JACIS : Journal Automation Computer Information System Volume: **4** Nomor **2**, **2024**, *Hal*: 76-87

Implementasi Black Box Testing Pada Game Ular Untuk Mendeteksi Bug

Fauziah Putri Utami[1], Hilya Zahra Alifa*[2] dan Muhammad Ainul Yaqin[3]

[1] Universitas Islam Negeri Maulana Malik Ibrahim Malang; putrifauziah807@gmail.com
[*2] Universitas Islam Negeri Maulana Malik Ibrahim Malang; hilyazahraa12@gmail.com,
[3] Universitas Islam Negeri Maulana Malik Ibrahim Malang; yaqinov@ti.uin-malang.ac.id

Abstrak: Permainan Ular adalah salah satu permainan klasik yang masih populer hingga saat ini, namun sering mengalami bug yang mengganggu pengalaman pengguna. Deteksi bug yang tidak terdeteksi selama tahap pengembangan menjadi permasalahan yang signifikan dan mendesak untuk diatasi. Penelitian ini bertujuan untuk menerapkan metode Black Box Testing pada permainan Ular guna mendeteksi bug yang mungkin terlewatkan. Sebuah sistem Black Box Testing dikembangkan untuk menjalankan skenario pengujian secara otomatis tanpa memperhatikan logika internal permainan. Pengujian dilakukan terhadap berbagai fitur dan fungsi permainan Ular, seperti pergerakan ular, mekanisme pertumbuhan ular, dan interaksi antara ular dengan elemen permainan lainnya. Teknik black box testing yang digunakan mencakup Equivalence Partitioning, Boundary Value Analysis, Use Case Testing, dan Decision Table Technique. Hasil pengujian menunjukkan bahwa metode Black Box Testing berhasil mendeteksi 1 bug dari 28 skenario uji, yaitu ketidaksesuaian kecepatan pergerakan ular sesuai dengan levelnya. Oleh karena itu, disimpulkan bahwa game ini masih memerlukan pengembangan lebih lanjut untuk meningkatkan kualitasnya.
Keywords: Black Box, Game, Bug

DOI: https://doi.org/10.47134/jacis
*Correspondensi: Muhammad Ainul Yaqin
Email: yaqinov@ti.uin-malang.ac.id

Receive: 5 Juli 2024
Accepted: 17 Juli 2024
Published: 17 Juli 2024

Copyright: © 2021 by the authors. Submitted for possible open access publication under the terms and conditions of the Creative Commons Attribution (CC BY) license (http://creativecommons.org/licenses/by/4.0/).

Abstrak: Snake Game is a classic game that remains popular to this day but often experiences bugs that disrupt user experience. Detecting bugs that are not identified during the development phase is a significant and urgent problem to address. This study aims to apply the Black Box Testing method to the Snake Game to detect potentially overlooked bugs. A Black Box Testing system was developed to execute testing scenarios automatically without considering the game's internal logic. Testing was conducted on various features and functions of the Snake Game, such as snake movement, snake growth mechanism, and the interaction between the snake and other game elements. The black box testing techniques used include Equivalence Partitioning, Boundary Value Analysis, Use Case Testing, and Decision Table Technique. The test results indicate that the Black Box Testing method successfully detected 1 bug out of 28 test scenarios, namely the inconsistency in snake movement speed according to its level. Therefore, it is concluded that the game still requires further development to improve its quality.
Keywords: Black Box, Snake Game, Bug.

PENDAHULUAN

Game ular merupakan salah satu permainan klasik yang populer di berbagai kalangan. Biasanya dimainkan di perangkat elektronik seperti ponsel cerdas atau komputer, game ini mengharuskan pemain mengendalikan seekor ular yang bergerak di atas layar. Tujuannya adalah membuat ular tersebut tumbuh lebih panjang dengan menangkap makanan yang tersebar di sekitarnya. Namun, pemain harus berhati-hati agar ularnya tidak menabrak dinding atau tubuhnya sendiri, karena hal itu akan menyebabkan permainan berakhir.

Meskipun terkesan sederhana, game ular sering kali mengandung bug yang dapat mempengaruhi pengalaman pengguna. Untuk memastikan apakah fungsi-fungsi dalam suatu perangkat lunak beroperasi dengan efektif, akurat, dan sesuai dengan spesifikasi yang telah ditetapkan, diperlukan pengujian terhadap game ular ini [1]. Salah satu metode yang efektif untuk mendeteksi bug adalah black box testing.

Permasalahan yang sering muncul dalam permainan ular meliputi deteksi tabrakan, di mana penting untuk memastikan apakah permainan dapat mengenali dengan tepat saat ular bertabrakan dengan dinding atau tubuhnya sendiri. Selain itu, perilaku ular seperti berbelok, memakan makanan, dan bertambah panjang juga perlu diuji untuk memastikan kepatuhan mereka terhadap aturan yang telah ditetapkan. Kegagalan dalam mendeteksi bug ini dapat mengganggu pengalaman bermain dan menurunkan kualitas permainan. Dengan menemukan kesalahan selama pengujian, maka perbaikan dapat dilakukan pada fungsi-fungsi tersebut untuk menghindari cacat sistem yang berkelanjutan dan terus meningkatkan kualitas perangkat lunak [2].

