BAB III

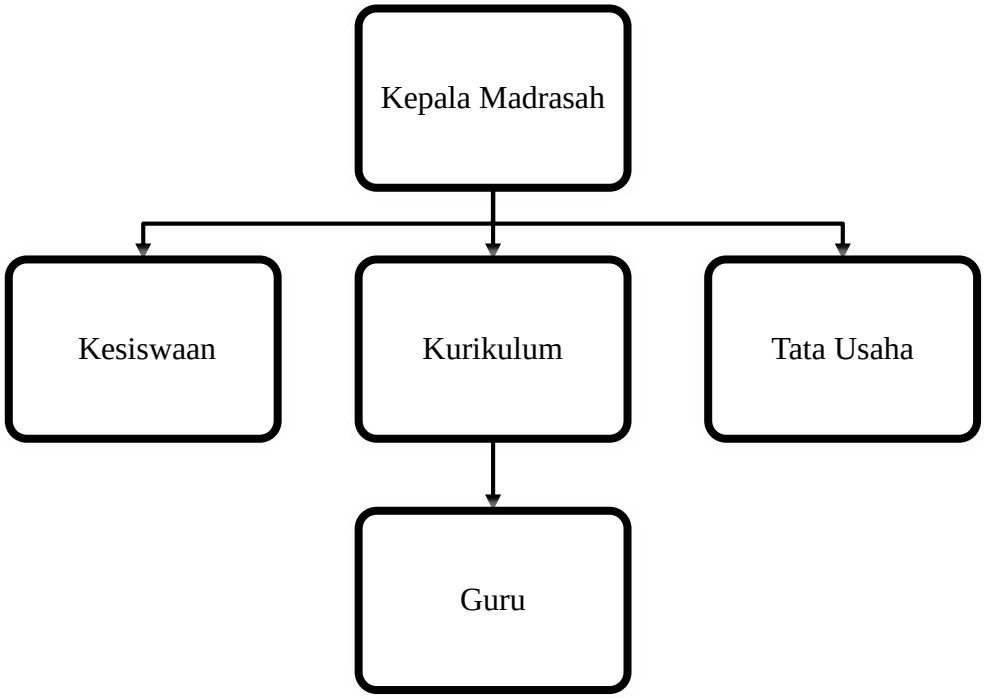
PERANCANGAN DAN PEMBAHASAN

3.1. Tinjauan Institusi/Perusahaan

3.1.1 Sejarah Institusi/Perusahaan

MTs Annida Al Islamy Bekasi terletak di Jl. Ir. KH. Mas Mansyur No. 91 Kel. Bekasi Jaya, Kec. Bekasi Timur, Kota Bekasi, Jawa Barat. Sekolah ini memiliki siswa kurang lebih 1690 siswa dengan 45 guru dan memiliki 30 kelas. Mata pelajaran yang di ajarkan sebanyak 57 pelajaran. sekolah ini juga terdapat 10 ekstrakurikuler.

3.1.2 Struktur Organisasi dan Fungsi



Sumber : MTs Annida Al Islamy Bekasi

Gambar III.1. Struktur Organisasi

Fungsi dari masing-masing jabatan :

1. Kepala Madrasah secara operasional melaksanakan pengelolaan kurikulum, peserta didik, ketenagaan, keuangan, sarana dan prasarana, hubungan sekolah-masyarakat dan ketatausahaan sekolah. Semua kegiatan-kegiatan operasional tersebut dilakukan melalui oleh seperangkat prosedur kerja yaitu perencanaan, pengorganisasian, penggerakan, dan pengawasan.
2. Kesiswaan berfungsi sebagai pengawas serta pengatur kegiatan, tata tertib dan perilaku siswa di sekolah.
3. Kurikulum berfungsi sebagai pedoman dalam pelaksanaan kegiatan belajar mengajar, memberi rambu-rambu yang dapat mengarahkan semua pihak untuk terlibat dalam pelaksanaan kurikulum MTs sesuai dengan pembagian tugas dan meningkatkan kualitas KBM (kegiatan belajar mengajar) dalam mencapai kompetensi siswa.
4. Tata Usaha berfungsi sebagai aktivitas administrasi adalah suatu kegiatan untuk mengadakan pencatatan dan penyusunan keterangan-keterangan dengan secara efektif dan efisien dengan menggunakan sarana dan prasarana sehingga keterangan-keterangan itu dapat dipergunakan secara langsung sebagai bahan informasi baik bagi pimpinan organisasi yang bersangkutan ataupun dapat dipergunakan oleh pihak luar organisasi yang membutuhkan.

3.2. Analisa Kebutuhan Software

Berdasarkan dari wawancara atau tanya jawab yang dilakukan dengan narasumber yaitu siswa kelas VII MTs Annida Al Islamy Bekasi. Selama ini belum ada media game yang menerangkan tentang bahaya rokok untuk kesehatan.

menurut mereka banyak sekali peringatan tentang bahaya rokok untuk kesehatan, mulai dari gambar di bungkus rokok, iklan di sosial media, media cetak dan serta ucapan lisan oleh guru tentang bahaya rokok untuk kesehatan namun tidak pernah ada penjelasan berupa bermain game.

Rata-rata para siswa di sekolah ini sudah memiliki dan terbiasa menggunakan smartphone yang di dalamnya sudah terinstal berbagai aplikasi termasuk beberapa aplikasi permainan favorit mereka. kebanyakan dari mereka memiliki smartphone dengan sistem operasi android, hanya beberapa saja yang menggunakan Iphone dan beberapa jenis smartphone lainnya. Diharapkan dengan adanya game interaktif ini bisa menerangkan tentang bahaya rokok untuk kesehatan dengan penuh harapan para siswa MTs Annida Al Islamy Bekasi tidak mau mencoba-coba menghisap rokok setelah mendapat penjelasan yang berada didalam game ini.

Sedangkan untuk memenuhi kebutuhan yang dijelaskan diatas, penulis membutuhkan hardware dan software untuk merancang game interaktif tersebut. Dibantu dengan materi dari berbagai sumber di internet dan penjelasan dalam wawancara dengan narasumber hingga terbentuk sebuah rancangan aplikasi animasi yang ingin dibuat. Dengan software *Construct Ver 2* semua kebutuhan seperti menyajikan kumpulan gambar yang tersusun sehingga menjadi sebuah game, kumpulan suara yang dibutuhkan dan visualisasi yang menarik bisa dibuat dengan software ini. Sedangkan penggunaan hardware dalam pembuatan game ini cukup dengan spesifikasi standar untuk memproses suatu tampilan grafik dengan baik.

3.3. Desain

3.3.1. Karakteristik *Software*

Dalam merancang sebuah sistem dalam hal ini adalah animasi pembelajaran harus berpedoman kepada karakteristik dan unsur aplikasi yaitu :

1. *Format*

   Game Interaktif ini terdiri dari 2 elemen yaitu permainan dan profil. Pada menu permainan pengguna akan disajikan gambar anak berseragam MTs dengan papan selancar. Pengguna wajib mendapatkan Simbol Dilarang Merokok yang terdapat pada papan bergerak dengan waktu yang telah ditentukan sebelumnya sesuai dengan levelnya. pada game interaktif ini terdapat 7 level, kesulitan di setiap levelnya terus bertambah. navigasi bagi pengguna cukup menyentuh objek anak MTs hingga objek mampu melewati rintangan dan mendapatkan Simbol Dilarang Merokok. pengguna akan naik level ketika semua Simbol Dilarang Merokok yang ada dalam level tersebut berhasil diraih oleh pengguna. Pada menu profil akan tersaji informasi pembuat game interaktif ini.

2. *Rules*

   Pada game interaktif ini, pengguna wajib mendapatkan semua Simbol Dilarang Merokok yang ada disetiap levelnya dengan batas waktu yang telah ditentukan oleh pembuat. Setelah menyelesaikan level, pengguna akan disajikan informasi tentang bahaya rokok untuk kesehatan bagi kesehatan dengan tombol lanjutkan yang berada dibagian bawah informasi tersebut.

3. *Policy*

   Ketika pengguna gagal menyelesaikan sampai level terakhir, maka pengguna wajib mengulang game tersebut ke level 1 lagi.

