BAB 4
IMPLEMENTASI DAN PENGUJIAN

4.1 Implementasi

Tahap ini merupakan tahapan dimana aplikasi yang sudah dianalisis dan dirancang akan diimplementasikan.

4.1.1 Implementasi Perangkat Keras

Perangkat keras yang akan digunakan untuk mengimplementasikan aplikasi yang dibuat, sebagai berikut :

1. Perangkat Mobile Android
2. CPU 1 GHz dual-core Cortex-A9
3. Memory internal 8 GB storage, 1 GB RAM
4. Ukuran Layar 480x800 pixel (WGA)

4.1.2 Implementasi Perangkat Lunak

Perangkat lunak yang akan digunakan untuk menunjang aplikasi berjalan pada perangkat keras yaitu sistem operasi Android 4.1 (Jelly Bean).

4.1.3 Implementasi Class

Implementasi *class* merupakan implementasi dari perancangan *class diagram* yang terbentuk dari *use case diagram* kedalam bentuk file fisik dengan ekstensi ".java". Implementasi *class* bisa dilihat pada tabel 4.1 dibawah ini :

Tabel 4.1 Implementasi Class Game Kabayan Behitung

No	Nama Class	Nama File Fisik
1	MainGameKabayanBerhitung	MainGameKabayanBerhitung.java
2	GameView	GameView.java
3	GameLoopThread	GameLoopThread.java
4	Scenes	Scenes.java
5	Menu	Menu.java
6	Life	Life.java
7	Intro	Intro.java
8	IntroGame1	IntroGame1.java
9	IntroGame2	IntroGame2.java
10	IntroPembukaLevel	IntroPembukaLevel.java
11	Petunjuk	Petunjuk.java
12	Pesan	Pesan.java
13	Level	Level.java
14	Level1	Level1.java
15	Level2	Level2.java
16	Level3	Level3.java

17	Ladang	Ladang.java
18	Graph	Graph.java
19	Pertanyaan	Pertanyaan.java
20	Sandi	Sandi.java
21	Kabayan	Kabayan.java
22	Font	Font.java
23	OpenImage	OpenImage.java
24	Image	Image.java
25	Sprite	Sprite.java
26	Audio	Audio.java
27	Sound	Sound.java
28	Music	Music.java
29	StateGame	StateGame.java
30	Assets	Assets.java

4.1.4 Implementasi Antarmuka

Pada bagian ini akan ditampilkan implementasi antarmuka pada *Game* Kabayan Berhitung.

4.1.4.1 Implementasi Antarmuka Menu Utama

Berikut adalah Implementasi Antarmuka Menu Utama.



Gambar 4.1 Implementasi Antarmuka Menu Utama

4.1.4.2 Implementasi Antarmuka Cerita Kabayan ke 1

Berikut adalah Implementasi Antarmuka Cerita Kabayan ke 1



Gambar 4.2 Implementasi Antarmuka Cerita Kabayan ke 1

4.1.4.3 Implementasi Antarmuka Cerita Kabayan ke 2

Berikut adalah Implementasi Antarmuka Cerita Kabayan ke 2.



Gambar 4.3 Implementasi Antarmuka Cerita Kabayan ke 2

4.1.4.4 Implementasi Antarmuka Intro Pembuka Pelatihan 1

Berikut adalah Implementasi Antarmuka Intro Pembuka Pelatihan 1



Gambar 4.4 Implementasi Antarmuka Inatro Pembuka Pelatihan 1

4.1.4.5 Implementasi Antarmuka Petunjuk Pelatihan 1

Berikut adalah Implementasi Antarmuka Petunjuk Pelatihan 1.



Gambar 4.5 Implementasi Petunjuk Pelatihan 1 ke 1



Gambar 4.6 Implementasi Petunjuk Pelatihan 1 ke 2

4.1.4.6 Implementasi Antarmuka Pelatihan 1

Berikut adalah Implementasi Antarmuka Pelatihan 1



Gambar 4.7 Implementasi Pelatihan 1

4.1.4.7 Implementasi Antarmuka Intro Pembuka Pelatihan 2

Berikut adalah Implementasi Antarmuka Intro Pembuka Pelatihan 2



Gambar 4.8 Implementasi Antarmuka Inatro Pembuka Pelatihan 2

4.1.4.8 Implementasi Antarmuka Petunjuk Pelatihan 2

Berikut adalah Implementasi Antarmuka Petunjuk Pelatihan 2.



