APLIKASI *SUDOKU SHARAF* MENGGUNAKAN METODE *LINEAR CONGRUENTIAL GENERATOR* SEBAGAI PEMBANGKIT DAN *DEPHT FIRST SEARCH* SEBAGAI PENYELESAIAN PERMAINAN BERBASIS ANDROID

SKRIPSI

Oleh:

HADIROTUL MUFIDA

NIM: 08650072

JURUSAN TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI
MAULANA MALIK IBRAHIM MALANG
2013

APLIKASI *SUDOKU SHARAF* MENGGUNAKAN METODE *LINEAR CONGRUENTIAL GENERATOR* SEBAGAIPEMBANGKITDAN *DEPHT FIRST SEARCH* SEBAGAI PENYELESAIAN PERMAINAN BERBASIS ANDROID

SKRIPSI

Diajukan kepada:
Fakultas Sains dan Teknologi
Universitas Islam Negeri Maulana Malik Ibrahim Malang
Untuk Memenuhi Salah Satu Persyaratan Dalam
Memperoleh Gelar Sarjana Komputer (S.Kom)

Oleh:
HADIROTUL MUFIDA
NIM: 08650072

JURUSAN TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI
MAULANA MALIK IBRAHIM MALANG
2013

LEMBAR PERSETUJUAN

APLIKASI *SUDOKU SHARAF* MENGGUNAKAN METODE *LINEAR CONGRUENTIAL GENERATOR* SEBAGAI PEMBANGKIT DAN *DEPHT FIRST SEARCH* SEBAGAI PENYELESAIAN PERMAINAN BERBASIS ANDROID

SKRIPSI

Oleh :

HADIROTUL MUFIDA
NIM. 08650072

TelahDisetujui, 13 September 2013

DosenPembimbing I	DosenPembimbing II
Hani Nurhayati, M.T	Fresy Nugroho, M.T
NIP. 197806252008012006	NIP. 197107222011011001

Mengetahui,
KetuaJurusanTeknikInformatika

Dr. Cahyo Crysdian
NIP. 197404242009011008

HALAMAN PENGESAHAN

APLIKASI *SUDOKU SHARAF* MENGGUNAKAN METODE *LINEAR CONGRUENTIAL GENERATOR*SEBAGAI PEMBANGKIT DAN *DEPHT FIRST SEARCH* SEBAGAI PENYELESAIAN PERMAINAN BERBASIS ANDROID

SKRIPSI

Oleh :

HADIROTUL MUFIDA
NIM. 08650072

Telah Dipertahankan Di Depan Dewan Penguji Skripsi
Dan Dinyatakan Diterima Sebagai Salah Satu Persyaratan
Untuk Memperoleh Gelar Sarjana Komputer (S.Kom)

Tanggal, 19 September 2013

Susunan Dewan Penguji:		Tanda Tangan
1. Penguji Utama	:Yunifa Miftachul Arif, M.T NIP. 19830616201101 1004	( )
2. Ketua Penguji	:Fachrul Kurniawan, M.MT NIP. 19771020200901001	( )
3. Sekretaris	: Hani Nurhayati, M.T NIP. 197806252008012006	( )
4. Anggota Penguji	: Fresy Nugroho M T NIP. 197107222011011001	( )

Mengetahui,
Ketua Jurusan Teknik Informatika

Dr. Cahyo Crysdian
NIP. 197404242009011008

HALAMAN PERNYATAAN

ORISINALITAS PENELITIAN

Saya yang bertanda tangan di bawah ini:

Nama : Hadirotul Mufida

NIM : 08650072

Fakultas / Jurusan : Sains dan Teknologi / Teknik Informatika

JudulPenelitian : **APLIKASI *SUDOKU SHARAF* MENGGUNAKAN METODE *LINEAR CONGRUENTIAL GENERATOR* SEBAGAI PEMBANGKIT DAN *DEPHT FIRST SEARCH* SEBAGAI PENYELESAIAN PERMAINAN BERBASIS ANDROID**

Menyatakan dengan sebenar-benarnya bahwa hasil penelitian saya ini tidak terdapat unsur-unsur penjiplakan karya penelitian atau karya ilmiah yang pernah dilakukan atau dibuat oleh orang lain, kecuali yang secara tertulis dikutip dalam naskah ini dan disebutkan dalam sumber kutipan dan daftar pustaka.

Apabila ternyata hasil penelitian ini terbukti terdapat unsur-unsur jiplakan, maka saya bersedia untuk mempertanggung jawabkan, serta diproses sesuai peraturan yang berlaku.

