BAB 4

IMPLEMENTASI DAN PENGUJIAN

Pada bab ini akan dilakukan implementasi dan pengujian terhadap sistem. Tahapan ini dilakukan setelah perancangan selesai dilakukan dan selanjutnya akan diimplementasikan pada bahasa pemrograman yang akan digunakan. Setelah implementasi maka dilakukan pengujian sistem dimana akan dilihat kekurangan-kekurangan pada aplikasi untuk pengembangan sistem selanjutnya.

4.1 Implementasi

Seteleh menganalisis dan merancang sistem yang akan dibangun secara rinci, maka selanjutnya menuju tahap implementasi.

Tujuan implementasi adalah untuk mengkonfirmasikan modul program perancangan kepada para pengguna sistem, sehingga pengguna dapat memberikan masukkan terhadap pembangunan sistem.

4.1.1 Kebutuhan Perangkat Keras

Spesifikasi perangkat keras (*hardware*) yang digunakan dalam pembangunan *game* edukasi IPA dan Matematika untuk siswa SMP dapat dilihat pada tabel berikut.

Tabel 4. 1 Perangkat Keras Yang Digunakan

NO	Perangkat Keras	Keterangan
1	Processor	Intel Pentium 4 CPU 2.80GHz 2.10GHz
2	Memory	DDR2 1GB
3	Harddisk	SATA 250 GB

4	Monitor	14 “
5	Mouse	Optic 1 buah
6	Keyboard	1 buah

4.1.2 Kebutuhan Perangkat Lunak

Spesifikasi perangkat lunak (*software*) yang digunakan dalam pembangunan *game* edukasi IPA dan Matematika untuk siswa SMP dapat dilihat pada tabel berikut :

Tabel 4. 2 Perangkat Lunak Yang Digunakan

NO	Perangkat Lunak	Keterangan
1	Windows 7	Sebagai sistem operasi
2	ActionScript	Sebagai bahasa pemrograman
3	Adobe Flash CS5	Sebagai program pembangun
4	Adobe Photoshop CS5	Untuk mengolah gambar
5	Microsoft Word	Untuk membuat laporan
6	Photoscape v3.5	Untuk mengolah gambar
7	Microsoft Visio 2007	Untuk Membuat Perancangan antarmuka
8	Adobe Soundboth CS5	Untuk Mengolah Suara

4.2 Pengujian

Pengujian sistem dilakukan bertujuan untuk menemukan kesalahan atau kekurangan pada perangkat lunak yang diuji. Pengujian bermaksud untuk mengetahui perangkat lunak yang dibuat sudah memenuhi kriteria yang sesuai dengan tujuan perancangan perangkat lunak tersebut.

Pengujian yang dilakukan yaitu pengujian *white box*, pengujian *alpha* dan pengujian beta. Pengujian *white-box* digunakan untuk menguji sistem. Pengujian *white-box* digunakan untuk meyakinkan semua perintah dan kondisi dieksekusi secara minimal. Pengujian alpha digunakan metode *black-box*. Pengujian beta digunakan untuk mengetahui tanggapan *user* terhadap aplikasi *game*, dengan melakukan kuisioner.

4.2.1 Pengujian *White-Box*

Pengujian *white-box* digunakan untuk meyakinkan semua perintah dan kondisi pada aplikasi dieksekusi secara minimal. Pengujian *white-box* menggunakan dua *tools* yaitu *flow graph* yang digunakan untuk menggambarkan alur dari algoritma dan *graph matrix* yang digunakan untuk menggenerasi *flow graph*. Adapun pengujian *white-box*nya adalah sebagai berikut:

1. Pengujian Algoritma *Greedy* untuk *game* biologi

a. *Flow Graph*

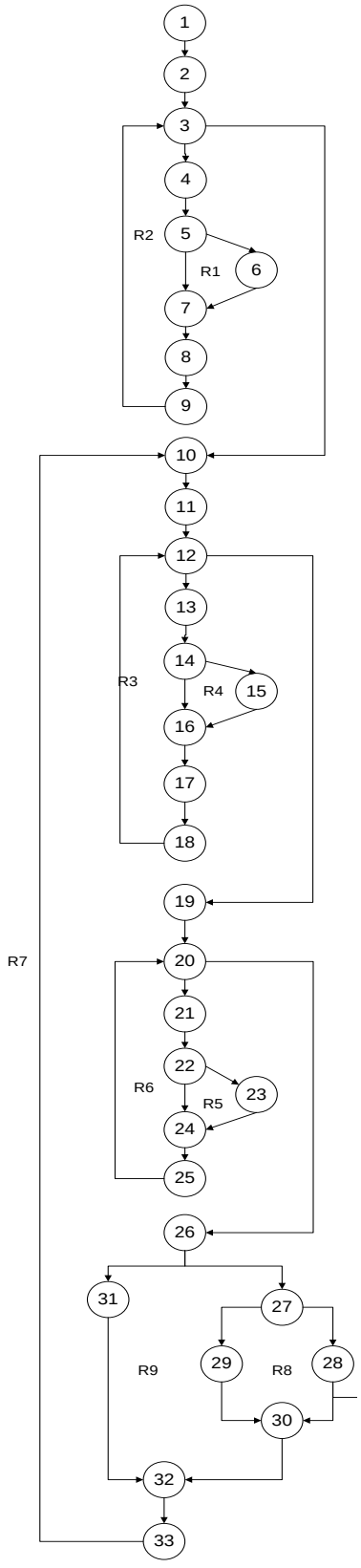
Berikut notasi *flow graph* algoritma *Greedy* yang digunakan :

```
1 : Buat variabel-variabel dan insialisasi
      variabel-variabel tersebut
      salah = 0
      i = 0
```

```
      j = 0
      panjang_karakter = jumlah huruf dari jawaban
      copy_jawaban =  jawaban
2 : Pilih huruf yang diambil dari kandidat
      (disimpan dalam variabel 'abjad')
3 : while j <= panjang_karakter
4 : ubah = copy(copy_jawaban, j, 1)
5 : if ubah != ' '
6 : replace (copy_jawaban, j, 1) = '*'
7 : endif
8 : j = j + 1
9 : endwhile
10 : while salah < 3
11 : k = 0
12 : while k <= panjang_karakter
13 : ada = 0
     bandingan = copy(jawaban, k, 1)
14 : if abjad = bandingan
15 : replace(copy_jawaban, k, 1) = abjad
       ada = ada + 1
16 : endif
17 : k = k + 1
18 : endwhile
19 : m = 0
20 : while m <= panjang_karakter
21 : bintang = copy(copy_jawaban, m, 1)
     b = 0
22 : if bintang = '*'
```

```
23 : b = b + 1
24 : endif
25 : endwhile
26 : if b != 0
27 : if ada != 0
28 : inputkan lagi nilai ke variabel abjad
29 : elseif ada = 0
       salah = salah + 1
       inputkan lagi nilai ke variabel abjad
30 : endif
31 : elseif b = 0
     salah = 5
32 : endif
33 : endwhile
```

Berikut ini adalah gambar *flow graph* algoritma *Greedy* yang digunakan*:*



Gambar 4. 1 *flow graph* algoritma *Greedy*

Keterangan :

 = Menggambarkan kondisi

 = Menggambarkan aksi

Dari Gambar 4.1 dapat dihitung *cyclomatic complexity* sebagai berikut :

Region = 9

V(G) = 40 *edge* - 33 *node* + 2 = 9

Keterangan :

V (G) = cyclomatic complexity

edge = panah

node = lingkaran

Jadi, *cyclomatic complexity* untuk gambar 4.1 adalah 9. Berdasarkan *cyclomatic complexity* tersebut, maka terdapat 9 *path* yan terdiri dari :