Gambar 4.9 Implementasi Petunjuk Pelatihan 2 ke 1



Gambar 4.10 Implementasi Petunjuk Pelatihan 2 ke 2

4.1.4.9 Implementasi Antarmuka Pelatihan 2 Graph 1

Berikut adalah Implementasi Antarmuka Pelatihan 2 Graph 1



Gambar 4.11 Implementasi Pelatihan 2 Graph 1

4.1.4.10 Implementasi Antarmuka Pelatihan 2 Graph 2

Berikut adalah Implementasi Antarmuka Pelatihan 2 Graph 2



Gambar 4.12 Implementasi Pelatihan 2 Graph 2

4.1.4.11 Implementasi Antarmuka Pelatihan 2 Graph 3

Berikut adalah Implementasi Antarmuka Pelatihan 2 Graph 3



Gambar 4.13 Implementasi Pelatihan 2 Graph 3

4.1.4.12 Implementasi Antarmuka Pelatihan 2 Pertanyaan

Berikut adalah Implementasi Antarmuka Pelatihan 2 Pertanyaan



Gambar 4.14 Implementasi Pelatihan 2 Pertanyaan

4.1.4.13 Implementasi Antarmuka Intro Pembuka Pelatihan 3

Berikut adalah Implementasi Antarmuka Intro Pembuka Pelatihan 3



Gambar 4.15 Implementasi Antarmuka Inatro Pembuka Pelatihan 3

4.1.4.14 Implementasi Antarmuka Petunjuk Pelatihan 3

Berikut adalah Implementasi Antarmuka Petunjuk Pelatihan 3.



Gambar 4.16 Implementasi Petunjuk Pelatihan 3 ke 1



Gambar 4.17 Implementasi Petunjuk Pelatihan 3 ke 2

4.1.4.15 Implementasi Antarmuka Pelatihan 3 Pintu 1

Berikut adalah Implementasi Antarmuka Pelatihan 3 Pintu 1



Gambar 4.18 Implementasi Pelatihan 3 Pintu 1

4.1.4.16 Implementasi Antarmuka Pelatihan 3 Pintu 2

Berikut adalah Implementasi Antarmuka Pelatihan 3 Pintu 2



Gambar 4.19 Implementasi Pelatihan 3 Pintu 2

4.1.4.17 Implementasi Antarmuka Pelatihan 3 Pintu 3

Berikut adalah Implementasi Antarmuka Pelatihan 3 Pintu 3



Gambar 4.20 Implementasi Pelatihan 3 Pintu 3

4.1.4.18 Implementasi Antarmuka Pelatihan 3 Pertanyaan

Berikut adalah Implementasi Antarmuka Pelatihan 3 Pertanyaan



Gambar 4.21 Implementasi Pelatihan 3 Pertanyaan

4.1.4.19 Implementasi Antarmuka Pelatihan 3 Sandi

Berikut adalah Implementasi Antarmuka Pelatihan 3 Sandi



Gambar 4.22 Implementasi Pelatihan 3 Sandi

4.1.5 Implementasi Pesan

Pada bagian ini akan ditampilkan implementasi pesan pada *Game* Kabayan Berhitung.

4.1.5.1 Implementasi Pesan Jawaban Benar

Berikut adalah Implementasi Pesan Jawaban Benar.



Gambar 4.23 Implementasi Pesan Jawaban Benar

4.1.5.2 Implementasi Pesan Jawaban Salah

Berikut adalah Implementasi Pesan Jawaban Salah.



Gambar 4.24 Implementasi Pesan Jawaban Salah

4.1.5.3 Implementasi Pesan Jawaban Waktu Habis

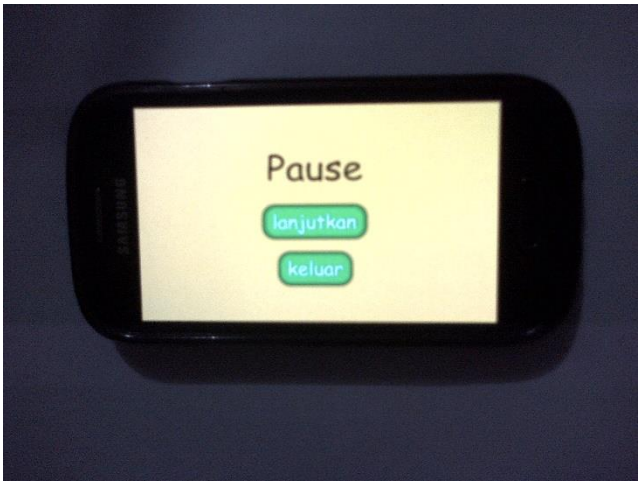
Berikut adalah Implementasi Pesan Jawaban Waktu Habis.