Malang, 13 September 2013

Yang MembuatPernyataan,

Hadirotul Mufida

NIM. 08650072

MOTTO

مَنْ جَدَّ وَجَدَ

“ Barang Siapa bersungguh-sungguh dia akan mendapatkannya”

Berusahalah jangan sampai terlengah walau sedetik
saja, karena atas kelengahan,
kita tak akan bisa dikembalikan seperti semula

“ DZIKIR, FIKIR DAN AMAL SHOLEH ”

PERSEMBAHAN

Karya ini saya persembahkan kepada:

Ayah Khusnul Khuluk dan Ibu Nurhayati yang telah memberikan kasih sayang yang tak terhingga, selalu memberi nasehat - nasehat dan motivasi serta selalu mendoakan anaknya tiada henti.

Dek Nia dan Dek Fia adik adikQ tercinta terimakasih atas dukungan dan motivasinya serta selalu mendoakan mbak fida

Sahabat – SahabatQ Suwedhy, mbak Ana, mas Sandy, mas Naseh, mas Gepenk, mas Ahda, mas Elong, mas Sony, mas Aam, mas Arif, neng Sofi, Betty Tul, Bayu Gombloh, Hari, Idung, adila, Rina, Devi, jaziir, Deeta, Novi, Nita, Luluk, Alien, Feni, Neneng dan sahabat – sahabatQ dimanapun kalian berada yang selalu mendo'akan , Mendukungku dan menyayangiku.

"Di dalam hariku mengisi kisahku denganmu sahabatku"

Dek izza, Bu Jamilah, ayahnya izza, Mamak, Bapak, Sakur, Hariri, serta keluarga besar bu Jamilah terimakasih atas tempat dan fasilitas serta kasih sayang yang telah diberikan kepada saya selama ini.

Seluruh sahabat dan keluarga-Ku PMII khususnya rayon pencerahan galileo, komisariat Sunan Ampel Malang, cabang PMII kota Malang

Seluruh sahabatku di jurusan Teknik Informatika UIN Maulana Malik Ibarahim Malang angkatan 2008

Serta rekan-rekan dan semua pihak yang tidak dapat disebutkan satu persatu, yang telah membantu penulis dari awal kuliah hingga tersusunnya skripsi ini.

Terimakasih

KATA PENGANTAR

Assalamu'alaikum Wr. Wb.

Segala puji syukur penulis panjatkan kehadirat Allah SWT, karena atas segala rahmat dan limpahan hidayah-Nya, skripsi yang berjudul "*Aplikasi Shorof Sudoku Menggunakan Metode Linear Congruential Generator Sebagai Pembangkit Dan Depht First Search Sebagai Penyelesaian Permainan Berbasis Android*" ini dapat penulis selesaikan sebagai salah satu syarat untuk memperoleh gelar sarjana pada program studi Teknik Informatika jenjang Strata-1 Universitas Islam Negeri (UIN) Maulana Malik Ibrahim Malang. Sholawat serta salam semoga senantiasa Allah limpahkan kepada Nabi Muhammad SAW, keluarga, sahabat dan seluruh umatnya yang rela berkorban demi kemajuan Islam.

Dalam penyelesaian skripsi ini, banyak pihak yang telah memberikan bantuan baik moril maupun materiil. Atas segala bantuan yang telah diberikan, penulis ingin menyampaikan doa dan ucapan terima kasih yang sedalam-dalamnya kepada:

1. Prof. Dr. H. Mudjia Rahardjo M.Si, selaku Rektor Universitas Islam Negeri (UIN) Maulana Malik Ibrahim Malang beserta seluruh staf. Dharma Bakti Bapak dan Ibu sekalian terhadap Universitas Islam Negeri Malang turut membesarkan dan mencerdaskan penulis.
2. Dr. Hj. Drh.Bayyinatul Muchtaromah, M.Si, selaku Dekan Fakultas Sains dan Teknologi Universitas Islam NegeriMaulana Malik Ibrahim Malang.
3. Dr. Cahyo Crysdian selaku Ketua Jurusan Teknik Informatika Fakultas Sains dan Teknologi UIN Maulana Malik Ibrahim Malang.
4. Hani Nurhayati, M.T selaku Dosen pembimbing skripsi penulis di jurusan Teknik Informatika UIN Maliki Malang. Beliaulah orang tua penulis di UIN Maliki Malang yang telah banyak memberikan bimbingan serta motifasi kepada penulis dalam menempuh jenjang pendidikan ini.
5. Fressy Nugroho M.T selaku Dosen Pembimbing Integrasi Sains dan Islam, beliau yang telah membimbing dan mengarahkan penulis dalam menyusun skripsi ini sehingga tiada dikotomi antara teknologi dan agama.