4. *Scenario*

   Pertama pengguna akan disajikan tampilan awal berupa tombol mulai yang berfungsi memulai game di level 1. Pengguna wajib menyelesaikan level dengan meraih semua Simbol Dilarang Merokok yang ada di dalam level 1 dengan batas waktu 15 detik pengguna wajib menyelesaikan level ini. Jika waktu habis pengguna harus mengulangnya kembali dari awal, jika pengguna berhasil menyelesaikan level 1, game interaktif ini akan menyajikan informasi tentang bahaya rokok untuk kesehatan dan tombol level selanjutnya hingga level 7.

5. *Events/Challenge*

   Pada game ini navigasi yang disajikan berupa tombol sentuh yang akan digunakan oleh user untuk mengarahkan objek yang loncat ke kiri dan ke kanan. pada bagian tertentu ada objek berupa Simbol Dilarang Merokok yang wajib diraih oleh pengguna dengan mengarahkan objek anak MTs tersebut.

6. *Roles*

   Pengguna wajib menyelesaikan level per level agar mendapat informasi seputar bahaya rokok untuk kesehatan.

7. *Decisions*

   Keputusan yang dibuat oleh pengguna hanya bertumpu pada navigasi yang tersedia untuk meraih Simbol Dilarang Merokok untuk kemudian menyelesaikan level per level.

8. *Level*

   Di dalam game interaktif ini terdapat 7 level yang harus di selesaikan dengan baik oleh pengguna.

9. *Score Model*

   Di dalam game interaktif ini setiap Simbol Dilarang Merokok yang didapat otomatis pengguna akan mendapatkan nilai 1, disetiap level memiliki jumlah Simbol Dilarang Merokok yang berbeda-beda tergantung dari levelnya.

10. *Indicators*

    Indikator yang digunakan adalah berupa objek berupa gambar Simbol Dilarang Merokok.

11. *Symbol*

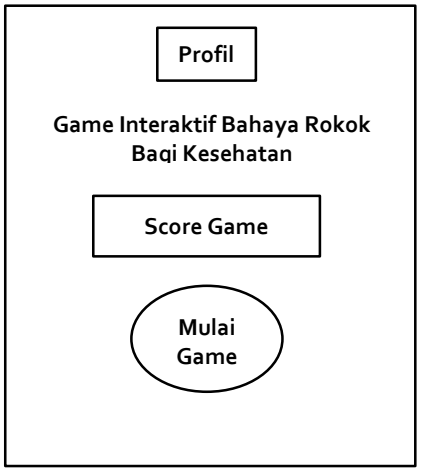
    Terdapat tombol panah sebagai penunjuk ke tampilan selanjutnya atau sebelumnya.Selain itu juga ada tombol menu yang digunakan untuk menuju ke menu-menu yang disediakan.

3.3.2 Perancangan *Story Board*

A. Storyboard Menu utama

   Berikut ini adalah gambaran dari storyboard menu utama seperti yang dijelaskan pada tabel di bawah ini :

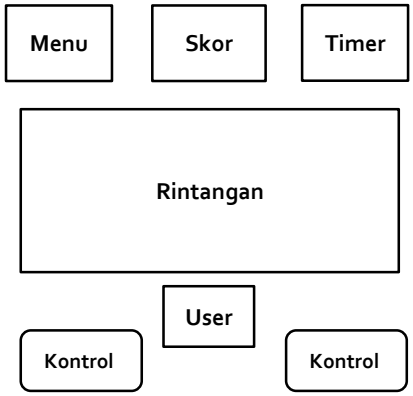
Tabel III.1. Storyboard Menu Home

VISUAL	SKETSA	AUDIO
Ketika game pertama kali dijalankan, akan langsung tampil sebuah tampilan untuk memulai game dan profil pembuat.		Music : Click.wav

B. Storyboard Menu Permainan Level 1-7

Berikut ini adalah gambaran dari storyboard menu materi seperti yang dijelaskan pada tabel di bawah ini :

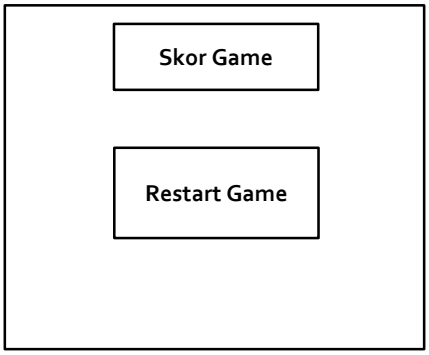
Tabel III.2. Storyboard Game Play

VISUAL	SKETSA	AUDIO
Pengguna diwajibkan meraih gambar Simbol Dilarang Merokok yang berada pada layar untuk menyelesaikan tiap level nya.		Music: Game_play.wav

C. Storyboard Game Over

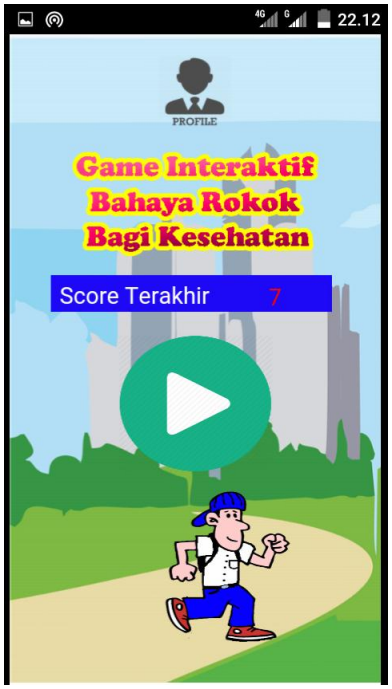
Berikut ini adalah gambaran jika pengguna gagal menyelesaikan game karena timer habis atau terjatuh seperti yang dijelaskan pada tabel di bawah ini :

Tabel III.3. Storyboard Game Over

VISUAL	SKETSA	AUDIO
Jika pengguna permainan gagal mendapatkan semua symbol Simbol Dilarang Merokok dengan batas waktu atau pemain terjatuh maka game akan selesai dan kembali lagi ke awal.		Music: Game Over.wav

3.3.3. *User Interface*

A. Tampilan *Home*



Gambar III.2. Tampilan *Home*

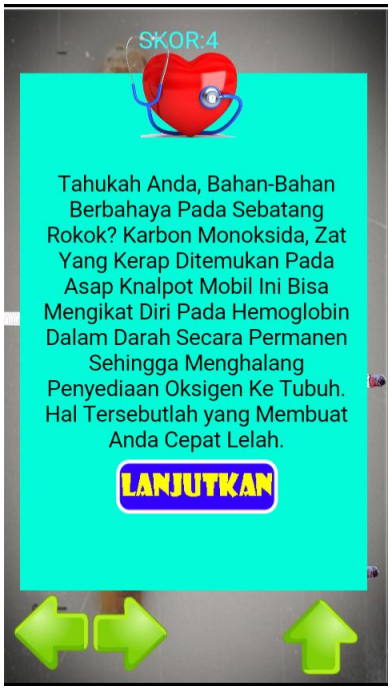
B. Tampilan *Menu Profil*

Gambar III.3. Tampilan *Menu Profil*

C. Tampilan *Game Play*

Gambar III.4. Tampilan *Game Play*

D. Tampilan *Info Kesehatan*



Gambar III.5. Tampilan *Info Kesehatan*

E. Tampilan *Game Over*

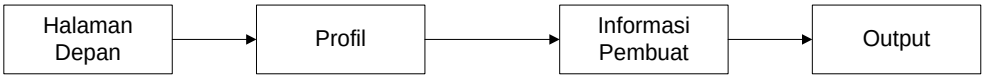
Gambar III.6. Tampilan *Game Over*

3.3.4. *State Transition* Diagram

A. *Scene* Menu Home

Gambar III.7. *Scene* Menu *Home*

B. *Scene* Menu Profil



Gambar III.8. S*cene Menu Profil*

C. *Scene* Game Play

Gambar III.9. *Scene* Game Play

D. *Scene* Game Over

Gambar III.10. *Scene* Game Over

3.4. Code generation

3.4.1.Testing

Aplikasi yang telah dibuat, selanjutnya diuji melalui teknik pengujian perangkat lunak yang meliputi pengujian *white box* dan *black box.*

