BAB II

TINJAUAN PUSTAKA

2.1. Teori Dasar

2.1.1. *Game*

Sudono dalam (Nama, Pamungkas, Mardiana, & Septama, 2019) mengungkapkan bermain merupakan suatu rangkaian kegiatan yang dapat memberikan informasi, hiburan, dan mengembangkan imajinasi pada anak yang dilakukan dengan atau tanpa menggunakan media. Menurut Arief (Sadiman, 2010), *game* adalah sesuatu yang bisa dimainkan dengan berdasarkan aturan tertentu sehingga ada yang menang dan ada yang kalah dalam konteks bukan serius dengan tujuan menghibur diri. *Game* merupakan sebuah aktivitas terstruktur yang merupakan hasil akhir dari proses multimedia dalam bentuk alat dengan tujuan bersenang-senang dan dapat digunakan sebagai media belajar.

Banyak *programmer-programmer* hebat yang berawal dari kegemaran terhadap bermain *game*. Meskipun kata *game* terdengar ditujukan untuk mainan anak, sebenarnya terdapat banyak manfaat yang bisa didapat dari bermain *game*. Salah satu manfaat tersebut meningkatkan kemampuan menalar atau logika. Dalam *game* terdapat permasalahan yang harus dicari jalan keluarnya untuk mencapai tahapan/tujuan tertentu, sehingga diperlukan kemampuan berpikir dan bernalar sampai dengan menyusun strategi untuk dapat menyelesaikan setiap permasalahan dalam *game*. Manfaat *game* sebagai hiburan yang cukup menantang dapat

membantu menyegarkan pikiran setelah melakukan aktivitas seharian, dan dapat digunakan untuk melatih refleks dan ketepatan pengguna terhadap sesuatu.

Menurut Wafda (Rifai, 2015), *game* dapat diklasifikasikan menjadi dua jenis, yaitu *game* 2D dan 3D. *Game* 2D memiliki keadaan ruang yang memiliki dua sisi, yaitu sisi X dan Y sehingga objek yang ditampilkan hanya dapat dilihat dari satu arah dan tidak dapat menampilkan bagian belakang dan samping sebuah objek. Sedangkan *game* 3D melibatkan 3 sisi, yaitu sisi X, sisi Y dan sisi Z sehingga objek dapat dilihat dari berbagai sudut menggunakan kamera. Saat ini *game* dengan desain 3 dimensi sangat diminati oleh masyarakat dibandingkan dengan *game* dengan desain 2 dimensi.

Sementara dilihat dari bagaimana cara pengguna bermain, *game* memiliki beberapa kategori diantaranya: *arcade, first person shooter, role Playing game, simulation, racing*, dan sebagainya. Salah satu kategori yang sering dimainkan adalah *game arcade*. *Game arcade* adalah jenis permainan yang bergantung pada ketangkasan tangan pemain dalam mengendalikan kontrol yang umumnya mempunyai karakteristik seperti memiliki konsep dan desain sederhana dan meningkatkan tingkat kesulitan pada setiap level.

2.1.2. *Game* Edukasi

Menurut Efwan (Efwan, Bunyamin, & Wahyudin, 2014), *game* edukasi adalah sebuah *game* berbasis digital yang dirancang sebagai bentuk pengayaan pendidikan dalam mendukung proses pembelajaran dan pengajaran. Efwan, Bunyamin, & Wahyudin melakukan penelitian yang menghasilkan sebuah media *game* edukasi 3 dimensi untuk mempelajari Bahasa inggris. Sedangkan menurut

Rahman (Rahman & Tresnawati, 2016), mengemukakan *game* edukasi merupakan permainan yang dikemas sedemikian rupa sehingga dapat merangsang daya pikir dan melatih meningkatkan konsentrasi penggunanya. Berdasarkan pendapat-pendapat diatas dapat disimpulkan *game* edukasi adalah suatu bentuk *game* yang dapat dimanfaatkan untuk mendukung proses belajar mengajar dengan cara yang lebih menyenangkan dan lebih kreatif serta untuk memberikan pengetahuan kepada penggunanya melalui media yang menarik.

Game dengan tujuan mendidik dimanfaatkan sebagai salah satu media edukasi yang memiliki karakteristik pola pembelajaran *learning by doing*. Sama halnya dengan proses belajar mengajar di kelas, materi yang disampaikan oleh guru kepada siswa tidak semua dapat memahami secara langsung. Namun melalui praktik, siswa dapat melihat, mengalami dan mengambil kesimpulan dari materi yang sudah dipraktikkan. Siswa cenderung lebih tertarik dan lebih mudah mengingat materi belajar melalui media interaktif dibandingkan belajar menggunakan buku (Hutabri & Putri, 2019). Ini berarti bila *game* memuat teori pembelajaran dapat lebih mudah dipahami melalui praktik. Dengan pola pembelajaran *learning by doing,* pengguna diharuskan untuk belajar sehingga mereka dapat memecahkan masalah yang ada.