6. Syahiduzzaman M.Kom, selaku Dosen wali memberikan bimbingan dan motivasi.
7. Seluruh Dosen Universitas Islam Negeri (UIN) Maliki Malang, khususnya Dosen Teknik Informatika, atas segala ilmu yang telah diberikan selama ini.
8. Seluruh *Civitas Akademika* Jurusan Teknik Informatika Fakultas Sains dan Teknologi Universitas Islam Negeri Maulana Malik Ibrahim Malang.
9. Drs. Khusnul Khuluq selaku Kepala Sekolah Madrasah Aliyah Negeri Kota Blitar, yang telah memberikan waktu dan tempat penulis untuk melakukan penelitian.
10. Para guru dan Jajaran Staf Madrasah Aliyah Negeri Kota Blitar, yang telah memberikan masukan dan motifasi dalam penyelesaian skripsi ini.
11. Ayah dan Ibuku tersayang, Adik dan seluruh keluarga besar di Blitar yang telah banyak memberikan doa, motivasi dan dorongan dalam penyelesaian skripsi ini.
12. Serta seluruh pihak yang secara langsung maupun tidak langsung membantu proses penyusunan skripsi ini yang tidak bisa penulis sebutkan satu persatu. Semoga Allah SWT membalas segala kebaikan dan bantuan yang telah diberikan.

Penulis menyadari bahwa skripsi ini masih banyak kekurangan dan jauh dari sempurna. Oleh karena itu saran dan kritik yang bersifat membangun sangat penulis harapkan untuk perbaikan ke depan.

Semoga skripsi ini dapat bermanfaat bagi pembaca dan dapat menambah pengetahuan kita semua, Amin.

Wassalamualaikum Wr. Wb.

Malang, 13 September 2013

Penulis

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

ABSTRAK

Mufida, Hadirotul. 2013. **Aplikasi Sudoku SharafMenggunakan Metode *Linear Congruential generator* Sebagai Pembangkit dan *Depht First Search* sebagai Penyelesaian Permainan berbasis Android**. Dosen Pembimbing: Hani Nurhayati, M.T, dan Fresy Nugroho, M.T.

Agama islam sudah ditetapkan oleh Allah SWT diturunkan melaluiRasul yang diutusdaribangsa Arab, sehinggakitabsuci Al-Qur'an pun diturunkan dalam bahasa Arab.Al-Qur'an sebagai kitab suci umat islam diturunkan dengan bahasa arab. Tentunya sudah menjadi kewajiban bagi umat islam untuk mempelajari bahasa arab agar dapat memahami kandungan dalam Al-Qur'an. Dalam mempelajari bahasa arab tentunya banyak cabang ilmu yang harus dipelajari, salah satunya yaitu Ilmu *Sharaf*.

Permasalahan siswa-siswi ketika mempelajari *sharaf* (bahasa arab) adalah model pembelajaran yang monoton dan tidak interaktif. Guru / ustadznya memberi penjelasan dan soal-soal *sharaf* sehingga seringkali siswa-siswi merasa bosan. Sehingga diperlukan model pembelajaran yang interaktif dengan model pembelajaran menggunakan *game* untuk mempelajari ilmu *sharaf*.

Berdasarkan latar belakang tersebut dilakukan penelitian untuk menciptakan aplikasi permainan Sudoku Sharaf dengan algoritma *Linear Congruential Generator* sebagai pembangkit dan *Depht First Search* sebagai penyelesaian permainan yang diharapkan dapat menjadi media pembelajaran Ilmu *Sharaf*. Dari hasil penelitian yang dilakukan menunjukkan bahwasanya algoritma *Linear Congruential Generator* mampu melakukan pengacakan soal *game Sudoku* sharaf dan algoritma *Depth First Search* (DFS) berhasil digunakan sebagai *solver* dengan tingkat keberhasilan 100%. Dari sisi pembelajaran *game Sudoku Sharaf*sangat membantu untuk mempelajari dan menghafal perubahan kata (*sharaf*) dalam bahasa arab hal ini terbukti dari penelitian yang dilakukan 90,91% responden pada jurusan Agama dan 91,43% pada jurusan IPA menyatakan bahwa *game Sudoku sharaf*membantu proses belajar dan menghafal *shighah.*

Kata Kunci: Ilmu Sharaf, *Articial Intelligent, Linear Congruential Generator, Depth First Search, Quisioner*

ABSTRACT

Mufida, Hadirotul. 2013. **An Application of Sharaf Sudoku Game Uses the Linear Congruential Generator for Generating and Depht First Search for Solving the Game**. Advisor : Hani Nurhayati, MT, and Fresy Nugroho, MT.