Gambar 4.25 Implementasi Pesan Waktu Habis

4.1.5.4 Implementasi Pesan Menu Pause

Berikut adalah Implementasi Pesan Menu Pause.



Gambar 4.26 Implementasi Pesan Menu Pause

4.1.5.5 Implementasi Pesan Menu Keluar

Berikut adalah Implementasi Pesan Menu Keluar.



Gambar 4.27 Implementasi Pesan Menu Keluar

4.1.5.6 Implementasi Pesan Gagal

Berikut adalah Implementasi Pesan Gagal.



Gambar 4.28 Implementasi Pesan Gagal

4.1.5.7 Implementasi Pesan Tamat

Berikut adalah Implementasi Pesan Tamat.



Gambar 4.29 Implementasi Pesan Tamat

4.2 Pengujian Sistem

Pengujian bermaksud untuk mengetahui perangkat lunak yang dibuat telah memenuhi kriteria yang sesuai dengan tujuan perancangan perangkat lunak tersebut.

4.2.1 Rencana Pengujian

Pengujian akan dilakukan melalui dua tahap, yaitu pengujian *alpha* dan pengujian *beta*.

1. Pengujian *Alpha*

Metode yang digunakan dalam pengujian *alpha* adalah *metode black box* yang fokus pada persyaratan fungsional dari perangkat lunak yang dibangun.

2. Pengujian *Beta*

Metode yang digunakan dalam pengujian *beta* merupakan pengujian yang dilakukan secara objektif terhadap pengguna.

4.2.2 Pengujian *Alpha*

Pengujian dilakukan terhadap aplikasi untuk memastikan bahwa aplikasi berjalan dengan benar sesuai dengan kebutuhan dan tujuan yang diharapkan. Proses pengujian akan menggunakan *Black Box Testing*.

4.2.2.1 *Scenario* Pengujian

Scenario pengujian menjelaskan pengujian terhadap sistem yang ada pada *game* Kabayan berhitung. Pengujian dilakukan dengan melakukan tes setiap *use case*, algoritma dan memungkinkan kesalahan yang terjadi pada setiap proses. Pengujian ini dilakukan secara *black box* yaitu pengujian dilakukan dengan hanya memperhatikan masukan ke sistem dan keluaran sistem. Tabel berikut ini menjelaskan *scenario* pengujian yang akan dilakukan.

Tabel 4.2 *Scenario* Pengujian

Use case	Butir Pengujian	Jenis Pengujian
Memulai Permainan	Menyentuh tombol mulai	Blackbox
Mencangkul Ladang	Memilih menyentuh ladang	Blackbox
Mencari Jalur Pada Graph	Memilih titik tujuan	Blackbox
	Mencapai titik akhir	Blackbox
Menjawab Pertanyaan	Menyentuh angka-angka untuk memasukan jawaban	Blackbox
	Menyentuh tombol sudah	Blackbox
	Menyentuh tombol hapus	Blackbox
Memasukan Sandi	Menyentuh kolom puluhan	Blackbox
	Menyentuh angka-angka untuk memasukan puluhan	Blackbox
	Menyentuh kolom satuan	Blackbox
	Menyentuh angka-angka untuk memasukan satuan	Blackbox

	Menyentuh tombol sudah	Blackbox
	Menyentuh tombol hapus	Blackbox
Algoritma	**Butir Pengujian**	**Jenis Pengujian**
Dijkstra	Menghasilkan jarak terpendek dan rute terpendek dari graph yang tertampil	Whitebox
Fisher-Yates	Menghasilkan jawaban yang teracak	Whitebox

4.2.2.2 Pengujian Blackbox

Tabel berikut adalah hasil dari pengujian *black box* yang dilakukan sesuai pengujian *scenario* pengujian yang telah dirancang sebelumnya.