Dengan menggunakan metode black box testing, pengujian dapat mengidentifikasi berbagai bug atau masalah dalam game ular tanpa harus memiliki pengetahuan mendalam tentang bagaimana kode programnya bekerja. Selain itu black box testing ternyata juga dapat melakukan verifikasi serta validasi perangkat lunak yang sedang dibangun [3]. Metode ini memungkinkan untuk menemukan dan memperbaiki masalah dengan lebih efisien, sehingga meningkatkan kualitas dan pengalaman pengguna dari game tersebut. Oleh karena itu, black box testing merupakan alat yang penting dalam memastikan kualitas dan keandalan perangkat lunak, termasuk pada game ular [4].

Dalam penelitian ini, penulis akan melakukan implementasi black box testing pada game ular, mulai dari perumusan skenario pengujian hingga penanganan bug yang terdeteksi. Ada beberapa bug yang belum ditemukan pada game ini, sehingga diperlukan pengujian lebih lanjut. Tujuan utama dari penelitian ini adalah untuk mendeteksi bug yang terdapat pada game ular dan meningkatkan kualitas permainan [5].

Beberapa penelitian sebelumnya telah melakukan pengujian perangkat lunak menggunakan metode black box dengan menerapkan teknik analisis nilai batas atau boundary value analysis. Salah satu penelitian menggunakan pendekatan black box testing dengan teknik analisis nilai batas pada aplikasi DANA. Fokus utama dari penelitian tersebut adalah memastikan bahwa semua fungsi aplikasi DANA berfungsi dengan baik dan bebas dari kesalahan dan bug. Hasil pengujian menunjukkan tingkat keberhasilan sebesar 95%, meskipun masih terdapat satu bidang yang mengalami kesalahan saat fungsi tersebut dijalankan. Hal ini menunjukkan bahwa aplikasi DANA belum optimal dan memerlukan perbaikan [6].

Penelitian sejenis menggunakan teknik boundary value analysis dalam pengujian black box pada Sistem Mobile Learning. Fokus utama dari penelitian tersebut adalah menguji menu-menu yang ada dalam aplikasi tersebut. Hasil pengujian menunjukkan tingkat keberhasilan

rata-rata sebesar 87,74%, yang menyimpulkan bahwa aplikasi tersebut masuk dalam kategori "sangat baik" berdasarkan hasil pengujian [7]. Kedua penelitian ini menunjukkan bahwa meskipun teknik analisis nilai batas dapat meningkatkan keandalan perangkat lunak, masih ada ruang untuk perbaikan dalam memastikan semua fungsi berjalan sempurna.
Meskipun penelitian sebelumnya telah menunjukkan keberhasilan dalam mendeteksi bug menggunakan black box testing, penelitian ini akan berusaha menemukan kebaruan dengan fokus pada pengujian game ular. Penulis akan mengidentifikasi masalah spesifik yang mungkin belum terdeteksi dan menerapkan solusi yang lebih efektif untuk meningkatkan kualitas dan keandalan game ini.

METODE

Metode penelitian yang efektif dalam menguji perangkat lunak, khususnya dalam konteks pengembangan game, memerlukan pendekatan yang sistematis dan terstruktur. Langkah-langkah yang menyeluruh diperlukan untuk memastikan bahwa proses pengujian dilakukan dengan baik dan menghasilkan hasil yang akurat.

Metode penelitian yang digunakan dalam penelitian ini adalah black box testing, yang merupakan salah satu teknik dalam pengujian perangkat lunak. Pendekatan ini fokus pada pengujian fungsional perangkat lunak tanpa memperhatikan struktur internal kode. Dalam black box testing, tester hanya memiliki pengetahuan tentang input dan output yang diharapkan dari perangkat lunak yang diuji, sementara detail implementasi internal tidak terungkap. Pengujian black box bertujuan untuk mengidentifikasi fungsi yang tidak tepat, kesalahan dalam antarmuka, kesalahan dalam struktur data, masalah kinerja, kesalahan dalam inisialisasi, dan masalah dalam terminasi [8]. Dengan demikian, black box testing menjadi pendekatan yang efektif dalam memvalidasi keandalan dan kinerja perangkat lunak tanpa memerlukan pengetahuan mendalam tentang struktur internalnya [9].