Islamic religion has been established by God revealed through the Apostle who is sent from the Arabs, so that the holy book Quran was revealed in Arabic. Al-Quran as a holy book of Islam revealed to the Arabic language. Surely, it is obligatory for Muslims to learn Arabic in order to understand the content of the Qur'an. In studying the Arabic language course many disciplines that must be learned, one of which is *Sharaf* Science.

Problems of the students when studying sharaf (Arabic) is a learning model that monotonous and not interactive. Teacher / ustadznya give explanations and questions sharaf that students often feel bored. Necessitating an interactive learning model with a model of learning using games to learn the science of sharaf.

Based on this research to create a Sudoku game application Sharaf Congruential Generator with Linear algorithms for generating and Depht First Search as completion of a game that is expected to be a medium of learning science Sharaf. From the results of research conducted shows that the algorithm is able to perform Linear Generator Congruential scrambling about Sudoku game and algorithms sharaf Depth First Search (DFS) was successfully used as the solver with a 100% success rate. Learning the game of Sudoku Sharaf very helpful to learn and memorize the word change (sharaf) in Arabic it is evident from research conducted 90.91% of respondents in the Department of Religion and 91.43% on majoring in science states that the game Sudoku sharaf help process shighah learn and memorize.

keywords :*Sharaf Science, Articial Intelligent, Linear Congruential Generator, Depth First Search, Quisioner*

BAB I

PENDAHULUAN

1.1 Latar Belakang

Agama islam sudah ditetapkan oleh Allah SWT diturunkan melalui Rasul yang diutus dari bangsa Arab, sehingga kitab suci Al-Qur'an pun diturunkan dalam bahasa Arab. Sebagaimana firman Allah dalam surat Yusuf ayat 2.

إِنَّآ أَنزَلۡنَٰهُ قُرۡءَٰنًا عَرَبِيّٗا لَّعَلَّكُمۡ تَعۡقِلُونَ ٢

Artinya: Sesungguhnya Kami menurunkannya berupa Al Quran dengan berbahasa Arab, agar kamu memahaminya. (Yusuf:2)

Ibnu katsir menafsirkan yang demikian itu (bahwa Al-qur'an diturunkan dalam bahasa arab) karena bahasa Arab adalah bahasa yang paling fasih, jelas, luas dan maknanya lebih mengena lagi cocok untuk jiwa manusia. (Tafisr Ibnu Katsir, 2011:15)

Sehingga, sudah menjadi konsekuensi logis bagi umat islam, jika ingin mendalami dan memahami kitab suci Al-qur'an harus memahami bahasa arab. Hal ini senada dengan pendapat yang diutarakan oleh syaikhu lislam Ibnu Taimiyah : "Sesungguhnya ketika Allah menurunkan kitab-Nya dan menjadikan rasul-Nya sebagai penyampai risalah dan hikmah, serta menjadikan generasi awal ini berkomunikasi dengan bahasa arab, maka tidak ada jalan lain untuk mengetahui dan memahami ajaran Islam kecuali

dengan bahasa arab. Oleh karena itu, memahami bahasa arab adalah bagian dari agama." (Ibnu Taimiyah:2012)

Untuk memahami bahasa arab harus belajar ilmu nahwu, yakni ilmu yang mempelajari kaidah - kaidah Bahasa Arab untuk mengetahui bentuk kata dan keadaan-keadaannya ketika masih satu kata (*Mufrod*) atau ketika sudah tersusun (*Murokkab*) (Qowa'idussorfiyah,2010:12). Termasuk didalamnya adalah pembahasan Sharaf. Yakni bagian dari ilmu nahwu yang menekankan kepada pembahasan bentuk kata dan keadaannya ketika mufrodnya (perubahan bentuk kata) (Qowa'idussorfiyah, 2010:12).

Permasalahan siswa-siswi ketika mempelajari *sharaf* (bahasa arab) adalah model pembelajaran yang monoton dan tidak interaktif. Guru / ustadznya memberi penjelasan dan soal-soal *sharaf* sehingga seringkali siswa-siswi merasa bosan. (Husnul, 2011). Sehingga diperlukan model pembelajaran yang interaktif dengan model pembelajaran menggunakan *game* untuk mempelajari ilmu *sharaf*.

Perkembangan teknologi yang cukup pesat sudah memungkinkan untuk membuat permainanan (*game*) pada sebuah aplikasi *mobile (handphone)*. Hal ini membuat para *developer* berlomba-lomba untuk meningkatkan fungsionalitas di perangkat *mobile* dengan cara membuat *game* atau aplikasilainya. Aplikasi *game* adalah salah satu aplikasi yang sangat popular dikalangan pelajar. Banyak pelajar yang menghabiskan waktunya berjam-jam hanya untuk bermain *game* di *handphon*nya.