Path 1 : 1-2-3-10-11-12-19-20-26-31-32-33

Path 2 : 1-2-3-10-11-12-19-20-26-27-29-30-32-33-10- …

Path 3 : 1-2-3-10-11-12-19-20-26-27-28-30-32-33-10- …

Path 4 : 1-2-3-10-11-12-19-20-21-22-23-24-25-20- …

Path 5 : 1-2-3-10-11-12-19-20-21-22-24-25-20- …

Path 6 : 1-2-3-10-11-12-13-14-15-16-17-18-12- …

Path 7 : 1-2-3-10-11-12-13-14-16-17-18-12- …

Path 8 : 1-2-3-4-5-6-7-8-9-3- …

Path 9 : 1-2-3-4-5-7-8-9-3- …

Tabel 4. 3 Tabel kondisi *path* algoritma *greedy*

No	Path	Kondisi	Hasil Pengujian
1	*Path 1 :* 1-2-3-10-11-12-19-20-26-31-32-33	*Path* ini dijalankan dalam kondisi kandidat yang dipih *user* dapat membentuk solusi jawaban dan membentuk sebuah solusi yang diharapkan.	[√] Berhasil [ ] Tidak Berhasil
2	*Path 2:* 1-2-3-10-11-12-19-20-26-27-29-30-32-33-10- …	*Path* ini dijalankan ketika menebak jawaban dengan benar namun solusi belum tersusun secara sempurna atau lengkap.	[√] Berhasil [ ] Tidak Berhasil
3	*Path 3:* 1-2-3-10-11-12-19-20-26-27-28-30-32-33-10- …	*Path* ini dijalankan ketika menebak jawaban dengan salah, sehinggan menambahkan pesan kesalahan.	[√] Berhasil [ ] Tidak Berhasil
4	*Path 4:* 1-2-3-10-11-12-19-20-21-22-23-	*Path* ini dijalankan ketika algoritma melakukan perulangan untuk mengecek apakah kandidat membentuk solusi dengan keputusan logika benar dan kandidat	[√] Berhasil [ ] Tidak Berhasil

	24-25-20-…	ditampilkan atau dimasukkan kedalam solusi.	
5	*Path 5 :* 1-2-3-10-11-12-19-20-21-22-24-25-20- …	*Path* ini dijalankan ketika algoritma melakukan perulangan untuk mengecek apakah kandidat membentuk solusi dengan keputusan logika salah.	[√] Berhasil [ ] Tidak Berhasil
6	*Path 6 :* 1-2-3-10-11-12-13-14-15-16-17-18-12- …	*Path* ini dijalankan ketika algoritma melakukan perulangan untuk mengecek apakah kandidat membentuk solusi dengan keputusan logika benar.	[√] Berhasil [ ] Tidak Berhasil
7	*Path 7* : 1-2-3-10-11-12-13-14-16-17-18-12- …	*Path* ini dijalankan ketika algoritma melakukan perulangan untuk mengecek apakah kandidat membentuk solusi dengan keputusan logika benar.	[√] Berhasil [ ] Tidak Berhasil
8	*Path 8 :* 1-2-3-4-5-6-7-8-9-3-…	*Path* ini dijalankan ketika algoritma melakukan perulangan untuk mengubah jawaban dengan tanda ‘*’ dengan keputusan logika benar.	[√] Berhasil [ ] Tidak Berhasil
9	*Path 9 :* 1-2-3-4-5-7-	*Path* ini dijalankan ketika algoritma melakukan perulangan untuk mengubah	[√] Berhasil

	8-9-3- …	jawaban dengan tanda ‘*’ dengan keputusan logika salah.	[ ] Tidak Berhasil

b. Graph Matrix

Graph matrix adalah matrik empat persegi yang mempunyai ukuran yang sama dengan jumlah *node* pada *flowgraph*. Masing-masing baris dan kolom mempunyai hubungan dengan *node* yang telah ditentukan dan pemasukan data matrik berhubungan dengan hubungan (*edge*) antar *node.*

Berikut ini adalah *graph matrix* untuk algoritma *greedy*:

Berdasarkan hasil pengujian *white-box* untuk algoritma *greedy* yang digunakan pada *game* biologi dapat disimpulkan bahwa :

1. Menjamin seluruh *independent path* di dalam modul yang dikerjakan sekurang-kurangnya sekali.
2. Mengerjakan seluruh keputusan logika dari sisi benar maupun salah.
3. Mengerjakan seluruh perulangan sesuai batasannya dan dalam batasan operasinalnya.