Teknik pengujian pada black Box testing ada banyak macamnya, yaitu:

a. Teknik Equivalence Partitioning adalah cara untuk membagi input data menjadi beberapa bagian atau partisi.
b. Teknik Boundary value analysis melibatkan penelusuran kesalahan dari luar atau dalam perangkat lunak, baik itu nilai minimum maupun maksimum dari kesalahan yang ditemukan.
c. Teknik Fuzzing adalah metode untuk menemukan bug atau gangguan dalam perangkat lunak dengan menginjeksikan data yang cacat.
d. Teknik Cause-Effect Graph menggunakan grafik sebagai referensi untuk menggambarkan hubungan antara penyebab dan efek dalam pengujian.
e. Teknik Orthogonal Array Testing digunakan ketika domain input relatif kecil namun kompleksitasnya cukup tinggi untuk skala besar.
f. Teknik All Pair Testing merancang semua pasangan dari kasus uji agar mencakup semua kemungkinan kombinasi diskrit dari seluruh pasangan inputnya.
g. Teknik State Transition berguna untuk menguji kondisi mesin dan navigasi dalam bentuk grafik.
h. Teknik Error Guessing merupakan pengujian yang dilakukan dengan cara mengidentifikasi kesalahan dalam aplikasi berdasarkan pengetahuan dan pengalaman penguji.
i. Teknik Use Case merupakan pengujian dimana kita menguji setiap fungsi perangkat lunak dengan menjalankan sistem dari tahap awal hingga tahap akhir.

j. Decision Table Technique adalah suatu metode yang diterapkan secara sistematis, di mana berbagai kombinasi input disusun dalam sebuah tabel. Teknik ini berguna untuk menguji fungsi yang menunjukkan keterkaitan logis antara dua atau lebih input.

Pada penelitian ini menggunakan beberapa Teknik pengujian pada black box testing yaitu Teknik Boundary value, Teknik Use Case Testing, Teknik Equivalence Partitioning, dan Teknik Decision Table.

Proses Melakukan Pengujian

Dalam melakukan software testing pada game ular ada beberapa tahapan kegiatan yang harus dilaksanakan. Berikut ini merupakan tahapannya:

a. Perencanaan Pengujian

Pada tahap ini, tujuan pengujian, cakupan, pendekatan, dan sumber daya yang dibutuhkan untuk pengujian direncanakan. Hal ini mencakup identifikasi fitur-fitur utama yang akan diuji, serta menentukan metode pengujian yang akan digunakan. Penelitian ini menggunakan beberapa teknik pengujian dalam black box testing, yaitu Teknik Boundary Value, Use Case Testing, Equivalence Partitioning, dan Decision Table. Berikut adalah penjelasan dari masing-masing teknik yang digunakan dalam penelitian ini:

1) Teknik Boundary value analysis

Boundary value analysis adalah teknik pengujian perangkat lunak yang berfokus pada nilai-nilai batas dari input domain [10]. Ide utamanya adalah bahwa kesalahan sering terjadi di sekitar batas-batas input, sehingga pengujian nilai-nilai ini dapat membantu menemukan bug atau kesalahan yang mungkin tidak terlihat ketika menggunakan nilai-nilai tengah atau biasa dari domain [11]. Teknik ini digunakan dalam pengujian pada tabel 3 dan tabel 4. Pada tabel 3, pengujian memeriksa batas-batas atau tembok pada permainan ular yang terdiri atas empat sisi: atas, bawah, kanan, dan kiri. Skenario ini harus memastikan bahwa permainan berakhir dengan "game over" ketika ular menabrak tembok. Sedangkan pada tabel 4, pengujian berkaitan dengan fitur penyimpanan skor, yang melibatkan pembuatan, penyimpanan, dan pembaruan skor tertinggi, untuk memastikan bahwa sistem dapat mengelola skor dengan benar.

2) Teknik Decision Table

Decision Table adalah teknik pengujian perangkat lunak yang digunakan untuk menangani kombinasi kondisi dan aksi yang kompleks [12]. Decision Table menyusun berbagai kondisi input dan tindakan yang sesuai dalam format tabel, memungkinkan pengujian yang sistematis dan terstruktur. Teknik ini digunakan dalam pengujian tabel 2 yang menunjukkan hubungan antara kondisi input dan tindakan yang diambil. Hal ini meliputi pengujian terhadap perilaku ular saat memakan objek di permainan, baik itu ayam untuk pertambahan panjang dan skor, maupun ekor ular itu sendiri yang seharusnya mengakibatkan kegagalan permainan.

3) Teknik Equivalence Partitioning

Equivalence Partitioning adalah teknik pengujian perangkat lunak yang membagi domain input menjadi beberapa kelompok atau partisi yang diharapkan memberikan perilaku yang sama [13]. Pengujian dilakukan dengan mengambil satu nilai representatif dari setiap partisi [14]. Teknik

ini digunakan pada tabel 1 yang berfokus pada pengujian pergerakan ular terkait dengan input dari pengguna, seperti menekan tombol arah pada keyboard, untuk memastikan bahwa ular dapat bergerak sesuai dengan input yang diberikan.