Berdasarkan hasil penelitian terdahulu, ditemukan bahwa *game* edukasi mendukung proses pendidikan (Rifai, 2015). *Game* edukasi unggul dalam banyak hal daripada metode pembelajaran tradisional. Beberapa keunggulannya yakni dengan *game* edukasi bisa membantu melatih pengguna untuk berlogika, menganalisis, dan memecahkan masalah yang dihadapi. Membantu dalam

merangsang stimulus otak serta mengembangkan imajinasi. Selain itu, adanya animasi multimedia yang dapat memacu daya ingat untuk menyimpan materi pelajaran dalam waktu yang lebih lama daripada metode pembelajaran tradisional (Nugroho, 2015).

2.1.3. Algoritma

Definisi algorima adalah suatu urutan tahapan / langkah-langkah logis dalam menyelesaikan suatu masalah yang disusun secara sistematis dan berdasarkan logika. Algoritma sangat erat kaitannya dengan kata logika, yaitu kemampuan seseorang dalam berpikir menggunakan alasan tentang suatu masalah yang menghasilkan suatu kebenaran, yang terbukti dan dapat diterima dengan akal. Logika sering dikaitkan dengan kecerdasan, seseorang yang mampu berlogika dengan baik sering disebut sebagai orang yang pintar. Logika sangat identik dengan konsep akal dan penalaran. Penalaran adalah bentuk pengetahuan tidak langsung yang didasarkan pada pernyataan berpikir yang mungkin benar dan mungkin juga salah (Manurung, 2016).

Definisi logika sangat sederhana yaitu ilmu yang mengajarkan cara berpikir sesuai dengan aturan yang tertentu yang telah diberlakukan. Pelajaran logis menciptakan kesadaran dalam menggunakan prinsip untuk berpikir secara sistematis. Kata algoritma diambil dari nama seorang ilmuwan Arab yang bernama Abu Jafar Muhammad Ibnu Musa Al Khuwarizmi penulis buku berjudul “Al Jabar Wal Muqabala” (Munir, 2011). Kata Al Khuwarizmi dibaca dengan Bahasa Inggris menjadi *Algorism* yang kemudian secara perlahan berubah menjadi *Algorithm* yang diserap dalam ke Bahasa Indonesia menjadi kata Algoritma.

Dalam menggunakan algoritma dibutuhkan beberapa pertimbangan. Pertama, algoritma harus benar. Ini berarti algoritma yang dikehendaki memberikan output yang sesuai dengan input yang telah diberikan. Bila *output* salah, algoritma yang digunakan bukan termasuk algoritma yang baik. Kedua, hasil dari algoritma mendekati nilai yang sebenarnya. Ketiga, keefisiensian algoritma. Algoritma yang baik tidak hanya mengeluarkan output yang benar, tetapi juga mencakup waktu dan memori yang digunakan. Bila algoritma tersebut membutuhkan waktu yang lama dan mengonsumsi memori yang banyak, maka algoritma tersebut bukan algoritma yang baik.

Algoritma secara garis besar dibedakan menjadi 2 bentuk penyajian, yaitu tulisan dan gambar. Salah satu bentuk penyajian dengan tulisan berupa *pseudocode* dimana kode yang mirip dengan kode pemrograman yang sebenarnya sehingga lebih mudah dikomunikasikan kepada pemrogram. Sedangkan penyajian dalam bentuk gambar dapat berupa *flowchart* yang memperlihatkan urutan dan hubungan antar proses beserta pernyataannya.

2.1.4. *Android*

Android adalah sebuah sistem operasi untuk perangkat *mobile* berbasis *linux* yang mencakup sistem operasi, *middleware*, dan aplikasi (Efendi, 2018). Kemajuan teknologi saat ini tentunya tidak lepas dari perkembangan teknologi yang semakin canggih. *Android* sendiri akan terus berkembang dan memperbaharui sistem operasinya dengan mengikuti perkembangan teknologi saat ini. Hal ini dapat dilihat dari adanya versi demi versi yang telah di *upgrade* setiap tiga sampai enam bulan kedepan yang terus dikembangkan oleh pihak *android*. Demi menghadapi pesaing

(kompetitor) lainnya supaya konsumen tetap berada atau merasa nyaman dengan *smartphone* yang dimilikinya, *android* juga terus memberi pelayanan yang memuaskan bagi pelanggannya. Berbagai fitur yang telah ditawarkan oleh android bisa menjadikannya platform *smartphone* handal yang menguasai pangsa pasar dunia (global) hingga saat ini.

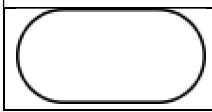
2.1.5. *Unified Modeling Language* (UML)

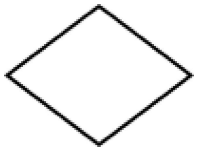
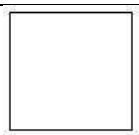
Unified Modeling Language (UML) merupakan suatu desain pemodelan secara visual mengenai arus sistem informasi. Umumnya UML digunakan oleh pengembang sistem sehingga perancangan dapat terarah dan terdokumentasi dengan baik. Berdasarkan buku "Rekayasa Perangkat Lunak", dalam memodelkan suatu sistem, UML terdiri dari 13 macam diagram yang didasarkan pada kebutuhan dan karakteristik sistem (Rosa & Shalahuddin, 2015). Masing-masing model menjelaskan cara kerja dari proses suatu sistem. Berikut beberapa model yang digunakan dalam penelitian ini dan penjelasannya.

a. *Flowchart*

Flowchart adalah suatu bagan dengan simbol-simbol tertentu yang menggambarkan urutan proses secara mendetail dan hubungan antara suatu proses (instruksi) dengan proses lainnya dalam suatu program.