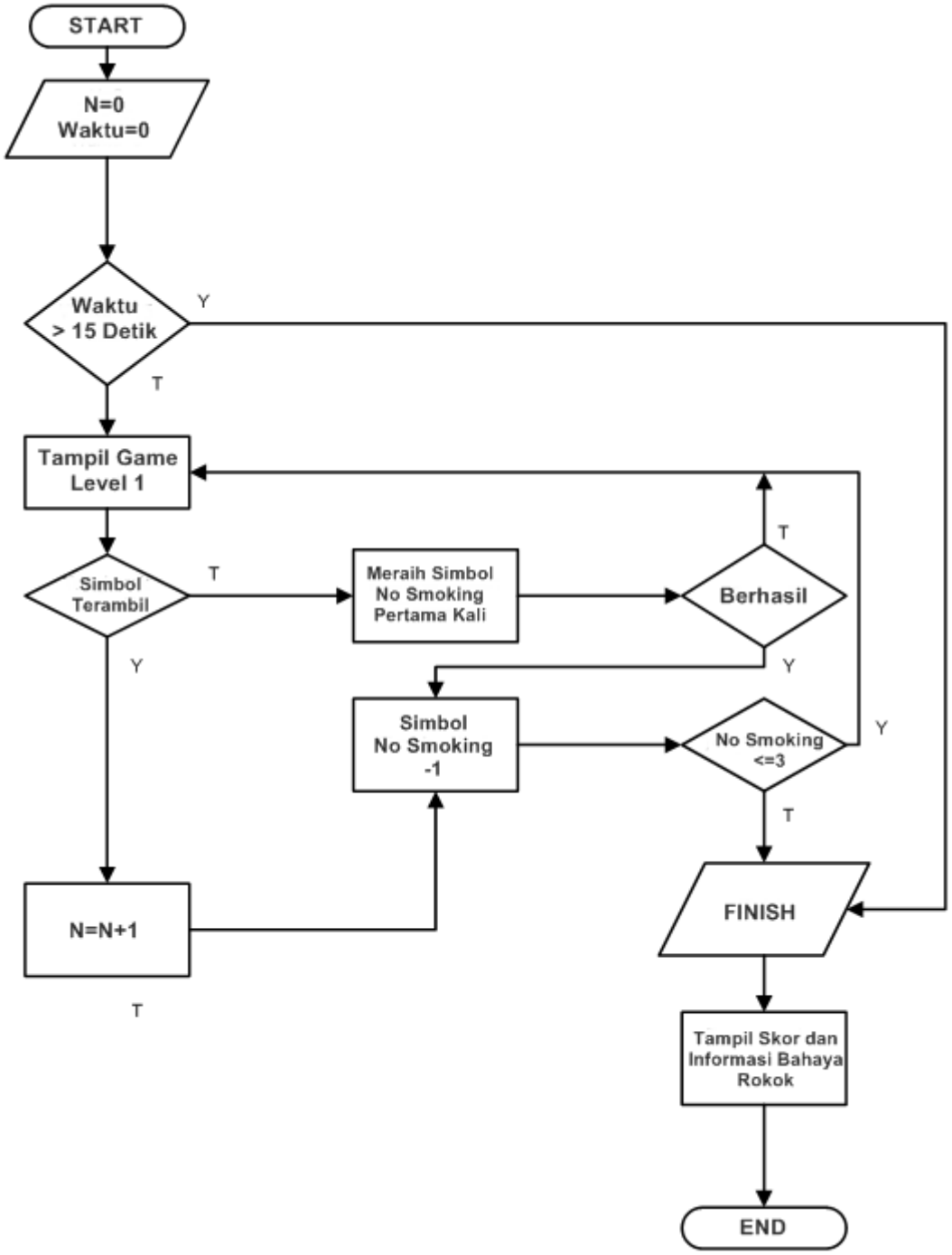
A. Pengujian White Box

Metode *white box* yaitu menguji perangkat lunak dari segi desain dan kode program apakah mampu menghasilkan fungsi-fungsi, masukan, dan keluaran yang sesuai dengan spesifikasi kebutuhan. Pengujian kotak putih dilakukan dengan memeriksa logika dari kode program, pembuatan kasus uji bisa mengikuti standar pengujian dari standar pemrograman seharusnya.

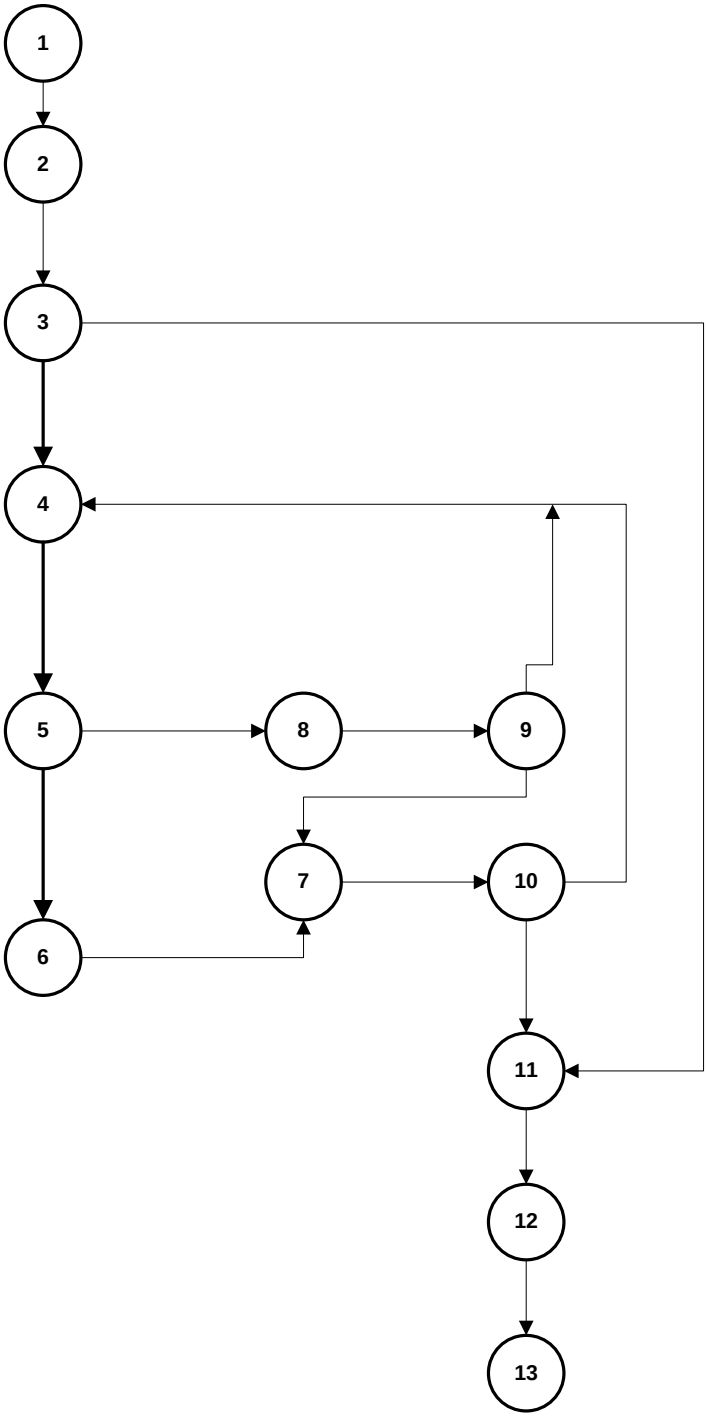
Tidak semua hal pengujian dilakukan terhadap keseluruhan program secara utuh, adapun yang akan dibahas adalah pengujian terhadap level 1.

Secara garis besar algoritma dari evaluasi sebagai berikut :

1. Pengguna harus mengambil simbol Simbol Dilarang Merokok dalam waktu 15 detik.
2. Setiap simbol Simbol Dilarang Merokok yang diambil akan mendapatkan skor 1, jika belum maka pengguna tidak dapat melanjutkan ke level selanjutnya.
3. Jika waktu habis, maka permainan akan selesai dan pengguna akan mengulang kembali ke level 1.
4. Pemain dapat melanjutkan ke *level* berikutnya jika dapat mendapatkan semua symbol Simbol Dilarang Merokok.



