0,0,1,1

# java.util,0,0,6,0,0,0,1,1

# javax.imageio,0,0,1,0,0,0,1,1

# javax.sound.sampled,0,0,3,0,0,0,1,1

# javax.swing,0,0,2,0,0,0,1,1

# org.junit,0,0,5,0,0,0,1,1