Tabel 2. 1 Simbol Notasi dan Relasi dalam Diagram *Use case*

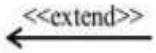
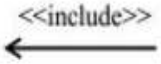
Simbol	Nama	Kegunaan
	Garis Alir	Menunjukkan arah aliran algoritma, dari satu proses ke proses berikutnya.
	Terminal	Menunjukkan awal atau akhir sebuah proses.

	Proses / Langkah	Menyatakan kegiatan yang akan terjadi dalam diagram alir.
	Titik Keputusan	Proses / langkah di mana perlu adanya keputusan atau adanya kondisi tertentu. Di titik ini selalu ada dua keluaran untuk melanjutkan aliran kondisi yang berbeda.
	Masukan / Keluaran	Digunakan untuk mewakili data masuk, atau data keluar. Hanya bisa dimulai dari masukan menuju keluaran, bukan sebaliknya.
	*Predefined Process*	Digunakan untuk menunjukkan suatu proses yang begitu kompleks, sehingga tidak bisa dijelaskan di diagram alir ini dan merujuk pada diagram alir yang terpisah.
	Persiapan / Inisialisasi	Menunjukkan operasi yang tidak memiliki efek khusus selain mempersiapkan sebuah nilai untuk langkah / proses berikutnya.
	Konektor dalam Halaman	Digunakan untuk menghubungkan satu proses ke proses lainnya, sama halnya seperti tanda panah dalam pengulangan. Namun hanya bisa menghasilkan satu keluaran.
	Konektor Luar Halaman	Berfungsi untuk menghubungkan satu proses ke proses lainnya, sama halnya seperti tanda panah, hanya saja untuk merujuk ke halaman yang berbeda.
	Kontrol / Inspeksi	Menunjukkan proses / langkah di mana ada inspeksi atau pengontrolan.

b. *Use case diagram*

Use case merepresentasikan hubungan antara fungsionalitas yang membentuk sistem secara teratur dengan aktor. *Use case diagram* menggunakan sudut pandang pengguna sistem untuk menggambarkan pola perilaku sistem dan urutan kegiatan yang berlangsung. Gambaran tersebut berupa gambaran singkat hubungan antara *use case*, sistem, dan *actor*. Masing-masing *use case* didefinisikan lebih rinci melalui sebuah skenario. Notasi dan relasi dari *use case* sebagai berikut:

Tabel 2. 2 Simbol Notasi dan Relasi dalam Diagram *Use Case*

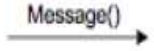
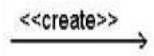
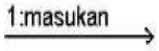
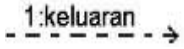
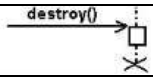
Simbol	Nama	Kegunaan
	*Actor*	Mewakili manusia, sistem, atau perangkat yang akan menggunakan berinteraksi dengan sistem.
	*Use case*	Menggambarkan hubungan *actor* dengan sistem
	Association	Komunikasi antara masing-masing aktor dan setiap *use case* dalam *use case* atau *use case* berinteraksi dengan aktor.
	*Extend*	Menunjukkan bahwa bagian dari suatu elemen adalah penambahan fungsional dari elemen lain jika kondisi tertentu terpenuhi
	*Include*	Menunjukkan bahwa suatu bagian dari elemen seluruhnya merupakan bagian dari elemen lainnya
	*Generalization*	Menunjukkan hubungan antara elemen yang lebih umum dengan elemen yang lebih spesifik.
	*Dependency*	Hubungan yang menunjukkan bahwa perubahan dalam satu elemen mempengaruhi elemen lainnya. Terdapat dua tipe yaitu *include* dan *extend*.

c. *Sequence diagram*

Diagram ini menjelaskan bagaimana mendefinisikan input dan output serta urutan interaksi antara sistem dan pengguna untuk sebuah *use case*. Secara garis besar diagram sekuen memberikan informasi mengenai urutan informasi diantara objek-objek yang berkomunikasi. Jumlah diagram sekuen yang akan dibuat sebanyak *use case* yang telah didefinisikan. Berikut simbol – simbol dalam diagram sekuen:

Tabel 2. 3 Simbol Notasi dan Relasi dalam Diagram Sekuen

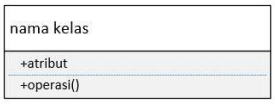
Simbol	Deskripsi
	Aktor: mewakili orang, proses maupun sistem yang berinteraksi dengan sistem yang akan dibuat. Dapat berupa objek orang maupun objek nama.