Gambar III.11. Bagan Alir Game Level 1



Gambar III.12. Grafik Alir Game Level 1

1

```
        if (a1 === a2)
                return 0;
        var s1 = Math.sin(a1);
        var c1 = Math.cos(a1);
        var s2 = Math.sin(a2);
        var c2 = Math.cos(a2);
        var n = s1 * s2 + c1 * c2;
        if (n >= 1)
                return 0;
        if (n <= -1)
                return cr.PI;
        return Math.acos(n);
};
cr.angleRotate = function (start, end, step)
{
        var ss = Math.sin(start);
        var cs = Math.cos(start);
        var se = Math.sin(end);
        var ce = Math.cos(end);
        if (Math.acos(ss * se + cs * ce) > step)
        {
                if (cs * se - ss * ce > 0)
                        return cr.clamp_angle(start + step);
                else
                        return cr.clamp_angle(start - step);
        }
        else
```

```
                return cr.clamp_angle(end);
};
cr.angleClockwise = function (a1, a2)
{
        var s1 = Math.sin(a1);
        var c1 = Math.cos(a1);
        var s2 = Math.sin(a2);
        var c2 = Math.cos(a2);
        return c1 * s2 - s1 * c2 <= 0;
};
cr.rotatePtAround = function (px, py, a, ox, oy, getx)
{
        if (a === 0)
                return getx ? px : py;
        var sin_a = Math.sin(a);
        var cos_a = Math.cos(a);
        px -= ox;
        py -= oy;
        var left_sin_a = px * sin_a;
        var top_sin_a = py * sin_a;
        var left_cos_a = px * cos_a;
        var top_cos_a = py * cos_a;
        px = left_cos_a - top_sin_a;
        py = top_cos_a + left_sin_a;
        px += ox;
        py += oy;
        return getx ? px : py;
```

```
    }
    cr.distanceTo = function(x1, y1, x2, y2)
    {
            var dx = x2 - x1;
var dy = y2 - y1;
            return Math.sqrt(dx*dx + dy*dy);
    };
    cr.xor = function (x, y)
    {
            return !x !== !y;
    };
    cr.lerp = function (a, b, x)
    {
            return a + (b - a) * x;
    };
    cr.unlerp = function (a, b, c)
    {
            if (a === b)
                    return 0;                       // avoid divide by 0
            return (c - a) / (b - a);
    };
    cr.anglelerp = function (a, b, x)
    {
            var diff = cr.angleDiff(a, b);
            if (cr.angleClockwise(b, a))
            {
                    return a + diff * x;
```

```
		}
		else
		{
			return a - diff * x;
		}
};
cr.qarp = function (a, b, c, x)
{
		return cr.lerp(cr.lerp(a, b, x), cr.lerp(b, c, x), x);
};
cr.cubic = function (a, b, c, d, x)
{
		return cr.lerp(cr.qarp(a, b, c, x), cr.qarp(b, c, d, x), x);
};
cr.cosp = function (a, b, x)
{
		return (a + b + (a - b) * Math.cos(x * Math.PI)) / 2;
};
cr.hasAnyOwnProperty = function (o)
{
		var p;
		for (p in o)
		{
			if (o.hasOwnProperty(p))
				return true;
		}
		return false;
```

```
};
cr.wipe = function (obj)
{
	var p;
	for (p in obj)
	{
		if (obj.hasOwnProperty(p))
			delete obj[p];
	}
};
var startup_time = +(new Date());
cr.performance_now = function()
{
	if (typeof window["performance"] !== "undefined")
	{
		var winperf = window["performance"];
		if (typeof winperf.now !== "undefined")
			return winperf.now();
		else if (typeof winperf["webkitNow"] !== "undefined")
			return winperf["webkitNow"]();
		else if (typeof winperf["mozNow"] !== "undefined")
			return winperf["mozNow"]();
		else if (typeof winperf["msNow"] !== "undefined")
			return winperf["msNow"]();
	}
	return Date.now() - startup_time;
};
```

```
	var isChrome = false;
	var isSafari = false;
	var isiOS = false;
	var isEjecta = false;
	if (typeof window !== "undefined")		// not c2 editor
	{
		isChrome = /chrome/i.test(navigator.userAgent) || /chromium/i.test(navigator.userAgent);
		isSafari = !isChrome && /safari/i.test(navigator.userAgent);
		isiOS = /(iphone|ipod|ipad)/i.test(navigator.userAgent);
		isEjecta = window["c2ejecta"];
	}
	var supports_set = ((!isSafari && !isEjecta && !isiOS) && (typeof Set !== "undefined" && typeof Set.prototype["forEach"] !== "undefined"));
	function ObjectSet_()
	{
		this.s = null;
		this.items = null;				// lazy allocated (hopefully results in better GC performance)
		this.item_count = 0;
		if (supports_set)
		{
			this.s = new Set();
		}
		this.values_cache = [];
		this.cache_valid = true;
		cr.seal(this);
	};
	ObjectSet_.prototype.contains = function (x)
```

```
{
        if (this.isEmpty())
                return false;
        if (supports_set)
                return this.s["has"](x);
        else
                return (this.items && this.items.hasOwnProperty(x));
};
ObjectSet_.prototype.add = function (x)
{
        if (supports_set)
        {
                if (!this.s["has"](x))
                {
                        this.s["add"](x);
                        this.cache_valid = false;
                }
        }
        else
        {
                var str = x.toString();
                var items = this.items;
                if (!items)
                {
                        this.items = {};
                        this.items[str] = x;
                        this.item_count = 1;
```

2

```
                this.cache_valid = false;
            }
            else if (!items.hasOwnProperty(str))
            {
                items[str] = x;
                this.item_count++;
                this.cache_valid = false;
            }
        }
};
ObjectSet_.prototype.remove = function (x)
{
        if (this.isEmpty())
            return;
        if (supports_set)
        {
            if (this.s["has"](x))
            {
                this.s["delete"](x);
                this.cache_valid = false;
            }
        }
        else if (this.items)
        {
            var str = x.toString();
            var items = this.items;
            if (items.hasOwnProperty(str))
```

```
                {
                        delete items[str];
                        this.item_count--;
                        this.cache_valid = false;
                }
        }
};
ObjectSet_.prototype.clear = function (/*wipe_*/)
{
        if (this.isEmpty())
                return;
        if (supports_set)
        {
                this.s["clear"]();                      // best!
        }
        else
        {
                this.items = null;                      // creates garbage; will
lazy allocate on next add()
                this.item_count = 0;
        }
        cr.clearArray(this.values_cache);
        this.cache_valid = true;
};
ObjectSet_.prototype.isEmpty = function ()
{
        return this.count() === 0;
};
```

```
ObjectSet_.prototype.count = function ()
{
	if (supports_set)
		return this.s["size"];
	else
		return this.item_count;
};
var current_arr = null;
var current_index = 0;
function set_append_to_arr(x)
{
	current_arr[current_index++] = x;
};
ObjectSet_.prototype.update_cache = function ()
{
	if (this.cache_valid)
		return;
	if (supports_set)
	{
		cr.clearArray(this.values_cache);
		current_arr = this.values_cache;
		current_index = 0;
		this.s["forEach"](set_append_to_arr);
;
		current_arr = null;
		current_index = 0;
	}
```

5

```
            else
            {
                var values_cache = this.values_cache;
                cr.clearArray(values_cache);
                var p, n = 0, items = this.items;
                if (items)
                {
                    for (p in items)
                    {
                        if (items.hasOwnProperty(p))
                            values_cache[n++] = items[p];
                    }
                }
;
            }
            this.cache_valid = true;
        };
        ObjectSet_.prototype.valuesRef = function ()
        {
            this.update_cache();
            return this.values_cache;
        };
        cr.ObjectSet = ObjectSet_;
        var tmpSet = new cr.ObjectSet();
        cr.removeArrayDuplicates = function (arr)
        {
            var i, len;
```