2. Pengujian Algoritma *Collision detection* pada *game* matematika

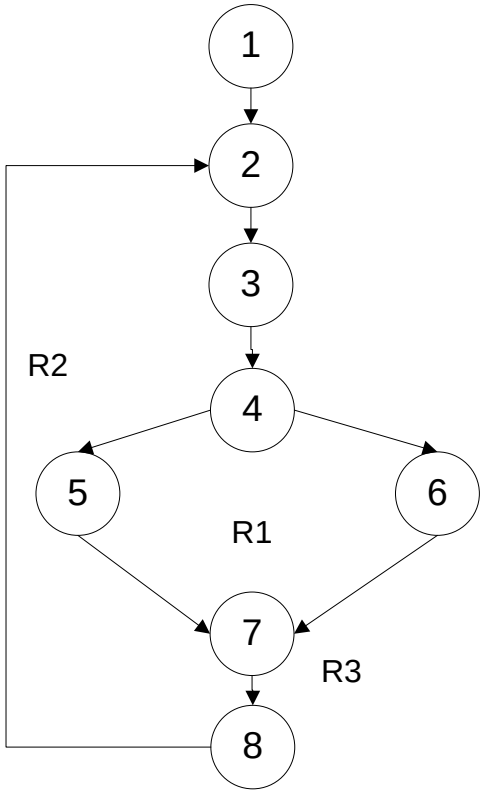
a. Flow Graph

Berikut notasi *flow graph* teknik *collision detection* yang digunakan :

```
1 : insialisasi variabel-variabel
      r (jari-jari)
      bx (titik x objek balon)
      by (titik y objek balon)
      gx (titik x objek garis)
      gy (titik y objek garis)
      naik = n
      jarak ( nilai awal jarak adalah by – gy)
      c (tumbukan) = gy + r
2 : while jarak > c
3 : jarak = by – n
4 : if jarak = c
5 : Terjadi tumbukan dan objek balon diganti dengan objek
      balon   meletus.
6 : else
       Tidak terjadi tumbukan
```

```
7 : endif
8 : endwhile
```

Berikut ini adalah *gambar flow graph* teknik *collision detection* yang digunakan:



Gambar 4. 2 *flow graph teknik collision detection*

Keterangan :

= Menggambarkan kondisi

= Menggambarkan aksi

Dari Gambar 4.2 dapat dihitung *cyclomatic complexity* sebagai berikut :

Region = 3

V(G) = 9 *edge* - 8 *node* + 2 = 3

Jadi, *cyclomatic complexity* untuk gambar 4.2 adalah 3

Berdasarkan *cyclomatic complexity* tersebut, maka terdapat 3 *path* yan terdiri dari :

Path 1 : 1-2-3-4-5-7-8

Path 2 : 1-2-3-4-6-7-8-2-…

Path 3 : 1-2-3-4-6-7-8

Tabel 4. 4 Tabel kondisi *path* teknik *collision detection*

No	Path	Kondisi	Hasil Pengujian
1	*Path 1 :* 1-2-3-4-5-7-8	Pada *path* 1, kondisi ini menunjukan algoritma melakukan perintah atau keputusan logika yang benar, dimana terjadi tumbukan antar objek yang terdapat pada *game* matematika.	[√] Berhasil [ ] Tidak Berhasil
2	*Path 2 :* 1-2-3-4-6-7-8-2-…	Pada *path* 2, terjadi ketika algoritma melakukan keputusan logika salah, sehingga melakukan perulangan, hingga kondisi perulangan berakhir yaitu variabel jarak lebih kecil daripada variabel c.	[√] Berhasil [ ] Tidak Berhasil
3	*Path 3 :* 1-2-3-4-6-7-8	Pada Path 3, terjadi ketika algoritma melakukan keputusan logika salah, dan tidak mengulangi lagi karena kondisi perulangan tidak terpenuhi.	[√] Berhasil [ ] Tidak Berhasil