	Objek: berpatisipasi dalam dengan mengirimkan atau menerima pesan
	Garis hidup: menunjuk objek-objek yang saling berinteraksi
	Waktu aktif: mendeklarasikan suatu objek dalam keadaan aktif dan interaktif.
	Pesan: memuat informasi-informasi mengenai kegiatan yang terjadi
	Pesan tipe *create*: menyatakan bahwa suatu objek dibentuk oleh objek yang lain, arah panah mengarah pada objek yang dibuat
	Pesan tipe *send*: menyatakan bahwa suatu objek mengirimkan data masuka ke objek lain
	Pesan tipe *return*: menyatakan bahwa suatu objek telah menjalankan atau menghasilkan suatu kembalian ke objek tertentu.
	Pesan tipe *destroy*: menyatakan suatu objek mengakhiri keadaan objek yang lain

d. *Class diagram*

Class diagram menunjukan kelas-kelas yang ada dari sebuah sistem dan hubungannya secara logika. Diagram ini digunakan untuk mendokumentasikan dan menggambarkan kelas-kelas yang akan dibuat untuk membangun sistem dan hubungan antara *class object* tersebut. Kelas-kelas tersebut dapat ditentukan/ditemukan dengan cara memeriksa objek-objek dalam *sequence diagram*. Berikut notasi-notasi yang ada dalam *class diagram*:

Tabel 2. 4 Simbol Notasi dan Relasi dalam Diagram Kelas

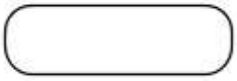
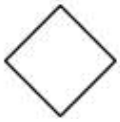
Simbol	Deskripsi
	Kelas: suatu kelas pada struktur sistem

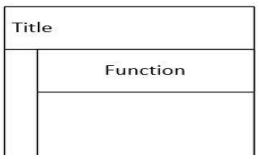
	Antarmuka: sebagai antarmuka komponen
	Asosiasi: Hubungan antara kelas dengan makna umum
	Asosiasi berarah: hubungan antara kelas dengan makna suatu kelas yang digunakan oleh kelas lain
	Generalisasi: relasi antarkelas dengan makna umum khusus
	Agregasi: Hubungan antar kelas dengan makna semua bagian

e. *Activity diagram*

Diagram aktivitas menjelaskan alur kerja sistem bisnis atau proses atau menu yang ditemukan dalam perangkat lunak. Diagram aktivitas menggambarkan aktivitas yang terjadi dalam sistem. Mulai dari awal tahapan sampai akhir sistem yang telah dirancang sesuai langkah-langkah proses kerja sistem. Diagram ini dibuat berdasarkan satu atau lebih *use case* yang ada dalam *use case diagram.*

Tabel 2. 5 Simbol Notasi dan Relasi dalam Diagram Aktivitas.

Simbol	Deskripsi
	Status awal: sebuah diagram aktivitas yang memiliki sebuah kondisi status awal
	Aktivitas: aktivitas yang dilakukan oleh sistem umumnya dimulai dengan kata kerja.
	Percabangan: asosiasi percabangan dimana jika ada lebih dari satu pilihan
	Penggabungan: asosiasi dimana lebih dari satu aktivitas digabungkan menjadi satu
	Status akhir: diagram aktivitas yang memiliki status akhir

	*Swimlane*: memisahkan organisasi bisnis yang bertanggung jawab atas aktivitas yang terjadi

2.2. Variabel

Algoritma berisi rangkaian langkah-langkah dalam menyelesaikan suatu masalah. Langkah-langkah tersebut digunakan untuk menyelesaikan masalah yang dapat berupa struktur sekuensial (*sequence*), struktur seleksi (*selection*) dan struktur perulangan (*repetition*). Ketiga jenis struktur tersebut akan membentuk konstruksi didalam suatu algoritma (Munir, 2011).

1. Struktur sekuensial (*sequence)*

Pada dasarnya sebuah program menjalankan proses dari dasar. Dalam struktur sekuensial, langkah-langkah dalam algoritma dilakukan atau dieksekusi secara berurutan dari awal hingga akhir secara berurutan. Dari yang langkah pertama dieksekusi ke langkah yang kedua, selanjutnya dari langkah kedua ke langkah yang ketiga, dan seterusnya. Urutan dalam eksekusi ini menentukan keadaan akhir algoritma. Bila urutan langkah eksekusinya dirubah maka ada kemungkinan memberikan hasil akhir yang berbeda.

Sebagai contoh dalam sebuah persoalan pertukaran isi dua gelas air. Gelas A yang berisi air berwarna hijau dan gelas B yang berisi air berwarna kuning dengan kondisi volume air sama-sama penuh. Bagaimana menukar isi air gelas A dan gelas B sehingga gelas B berisi air berwarna hijau dan gelas A berisi warna kuning. Bila algoritma solusi dari persoalan diatas dituliskan seperti berikut:

a. Menuang air berwarna hijau dari gelas A ke gelas B yang berisi air berwarna kuning.
b. Menuang air berwarna kuning dari gelas B ke gelas A.

Maka algoritma tersebut memberikan hasil berupa gelas B tidak berisi air karena kosong dan gelas A berisi air berwarna kuning. Sehingga algoritma diatas tidaklah benar. Untuk menyelesaikan permasalah tersebut, dibutuhkan sebuah gelas kosong yaitu gelas C yang digunakan sebagai tempat penampungan sementara. Dengan begitu terdapat 3 langkah dalam menyelesaikan persoalan tersebut. Langkah-langkah tersebut berupa:

a. Menuang air berwarna hijau dari gelas A ke dalam sebuah gelas C kosong.
b. Menuang air berwarna kuning dari gelas B ke gelas A yang sudah kosong.
c. Menuang kembali air berwarna hijau dari gelas C ke gelas B yang kosong.