Tabel 4.3 Pengujian Blackbox

Memulai Permainan			
Sekenario	Hasil yang diharapkan	Hasil pengujian	
Menyentuh tombol mulai	Menampilkan cerita Kabayan	[√] diterima	[ ] ditolak
Mencangkul ladang			
Sekenario	Hasil yang diharapkan	Hasil pengujian	
Memilih menyentuh ladang	Menampilkan angka pada ladang yang disentuh dan Kabayan mencangkul	[√] diterima	[ ] ditolak
Mencari Jalur Pada Graph			
Sekenario	Hasil yang diharapkan	Hasil pengujian	
Memilih titik tujuan	Menambah atau mengurangi jarak yang sudah ditempuh pemain dan menampilkan Kabayan bergerak ke titik yang dipilih	[√] diterima	[ ] ditolak
Mencapai titik akhir pada graph	Memeriksa jarak yang sudah dilalui pemain	[√] diterima	[ ] ditolak
	Menampilkan pesan	[√] diterima	[ ] ditolak
Menjawab Pertanyaan			
Sekenario	Hasil yang diharapkan	Hasil pengujian	
Menyentuh angka-angka untuk memasukan jawaban	Menampilkan angka yang sudah disentuh pemain di kolom jawaban	[√] diterima	[ ] ditolak
Menyentuh tombol sudah	Memeriksa jawaban	[√] diterima	[ ] ditolak
	Menampilkan pesan	[√] diterima	[ ] ditolak
Menjawab Pertanyaan			
Sekenario	Hasil yang diharapkan	Hasil pengujian	
Menyentuh kolom puluhan	Menampilkan kolom puluhan tersorot	[√] diterima	[ ] ditolak

Menyentuh angka-angka untuk memasukan puluhan	Menampilkan angka yang disentuh pada kolom puluhan	[√] diterima	[ ] ditolak
Menyentuh kolom satuan	Menampilkan kolom satuan tersorot	[√] diterima	[ ] ditolak
Menyentuh angka-angka untuk memasukan satuan	Menampilkan angka yang disentuh pada kolom satuan	[√] diterima	[ ] ditolak
Menyentuh tombol sudah	Memeriksa sandi	[√] diterima	[ ] ditolak
	Menampilkan pesan	[√] diterima	[ ] ditolak
Menyentuh tombol hapus	Jika kolom puluhan tersorot, maka menghapus angka pada kolom puluhan	[√] diterima	[ ] ditolak
	Jika kolom satuan tersorot, maka menghapus angka pada kolom satuan	[√] diterima	[ ] ditolak

4.2.2.3 Pengujian Whitebox

Pengujian *white box* merupakan pengujian yang didasarkan pada pengecekan terhadap detail perancangan, menggunakan struktur *control* dari desain aplikasi secara prosedural untuk membagi pengujian ke dalam beberapa kasus pengujian untuk mengetahui apakah setiap node pada *independent paths* dikerjakan. Serta untuk menguji alur logika, perulangan, dan validasi parameter masukannya.

a Pengujian terhadap Algoritma Dijkstra

Pengujian untuk kinerja algoritma Dijkstra pada game Kabayan berhitung akan dijelaskan, sebagai berikut.

1. Sources Code Dijkstra pada *game* Kabayan berhitung

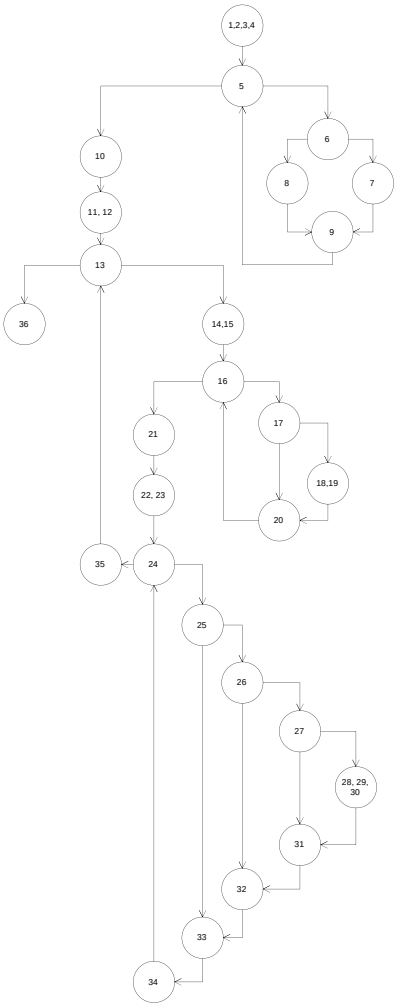
Pada tahap ini sources code akan diuji dan diberi nomor untuk menentukan graph yang terbentuk.

```
public void dijkstra(int[][] d, int from) {
1.    Set<Integer> set = new HashSet<Integer>();
2.    set.add(from);
3.    boolean[] closed = new boolean[d.length];
4.    distances = new int[d.length];
5.    for (int i = 0; i < d.length; i++) {
6.       if (i != from) {
7.          distances[i] = Integer.MAX_VALUE;
         } else {
8.          distances[i] = 0;
9.       }
10.   }
11.   predecessors = new int[d.length];
12.   predecessors[from] = -1;
13.   while (!set.isEmpty()) {
```

```
14.       int minDistance = Integer.MAX_VALUE;
15.       int node = -1;
16.       for (Integer i : set) {
17.          if (distances[i] < minDistance) {
18.             minDistance = distances[i];
19.             node = i;
20.          }
21.       }
22.       set.remove(node);
23.       closed[node] = true;
24.       for (int i = 0; i < d.length; i++) {
25.          if (d[node][i] != Integer.MAX_VALUE) {
26.             if (!closed[i]) {
27.                if (distances[node] + d[node][i] < distances[i]) {
28.                     distances[i] = distances[node] + d[node][i];
29.                     predecessors[i] = node;
30.                     set.add(i);
31.                }
32.             }
33.          }
34.       }
35.    }
36.}
```