4) Teknik Use Case Testing
Use Case Testing adalah teknik pengujian perangkat lunak yang berfokus pada pengujian skenario penggunaan dari perspektif pengguna [15]. Teknik ini menggunakan Use Case atau skenario penggunaan untuk menentukan kasus uji yang mencakup interaksi antara pengguna dan sistem. Teknik ini digunakan pada tabel 5 yang menguji kecepatan game pada level yang berbeda. Pada game ini terdapat tiga level kecepatan, yaitu easy, medium, dan hard. Tabel ini digunakan untuk memastikan bahwa game ular dapat berjalan dengan kecepatan yang sesuai dengan level yang dipilih.

b. Desain Skenario Pengujian
Pada tahap ini, serangkaian skenario pengujian dirancang yang mencakup berbagai aspek game. Ini termasuk pengujian fungsionalitas inti, fitur tambahan, dan situasi yang mungkin terjadi.

c. Implementasi Skenario Pengujian
Setelah merancang skenario pengujian, langkah selanjutnya adalah menjalankannya pada game yang diuji. Skenario yang telah dibuat sebelumnya akan dijalankan dan hasilnya akan dicatat. Ini mencakup pengujian fungsionalitas game dan memeriksa apakah hasilnya sesuai dengan harapan.

d. Analisis Hasil Pengujian
Setelah menjalankan skenario pengujian, hasilnya akan dianalisis. Tujuannya adalah untuk menemukan bug atau masalah lain yang mungkin muncul selama pengujian. Dilakukan pemeriksaan apakah game berperilaku seperti yang diharapkan dan mengidentifikasi area dimana masalah atau kekurangan terjadi.

e. Pelaporan Hasil
Tahap terakhir adalah pembuatan laporan hasil pengujian. Laporan ini akan mencakup temuan bug yang ditemukan selama pengujian, serta rekomendasi untuk perbaikan. Laporan ini akan menjadi dokumen penting yang dapat digunakan oleh pengembang untuk meningkatkan kualitas game.

Rancangan Skenario Pengujian

Rancangan skenario pengujian adalah proses yang sistematis untuk menguji fungsionalitas dan kinerja suatu sistem perangkat lunak atau aplikasi. Tujuannya adalah untuk memastikan bahwa aplikasi tersebut berfungsi sesuai dengan yang diharapkan dan dapat diandalkan dalam berbagai situasi. Rancangan skenario pengujian mencakup identifikasi berbagai kasus uji, proses pengujian, serta hasil yang diharapkan dari setiap pengujian

Tabel 1 berfokus pada pengujian pergerakan ular terkait dengan input dari pengguna, seperti menekan tombol arah pada keyboard. Tabel ini menggunakan Teknik Equivalence Partitioning untuk memastikan bahwa ular dapat bergerak sesuai dengan input yang diberikan.

Tabel 1. Pengujian Pergerakan Ular

Test ID	Partisi Input	Input	Kondisi Awal Ular	Hasil yang diharapkan
A01	Arah valid	*'Up'*	Bergerak ke bawah	Ular tetap bergerak ke bawah (input ditolak)
A02	Arah valid	*'Up'*	Bergerak ke kiri	Ular bergerak ke atas

A03	Arah valid	*'Up'*	Bergerak ke kanan	Ular bergerak ke atas
A04	Arah valid	*'Down'*	Bergerak ke atas	Ular tetap bergerak ke atas (input ditolak)
A05	Arah valid	*'Down'*	Bergerak ke kiri	Ular bergerak ke bawah
A06	Arah valid	*'Down'*	Bergerak ke kanan	Ular bergerak ke bawah
A07	Arah valid	*'Left'*	Bergerak ke kanan	Ular tetap bergerak ke kanan (input ditolak)
A08	Arah valid	*'Left'*	Bergerak ke atas	Ular bergerak ke kiri
A09	Arah valid	*'Left'*	Bergerak ke bawah	Ular bergerak ke kiri
A10	Arah valid	*'Right'*	Bergerak ke kiri	Ular tetap bergerak ke kiri (input ditolak)
A11	Arah valid	*'Right'*	Bergerak ke atas	Ular bergerak ke kanan
A12	Arah valid	*'Right'*	Bergerak ke bawah	Ular bergerak ke kanan
A13	Input tidak valid	'A'	Bergerak ke atas	Ular tetap bergerak ke atas (input diabaikan)
A14	Input tidak valid	'1'	Bergerak ke bawah	Ular tetap bergerak ke bawah (input diabaikan)
A15	Input tidak valid	'#'	Bergerak ke kiri	Ular tetap bergerak ke kiri (input diabaikan)
A16	Input tidak valid	'Z'	Bergerak ke kanan	Ular tetap bergerak ke kanan (input diabaikan)

Tabel 2 diuji menggunakan Teknik Decision Table yang menunjukkan hubungan antara kondisi input dan tindakan yang diambil. Hal ini meliputi pengujian terhadap perilaku ular saat memakan objek di permainan, baik itu ayam untuk pertambahan panjang dan skor, maupun ekor ular itu sendiri yang seharusnya mengakibatkan kegagalan permainan.