Dari persoalan diatas, algoritma yang digunakan terdiri atas 3 buah pernyataan. Tiap pernyataan akan dieksekusi secara berurutan. Hasil dari algoritma yang telah dieksekusi adalah gelas A berisi air berwarna kuning dari gelas B dan gelas B berisi air berwarna hijau dari gelas A.

2. Struktur seleksi (*selection*)

Dalam struktur seleksi, seleksi langkah-langkah ditetapkan berdasarkan suatu kondisi atau untuk membuat suatu keputusan. Algoritma mengeksekusi instruksi berikutnya jika kondisi yang ditetapkan terpenuhi. Dalam struktur ini, tidak semua instruksi dieksekusi, instruksi akan dijalankan jika memenuhi persyaratan sehingga hanya satu instruksi dari dua atau lebih intruksi yang disediakan akan dieksekusi.

Sebagai contoh, persoalan dalam berkendara diperempatan yang terdapat lampu lalu lintas. Untuk lampu lalu lintas terdapat aturan-aturan yang harus ditaati oleh pengendara. Aturan-aturan tersebut dapat berupa:

a. Jika lampu dari lampu lalu lintas berwarna merah, maka para pengendara harus berhenti.
b. Jika lampu dari lampu lalu lintas berwarna kuning sesudah lampu hijau padam, maka para pengendara bersiap untuk berhenti.
c. Jika lampu dari lampu lalu lintas berwarna kuning sesudah lampu merah padam, maka para pengendara bersiap untuk jalan.
d. Jika lampu dari lampu lalu lintas berwarna, maka para pengendara diperbolehkan untuk jalan.

Pernyatan-pernyataan diatas dapat dituliskan dalam pernyataan kondisional. Adapun pernyataan kondisional dapat ditulis sebagai berikut:

if kondisi *then* aksi

Jika diterjemahkan dalam Bahasa Indonesia, "*if*" berarti "jika", "*then*" berarti "maka", "kondisi" berarti syarat yang dapat ditentukan bernilai benar atau salah, "aksi" berarti bila kondisi benar akan dieksekusi. Sedangkan bila kondisi bernilai salah, maka aksi tidak dieksekusi. Dalam struktur seleksi, kata "*if*" dan "*then*" adalah kata kunci. Struktur seleksi *if-then* hanya memberikan hanya satu pilihan aksi bila kondisi terpenuhi (bernilai benar), dan tidak memberikan pilihan lain bila kondisi tidak terpenuhi (bernilai salah).

Bentuk lain dari algoritma dalam struktur seleksi yang memberikan lebih dari 1 aksi berupa *if-then-else* yang dapat dituliskan seperti berikut:

If kondisi *then* aksi 1

else aksi 2

Disini "*else*" berarti "selain itu". Jika syarat dalam kondisi terpenuhi (bernilai benar) maka aksi 1 dieksekusi, selain itu jika kondisi tidak terpenuhi (bernilai salah) maka intruksi mengeksekusi aksi 2. Contoh algoritma dari struktur diatas dapat berupa:

a. *If* hari sudah malam *then* hidupkan lampu *else* matikan lampu.
b. *If* air di bak mandi sudah penuh *then* matikan air *else* buka keran air.

Untuk struktur seleksi yang lebih rumit yang memberikan lebih dari 2 aksi dapat berupa seleksi bersarang yaitu *if-then-elseif-then-else* yang dapat dituliskan seperti berikut:

If kondisi 1 *then* aksi 1

else if kondisi 2 *then* aksi 2

else aksi 3

Dari penulisan tersebut dapat diartikan jika kondisi 1 terpenuhi (bernilai benar) maka eksekusi aksi 1 dan jika kondisi 1 bernilai salah periksa kondisi 2. Jika kondisi 2 bernilai benar maka eksekusi aksi 2, selain itu jika kondisi 1 dan 2 bernilai salah, maka eksekusi aksi 3. Contoh algoritma dari struktur diatas dapat berupa:

a. *If* waktu menunjukkan pukul 12 siang *then* waktunya makan siang *else if* waktu menunjukkan pukul 6 malam *then* waktunya makan malam *else* waktunya istirahat.
b. *If* cuaca hari ini mendung *then* sedia payung *else if* cuaca hari ini hujan *then* buka payung *else* cuaca cerah.

Dari beberapa struktur seleksi yang telah dijabarkan dapat memberikan keuntungan yaitu kemampuan dalam mengikuti jalur aksi yang kemungkinan berbeda berdasarkan syarat kondisi yang ditentukan.

3. Struktur perulangan (*repetition*)

Dalam struktur *repetition*, langkah-langkah tertentu dieksekusi berulang kali hingga syarat suatu kondisi terpenuhi. Terdapat beberapa macam konstruksi perulangan yang berbeda. Dari beberapa konstruksi tersebut, ada yang dapat dipakai untuk masalah yang sama dan ada yang hanya cocok digunakan untuk masalah tertentu. Pemilihan dalam menggunakan konstruksi perulangan untuk masalah tertentu dapat memperngaruhi kebenaran algoritma.