Oleh karena itu, sangat penting membangun sebuah aplikasi game yang mengandung unsure pembelajaran pada sebuah *handphone.* Salah satu game berbasis pembelajaran adalah *game Sudoku Albata* yang dibuat dan diteliti oleh RiyadlilAbrar (Abrar, 2012) berbasis android. Metode yang digunakan adalah *Harmony Search* sebagai pembangkit dan penyelesai permainan. Dengan menggunakan metode tersebut, permainan *sudoku* dengan jumlah kotak kosong 35-40 dapat diselesaikan dalam waktu tercepat antara 1-15 detik, dengan jumlah kotak kosong 40-45 dapat diselesaikan dalam waktu antara 2-12 detik, dan jumlah kotak kosong 45-50 dapat diselesaikan dalam waktu antara 5-35 detik.

Penelitian lainya dilakukan oleh Ahmad Baihaqi (Baihaqi, 2013) yang berjudul Aplikasi permainan sudoku *sharaf* menggunakan *Branch and Bound* sebagai penyelesai permainan. Ahmad baihaqi mengganti angka pada permainan sudoku menjadi *shighah* dalam ilmu *sharaf,* serta membagi permainan menjadi 3 level, yakni: mudah, sedang dan sulit. Algoritma *Branch and Bound* berhasil diimplementasikan sebagai *solver* permainan Sudoku dengan keberhasilan kecepatan waktu yang berbeda-beda, tergantung level permainan.

Berdasarkan penelitian yang dilakukan oleh Ahmad Baihaqi tersebut, peneliti ingin membandingkan keberhasilan algoritma *Branch and bounch* dengan algoritma *Depth First Search* dari segi kecepatan untuk melakukan *solver* permainan *Sudoku sharaf*.

.

1.2 Rumusan Masalah

Berdasarkan uraian pada latar belakang, maka rumusan masalah penelitian ini adalah

1. Bagaimana membuat Aplikasi *Game Sudoku Sharaf* menggunakan metode *Linear Congruential Generator (LCG)* Sebagai Pembangkit ?
2. Bagaimana membuat Aplikasi *Game Sudoku Sharaf* menggunakan *Depht First Search* sebagai penyelesai permainan *Sudoku Sharaf* berbasis android ?
3. Bagaimana keberhasilan algoritma *Depth First Search* sebagai *solver* permainan *Sudoku sharaf* bila dibandingkan dengan algoritma *Branch And Bound*?

1.3 Tujuan Penelitian

Berdasarkan rumusan masalah, maka tujuan penelitian ini adalah

1. Membuat membuat Aplikasi *Game Sudoku Sharaf* menggunakan metode *Linear Congruential Generator (LCG)* Sebagai Pembangkit dalam pengacakan pembuatan soal *Sudoku Sharaf*.
2. Membuat membuat Aplikasi *Game Sudoku Sharaf* menggunakan *Depht First Search* sebagai penyelesai permainan *Sudoku Sharaf* berbasis android?
3. Membandingkan keberhasilan algoritma *Depth First Search* dengan algoritma *Branch And Bound* sebagai *solver* permainan *Sudoku sharaf*.

1.4 Batasan Masalah

Batasan masalah pada penelitian ini sebagai berikut :

1. Aplikasi *Game Sudoku sharaf* dibangun menggunakan bahasa pemrogaman android dan berbasis *android mobile*.
2. *Game Sudoku sharaf* hanya bisa dimainkan oleh satu *player* (*single player)*.
3. Kata pada rumusan *sharaf* yang digunakan hanya berjumlah 9 kata dan digunakan untuk menggantikan angka pada permainan Sudoku yang sebenarnya.
4. Dimensi kotak pada *game Sudoku sharaf* berjumlah 9x9 kotak dan setiap *grid* berjumlah 3 x 3.
5. Metode yang digunakan untuk melakukan pembangkitan posisi *puzzle game Sudoku sharaf* dengan menggunakan Algoritma *Linear Congruential Generato*r, sedangkan untuk melakukan penyelesaian permainan (*solver*) dengan menggunakan algoritma dengan menggunakan algoritma *Depht First Search.*

1.5 Manfaat Penelitian

Manfaat dari penelitian ini adalah dihasilkan sebuah aplikasi *game* yang digunakan sebagai media pembelajaran ilmu *sharaf*, sehingga memudahkan dalam memahami perubahan kata dalam bahasa arab. Selain itu, juga bisa digunakan regerensi untuk mengukur keberhasilan algoritma *Depth First Search* (DFS) sebagai *solver* dalam permainan sudoku.