6

```
            for (i = 0, len = arr.length; i < len; ++i)
            {
                tmpSet.add(arr[i]);
            }
            cr.shallowAssignArray(arr, tmpSet.valuesRef());
            tmpSet.clear();
        };
        cr.arrayRemoveAllFromObjectSet = function (arr, remset)
        {
            if (supports_set)
                cr.arrayRemoveAll_set(arr, remset.s);
            else
                cr.arrayRemoveAll_arr(arr, remset.valuesRef());
        };
        cr.arrayRemoveAll_set = function (arr, s)
        {
            var i, j, len, item;
            for (i = 0, j = 0, len = arr.length; i < len; ++i)
            {
                item = arr[i];
                if (!s["has"](item))                    // not an item to remove
                    arr[j++] = item;                    // keep it
            }
            cr.truncateArray(arr, j);
        };
        cr.arrayRemoveAll_arr = function (arr, rem)
```

```
        {
                var i, j, len, item;
                for (i = 0, j = 0, len = arr.length; i < len; ++i)
                {
                        item = arr[i];
                        if (cr.fastIndexOf(rem, item) === -1)      // not an item to remove
                                arr[j++] = item;                                        // keep
it
                }
                cr.truncateArray(arr, j);
        };
        function KahanAdder_()
        {
                this.c = 0;
    this.y = 0;
    this.t = 0;
    this.sum = 0;
                cr.seal(this);
        };
        KahanAdder_.prototype.add = function (v)
        {
                this.y = v - this.c;
           this.t = this.sum + this.y;
           this.c = (this.t - this.sum) - this.y;
           this.sum = this.t;
        };
   KahanAdder_.prototype.reset = function ()
   {
```

```
        this.c = 0;
        this.y = 0;
        this.t = 0;
        this.sum = 0;
    };
        cr.KahanAdder = KahanAdder_;
        cr.regexp_escape = function(text)
        {
                return text.replace(/[-[\]{}()*+?.,\\^$|#\s]/g, "\\$&");
        };
        function CollisionPoly_(pts_array_)
        {
                this.pts_cache = [];
                this.bboxLeft = 0;
                this.bboxTop = 0;
                this.bboxRight = 0;
                this.bboxBottom = 0;
                this.convexpolys = null;            // for physics behavior to cache
separated polys
                this.set_pts(pts_array_);
                cr.seal(this);
        };
        CollisionPoly_.prototype.set_pts = function(pts_array_)
        {
                this.pts_array = pts_array_;
                this.pts_count = pts_array_.length / 2;                    // x, y, x, y... in
array
                this.pts_cache.length = pts_array_.length;
```

7

```
        this.cache_width = -1;
        this.cache_height = -1;
        this.cache_angle = 0;
};
CollisionPoly_.prototype.is_empty = function()
{
        return !this.pts_array.length;
};
CollisionPoly_.prototype.update_bbox = function ()
{
        var myptscache = this.pts_cache;
        var bboxLeft_ = myptscache[0];
        var bboxRight_ = bboxLeft_;
        var bboxTop_ = myptscache[1];
        var bboxBottom_ = bboxTop_;
        var x, y, i = 1, i2, len = this.pts_count;
        for ( ; i < len; ++i)
        {
                i2 = i*2;
                x = myptscache[i2];
                y = myptscache[i2+1];
                if (x < bboxLeft_)
                        bboxLeft_ = x;
                if (x > bboxRight_)
                        bboxRight_ = x;
                if (y < bboxTop_)
                        bboxTop_ = y;
```

```
                    if (y > bboxBottom_)
                        bboxBottom_ = y;
            }
            this.bboxLeft = bboxLeft_;
            this.bboxRight = bboxRight_;
            this.bboxTop = bboxTop_;
            this.bboxBottom = bboxBottom_;
};
CollisionPoly_.prototype.set_from_rect = function(rc, offx, offy)
{
            this.pts_cache.length = 8;
            this.pts_count = 4;
            var myptscache = this.pts_cache;
            myptscache[0] = rc.left - offx;
            myptscache[1] = rc.top - offy;
            myptscache[2] = rc.right - offx;
            myptscache[3] = rc.top - offy;
            myptscache[4] = rc.right - offx;
            myptscache[5] = rc.bottom - offy;
            myptscache[6] = rc.left - offx;
            myptscache[7] = rc.bottom - offy;
            this.cache_width = rc.right - rc.left;
            this.cache_height = rc.bottom - rc.top;
            this.update_bbox();
};
CollisionPoly_.prototype.set_from_quad = function(q, offx, offy, w, h)
{
```

```
            this.pts_cache.length = 8;
            this.pts_count = 4;
            var myptscache = this.pts_cache;
            myptscache[0] = q.tlx - offx;
            myptscache[1] = q.tly - offy;
            myptscache[2] = q.trx - offx;
            myptscache[3] = q.try_ - offy;
            myptscache[4] = q.brx - offx;
            myptscache[5] = q.bry - offy;
            myptscache[6] = q.blx - offx;
            myptscache[7] = q.bly - offy;
            this.cache_width = w;
            this.cache_height = h;
            this.update_bbox();
        };
        CollisionPoly_.prototype.set_from_poly = function (r)
        {
            this.pts_count = r.pts_count;
            cr.shallowAssignArray(this.pts_cache, r.pts_cache);
            this.bboxLeft = r.bboxLeft;
            this.bboxTop - r.bboxTop;
            this.bboxRight = r.bboxRight;
            this.bboxBottom = r.bboxBottom;
        };
        CollisionPoly_.prototype.cache_poly = function(w, h, a)
        {
            if (this.cache_width === w && this.cache_height === h &&
this.cache_angle === a)
```

```
            return;             // cache up-to-date
        this.cache_width = w;
        this.cache_height = h;
        this.cache_angle = a;
        var i, i2, i21, len, x, y;
        var sina = 0;
        var cosa = 1;
        var myptsarray = this.pts_array;
        var myptscache = this.pts_cache;
        if (a !== 0)
        {
            sina = Math.sin(a);
            cosa = Math.cos(a);
        }
        for (i = 0, len = this.pts_count; i < len; i++)
        {
            i2 = i*2;
            i21 = i2+1;
            x = myptsarray[i2] * w;
            y = myptsarray[i21] * h;
            myptscache[i2] = (x * cosa) - (y * sina);
            myptscache[i21] = (y * cosa) + (x * sina);
        }
        this.update_bbox();
};
CollisionPoly_.prototype.contains_pt = function (a2x, a2y)
{
```

```
        var myptscache = this.pts_cache;
        if (a2x === myptscache[0] && a2y === myptscache[1])
                return true;
        var i, i2, imod, len = this.pts_count;
        var a1x = this.bboxLeft - 110;
        var a1y = this.bboxTop - 101;
        var a3x = this.bboxRight + 131
        var a3y = this.bboxBottom + 120;
        var b1x, b1y, b2x, b2y;
        var count1 = 0, count2 = 0;
        for (i = 0; i < len; i++)
        {
                i2 = i*2;
                imod = ((i+1)%len)*2;
                b1x = myptscache[i2];
                b1y = myptscache[i2+1];
                b2x = myptscache[imod];
                b2y = myptscache[imod+1];
                if (cr.segments_intersect(a1x, a1y, a2x, a2y, b1x, b1y, b2x, b2y))
                        count1++;
                if (cr.segments_intersect(a3x, a3y, a2x, a2y, b1x, b1y, b2x, b2y))
                        count2++;
        }
        return (count1 % 2 === 1) || (count2 % 2 === 1);
};
CollisionPoly_.prototype.intersects_poly = function (rhs, offx, offy)
{
```