Komputer dapat mengerjakan pekerjaan yang sama tanpa kenal lelah, yang berbeda dengan manusia yang cepat lelah mengerjakan pekerjaan yang berulang-ulang. Sebagai contoh dalam sebuah persoalan untuk menuliskan kalimat yang sama sebanyak 100 kalimat. Bagaimana komputer dapat menuliskan kalimat sebanyak 100 kali. Bagi manusia, solusi yang dapat digunakan dengan menuliskan 1 kalimat per 1 kalimat hingga mencapai 100 kalimat. Berbeda dengan komputer, bila komputer juga menuliskan 1 per 1 kalimat maka menjadi tidak efisien. Oleh karena itu, komputer dapat menggunakan notasi dalam struktur perulangan yang dapat berupa *repeat-until, for* dan *while*. Berikut penjelasannya:

a. Penulisan *repeat-until*: *repeat* aksi *until* kondisi.

   Perulangan ini terjadi berdasarkan nilai benar atau salah. Aksi akan dieksekusi sebanyak kondisi bernilai salah. Jika kondisi sudah bernilai benar, maka aksi akan berhenti dieksekusi.

b. Penulisan *for: for* peubah ← nilai awal *to* nilai akhir *do* aksi *endfor*.

   Pernyataan *for* digunakan untuk menghasilkan perulangan sebanyak yang telah ditentukan. Jumlah perulangan dapat ditentukan sebelum eksekusi dilakukan. Untuk melakukan perulangan, dibutuhkan sebuah peubah yang nilainya akan bertambah setiap kali perulangan terjadi. Jika peubah tersebut sudah mencapai jumlah yang telah ditentukan, maka perulangan akan berhenti.

c. Penulisan *while*: *while* kondisi *do* aksi *endwhile*.

   Aksi akan di eksekusi berulang kali selama kondisi bernilai benar. Bila kondisi masih bernilai salah maka perulangan akan berhenti. Dalam perulangan ini harus diperhatikan bahwa kondisi akan menghasilkan perulangan yang berhenti. Perulangan yang tidak pernah berhenti mengartikan bahwa logika dari algoritma tersebut salah.

Dari persoalan sebelumnya bila digunakan ketiga konstruksi diatas maka dapat dituliskan sebagai berikut:

a. Untuk *repeat-until*

   repeat kalimat until total kalimat sama dengan 100.

b. Untuk *for*

   for peubah ← 1 *to* 100 *do* kalimat *endfor*.

c. Untuk *while*

   while x < 100

   do kalimat

   x ← *x* + *1 endwhile*

2.3. Software Pendukung

2.3.1. *Unity*

Dalam buku yang berjudul "*Programming a Game With Unity*" yang ditulis oleh Andre Ifante (Infante, 2014) mengungkapkan kehadiran *videogame* memunculkan banyak *tools* yang dikembangkan sebagai editor untuk pengembangkan *game* modern kepada individu maupun tim *programmer* dan perancang. Salah satu *tools* tersebut adalah *unity* yang dapat digunakan secara mudah dan cepat, dengan biaya yang murah serta dilengkapi fitur-fitur yang membuatnya ideal dalam pengembangan sebuah *game. Unity* memiliki antarmuka yang sederhana dan mudah dipelajari serta memiliki tingkat grafis yang tinggi sehingga dapat menghasilkan *game* dengan cepat dan berkualitas.

Kelebihan dari *unity* yaitu dapat digunakan untuk membangun *game* 2D dan 3D. *Unity* mampu berjalan untuk hampir semua jenis sistem operasi. *Scripting* dalam *Unity* 3D juga mudah dipelajari dan cukup sederhana. *Unity* memiliki *framework* lengkap untuk pengembangan profesional. Untuk menjalankan *engine* ini, disediakan beberapa pilihan dalam menggunakan bahasa pemrograman seperti C#, boo dan javascript. Berdasarkan uraian diatas, maka dapat disimpulkan *Unity* merupakan suatu *software editor engine* yang mudah dioperasikan dalam hal mengembangkan *game* yang dapat berjalan diberbagai macam platform. Untuk itu, dalam penelitian ini peneliti menggunakan *software unity* dalam mengembangkan *game* edukasi untuk pengenalan dasar algoritma berbasis *android*.

2.3.2. *Blender*

Dalam buku berjudul "*The Beginners Guide To Blender*" oleh Jonathan Lampel (Lampel, 2015), menguraikan sebuah *software* 3D bernama *blender*. Jonathan menjelaskan *software blender* merupakan *software open source* yang sering digunakan dalam pembuatan model cetak 3D, film animasi, efek visual dan lain sebagainya. Selain itu *Blender* juga memiliki berbagai macam fitur fitur yang memudahkan penggunanya termasuk *modeling*, *rendering, texturing*, *rigging, scripting, composite,* post-produksi dan banyak lagi. Dalam penelitian ini, peneliti menggunakan *Blender* sebagai media untuk membuat beberapa model yang akan digunakan dalam mengembangkan *game* edukasi untuk pengenalan dasar algoritma.

2.3.3. *Adobe Ilustrator*

Dalam *ebook* berjudul "Otodidak *Adobe Ilustator*" oleh Jubilee Enterprise (Enterprise, 2018) menjelaskan sebuah *software* desain grafis yang *bernama Adobe Ilustrator* yang dimanfaatkan untuk membuat gambar dan ilustrasi dalam bentuk vektor. Untuk objek vektor memiliki keunggulan dalam hal detail ketajaman gambar. *Adobe illustrator* sering digunakan untuk membuat desain grafis, melukis sebuah objek, menata tulisan, membuat desain website dan lain sebagainya. Dalam penelitian ini, peneliti menggunakan *Adobe illustrator* sebagai alat bantu untuk membuat beberapa gambar dan *icon* yang akan digunakan dalam mengembangkan *game* edukasi.