1.6 SistematikaPenulisan

Penulisan skripsi ini tersusun dalam lima bab dengan sistematika penulisan sebagai berikut :

BAB I PENDAHULUAN

Pendahuluan, membahas tentang latar belakang masalah, rumusan masalah, batasan masalah, tujuan penyusunan tugas akhir, metedologi, dan sistematika penyusunan tugas akhir.

BAB II LANDASAN TEORI

Landasan teori berisikan beberapa teori yang mendasari dalam penyusunan tugas akhir ini. Adapun yang dibahas dalam bab ini adalah dasar teori yang berkaitan dengan pembahasan tentang perubahan bentuk pada *mufrodat* (kata kerja), Permainan Sudoku, dan *Linear Congruential Generator* Sebagai Pembangkit Dan *Depht First Search* Sebagai Penyelesaian.

BAB III ANALISA DAN PERANCANGAN

Menganalisa kebutuhan sistem untuk membuat *game* meliputi spesifikasi kebutuhan *software* dan langkah-langkah pembuatan Sudoku Sharaf.

BAB IV HASIL DAN PEMBAHASAN

Menjelaskan tentang pengujian *Sudoku Sharaf* yang telah diterapkan dalam pembuatan *game*.

BAB V KESIMPULAN DAN SARAN

Berisi kesimpulan dari penelitian dan saran untuk pengembangan penelitian.

1.7 Metode Penelitian

Untuk mencapai tujuan yang telah dirumuskan sebelumnya, maka metodologi pengumpulan data yang dilakukan dalam penulisan skripsi ini adalah *library research* yaitu suatu cara penelitian dan pengumpulan data teoritis dari buku-buku, artikel, jurnal dan berbagai literatur yang mendukung penyusunan skripsi. Pengumpulan data juga dilakukan dengan melakukan uji coba aplikasi pada pengguna yang disertai dengan angket.

BAB II

TINJAUAN PUSTAKA

2.1. Ilmu Sharaf

2.1.1 Pengertian Ilmu Sharaf

Ilmu Sharaf adalah ilmu yang mempelajari perubahan kalimah untuk memperoleh makna baru dari kata dasarnya. ilmu ini hanya mempelajari kalimah kalimah yang bisa di *tashrif* (memiliki asal *mustaq*), sehingga kalimah yang *jamid* dan kalimah huruf tidak menjadi pembahasan dalam ilmu *sharaf*. dalam *tashrif sharaf*, ada dua cara pentashrifan, yang pertama adalah dengan *tashrif istilahi* dan tashrif lughowi. *Tashrif Istilahi* adalah *tashrif* yang tersusun dari beberapa kalimah *isim* dan kalimah *fi'il*, dengan perubahan dengan huruf tambahan tertentu dapat menimbulkan makna baru,tetapi tidak berubah dari kata dasrnya. seperti contohnya ”pukul” apabila kita ingin merubah kata itu menjadi memukul bentuk kata kerja maka harus dirubah menjadi *fiil* yang asli *mustaq*nya ضَرْبٌ akan menjadi ضَرَبَ atau يَضْرِبُ. sedangkan *Tashrif Lughowi* adalah *tashrif* yang disusun berdasarkan *dhomir* (kata ganti) seperti contohnya satu pemukul bila dirubah menjadi dua pemukul maka *tashrif lughowi*-lah kita bisa mendapatkan perubahan itu (Mufarakhoh, 2011).

2.1.2 Pengertian Tashrif

Dalam Ilmu *Sharaf*, Para Ulama telah membagi *tashrif* ini menjadi dua macam, yaitu *Tashrif Lughowi* dan *Tashrif Istilahi*

- *Tashrif Lughowi* adalah berubah atau mengubah, dari bentuk aslinya kepada bentuk yang lain.
- Tashrif Istilahi adalah berubahnya bentuk asal pertama *fi'il madhi* menjadi *fi'il mudhore'* menjadi *masdhar* menjadi *isim fa'il* menjadi *isim maf'ul* menjadi *fi'il amar* menjadi *fi'il nahi* menjadi *isim zaman* menjadi *isim makan* dan seterusnya sampai *isim alat.*

2.1.3 Pembagian Tashrif

2.1.3.1 Tashrif Lughowy

Pengertian *Tashrif Lughowy* adalah Ilmu yang mempelajari tentang perubahan kalimat dari bentuk satu ke bentuk lainnya dengan meninjau *mufrod tasniyah* jamaknya, *mudzakkar mu'annats*-nya, serta *ghoib khitob* dan *takallum*-nya.