b. Graph Matrix

Berikut ini adalah *graph matrix* teknik *collision detection* yang digunakan

Tabel 4. 5 Tabel *graph matrix* teknik *collision detection*

	1	2	3	4	5	6	7	8	
1		1							0
2			1						0
3				1					0
4					1	1			1
5							1		0
6							1	1	1
7								1	0
8		1							0
								SUM (E) + 1	3

Berdasarkan hasil pengujian *white-box* untuk metode *collision detection* yang digunakan pada *game* matematika dapat disimpulkan bahwa :

1. Menjamin seluruh *independent path* di dalam modul yang dikerjakan sekurang-kurangnya sekali.
2. Mengerjakan seluruh keputusan logika dari sisi benar maupun salah.
3. Mengerjakan seluruh perulangan sesuai batasannya dan dalam batasan operasinalnya.

4.2.2 Kesimpulan Pengujian *White-Box*

Setelah melakukan pengujian algortima *greddy* maupun teknik *collision detection* dengan *white-box* dapat disimpulkan bahawa dua algoritma yang disebutkan tadi :

4. Menjamin seluruh *independent path* di dalam modul yang dikerjakan sekurang-kurangnya sekali.
5. Mengerjakan seluruh keputusan logika dari sisi benar maupun salah.
6. Mengerjakan seluruh perulangan sesuai batasannya dan dalam batasan operasinalnya.
7. Mengerjakan seluruh struktur data internal yang menjamin validitas.

4.2.3 Pengujian *Alpha*

Pengujian *fungsional* yang digunakan untuk menguji sistem yang baru adalah metode pengujian *alpha*. Pengujian *alpha* dilakukan dengan menggunakan metode *black box.* Pengujian *black box* berfokus pada persyaratan fungsional perangkat lunak.

1. Rencana Pengujian

Rencana pengujian yang akan dilakukan dapat dilihat pada tabel berikut :

Tabel 4. 6 Tabel Pengujian Aplikasi *Game* edukasi IPA dan Matematika untuk siswa SMP

Pengujian Aplikasi *Game* edukasi IPA dan Matematika untuk siswa *SMP*			
NO	**Item Uji**	**Detail Pengujian**	**Jenis Pengujian**
1	Menu Utama	Memilih Tombol *Game*	*Black box*

		Matematika	
		Memilih Tombol *Game* Biologi	*Black box*
		Memilih Tombol *Game* Fisika	*Black box*
		Memilih Tombol Petunjuk	*Black box*
		Memilih Tombol Tentang Saya	*Black box*
		Memilih Tombol Perpustakaan	*Black box*
		Memilih Tombol Keluar	*Black box*
2	Tentang Saya	Memilih Tombol Kembali	*Black box*
3	Petunjuk	Memilih Tombol Petunjuk *Game* Matematika	*Black box*
		Memilih Tombol Petunjuk *Game* Biologi	*Black box*
		Memilih Tombol Petunjuk *Game Fisika*	*Black box*

		Memilih Tombol Keluar	*Black box*
4	Petunjuk *Game* Matematika	Memilih Tombol Kembali	*Black box*
5	Petunjuk *Game* Biologi	Memilih Tombol Kembali	*Black box*
6	Petunjuk *Game* Fisika	Memilih Tombol Kembali	*Black box*
7	Perpustakaan	Memilih Tombol Materi Matematika	*Black box*
		Memilih Tombol Materi Biologi	*Black box*
		Memilih Tombol Materi Fisika	*Black box*
8	Perpustakaan Matematika	Memilih Tombol Materi Bangun Datar	*Black box*
		Memilih Tombol Materi Aljabar	*Black box*
		Memilih Tombol Kembali	*Black box*
9	Perpustakaan Biologi	Memilih Tombol Materi Ekosistem	*Black box*