2.3.4. *Microsoft Visio*

Agar pengembangan *game* edukasi lebih terarah dan dapat memberikan gambaran maka digunakan rancangan diagram UML menggunakan *software* pendukung berupa *Microsoft Visio*. Dalam buku yang berjudul "*Microsoft Visio*" yang diterbitkan oleh Tutorials Point (Tutorials Point (I) Pvt. Ltd., 2017) menjelaskan *Microsoft Visio* adalah sebuah *tools* diagram yang memungkinkan untuk membuat diagram mulai dari tingkatan simple sampai dengan tingkatan kompleks, yang membantu dalam visualisasi data dan pemodelan proses.

Microsoft Visio juga membantu untuk membuat bagan organisasi terperinci, *floor plan*, diagram pivot, dan sebagainya. Berdasarkan uraian tersebut, maka dapat disimpulkan *Microsoft Visio* merupakan sebuah perangkat lunak komputer keluaran perusahaan *Microsoft* yang dapat membantu dalam pembuatan diagram dengan banyak fasilitas sehingga informasi dan sistem dapat digambarkan dalam bentuk suatu diagram disertai penjelasan singkat.

2.4. Penelitian Terdahulu

Penelitian terdahulu yang berkaitan tentang pengembangan *game* edukasi berbasis *Android* yang berhubungan dengan pengenalan dasar algoritma sebagai berikut:

1. Penelitian yang dilakukan oleh Lindberg dan Laine dengan judul "***Formative evaluation of an adaptive game for engaging learners of programming concepts in K-12***" - International Journal of Serious *Game*s, Volume: 5, Nomor: 2, Juni 2018 ISSN: 2384-8766. Penelitian tersebut mengangkat masalah mengenai permintaan global untuk *programmer* meningkat sehingga

beberapa negara telah mengintegrasikan pemrograman ke dalam kurikulum K-12. Metode yang digunakan untuk permasalahan tersebut berupa penyajian materi pembelajaran melalui permainan, yaitu *game* edukasi pemrograman. Hasil dari 32 siswa kelas 6 Korea yang memainkan *game* dan 32 siswa kelas 6 yang mempelajari konsep yang sama menggunakan *handout* menujukkan pembelajaran sama efektifnya pada kedua kelompok. (Lindberg & Laine, 2018).

2. Penelitian yang dilakukan oleh Yogi Udjaja dengan judul "***Gamification Assisted Language Learning For Japanese Language Using Expert Point Cloud Recognize*r**" - International Journal of Computer *Game*s Technology, Volume: 2018, Nomor: 1, Desember 2018 ISSN: 16877055. Penelitian tersebut dilatarbelakangi oleh tingginya calon siswa Internasional yang melanjutkan pendidikan di Jepang yang terkendala oleh keterampilan berbahasa Jepang. Metode yang digunakan dalam penelitan tersebut berupa *Gamifcation Assisted Language Learning* (GALL) yang menyajikan materi pembelajaran melalui *game* untuk merangsang sistem saraf sensorik dan motorik dan memotivasi siswa (pemain) untuk belajar lebih keras. Hasil secara keseluruhan dalam hitungan seminggu terbukti dapat meningkatkan kemampuan pemain dari 20% hingga 100%. (Udjaja, 2018).
3. Penelitian yang dilakukan oleh Tumpal Halomoan Manurung dengan judul "**Perancangan Aplikasi Pembelajaran Logika dan Algoritma dengan menggunakan Metode *Computer Based Instruction***" - Jurnal Riset Komputer (JURIKOM) Volume: 3, Nomor: 1, Februari 2016 ISSN: 2407-

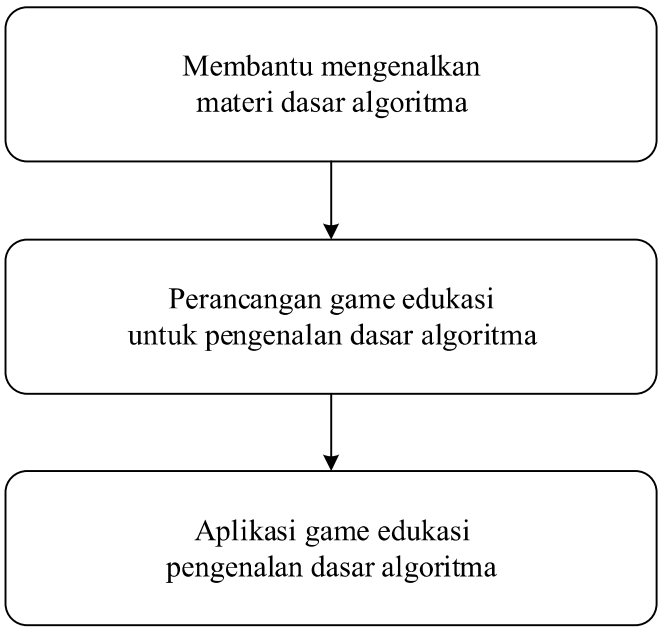
389X. Permasalahan yang diangkat dalam penelitian ini terkait dengan masih banyak orang umum yang dapat menciptakan suatu program namun masih ada yang belum paham dasar logika dan algoritma. Metode yang digunakan oleh peneliti yaitu *Computer Based Instruction* yang menggunakan komputer sebagai media penyampaian materi. Hasil penelitian memberikan kesimpulan bahwa aplikasi memudahkan pengguna untuk memahami pembelajaran logika dan algoritma menggunakan sarana yang menarik dan mudah dipahami dalam bentuk teks, gambar dan video, serta membantu mengarahkan dan memaksimalkan proses belajar mengajar. (Manurung, 2016).