Tabel 2. Pengujian Perilaku Ular

Test ID	Kondisi	Aturan 1	Aturan 2	Aturan 3	Aturan 4
B01	Memakan Ayam (T/F)	F	T	F	T
B02	Memakan Ekor Ular (T/F)	F	F	T	T
B03	Tidak Memakan Apa-apa (T/F)	T	T	T	F

Keterangan:
T : Ular berhasil melakukan aksi
F : Ular gagal melakukan aksi
E : Ular menampilkan reaksi atau perilaku tertentu
H : Ular tidak menampilkan reaksi atau perilaku tertentu

Interpretasi:
Aturan 1 : Ular tidak memakan ayam dan tidak memakan ekor ular, menampilkan reaksi atau perilaku tertentu
Aturan 2 : Ular memakan ayam, tetapi tidak memakan ekor ular, menampilkan reaksi atau perilaku tertentu
Aturan 3 : Ular tidak memakan ayam, tetapi memakan ekor ular, menampilkan reaksi atau perilaku tertentu
Aturan 4 : Ular memakan ayam dan memakan ekor ular, tidak menampilkan reaksi atau perilaku tertentu

Tabel 3 memeriksa batas-batas atau tembok pada permainan ular yang terdiri atas

empat sisi, yaitu atas, bawah, kanan, dan kiri. Pengujian ini menggunakan Teknik *Boundary value analysis*. Skenario ini harus memastikan bahwa permainan berakhir dengan "*game over*" ketika ular menabrak tembok.

Tabel 3. Pengujian Batas Tembok Terhadap Ular

Test ID	Rincian Pengujian	Hasil yang Diharapkan
C01	Ular menabrak batas tembok bagian atas	Ular mati karena menabrak batas atas
C02	Ular menabrak batas tembok bagian bawah	Ular mati karena menabrak batas bawah
C03	Ular menabrak batas tembok bagian kanan	Ular mati karena menabrak batas kanan
C04	Ular menabrak batas tembok bagian kiri	Ular mati karena menabrak batas kiri

Tabel 4 berkaitan dengan pengujian fitur penyimpanan skor, yang melibatkan pembuatan, penyimpanan, dan pembaruan skor tertinggi. Pada Tabel ini, digunakan teknik *Boundary value analysis* untuk memastikan bahwa sistem dapat mengelola skor dengan benar.

Tabel 4. Pengujian Penyimpanan Skor

Test ID	Rincian Pengujian	Hasil yang Diharapkan
D01	Membuat rekor skor yang lebih tinggi dari sebelumnya Skor baru > Skor sebelumnya	Skor baru tersimpan
D02	Membuat rekor skor yang lebih rendah dari sebelumnya Skor baru < skor sebelumnya	Skor baru tidak tersimpan dan yang tertera adalah skor terbaik sebelumnya

Tabel 5 menguji kecepatan game pada level yang berbeda. Pada game ini terdapat tiga level kecepatan, yaitu *easy, medium,* dan *hard*. Tabel ini menggunakan Teknik *Use Case* Testing untuk memastikan bahwa game ular dapat berjalan dengan kecepatan yang sesuai dengan level yang dipilih.

Tabel 5. Pengujian Pada Level Kecepatan Yang Berbeda

Test ID	Rincian Pengujian	Langkah-langkah	Hasil yang Diharapkan
E01	Kecepatan level '*Easy*'	1. Buka Permainan 2. Pilih level '*Easy*' 3. Mainkan Permainan	Ular bergerak dengan lambat
E02	Kecepatan level '*Normal*'	1. Buka Permainan 2. Pilih level '*Normal*' 3. Mainkan Permainan	Ular bergerak dengan lebih cepat dari level '*Easy*'
E03	Kecepatan Level '*Hard*'	1. Buka Permainan 2. Pilih level '*Hard*' 3. Mainkan Permainan	Ular bergerak dengan sangat cepat

HASIL DAN PEMBAHASAN

Pengujian merupakan tahap krusial dalam pengembangan perangkat lunak atau sistem, di mana berbagai fitur dan fungsionalitas diuji untuk memastikan bahwa sistem tersebut berfungsi dengan baik sesuai dengan spesifikasi yang telah ditetapkan. Dalam kasus ini, pengujian dilakukan terhadap sebuah permainan ular, dimana berbagai aspek seperti pergerakan ular, interaksi dengan objek dalam permainan, respons terhadap kondisi tertentu, dan kemampuan penyimpanan skor dievaluasi. Melalui hasil pengujian ini, kita akan mengetahui sejauh mana implementasi permainan ular ini memenuhi harapan dan standar

yang telah ditetapkan. Berikut adalah penjelasan mengenai hasil dan pembahasan dari setiap tabel:

Tabel 6 menunjukkan hasil bahwa ular dapat bergerak sesuai dengan arah yang diinginkan oleh pemain, baik itu ke atas, ke bawah, ke kanan, atau ke kiri. Kesimpulannya adalah bahwa pergerakan ular sesuai dengan yang diharapkan dan sistem berfungsi dengan baik.