		Memilih Tombol Kembali	*Black box*
10	Perpustakaan Fisika	Memilih Tombol Materi Gerak Lurus	*Black box*
		Memilih Tombol Kembali	*Black box*
11	*Game* Matematika	Objek balon bergerak	*Black box*
		Memilih Tombol Main Lagi	*Black box*
		Memilih Tombol Lanjutkan	*Black box*
		Memilih Tombol Kembali	*Black box*
12	*Game* Biologi	Tombol Kandidat	*Black box*
		Memilih Tombol Main Lagi	*Black box*
		Memilih Tombol Kembali	*Black box*
13	*Game* Fisika	Tombol Benar	*Black box*
		Tombol Salah	*Black box*
		Memilih Tombol Main Lagi	*Black box*
		Memilih Tombol	*Black box*

		Keluar	
14	Keluar	Konfirmasi Keluar dari *game*	*Black box*

2. Kasus dan Hasil Pengujian

Berikut adalah kasus untuk menguji perangkat lunak yang sudah dibangun menggunakan metode *Black Box* berdasarkan Tabel Rencana pengujian yang telah dibuat.

Tabel 4. 7 Tabel Hasil Pengujian Aplikasi *Game* edukasi IPA dan Matematika untuk siswa SMP

Hasil Pengujian Aplikasi *Game* edukasi IPA dan Matematika untuk siswa *SMP*				
No	**Kasus / Diuji**	**Skenario Uji**	**Hasil yang diharapkan**	**Hasil Pengujian**
1.	Menu Utama	Memilih Tombol *Game* Matematika	Menampilkan *Game* Matematika	[√] Berhasil [ ] Tidak Berhasil
		Memilih Tombol *Game* Biologi	Menampilkan *Game* Biologi	[√] Berhasil [ ] Tidak Berhasil
		Memilih Tombol *Game* Fisika	Menampilkan *Game* Fisika	[√] Berhasil [ ] Tidak Berhasil

		Memilih Tombol Petunjuk	Menampilkan menu Petunjuk untuk mengetahui cara bermain	[√] Berhasil [ ] Tidak Berhasil
		Memilih Tombol Tentang Saya	Menampilkan informasi yang membuat *game*	[√] Berhasil [ ] Tidak Berhasil
		Memilih Tombol Perpustakaan	Menampilkan menu atau katalog perpustakaan	[√] Berhasil [ ] Tidak Berhasil
		Memilih Tombol Keluar	Menampilkan pesan konfirmasi apakah mau keluar dari *game* atau masih tetap bermain *game*	[√] Berhasil [ ] Tidak Berhasil
2	Tentang Saya	Memilih Tombol Kembali	Aplikasi akan keluar dari tampilan Tentang Saya dan kembali ke menu utama	[√] Berhasil [ ] Tidak Berhasil
3	Petunjuk	Memilih Tombol Petunjuk *Game*	Menampilkan Animasi Petunjuk	[√] Berhasil

		Matematika	*Game* matematika	[ ] Tidak Berhasil
		Memilih Tombol Petunjuk *Game* Biologi	Menampilkan Animasi Petunjuk *Game* Biologi	[√] Berhasil [ ] Tidak Berhasil
		Memilih Tombol Petunjuk *Game Fisika*	Menampilkan Animasi Petunjuk *Game* Fisika	[√] Berhasil [ ] Tidak Berhasil
		Memilih Tombol Keluar	Aplikasi akan keluar dari Petunjuk dan kembali ke menu utama	[√] Berhasil [ ] Tidak Berhasil
4	Petunjuk *Game* Matematika	Memilih Tombol Kembali	Aplikasi akan keluar dari Petunjuk *game* matematika dan kembali ke menu Petunjuk	[√] Berhasil [ ] Tidak Berhasil
5	Petunjuk *Game* Biologi	Memilih Tombol Kembali	Aplikasi akan keluar dari Petunjuk *game* biologi dan kembali ke menu	[√] Berhasil [ ] Tidak Berhasil