4. Penelitian yang dilakukan oleh Yoyon Efendi dengan judul "**Rancangan Aplikasi *Game* Edukasi berbasis *Mobile* menggunakan *App Inventor***" - Jurnal Intra-Tech Volume 2 Nomor: 1, April 2018 ISSN: 2549-0222. Penelitian tersebut mengangkat masalah yang ditemui pendidik dalam memberi pelajaran yang berbasis buku cetak berupa kurangnya kemauan belajar anak. Tujuan dari *game* edukasi bagai anak prasekolah untuk memberikan stimulasi terhadap perkembangan anak baik fisik, motorik, kemampuan berpikir, serta psikososial dan keberanian. Metode yang digunakan berupa model *research and development* yang sudah terstandarisasi. Hasil dari penelitian ini menunjukkan bahwa aplikasi *game* yang berisi suara, gambar dan kuis dapat memperkenalkan materi dengan cara yang menarik sehingga mudah diterima dan dipahami oleh anak yang masih dalam usia dini (Efendi, 2018).

5. Penelitian yang dilakukan oleh Anik Vega Vitianingsih dengan judul "***Game* Edukasi sebagai Media Pembelajaran Pendidikan Anak Usia Dini**" - Jurnal INFORM Volume 1, Nomor: 1, 2016 ISSN: 2502-3470. Penelitian ini mengangkat masalah dengan mengganti metode pembelajaran konvensional menjadi *game* yang memuat simulasi sehingga dapat mengembangan kreativitas anak. Tujuan dari penelitian ini untuk membuat suatu aplikasi *game* edukasi sebagai media alternatif dalam menyampaikan pembelajaran. Metode yang digunakan dalam penelitian ini berupa metode *waterfall life cycle*. Hasil dari penelitian ini menunjukkan *game* edukasi dapat membantu dalam mengubah metode pembelajaran dari konvensional menjadi pembelajaran melalui media *game* (Vitianingsih, 2016).
6. Penelitian yang dilakukan oleh Ariadie Chandra Nugraha, Moh.Khairudin, dan Deny Budi Hertanto dengan judul "**Rancang Bangun *Game* Edukasi sebagai Media Pembelajaran Mata Kuliah Praktik Teknik Digital**" - Jurnal Edukasi Elektro, Volume 1, Nomor: 1, Mei 2017 ISSN: 2548-8260. Penelitian ini dilatarbelakangi oleh metode atau cara pembelajaran konvensional oleh dosen yang mengakibatkan tidak sedikit mahasiswa yang tidak menerima materi secara keseluruhan. Metode yang diterapkan dalam penelitian ini menggunakan metode *ADDIE* yang terdiri dari tahapan analisis, desain, pengembangan, implementasi, dan evaluasi. Hasil dari penelitian ini menunjukkan bahwa *game* edukasi yang telah dibangun layak untuk digunakan sebagai alat bantu pembelajaran (Nugraha, Khairudin, & Hertanto, 2017).

7. Penelitian yang dilakukan oleh De Firman Aonillah Efwan, Bunyamin dan Wahyudin dengan judul "**Pengembangan *Game* Edukasi 3 Dimensi Bahasa Inggris sebagai Media Pembelajaran untuk Anak**"- Jurnal TEKNOIF Volume. 3 Nomor: 1 April 2015 ISSN: 2338-2724. Penelitian tersebut dilatarbelakangi oleh pentingnya Bahasa Inggris yang menjadi Bahasa Internasional yang semestinya sudah diperkenalkan kepada masyarkat terutama kalangan anak. Metode yang digunakan dalam penelitian ini adalah *multimedia development life cycle* (MDLC). Hasil dari penelitian ini adalah sebuah *game* edukasi yang dapat menambah pembendaharaan kata dalam Bahasa inggris, terampil dalam *writing*, terlatih dalam *listening*, serta pengucapan (Efwan et al., 2014).

2.5. Kerangka Pemikiran



Gambar 2. 1 Bagan Kerangka Pemikiran

Game edukasi untuk pengenalan dasar algoritma ini dimulai dari adanya kesulitan dalam memahami materi algoritma yang masih terbatas oleh teori dan alat bantu ajar yang kurang memadai, kemudian adanya peluang untuk memanfaatkan *smartphone* dengan sistem operasi *android*. Untuk itu, dirancanglah sebuah media *game* edukasi untuk pengenalan dasar algoritma yang mencakup materi struktur sekuensial, struktur seleksi, dan struktur perulangan. Hasil dari penelitian ini berupa sebuah *game* edukasi sehingga dapat mempermudah siswa untuk memahami materi algoritma.