9

```
var rhspts = rhs.pts_cache;
var mypts = this.pts_cache;
if (this.contains_pt(rhspts[0] + offx, rhspts[1] + offy))
        return true;
if (rhs.contains_pt(mypts[0] - offx, mypts[1] - offy))
        return true;
var i, i2, imod, leni, j, j2, jmod, lenj;
var a1x, a1y, a2x, a2y, b1x, b1y, b2x, b2y;
for (i = 0, leni = this.pts_count; i < leni; i++)
{
        i2 = i*2;
        imod = ((i+1)%leni)*2;
        a1x = mypts[i2];
        a1y = mypts[i2+1];
        a2x = mypts[imod];
        a2y = mypts[imod+1];
        for (j = 0, lenj = rhs.pts_count; j < lenj; j++)
        {
                j2 = j*2;
                jmod = ((j+1)%lenj)*2;
                b1x = rhspts[j2] + offx;
                b1y = rhspts[j2+1] + offy;
                b2x = rhspts[jmod] + offx;
                b2y = rhspts[jmod+1] + offy;
                if (cr.segments_intersect(a1x, a1y, a2x, a2y, b1x, b1y,
b2x, b2y))
                        return true;
        }
```

```
        }
        return false;
    };
    CollisionPoly_.prototype.intersects_segment = function (offx, offy, x1, y1, x2,
y2)
    {
        var mypts = this.pts_cache;
        if (this.contains_pt(x1 - offx, y1 - offy))
            return true;
        var i, leni, i2, imod;
        var a1x, a1y, a2x, a2y;
        for (i = 0, leni = this.pts_count; i < leni; i++)
        {
            i2 = i*2;
            imod = ((i+1)%leni)*2;
            a1x = mypts[i2] + offx;
            a1y = mypts[i2+1] + offy;
            a2x = mypts[imod] + offx;
            a2y = mypts[imod+1] + offy;
            if (cr.segments_intersect(x1, y1, x2, y2, a1x, a1y, a2x, a2y))
                return true;
        }
        return false;
    };
    CollisionPoly_.prototype.mirror = function (px)
    {
        var i, leni, i2;
        for (i = 0, leni = this.pts_count; i < leni; ++i)
```

```
        {
            i2 = i*2;
            this.pts_cache[i2] = px * 2 - this.pts_cache[i2];
        }
};
CollisionPoly_.prototype.flip = function (py)
{
        var i, leni, i21;
        for (i = 0, leni = this.pts_count; i < leni; ++i)
        {
            i21 = i*2+1;
            this.pts_cache[i21] = py * 2 - this.pts_cache[i21];
        }
};
CollisionPoly_.prototype.diag = function ()
{
        var i, leni, i2, i21, temp;
        for (i = 0, leni = this.pts_count; i < leni; ++i)
        {
            i2 = i*2;
            i21 = i2+1;
            temp = this.pts_cache[i2];
            this.pts_cache[i2] = this.pts_cache[i21];
            this.pts_cache[i21] = temp;
        }
};
cr.CollisionPoly = CollisionPoly_;
```

```
function SparseGrid_(cellwidth_, cellheight_)
{
	this.cellwidth = cellwidth_;
	this.cellheight = cellheight_;
	this.cells = {};
};
SparseGrid_.prototype.totalCellCount = 0;
SparseGrid_.prototype.getCell = function (x_, y_, create_if_missing)
{
	var ret;
	var col = this.cells[x_];
	if (!col)
	{
		if (create_if_missing)
		{
			ret = allocGridCell(this, x_, y_);
			this.cells[x_] = {};
			this.cells[x_][y_] = ret;
			return ret;
		}
		else
			return null;
	}
	ret = col[y_];
	if (ret)
		return ret;
	else if (create_if_missing)
```

```
		{
			ret = allocGridCell(this, x_, y_);
			this.cells[x_][y_] = ret;
			return ret;
		}
		else
			return null;
	};
	SparseGrid_.prototype.XToCell = function (x_)
	{
		return cr.floor(x_ / this.cellwidth);
	};
	SparseGrid_.prototype.YToCell = function (y_)
	{
		return cr.floor(y_ / this.cellheight);
	};
	SparseGrid_.prototype.update = function (inst, oldrange, newrange)
	{
		var x, lenx, y, leny, cell;
		if (oldrange)
		{
			for (x = oldrange.left, lenx = oldrange.right; x <= lenx; ++x)
			{
				for (y = oldrange.top, leny = oldrange.bottom; y <= leny;
++y)
				{
					if (newrange && newrange.contains_pt(x, y))
						continue;		// is still in this cell
```

```
                                        cell = this.getCell(x, y, false);     // don't create if
missing
                                        if (!cell)
                                                continue;           // cell does not exist yet
                                        cell.remove(inst);
                                        if (cell.isEmpty())
                                        {
                                                freeGridCell(cell);
                                                this.cells[x][y] = null;
                                        }
                                }
                        }
                }
                if (newrange)
                {
                        for (x = newrange.left, lenx = newrange.right; x <= lenx; ++x)
                        {
                                for (y = newrange.top, leny = newrange.bottom; y <=
leny; ++y)
                                {
                                        if (oldrange && oldrange.contains_pt(x, y))
                                                continue;           // is still in this cell
                                        this.getCell(x, y, true).insert(inst);
                                }
                        }
                }
        };
        SparseGrid_.prototype.queryRange = function (rc, result)
```

10

```
{
	var x, lenx, ystart, y, leny, cell;
	x = this.XToCell(rc.left);
	ystart = this.YToCell(rc.top);
	lenx = this.XToCell(rc.right);
	leny = this.YToCell(rc.bottom);
	for ( ; x <= lenx; ++x)
	{
		for (y = ystart; y <= leny; ++y)
		{
			cell = this.getCell(x, y, false);
			if (!cell)
				continue;
			cell.dump(result);
		}
	}
};
cr.SparseGrid = SparseGrid_;
function RenderGrid_(cellwidth_, cellheight_)
{
	this.cellwidth = cellwidth_;
	this.cellheight = cellheight_;
	this.cells = {};
};
RenderGrid_.prototype.totalCellCount = 0;
RenderGrid_.prototype.getCell = function (x_, y_, create_if_missing)
{
```

```
        var ret;
        var col = this.cells[x_];
        if (!col)
        {
                if (create_if_missing)
                {
                        ret = allocRenderCell(this, x_, y_);
                        this.cells[x_] = {};
                        this.cells[x_][y_] = ret;
                        return ret;
                }
                else
                        return null;
        }
        ret = col[y_];
        if (ret)
                return ret;
        else if (create_if_missing)
        {
                ret = allocRenderCell(this, x_, y_);
                this.cells[x_][y_] = ret;
                return ret;
        }
        else
                return null;
};
RenderGrid_.prototype.XToCell = function (x_)
```

```
{
	return cr.floor(x_ / this.cellwidth);
};
RenderGrid_.prototype.YToCell = function (y_)
{
	return cr.floor(y_ / this.cellheight);
};
RenderGrid_.prototype.update = function (inst, oldrange, newrange)
{
	var x, lenx, y, leny, cell;
	if (oldrange)
	{
		for (x = oldrange.left, lenx = oldrange.right; x <= lenx; ++x)
		{
			for (y = oldrange.top, leny = oldrange.bottom; y <= leny;
++y)
			{
				if (newrange && newrange.contains_pt(x, y))
					continue;		// is still in this cell
				cell = this.getCell(x, y, false);	// don't create if
missing
				if (!cell)
					continue;		// cell does not exist yet
				cell.remove(inst);
				if (cell.isEmpty())
				{
					freeRenderCell(cell);
					this.cells[x][y] = null;
```

```
                                        }
                                }
                        }
                }
                if (newrange)
                {
                        for (x = newrange.left, lenx = newrange.right; x <= lenx; ++x)
                        {
                                for (y = newrange.top, leny = newrange.bottom; y <=
leny; ++y)
                                {
                                        if (oldrange && oldrange.contains_pt(x, y))
                                                continue;       // is still in this cell
                                        this.getCell(x, y, true).insert(inst);
                                }
                        }
                }
        };
        RenderGrid_.prototype.queryRange = function (left, top, right, bottom, result)
        {
                var x, lenx, ystart, y, leny, cell;
                x = this.XToCell(left);
                ystart = this.YToCell(top);
                lenx = this.XToCell(right);
                leny = this.YToCell(bottom);
                for ( ; x <= lenx; ++x)
                {
                        for (y = ystart; y <= leny; ++y)
```

```
                {
                        cell = this.getCell(x, y, false);
                        if (!cell)
                                continue;
                        cell.dump(result);
                }
        }
};
RenderGrid_.prototype.markRangeChanged = function (rc)
{
        var x, lenx, ystart, y, leny, cell;
        x = rc.left;
        ystart = rc.top;
        lenx = rc.right;
        leny = rc.bottom;
        for ( ; x <= lenx; ++x)
        {
                for (y = ystart; y <= leny; ++y)
                {
                        cell = this.getCell(x, y, false);
                        if (!cell)
                                continue;
                        cell.is_sorted = false;
                }
        }
};
cr.RenderGrid = RenderGrid_;
```