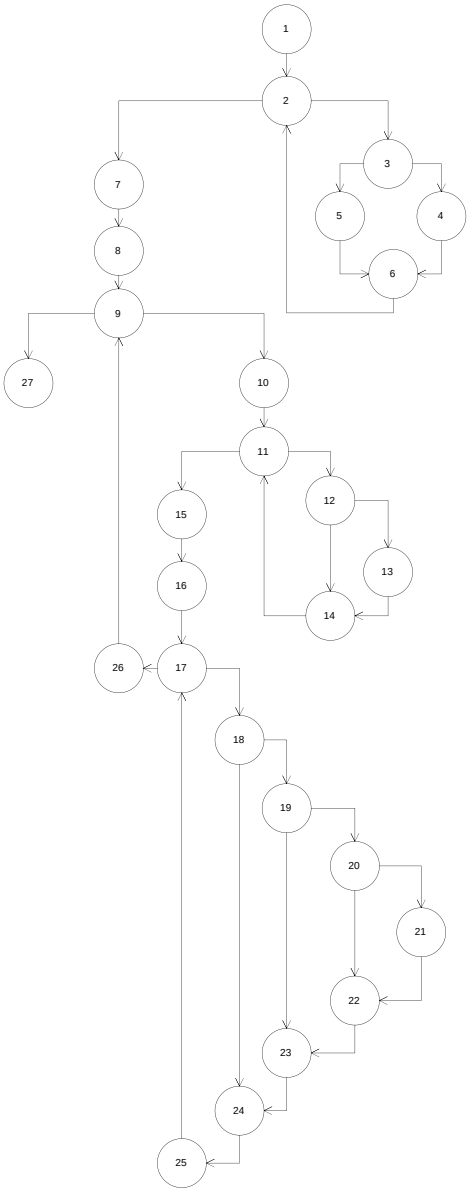
2. Flow Graph

Pada tahap ini sources code yang sudah diberi nomor sesuai dengan proses yang dilakukan, akan dibuatkan sebuah graph alur proses.



Gambar 4.30 *flow graph* Dijkstra

Setelah dibuat *flow graph* dengan setiap proses didalam satu node, maka perlu dilakukan penyederhanan bahwa dalam 1 node dapat mewakili beberapa proses. Penyederhanaan tersebut akan dijelaskan, sebagai berikut.



Gambar 4.31 Penyederhanaan *flow graph* Dijkstra

3. Cyclomatic Complexity

Pada tahap ini merupakan tahap pengukuran yang menyediakan ukuran kuantitatif dari kompleksitas logika. Tahap ini digunakan untuk menentukan *independent paths*.

E = 35

N = 27

V(G) = E - N + 2

V(G) = 35 - 27 + 2

V(G) = 10

Keterangan :

E : Jumlah busur pada flowgraph

N : Jumlah simpul pada flowgraph

V(G) : Cyclomatic Complexity

4. Independent Path

   Berikut adalah *independent pahts* yang terjadi.

Tabel 4.4 Independent Path Dijkstra

No Path	Path
1	1-2-3-4-6-2-7-8-9-10-11-12-13-14-11-15-16-17-18-19-20-21-22-23-24-25-17-26-9-27
2	1-2-3-5-6-2-7-8-9-10-11-12-13-14-11-15-16-17-18-19-20-21-22-23-24-25-17-26-9-27
3	1-2-7-8-9-10-11-12-13-14-11-15-16-17-18-19-20-21-22-23-24-25-17-26-9-27
4	1-2-7-8-9-10-11-12-14-11-15-16-17-18-19-20-21-22-23-24-25-17-26-9-27
5	1-2-7-8-9-10-11-15-16-17-18-19-20-21-22-23-24-25-17-26-9-27
6	1-2-7-8-9-10-11-15-16-17-18-19-20-22-23-24-25-17-26-9-27
7	1-2-7-8-9-10-11-15-16-17-18-19-23-24-25-17-26-9-27
8	1-2-7-8-9-10-11-15-16-17-18-24-25-17-26-9-27
9	1-2-7-8-9-10-11-15-16-17-26-9-27
10	1-2-7-8-9-27

5. Graph Matrices Testing

Berikut pemaparan dari hasil graph ke bentuk matrik.