Tabel 6. Hasil Pengujian Pergerakan Ular

Test ID	Partisi Input	Input	Kondisi Awal Ular	Hasil yang diharapkan	Hasil pengujian	Kesimpulan
A01	Arah valid	*'Up'*	Bergerak ke bawah	Ular tetap bergerak ke bawah (input ditolak)	Ular tetap bergerak ke bawah	Sesuai
A02	Arah valid	*'Up'*	Bergerak ke kiri	Ular bergerak ke atas	Ular bergerak ke atas	Sesuai
A03	Arah valid	*'Up'*	Bergerak ke kanan	Ular bergerak ke atas	Ular bergerak ke atas	Sesuai
A04	Arah valid	*'Down'*	Bergerak ke atas	Ular tetap bergerak ke atas (input ditolak)	Ular tetap bergerak ke atas	Sesuai
A05	Arah valid	*'Down'*	Bergerak ke kiri	Ular bergerak ke bawah	Ular bergerak ke bawah	Sesuai
A06	Arah valid	*'Down'*	Bergerak ke kanan	Ular bergerak ke bawah	Ular bergerak ke bawah	Sesuai
A07	Arah valid	*'Left'*	Bergerak ke kanan	Ular tetap bergerak ke kanan (input ditolak)	Ular tetap bergerak ke kanan	Sesuai
A08	Arah valid	*'Left'*	Bergerak ke atas	Ular bergerak ke kiri	Ular bergerak ke kiri	Sesuai
A09	Arah valid	*'Left'*	Bergerak ke bawah	Ular bergerak ke kiri	Ular bergerak ke kiri	Sesuai
A10	Arah valid	*'Right'*	Bergerak ke kiri	Ular tetap bergerak ke kiri (input ditolak)	Ular tetap bergerak ke kiri	Sesuai
A11	Arah valid	*'Right'*	Bergerak ke atas	Ular bergerak ke kanan	Ular bergerak ke kanan	Sesuai
A12	Arah valid	*'Right'*	Bergerak ke bawah	Ular bergerak ke kanan	Ular bergerak ke kanan	Sesuai
A13	Input tidak valid	'A'	Bergerak ke atas	Ular tetap bergerak ke atas (input diabaikan)	Ular tetap bergerak ke atas	Sesuai
A14	Input tidak valid	'1'	Bergerak ke bawah	Ular tetap bergerak ke bawah (input diabaikan)	Ular tetap bergerak ke bawah	Sesuai
A15	Input tidak valid	'#'	Bergerak ke kiri	Ular tetap bergerak ke kiri (input diabaikan)	Ular tetap bergerak ke kiri	Sesuai
A16	Input tidak valid	'Z'	Bergerak ke kanan	Ular tetap bergerak ke kanan (input diabaikan)	Ular tetap bergerak ke kanan	Sesuai

Tabel 7 menunjukkan hasil bahwa perilaku ular sesuai dengan yang diharapkan, di mana ular bertambah panjang dan skor bertambah saat memakan ayam, dan ular mati serta permainan berakhir ketika memakan ekornya sendiri. Untuk kasus ketika ular tidak memakan apa-apa, ular tetap hidup dan permainan masih berlanjut, sesuai dengan yang diharapkan.

Tabel 7. Hasil Pengujian Perilaku Ular

Test ID	Kondisi	Aturan 1	Aturan 2	Aturan 3	Aturan 4
B01	Memakan Ayam (T/F)	F	T	F	T
B02	Memakan Ekor Ular (T/F)	F	F	T	T

B03	Tidak Memakan Apa-apa (T/F)	T	T	T	F
	Output (E/H)	E	E	E	H
	Keterangan	Ular menampilkan reaksi atau perilaku tertentu	Ular menampilkan reaksi atau perilaku tertentu	Ular menampilkan reaksi atau perilaku tertentu	Ular tidak menampilkan reaksi atau perilaku tertentu
	Kesimpulan	Lolos	Lolos	Lolos	Lolos

Tabel 8 menunjukan hasil pengujian bahwa ular mati dan permainan berakhir saat menabrak tembok, dan permainan berlanjut normal jika ular tidak menabrak tembok, sesuai dengan ekspektasi.

Tabel 8. Hasil Pengujian Batas Tembok Terhadap Ular

Test ID	Rincian Pengujian	Hasil yang Diharapkan	Hasil Pengujian	Kesimpulan
C01	Ular menabrak batas tembok bagian atas	Ular mati karena menabrak batas atas	Ular mati setelah menabrak batas kemudian langsung muncul halaman game over	Sesuai
C02	Ular menabrak batas tembok bagian bawah	Ular mati karena menabrak batas bawah	Ular mati setelah menabrak batas kemudian langsung muncul halaman game over	Sesuai
C03	Ular menabrak batas tembok bagian kanan	Ular mati karena menabrak batas kanan	Ular mati setelah menabrak batas kemudian langsung muncul halaman game over	Sesuai
C04	Ular menabrak batas tembok bagian kiri	Ular mati karena menabrak batas kiri	Ular mati setelah menabrak batas kemudian langsung muncul halaman game over	Sesuai

Tabel 9 menunjukkan hasil bahwa skor baru tersimpan ketika pemain mencetak skor tertinggi baru, dan skor tidak berubah jika skor yang baru dicetak lebih rendah dari sebelumnya, yang sesuai dengan harapan.