2.1.3.2 Tashrif Istilahi

Tashrif Istilahi adalah *Tashrif*-an untuk mengetahui bentuk *shighah* dari suatu kalimat. Atau dalam bahasa pesantrennya yaitu *tashrif*-an / *qiyasi* /

ejaan yang dimulai dari *shighah fi'il madhi* dan di akhiri sampai *isim alat.*

Contoh : عَلَمَ – يَعْلَمُ – عِلْمٌ – عَالِمٌ – مَعْلُوْمٌ – مَعْلَمٌ – عَلِيْمٌ - عَلاَّمٌ

الماضي	المضارع	المصدر	المصدر	الفاعل	المفعول	الأمر
فَعَلَ	يَفْعَلُ	فَعْلاً	مَفْعَلاً	فَاعِلٌ	مَفْعُوْلٌ	افْعَلْ
فَعَّلَ	يُفَعِّلُ	تَفْعِيْلاً	تَفْعِلَةً	مُفَعِّلٌ	مُفَعَّلٌ	فَعِّلْ
أَفْعَلَ	يُفْعِلُ	إِفْعَالاً	مفعلا	مُفْعِلٌ	مُفْعَلٌ	أَفْعِلْ
فَاعَلَ	يُفَاعِلُ	مُفَاعَلَةً	فِعَالاً	مُفَاعِلٌ	مُفَاعَلٌ	فَاعِلْ
تَفَعَّلَ	يَتَفَعَّلُ	تَفَعُّلاً	مُتَفَعَّلاً	مُتَفَعِّلٌ	مُتَفَعَّلٌ	تَفَعَّلْ
افْتَعَلَ	يَفْتَعِلُ	افْتِعَالاً	مُفْتَعَلاً	مُفْتَعِلٌ	مُفْتَعَلٌ	افْتَعِلْ
انْفَعَلَ	يَنْفَعِلُ	انْفِعَالاً	مُنْفَعَلاً	مُنْفَعِلٌ	مُنْفَعَلٌ	انْفَعِلْ
تَفَاعَلَ	يَتَفَاعَلُ	تَفَاعُلاً	مُتَفَاعَلاً	مُتَفَاعِلٌ	مُتَفَاعَلٌ	تَفَاعَلْ
اسْتَفْعَلَ	يَسْتَفْعِلُ	اسْتِفْعَالاً	مُسْتَفْعَلاً	مُسْتَفْعِلٌ	مُسْتَفْعَلٌ	اسْتَفْعِلْ

Tabel 2.1 Perubahan Bentuk Kata menurut Isimnya (Contoh *Tashrif Istilahi*)
(Sumber : http://arabic.manesa.blogspot.com/ringkasan sharaf istilahi)

2.2. Permainan Sudoku

Permainan ini sudah dikenal di media masa sejak abad ke 19, ketika permainan *Puzzle Magic Squares* di Perancis ramai dibicarakan. *Magic Square* sendiri berbeda dengan *Sudoku*, karena *Sudoku* terdiri dari 9 angka dan lebih membutuhkan kemampuan aritmatika daripada kemampuan logika untuk menyelesaikan permainan. Awalnya *Sudoku* hanya mengenal baris dan kolom saja tapi dalam perkembangannya ada yang namanya *region*.

Menurut *Will Shortz*, Sudoku *Modern* kebanyakan di desain oleh Horward Gams, arsitek berumur 74 tahun dan sekaligus *freelance* konstruktor *puzzle*, dan pertama kali permainan *Sudoku* dipublikasikan tahun 1979 oleh majalah Dell sebagai *Number Place*. Kemudian ia meninggal pada tahun 1989 sebelum melakukan perubahan – perubahan *Sudoku*.

Puzzle tersebut diperkenalkan di Jepang oleh Nikoli dalam majalah bulanan Nikolist sekitar April 1984 sebagai "*As Suuji wa dokushin ni kagiru*" yang dapat diterjemahkan sebagai "Angka harus terjadi sekali", dan akhirnya nama tersebut dinamakan menjadi Sudoku oleh Maki Kaji, nama tersebut didapatkan dari bahasa Kanji.

Tahun 1997, sudah banayk ditemukan *puzzle* Sudoku dalam toko buku Jepang, dan dalam 6 tahun berikutnya Sudoku dikembangkan menjadi program komputer. Peningkatan permainan Sudoku semakin cepat diketahui umum melalui majalah-majalah dan koran dari berbagai dunia. Dan tahun 2005 untuk pertama kalinya Sudoku menjadi bagian dari acara TV berjudul Sudoku *Show*, Sudoku *Live*.