Tabel 4.5 Graph Matrices Testing Dijkstra

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	Perhitungan
1		1																										0
2			1				1																					1
3				1	1																							1
4						1																						0
5						1																						0
6		1																										0
7								1																				0
8									1																			0
9										1																	1	1
10											1																	0
11												1			1													1
12													1	1														1
13														1														0
14											1																	0
15																1												0
16																	1											0
17																		1								1		1
18																			1					1				1
19																				1			1					1
20																					1	1						1
21																						1						0
22																							1					0
23																								1				0
24																									1			0
25																	1											0
26									1																			0
27																												0
Jumlah																												9

V(G) = Jumlah *graph matrices* + 1

V(G) = 10

6. Hasil Pengujian

Setelah melakukan pengujian dapat disimpulkan hasil dari pengujian yang dilakukan, sebagai berikut.

Tabel 4.6 Hasil Pengujian Dijkstra

No Path	Hasil yang diharapkan	Hasil Sesuai Uji	Keterangan
1	Menginisialisasi jarak dengan nilai tak terhingga	Menginisialisasi jarak dengan nilai tak terhingga	[√] alur terlewati [ ] alur tidak terlewati
2	Menginisialisasi jarak awal denan nilai 0	Menginisialisasi jarak awal denan nilai 0	[√] alur terlewati [ ] alur tidak terlewati
3	Memeriksa bahwa di dalam himpunan node atau vertex masih belum kosong	Memeriksa bahwa di dalam himpunan node atau vertex masih belum kosong	[√] alur terlewati [ ] alur tidak terlewati
4	Memeriksa seluruh himpunan yang menjadi	Memeriksa seluruh himpunan yang	[√] alur terlewati [ ] alur tidak terlewati

	nilai sementara.	menjadi nilai sementara.	
5	Memeriksa jarak terkecil yang pada himpunan dengan jarak minimal	Memeriksa jarak terkecil yang pada himpunan dengan jarak minimal	[√] alur terlewati [ ] alur tidak terlewati
6	Memeriksa tetangga dari node atau vertex yang diperiksa	Memeriksa tetangga dari node atau vertex yang diperiksa	[√] alur terlewati [ ] alur tidak terlewati
7	Membandingkan tetangga dari vertex yang diperiksa dengan nilai terbesar dari Integer	Membandingkan tetangga dari vertex yang diperiksa dengan nilai terbesar dari Integer	[√] alur terlewati [ ] alur tidak terlewati
8	Memeriksa tetangga dari vertex yang diperiksa tidak dalam himpunan closed	Memeriksa tetangga dari vertex yang diperiksa tidak dalam himpunan closed	[√] alur terlewati [ ] alur tidak terlewati
9	Membandingkan jarak yang sudah ditambah dengan vertex tetangga dengan vertex yang sebelumnya yang menjadi minimal	Membandingkan jarak yang sudah ditambah dengan vertex tetangga dengan vertex yang sebelumnya yang menjadi minimal	[√] alur terlewati [ ] alur tidak terlewati
10	Memeriksa bahwa himpunan sudah kosong	Memeriksa bahwa himpunan sudah kosong	[√] alur terlewati [ ] alur tidak terlewati

b Pengujian terhadap Algoritma Fisher-Yates

Pengujian untuk kinerja algoritma Fisher-Yates pada game Kabayan berhitung akan dijelaskan, sebagai berikut.

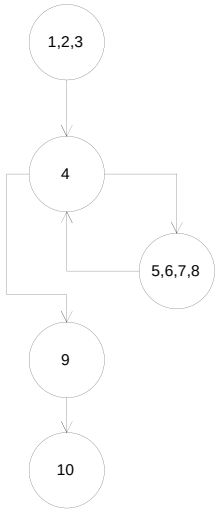
1. Sources Code Fisher-Yates pada *game* Kabayan berhitung

Pada tahap ini sources code akan diuji dan diberi nomor untuk menentukan graph yang terbentuk.

```
public int[] fisherYates(int[] array) {
1.   int m = 20;
2.   int i = 0;
3.   int t = 0;
4.   while (m > 0) {
5.      i = (int) (Math.random() * m--);
6.      t = array[m];
7.      array[m] = array[i];
8.      array[i] = t;
9.   }
10.  return array;
   }
```

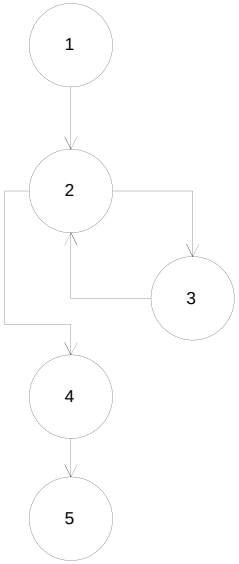
2. Flow Graph

Pada tahap ini sources code yang sudah diberi nomor sesuai dengan proses yang dilakukan, akan dibuatkan sebuah graph alur proses.