Tabel 9. Hasil Pengujian Penyimpanan Skor

Test ID	Rincian Pengujian	Hasil yang Diharapkan	Hasil Pengujian	Kesimpulan
D01	Membuat rekor skor yang lebih tinggi dari sebelumnya	Skor baru tersimpan	Skor baru tersimpan	Sesuai
D02	Membuat rekor skor yang lebih rendah dari sebelumnya	Skor baru tidak tersimpan dan yang tertera adalah skor terbaik sebelumnya	Skor baru tidak tersimpan dan yang tertera adalah skor terbaik sebelumnya	Sesuai

Tabel 10 menunjukkan bahwa ular bergerak dengan kecepatan yang bervariasi pada setiap level. Terdapat tiga level berbeda: easy, medium, dan hard. Jika ular bergerak dengan kecepatan yang berbeda pada masing-masing level tersebut, maka hasil pengujian sesuai dengan ekspektasi.

Tabel 10. Hasil Pengujian Pada Level Kecepatan Yang Berbeda

Test ID	Rincian Pengujian	Langkah-langkah	Hasil yang Diharapkan	Hasil Pengujian	Kesimpulan
E01	Kecepatan level 'Easy'	1. Buka Permainan 2. Pilih level "Easy" 3. Mainkan Permainan	Ular bergerak dengan lambat	Ular bergerak dengan lambat	Sesuai
E02	Kecepatan level 'Normal'	1. Buka Permainan 2. Pilih level "Normal" 3. Mainkan Permainan	Ular bergerak dengan lebih cepat dari level 'Easy'	Ular bergerak lebih cepat dari level sebelumnya	Sesuai
E03	Kecepatan Level 'Hard'	1. Buka Permainan 2. Pilih level "Hard" 3. Mainkan Permainan	Ular bergerak dengan sangat cepat	Tidak ada perubahan kecepatan, kecepatan sama dengan level 'Easy'	Tidak sesuai

Hasil penelitian ini menunjukkan kinerja dan fungsionalitas dari berbagai aspek sistem permainan ular yang diuji. Tabel 6 menunjukkan hasil pengujian pergerakan ular. Ular bergerak sesuai dengan arah input kecuali jika input berlawanan dengan arah gerakan saat ini (misalnya, bergerak ke bawah tidak bisa langsung diganti menjadi ke atas). Input yang tidak valid diabaikan. Dari hasil ini, dapat disimpulkan bahwa pergerakan ular berfungsi dengan benar. Tabel 7 menggambarkan hasil pengujian perilaku ular. Ular bertambah panjang dan skor bertambah saat memakan ayam, serta mati jika memakan ekornya sendiri. Ular tetap hidup jika tidak memakan apa-apa. Kesimpulannya, perilaku ular sesuai dengan yang diharapkan. Tabel 8 membahas pengujian batas tembok terhadap ular. Ular mati dan permainan berakhir saat menabrak tembok, yang menunjukkan bahwa sistem menangani batas permainan dengan benar. Tabel 9 menyajikan hasil pengujian penyimpanan skor. Skor baru tersimpan jika lebih tinggi dari skor sebelumnya, dan skor tidak berubah jika skor baru lebih rendah dari skor sebelumnya. Ini menunjukkan bahwa fitur penyimpanan skor berfungsi dengan baik. Tabel 10 menampilkan pengujian pada level kecepatan yang berbeda. Ular bergerak dengan kecepatan yang sesuai pada level 'Easy' dan 'Normal', tetapi tidak menunjukkan perubahan kecepatan yang diharapkan pada level 'Hard'. Kesimpulannya, perlu dilakukan perbaikan pada penyesuaian kecepatan di level 'Hard'.

SIMPULAN

Berdasarkan hasil penelitian yang disajikan dalam tabel 6 sampai dengan tabel 10, kesimpulan yang dapat diambil adalah bahwa implementasi fitur-fitur dalam permainan ular telah sesuai dengan harapan yang telah ditetapkan. Dari pengujian pergerakan ular, pemberian makanan kepada ular, interaksi dengan tembok, hingga penyimpanan skor, semua hasilnya sesuai dengan yang diharapkan. Namun, pada pengujian kecepatan berdasarkan pemilihan level, terdapat bug pada test ID E03 dimana hasil yang di harapkan tidak sesuai dengan hasil pengujian. Hal ini menunjukkan bahwa pengembangan permainan belum memenuhi standar yang telah ditetapkan, sehingga dapat disimpulkan bahwa game perlu dilakukan pengembangan lebih lanjut.

DAFTAR PUSTAKA

[1] A. Ijudin and A. Saifudin, "Pengujian Black Box pada Aplikasi Berita Online dengan Menggunakan Metode Boundary Value Analysis," *Jurnal Informatika Universitas Pamulang*, vol. 5, no. 1, pp. 8–12, 2020.