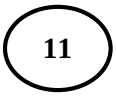
```
var gridcellcache = [];
function allocGridCell(grid_, x_, y_)
{
	var ret;
	SparseGrid_.prototype.totalCellCount++;
	if (gridcellcache.length)
	{
		ret = gridcellcache.pop();
		ret.grid = grid_;
		ret.x = x_;
		ret.y = y_;
		return ret;
	}
	else
		return new cr.GridCell(grid_, x_, y_);
};
function freeGridCell(c)
{
	SparseGrid_.prototype.totalCellCount--;
	c.objects.clear();
	if (gridcellcache.length < 1000)
		gridcellcache.push(c);
};
function GridCell_(grid_, x_, y_)
{
	this.grid = grid_;
	this.x = x_;
```

```
        this.y = y_;
        this.objects = new cr.ObjectSet();
};
GridCell_.prototype.isEmpty = function ()
{
        return this.objects.isEmpty();
};
GridCell_.prototype.insert = function (inst)
{
        this.objects.add(inst);
};
GridCell_.prototype.remove = function (inst)
{
        this.objects.remove(inst);
};
GridCell_.prototype.dump = function (result)
{
        cr.appendArray(result, this.objects.valuesRef());
};
cr.GridCell = GridCell_;
var rendercellcache = [];
function allocRenderCell(grid_, x_, y_)
{
        var ret;
        RenderGrid_.prototype.totalCellCount++;
        if (rendercellcache.length)
        {
```

```
                ret = rendercellcache.pop();
                ret.grid = grid_;
                ret.x = x_;
                ret.y = y_;
                return ret;
        }
        else
                return new cr.RenderCell(grid_, x_, y_);
};
function freeRenderCell(c)
{
        RenderGrid_.prototype.totalCellCount--;
        c.reset();
        if (rendercellcache.length < 1000)
                rendercellcache.push(c);
};
function RenderCell_(grid_, x_, y_)
{
        this.grid = grid_;
        this.x = x_;
        this.y = y_;
        this.objects = [];            // array which needs to be sorted by Z order
        this.is_sorted = true;        // whether array is in correct sort order or not
        this.pending_removal = new cr.ObjectSet();
        this.any_pending_removal = false;
};
RenderCell_.prototype.isEmpty = function ()
```

```
	{
		if (!this.objects.length)
		{
;
;
			return true;
		}
		if (this.objects.length > this.pending_removal.count())
			return false;
;
		this.flush_pending();		// takes fast path and just resets state
		return true;
	};
	RenderCell_.prototype.insert = function (inst)
	{
		if (this.pending_removal.contains(inst))
		{
			this.pending_removal.remove(inst);
			if (this.pending_removal.isEmpty())
				this.any_pending_removal = false;
			return;
		}
		if (this.objects.length)
		{
			var top = this.objects[this.objects.length - 1];
			if (top.get_zindex() > inst.get_zindex())
				this.is_sorted = false;		// 'inst' should be somewhere beneath 'top'
```

```
                this.objects.push(inst);
        }
        else
        {
                this.objects.push(inst);
                this.is_sorted = true;
        }
;
};
RenderCell_.prototype.remove = function (inst)
{
        this.pending_removal.add(inst);
        this.any_pending_removal = true;
        if (this.pending_removal.count() >= 30)
                this.flush_pending();
};
RenderCell_.prototype.flush_pending = function ()
{
;
        if (!this.any_pending_removal)
                return;         // not changed
        if (this.pending_removal.count() === this.objects.length)
        {
                this.reset();
                return;
        }
        cr.arrayRemoveAllFromObjectSet(this.objects, this.pending_removal);
```



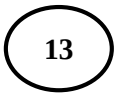
```
        this.pending_removal.clear();
        this.any_pending_removal = false;
};
function sortByInstanceZIndex(a, b)
{
        return a.zindex - b.zindex;
};
RenderCell_.prototype.ensure_sorted = function ()
{
        if (this.is_sorted)
                return;             // already sorted
        this.objects.sort(sortByInstanceZIndex);
        this.is_sorted = true;
};
RenderCell_.prototype.reset = function ()
{
        cr.clearArray(this.objects);
        this.is_sorted = true;
        this.pending_removal.clear();
        this.any_pending_removal = false;
};
RenderCell_.prototype.dump = function (result)
{
        this.flush_pending();
        this.ensure_sorted();
        if (this.objects.length)
                result.push(this.objects);
```

```
};
cr.RenderCell = RenderCell_;
var fxNames = [ "lighter",
                                    "xor",
                                    "copy",
                                    "destination-over",
                                    "source-in",
                                    "destination-in",
                                    "source-out",
                                    "destination-out",
                                    "source-atop",
                                    "destination-atop"];
cr.effectToCompositeOp = function(effect)
{
        if (effect <= 0 || effect >= 11)
                return "source-over";
        return fxNames[effect - 1];        // not including "none" so offset by 1
};
cr.setGLBlend = function(this_, effect, gl)
{
        if (!gl)
                return;
        this_.srcBlend = gl.ONE;
        this_.destBlend = gl.ONE_MINUS_SRC_ALPHA;
        switch (effect) {
        case 1:             // lighter (additive)
                this_.srcBlend = gl.ONE;
```

```
            this_.destBlend = gl.ONE;
            break;
case 2:     // xor
            break;  // todo
case 3:     // copy
            this_.srcBlend = gl.ONE;
            this_.destBlend = gl.ZERO;
            break;
case 4:     // destination-over
            this_.srcBlend = gl.ONE_MINUS_DST_ALPHA;
            this_.destBlend = gl.ONE;
            break;
case 5:     // source-in
            this_.srcBlend = gl.DST_ALPHA;
            this_.destBlend = gl.ZERO;
            break;
case 6:     // destination-in
            this_.srcBlend = gl.ZERO;
            this_.destBlend = gl.SRC_ALPHA;
            break;
case 7:     // source-out
            this_.srcBlend = gl.ONE_MINUS_DST_ALPHA;
            this_.destBlend = gl.ZERO;
            break;
case 8:     // destination-out
            this_.srcBlend = gl.ZERO;
            this_.destBlend = gl.ONE_MINUS_SRC_ALPHA;
```

```
                break;
        case 9:         // source-atop
                this_.srcBlend = gl.DST_ALPHA;
                this_.destBlend = gl.ONE_MINUS_SRC_ALPHA;
                break;
        case 10:// destination-atop
                this_.srcBlend = gl.ONE_MINUS_DST_ALPHA;
                this_.destBlend = gl.SRC_ALPHA;
                break;
        }
};
cr.round6dp = function (x)
{
        return Math.round(x * 1000000) / 1000000;
};
/*
var localeCompare_options = {
        "usage": "search",
        "sensitivity": "accent"
};
var has_localeCompare = !!"a".localeCompare;
var localeCompare_works1 = (has_localeCompare && "a".localeCompare("A", undefined, localeCompare_options) === 0);
var localeCompare_works2 = (has_localeCompare && "a".localeCompare("á", undefined, localeCompare_options) !== 0);
var supports_localeCompare = (has_localeCompare && localeCompare_works1 && localeCompare_works2);
*/
cr.equals_nocase = function (a, b)
```

```
{
        if (typeof a !== "string" || typeof b !== "string")
                return false;
        if (a.length !== b.length)
                return false;
        if (a === b)
                return true;
        /*
        if (supports_localeCompare)
        {
                return (a.localeCompare(b, undefined, localeCompare_options)
=== 0);
        }
        else
        {
        */
                return a.toLowerCase() === b.toLowerCase();
};
cr.isCanvasInputEvent = function (e)
{
        var target = e.target;
        if (!target)
                return true;
        if (target === document || target === window)
                return true;
        if (document && document.body && target === document.body)
                return true;
        if (cr.equals_nocase(target.tagName, "canvas"))
```



```
            return true;
        return false;
```