Gambar 4.32 *flow graph* Fisher-Yates

Setelah dibuat *flow graph* dengan setiap proses didalam satu node, maka perlu dilakukan penyederhanan bahwa dalam 1 node dapat mewakili beberapa proses. Penyederhanaan tersebut akan dijelaskan, sebagai berikut.



Gambar 4.33 Penyederahanaan *flow graph* Fisher-Yates

3. Cyclomatic Complexity

Pada tahap ini merupakan tahap pengukuran yang menyediakan ukuran kuantitatif dari kompleksitas logika. Tahap ini digunakan untuk menentukan *independent paths*.

E = 5

N = 5

V(G) = E - N + 2

V(G) = 5 - 5 + 2

V(G) = 2

Keterangan :

E : Jumlah busur pada flowgraph

N : Jumlah simpul pada flowgraph

V(G) : Cyclomatic Complexity

4. Independent Path

Berikut adalah *independent pahts* yang terjadi.

Tabel 4.7 *independent pahts* Fisher-Yates

No Path	Path
1	1-2-3-2-4-5
2	1-2-4-5

5. Graph Matrices Testing

Berikut pemaparan dari hasil graph ke bentuk matrik.

Tabel 4.8 Graph Matrices Testing Fisher Yates

	1	2	3	4	5	Perhitungan
1		1				0
2			1	1		1
3		1				0
4					1	0
5						0
Jumlah						1

V(G) = Jumlah *graph matrices* + 1

V(G) = 2

6. Hasil Pengujian

Setelah melakukan pengujian dapat disimpulkan hasil dari pengujian yang dilakukan, sebagai berikut.

Tabel 4.9 Hasil Pengujian Fisher-Yates

No Path	Hasil yang diharapkan	Hasil Sesuai Uji	Keterangan
1	Menghasilkan array yang sudah di-shuffle	Menghasilkan array yang sudah di-shuffle	[√] alur terlewati [ ] alur tidak terlewati
2	Array tidak dishuffle karena array kosong	Array tidak dishuffle karena array kosong	[√] alur terlewati [ ] alur tidak terlewati

4.2.3 Pengujian *Beta*

Pengujian *beta* merupakan pengujian yang dilakukan secara objektif. Pengujian dilakukan terhadap 30 responden yang merupakan siswa sekolah dasar kelas 2 – 4 (8 – 10 tahun). Responden disesuaikan dengan analisis pengguna pada pembangunan aplikasi *game* Kabayan berhitung ini.

4.2.3.1 Pretest-Posttest

Pretest dan Postest merupakan pengujian terhadap pengguna untuk mengetahui kemapuan pengguna sebelum dan sesudah menggunakan program atau aplikasi. Hasil yang didapat akan menjadi bahan perbandingan dari tujuan penelitian ini.

1. *Scenario* Pengujian *Pretest-Postest*

Pengujian akan terbagi 2 yaitu pretest (pengujian sebelum menjalankan aplikasi) dan pretest (pengujian ketika menjalankan aplikasi). Pengujian yang dilakukan sebagai berikut :

1. *Pretest*

Pada pengujian *pretest* siswa akan diberikan soal pilihan ganda dengan materi yang sama dengan aplikasi *game* yang akan dimainkan. Soal yang terdapat pada pengujian *pretest* telah ditentukan sebelumnya dan setiap siswa akan diberikan soal yang sama. Dalam pengerjaannya siswa yang akan diuji satu persatu. Kriteria penilaian dalam *pretest* akan dijelaskan sebagai berikut :

Tabel 4.10 Keriteria Penilaian Pada Pengujian *Pretest*

Materi Yang Diuji	Jumlah Soal	Nilai Setiap Menjawab 1 Soal	Total Nilai Jika Menjawab Semua Soal Dengan Benar
Pertambahan dan Pengurangan	5	5	10
Perkalian dan Pembagian	3	10	30
Campuran	3	15	45
Jumlah			100