			Petunjuk	
6	Petunjuk *Game* Fisika	Memilih Tombol Kembali	Aplikasi akan keluar dari Petunjuk *game* fisika dan kembali ke menu Petunjuk	[√] Berhasil [ ] Tidak Berhasil
7	Perpustakaan	Memilih Tombol Materi Matematika	Menampilkan daftar materi matematika	[√] Berhasil [ ] Tidak Berhasil
		Memilih Tombol Materi Biologi	Menampilkan daftar materi Biologi	[√] Berhasil [ ] Tidak Berhasil
		Memilih Tombol Materi Fisika	Menampilkan daftar materi Fisika	[√] Berhasil [ ] Tidak Berhasil
8	Perpustakaan Matematika	Memilih Tombol Materi Bangun Datar	Menampilkan materi yang membahas tentang bangun datar	[√] Berhasil [ ] Tidak Berhasil
		Memilih Tombol Materi Aljabar	Menampilkan materi dan animasi yang membahas aljabar.	[√] Berhasil [ ] Tidak Berhasil

		Memilih Tombol Kembali	Aplikasi akan kembali ke menu perpustakaan	[√] Berhasil [ ] Tidak Berhasil
9	Perpustakaan Biologi	Memilih Tombol Materi Ekosistem	Menampilkan animasi yang membahas tentang ekosistem	[√] Berhasil [ ] Tidak Berhasil
		Memilih Tombol Kembali	Aplikasi akan kembali ke menu perpustakaan	[√] Berhasil [ ] Tidak Berhasil
10	Perpustakaan Fisika	Memilih Tombol Materi Gerak Lurus	Menampilkan animasi yang membahas tentang materi dari gerak lurus	[√] Berhasil [ ] Tidak Berhasil
		Memilih Tombol Kembali	Aplikasi akan kembali ke menu perpustakaan	[√] Berhasil [ ] Tidak Berhasil
11	*Game* Matematika	Objek balon bergerak	Jika balon yang dipilih benar maka nilai pemain akan bertambah 100, sedangkan jika	[√] Berhasil [ ] Tidak Berhasil

			salah makan nilai dikurangi 50, kecuali nilai masih nol, ketika salah menjawab nilainya akan tetap nol.	
		Memilih Tombol Main Lagi	Mengulang dan bermain *game* matematika dari level satu.	[√] Berhasil [ ] Tidak Berhasil
		Memilih Tombol Lanjutkan	Malanjutkan *game* matematika ke level yang selanjutnya	[√] Berhasil [ ] Tidak Berhasil
		Memilih Tombol Kembali	Aplikasi akan kembali dan menampilkan menu utama.	[√] Berhasil [ ] Tidak Berhasil
12	*Game* Biologi	Tombol Kandidat	Setiap tebakan kandidat yang membentuk solusi maka kandidat akan ditampilkan	[√] Berhasil [ ] Tidak Berhasil

			di baris solusi sedangkan tebakan kandidat yang salah tidak akan ditampilkan dan menampilkan pesan kesalahan	
		Memilih Tombol Main Lagi	Memulai *game* biologi dari pertanyaan atau level pertama	[√] Berhasil [ ] Tidak Berhasil
		Memilih Tombol Kembali	Keluar dari *game* biologi dan kembali ke menu utama	[√] Berhasil [ ] Tidak Berhasil
13	*Game* Fisika	Tombol Benar	Jika tombol di klik, aplikasi akan mengecek apakah pernyataan benar jika benar maka nilai akan bertambah 100 sedangkan jika	[√] Berhasil [ ] Tidak Berhasil