Kompleksitas Siklomatis (pengukuran kuantitatif terhadap kompleksitas logis suatu program) dari grafik alir dapat diperoleh dengan perhitungan:

Dimana : $$V(G) = E - N + 2$$

E = Jumlah edge grafik alir yang ditandakan dengan gambar panah

N = Jumlah simpul grafik alir yang ditandakan dengan gambar lingkaran

Sehingga kompleksitas siklomatisnya

V (G) = 25-18+2= 9

Basis set yang dihasilkan dari jalur independent secara linier adalah jalur sebagai berikut :

1-2-3-4-5-6-17-18-5-6-7-8-10-11-12-13-14-15-3

1-2-3-4-5-6-18-19-10-11-12-13-14-16

1-2-3-4-5-6-7-9-10-11-12-13-14-15-3

1-2-3-4-12-13-14-15-3

1-2-3-4-12-13-16-14

1-2-3-4-5-6-7-9-10-11-12-13-14-15-3

1-2-3-4-5-6-7-8-10-11-12-13-14-16-17

1-2-3-4-5-6-7-9-10-11-12-13-14-16-17

Ketika aplikasi dijalankan, maka terlihat bahwa salah satu basis set yang dihasilkan adalah 1-2-3-4-5-6-18-19-5-6-7-8-10-11-12-13-14-15-3 dan terlihat bahwa simpul telah dieksekusi satu kali.

B. Pengujian Black Box

Dalam pengujian berikutnya dilakukan untuk memastikan bahwa suatu event atau masukan akan menjalankan proses yang tepat dan menghasilkan output sesuai dengan rancangan yang dibuat.

Tabel III.4. Pengujian Black Box

INPUT/ EVENT	PROSES	OUTPUT/ NEXT STAGE	HASIL PENGUJIAN
Tombol Play	On (press) { gotoAndStop(action.play)}:	Mulai Bermain Game	Sesuai
Tombol profil	On (press) { gotoAndStop(action.profil)}:	Menampilkan profil	Sesuai
Tombol Lanjutkan level	On (press) { gotoAndStop(action.nextlevel)}:	Melanjutkan ke level selanjutnya	Sesuai
Tombol Ulangi	On (press) { gotoAndStop(action.loop.state)}:	Mengulang permainan	Sesuai
Tombolkeluar	On (press) { fscommand(“quit”,””)}:	Keluar / menutup program	Sesuai

3.4.2. *Support*

Tabel III.6. Kebutuhan Hardware dan Software

Kebutuhan	Keterangan
Sistem Operasi	Android Jelybean atau sesudahnya
Processor	Snapdragon 800 Mhz
Memori RAM	256 MB atau Selebihnya
Harddisk	4 GB
Software	Construct Ver. 2

Catatan : Spesifikasi yang ditampilkan adalah spesifikasi minimal yang disarankan

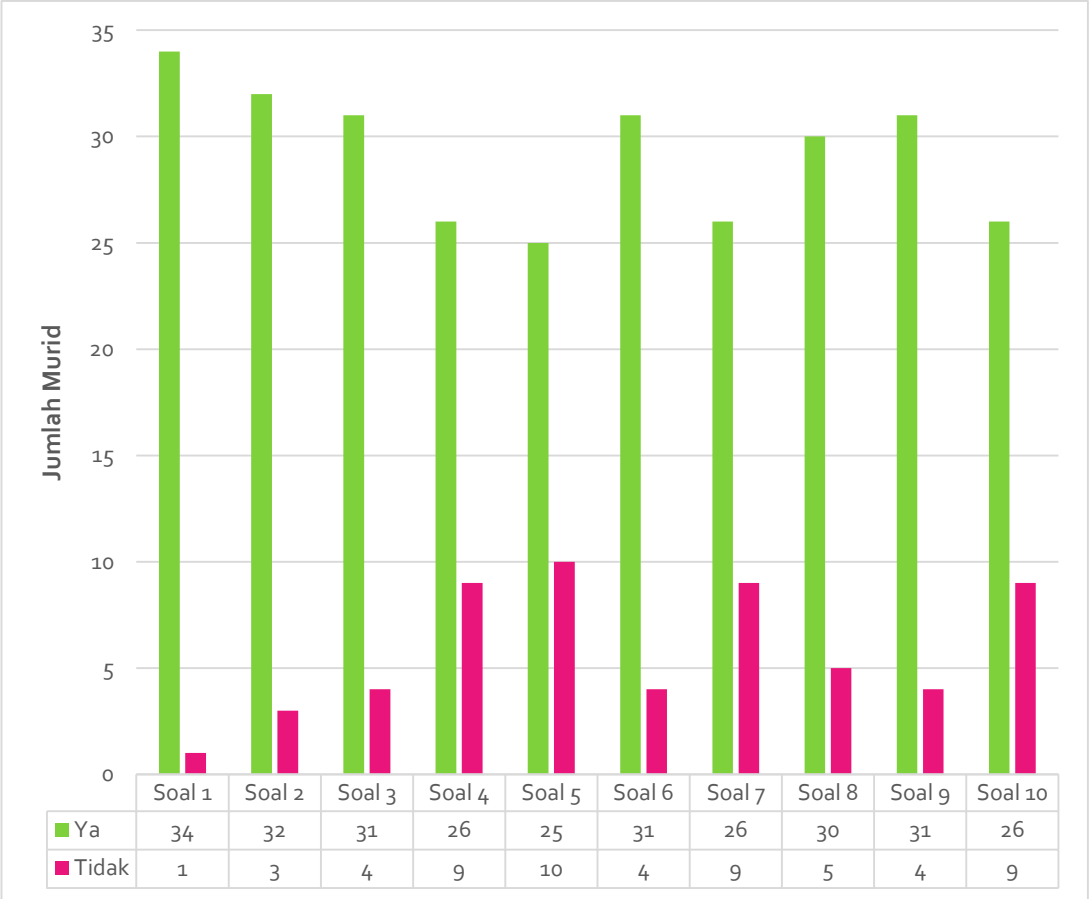
3.5. Hasil Pengolahan Data Kuisioner Animasi Interaktif

Kuisioner diberikan kepada siswa kelas VII yang terdiri dari 35 siswa dan dengan jumlah kuisioner terisi sebanyak 35 lembar kuisioner. Siswa diberikan pertanyaan tentang bagaimana pendapat mereka setelah aplikasi ini dijalankan yang terdiri dari 10 pertanyaan. Siswa hanya mempunyai tugas untuk memberikan tanda *ceklist* disamping pertanyaan yang diberikan.

No.	Pertanyaan Untuk Siswa / Pengguna	Ya	Tidak
1	Menurut anda apakah aplikasi ini sangat mudah digunakan?		
2	Apakah aplikasi ini dapat membantu anda memahami tentang bahaya rokok bagi kesehatan ?		
3	Apakah setelah mencoba aplikasi ini anda mengetahui macam-macam penyakit dan bahaya merokok ?		
4	Apakah aplikasi ini membuat anda menjauhi rokok ?		
5	Menurut anda aplikasi ini menarik untuk dimainkan ?		
6	Apakah penyampaian informasi bahaya rokok mudah anda mengerti ?		
7	Apakah gambar dan animasi yang terdapat di aplikasi ini menarik ?		
8	Apakah aplikasi ini membuat anda lebih peduli tentang bahaya rokok di masyarakat ?		
9	Apakah bahasa yang ada di aplikasi ini mudah dimengerti?		
10	Apakah aplikasi ini akan anda bagikan untuk keluarga dan teman ?		

Ket: Beri tanda ceklist (√) pada jawaban yang di pilih

Tabel III.7. Kuisioner Game Interaktif Bahaya Rokok



Gambar III.13. Grafik Kuisioner Para Siswa

Dari hasil kuisioner pada gambar III.13 dapat diambil kesimpulan bahwa sebagian besar siswa dapat menggunakan aplikasi ini dengan mudah dan dapat mudah dalam mengetahui tentang bahaya rokok bagi kesehatan, dapat membuat para siswa lebih peduli terhadap bahaya rokok untuk kesehatan diri sendiri dan orang-orang di sekitar dan menjadi alternatif penyampaian informasi kepada anak-anak dengan cara yang menyenangkan dan mudah dimengerti.