2. *Postest*

Pada pengujian *postest* siswa akan menjalankan aplikasi *game* yang dibuat yaitu *game* Kabayan Berhitung. Pada pengujiannya setiap siswa akan memainkan *game* ini satu persatu. Materi yang akan diuji pada siswa sama dengan pengujian *pretest*. Perbedaan dari pengujian *pretest* selain menjalankan *game*, terdapat pada soal yang akan diuji. Soal yang akan diuji semuanya diberikan secara acak, sehingga kemungkinan kecil untuk soal yang diuji pada *pretest* akan sama dengan pada *postest*. Pada pelatihan 1 waktu yang digunakan untuk menjawab tidak diaktifkan, karena pada pengujian *pretest* siswa tidak diberikan waktu untuk menjawab. Kriteria penilaian pada *pretest* akan sebagai berikut :

Tabel 4.11 Keriteria Penilaian Pada Pengujian *Postest*

Pelatihan	Materi Yang Diuji	Jumlah Soal	Nilai Setiap Menjawab 1 Soal	Total Nilai Jika Menjawab Semua Soal Dengan Benar
1	Pertambahan dan Pengurangan	5	5	10
2	Perkalian dan Pembagian	3	10	30
3	Campuran	3	15	45
Jumlah				100

2. Hasil Pengujian *Pretest-Posttest*

Hasil pengujian *pretest-postest* akan dijelaskan sebagai berikut.

Tabel 4.12 Hasil Pretest Posttest

No Responden	Hasil Nilai							
	Penjumlahan dan Pengurangan		Perkalian dan Pembagian		Campuran		Total	
	Pretest	Postest	Pretest	Postest	Pretest	Postest	Pretest	Postest
1	25	25	20	30	30	45	75	100
2	25	25	20	30	30	30	75	85

3	25	25	30	30	45	45	100	100
4	25	25	30	30	15	30	70	85
5	25	25	20	20	45	45	90	90
6	25	25	20	30	45	30	90	85
7	25	25	20	30	30	45	75	100
8	25	25	10	20	30	30	65	75
9	25	25	30	30	15	30	70	85
10	25	25	20	30	30	30	75	85
11	20	25	10	20	30	30	60	75
12	25	25	20	30	30	30	75	85
13	25	25	30	30	45	30	100	85
14	25	25	30	30	45	45	100	100
15	25	25	10	30	15	30	50	85
16	25	25	10	20	30	30	65	75
17	25	25	30	30	45	45	100	100
18	25	25	30	30	45	45	100	100
19	25	25	30	30	45	45	100	100
20	25	25	30	30	30	45	85	100
21	20	25	10	20	15	30	45	75
22	25	25	30	30	30	30	85	85
23	25	25	30	20	15	30	70	75
24	25	25	30	30	45	45	100	100
25	25	25	30	30	15	30	70	85
26	25	25	30	30	30	45	85	100
27	25	25	30	30	45	45	100	100
28	25	25	30	30	30	30	85	85
29	25	25	30	30	45	45	100	100
30	25	25	20	30	30	30	75	85

Dari hasil yang didapat, kemudian dihitung rata-rata dari setiap pengujian pretest dan postest untuk mengetahui peningkatan nilai yang terjadi pada setiap materi yang diuji. Hasil yang didapat akan direperesentasikan, sebagai berikut.

Tabel 4.13 Persentase Peningkatan

Jenis Test	Nilai rata-rata pretest	Nilai rata-rata postest	Peningkatan nilai	Presentase peningkatan
Penjumlahan dan Pengurangan	24,67	25,00	0,33	1,35%
Perkalian dan Pembagian	24,00	28,00	4,00	16,67%
Campuran	32,50	36,50	4,00	12,31%
Total	81,17	89,50	8,33	10,27%

4.2.3.2 Kesimpulan Pengujian *Beta*

Berdasarkan hasil pengujian *beta* dalam bentuk *pretst-posttest*, dapat disimpulkan sebagai berikut.

1. Pada operasi hitung pertambahan dan pengurangan didapatkan hasil presentase peningkatan hanya 1,35%. Dikarenakan responden sudah mampu untuk melakukan operasi hitung pertambahan dan pengurangan.
2. Pada operasi hitung perkalian dan pembagian didapatkan hasil presentase peningkatan sebesar 16,67%.
3. Pada operasi hitung campuran didapatkan hasil presentase peningkatan sebesar 12,31%.
4. Dari hasil secara keseluruhan didapatkan hasil presentase peningkatan sebesar 10,27%. Peningkatan ini membuktikan bahwa minat belajar anak dan pembelajaran melalui media *game mobile* terbukti efektif.