[2] D. Ahrizal, M. K. Miftah, R. Kurniawan, T. Zaelani, and Yulianti, "Pengujian Perangkat Lunak Sistem Informasi Peminjaman PlayStation dengan Teknik Boundary Value Analysis Menggunakan Metode Black Box Testing," *Jurnal Informatika Universitas Pamulang*, vol. 5, no. 1, pp. 73–77, 2020, [Online]. Available: http://openjournal.unpam.ac.id/index.php/informatika73

[3] S. Maji and J. Frank, "What is in the black box? – A perspective on software in cryoelectron microscopy," *Biophys J*, vol. 120, no. 20, pp. 4307–4311, Oct. 2021, doi: 10.1016/j.bpj.2021.09.015.

[4] S. Nidhra and J. Dondeti, "Black Box and White Box Testing Techniques - A Literature Review," *International Journal of Embedded Systems and Applications*, vol. 2, no. 2, pp. 29–50, Jun. 2012, doi: 10.5121/ijesa.2012.2204.

[5] A. Kurniawan, A. Maulana, V. R. Sukma, W. Keumala, and A. Saifudin, "Pengujian Black Box pada Aplikasi Penjualan Berbasis Web Menggunakan Metode Equivalents Partitions (Studi Kasus: PT Arap Store)," *Jurnal Teknologi Sistem Informasi dan Aplikasi* , vol. 3, no. 1, pp. 50–56, 2020, [Online]. Available: http://openjournal.unpam.ac.id/index.php/JTSI50

[6] I. Permatasari, F. Adhania, S. A. Putri, and S. R. C. Nursari, "Pengujian Black Box Menggunakan Metode Analisis Nilai Batas pada Aplikasi DANA," *Konvergensi Teknologi dan Sistem Informasi*, vol. 3, no. 2, pp. 373–387, 2023.

[7] C. Fadhana, Anwar, and M. Nasir, "Rancang Bangun Sistem Mobile Learning Sebagai Media Pembelajaran Dengan Metode Pengujian Black Box Testing," *Jurnal Teknologi Rekayasa Informasi dan Komputer*, vol. 2, no. 2, 2019.

[8] F. C. Ningrum, D. Suherman, S. Aryanti, H. A. Prasetya, and A. Saifudin, "Pengujian Black Box pada Aplikasi Sistem Seleksi Sales Terbaik Menggunakan Teknik Equivalence Partitions," *Jurnal Informatika Universitas Pamulang*, vol. 4, no. 4, pp. 125–130, 2019, [Online]. Available: http://openjournal.unpam.ac.id/index.php/informatika

[9] R. R. N. Fikri, I. Indera, A. Rahardi, and I. Agus, "Pengujian Blackbox pada Sistem Informasi Komunitas Pecinta Kucing di Bandar Lampung," *TEKNIKA*, vol. 18, no. 1, pp. 25–34, 2024.

[10] Y. D. Wijaya and M. W. Astuti, "PENGUJIAN BLACKBOX SISTEM INFORMASI PENILAIAN KINERJA KARYAWAN PT INKA (PERSERO) BERBASIS EQUIVALENCE PARTITIONS," *Jurnal Digital Teknologi Informasi*, vol. 4, no. 1, pp. 22–26, 2021.

[11] S. D. S. Saian, N. L. Kakihary, and T. Wahyono, "PENGUJIAN CONTENT MANAGEMENT SYSTEM (CMS) SEKOLAHKU MENGGUNAKAN METODE BLACK BOX TESTING DENGAN TEKNIK BOUNDARY VALUE ANALYSIS," *Jurnal Penerapan Teknologi Informasi dan Komunikasi*, vol. 1, no. 2, pp. 100–113, 2022, [Online]. Available: https://sekolahku.web.id/

[12] A. Firmansyah, M. A. Arief, M. D. F. Falah, O. D. Dharmawan, and J. Riyanto, "Pengujian Aplikasi Sistem Penilaian Mahasiswa Dengan Menggunakan Teknik Boundary Value Analysis," *Scientia Sacra: Jurnal Sains, Teknologi dan Masyarakat*, vol. 2,

no. 1, pp. 175–179, 2022, [Online]. Available: http://pijarpemikiran.com/index.php/Scientia

[13] N. M. Arofiq, A. Laksana, and A. Saifudin, "Pengujian Sistem Schedule Planning Produksi Dengan Metode Black Box Testing pada PT. Smartfren Telecom TBK Untuk Pemula," *Teknologi, Bisnis Dan Pendidikan*, vol. 1, no. 1, pp. 71–79, 2023, [Online]. Available: https://jurnalmahasiswa.com/index.php/teknobis

[14] S. D. Pratama, Lasimin, and M. N. Dadaprawira, "Pengujian Black Box Testing Pada Aplikasi Edu Digital Berbasis Website Menggunakan Metode Equivalence Dan Boundary Value," *Jurnal Teknologi Sistem Informasi dan Sistem Komputer TGD*, vol. 6, no. 2, pp. 560–569, 2023, [Online]. Available: https://ojs.trigunadharma.ac.id/index.php/jsk/index

[15] Meenu and Navita, "Study and Analysis of Software Testing," *International Journal on Recent and Innovation Trends in Computing and Communication* , vol. 3, no. 12, pp. 6674–6678, 2015, [Online]. Available: http://www.ijritcc.org