			salah nilai dikurangi 50	
		Tombol Salah	Jika tombol di klik, aplikasi akan mengecek apakah pernyataan salah jika pernyataan memang salah maka nilai akan bertambah 100 sedangkan jika pernyataan benar nilai dikurangi 50	[√] Berhasil [ ] Tidak Berhasil
		Memilih Tombol Main Lagi	Akan menampilkan pernyataan pertama dari *game* fisika.	[√] Berhasil [ ] Tidak Berhasil
		Memilih Tombol Keluar	Keluar dari *game* biologi dan kembali ke menu utama	[√] Berhasil [ ] Tidak Berhasil
14	Keluar	Konfirmasi	Akan keluar dari	[√] Berhasil

		Keluar dari *game*	*game* jika menekan tombol ya, dan akan kembalo ke menu utama jika menekan tombol tidak	[ ] Tidak Berhasil

4.2.4 Kesimpulan Pengujian *Alpha*

Berdasarkan hasil pengujian *alpha* (fungsional) dengan kasus uji diatas dapat ditarik kesimpulan bahwa dalam pembangunan aplikasi *game* edukasi IPA dan Matematika untuk siswa *SMP* tidak terdapat kesalahan proses dan secara fungsional mengeluarkan hasil yang sesuai dengan yang diharapkan.

4.2.5 Pengujian *Beta*

Pengujian *beta* merupakan pengujian yang dilakukan secara objektif dimana diuji secara langsung ke lapangan, dengan menggunakan kuesioner mengenai tanggapan *user* terhadap *game* yang telah dibangun. Adapun metode penilaian pengujian yang digunakan yaitu metode kuantitatif berdasarkan data sampel dari *user*.

Untuk mengetahui tanggapan dan penilaian pemain terhadap *game* ini. Telah disebarkan kuesioner kepada responden berdasarkan target *user* dan dilakukan perhitungan jawaban terbanyak.

Adapun berikut pertanyaan kuisioner yang dibagikan kepada *user* adalah sebagai berikut :

1. Apakah aplikasi *game* edukasi IPA dan Matematika untuk siswa *SMP* bisa dijadikan alternatif untuk dijadikan media edukasi baru untuk belajar mata pelajaran IPA ?
2. Apakah aplikasi *game* edukasi IPA dan Matematika untuk siswa SMP dapat menghilangkan rasa jenuh anda ketika belajar mata pelajaran IPA ?
3. Apakah aplikasi yang dibuat menjadikan anda tertarik terhadap mata pelajaran IPA?
4. Apakah aplikasi *game* edukasi IPA dan Matematika untuk siswa *SMP* dapat meningkatkan pengetahuan dan pemahaman anda terhadap mata pelajaran IPA?
5. Apakah anda setuju aplikasi *game* edukasi IPA dan Matematika untuk siswa SMP bermanfaat dan perlu dikembangkan lagi?

Tabel 4. 8 Jawaban Kuisioner

No Jawaban	Keterangan
A.	Ya
B.	Tidak

Tabel-tabel diatas adalah jawaban dari setiap pertanyaan kuisioner yang akan dibagikan kepada *user*.

4.2.6 Hasil Kuisioner

Data dari hasil pengisian kuisioner selanjutnya akan dihitung dan dilihat hasilnya. Berikut adalah hasil dari kuisioner:

Tabel 4. 9 Hasil Kuisioner

Kuisioner	Responder	Ya	Tidak
1	15	8	7
2	15	11	4

3	15	12	3
4	15	10	5
5	15	11	4

4.2.7 Kesimpulan Pengujian Beta

Dari pengujian *beta* yang telah dilakukan yaitu dengan pengujian pilihan jawaban dari kuisioner yang telah dibagikan di lapangan didapat kesimpulan bahwa program aplikasi yang telah dibuat dinilai dapat membantu meningkatkan dan menumbuhkan minat siswa terhadapa mata pelajaran IPA, serta dapat menambah wawasan siswa terhadap mata pelajaran IPA.