Sudoku adalah sebuah permainan atau logika yang berbasis kombinatorial penempatan angka. Tujuan dari permainan ini adalah mengisi kotak-kotak dengan angka sehingga setiap kolom, baris dan area (kotak kecil yang terbentuk dari susunan kotak seperti kotak 3 x 3 pada *Sudoku* 9 x 9) berisi semua antara angka 1 sampai 9.

Sudoku bisa memiliki beberapa bentuk, antara lain *Sudoku* gambar, angka dan huruf. Yang paling populer diataranya adalah *Sudoku* angka. Walaupun *Sudoku* dapat berbentuk angka, *Sudoku* tidak berhubungan dengan operasi matematika. Logikalah

yang menentukan dimana angka harus diletakkan dan di sinilah letak keasyikannya. Setiap kali kita bisa menyelesaikannya, kita pasti ingin mengerjakannya terus-menerus.

5	3			7				
6			1	9	5			
	9	8					6	
8				6				3
4			8		3			1
7				2				6
	6					2	8	
			4	1	9			5
				8			7	9

5	3	4	6	7	8	9	1	2
6	7	2	1	9	5	3	4	8
1	9	8	3	4	2	5	6	7
8	5	9	7	6	1	4	2	3
4	2	6	8	5	3	7	9	1
7	1	3	9	2	4	8	5	6
9	6	1	5	3	7	2	8	4
2	8	7	4	1	9	6	3	5
3	4	5	2	8	6	1	7	9

Gambar 2.1 Teka-teki Sudoku dan Solusi Penyelesainnya
(Sumber : Arif Dian :2010: Amikom Yogyakarta)

Selain memiliki beberapa bentuk, *Sudoku* juga memiliki beragam tipe berdasarkan banyaknya kotak *Sudoku*, mulai dari yang paling mudah yaitu *Sudoku* 2x2, *Sudoku* 4x4, *Sudoku* 6x6 hingga / *Sudoku* 9x9.

Berikut ini adalah peraturan permainan *Sudoku*:

1) Setiap angka/gambar/huruf hanya dapat muncul sekali dalam setiap baris.
2) Setiap angka/gambar/huruf hanya dapat muncul sekali dalam setiap kolom.
3) Setiap angka/gambar/huruf hanya dapat muncul sekali dalam setiap kotak kecil.

2.3. Metode *Linear Congruential Generator*

Dalam permainan *sudoku*, pembangkit bilangan secara acak sebagai soal yang akan ditampilkan sangatlah penting untuk menentukan tingkat kesulitan permainan tersebut. Oleh karena itu dibutuhkan sebuah generator (pembangkit) yang mampu membangkitkan bilangan acak dan dalam penelitian ini akan menggunakan *Linear Congruential Generator*.

Menurut Riani L. (2010:3), bilangan acak adalah bilangan yang tidak dapat diprediksi kemunculannya. Pada zaman dahulu bilangan acak diperoleh dengan cara melempar dadu atau mengocok kartu. Pada zaman modern bilangan acak diperoleh dengan cara membentuk bilangan acak secara numeric / aritmatik (menggunakan komputer), disebut "*Pseudo Random Number*" (bilangan *pseudo* acak).

Pembangkit bilangan acak harus (Riani L, 2010:2):

- Berdistribusi *uniform* (0,1) dan tidak berkorelasi antar bilangan
- Membangkitkan secara cepat dan *storage* tidak besar
- Dapat di-"*reproduce*"

Periode besar, karena kemungkinan bilangan acak dibangkitkan berulang.

Bentuk rumus pada *Linear Congruential Generator :*

$$Z_i = (aZ_{i-1} + c) \text{mod } m$$

Dimana:

Z_i = bilangan acak ke-*i* dari deretnya

Z_{i-1} = bilangan acak sebelumnya

a = factor pengali

c = *increment*

m = *modulus*

2.4. Metode *Depht First Search*

2.4.1 Pengertian Pencarian

Komputer melakukan proses penalaran atau berpikir sebaik manusia untuk memecahkan berbagai masalah dengan melakukan pencarian atau penelusuran pengetahuan. Hal tersebut dinamakan kecerdasan buatan atau *Artificial Intelligent* yang termasuk salah satu bagian ilmu dalam komputer.

Hal terpenting dalam menentukan keberhasilan sistem berdasarkan kecerdasan adalah kesuksesan dalam pencarian dan pencocokan. Pada dasarnya ada dua teknik pencarian atau pelacakan yang digunakan yaitu pencarian buta (*blind search*) dan pencarian terbimbing (*heuristic search*) (Kusumadewi, Sri.2003:23).

Beberapa metode pencarian tersebut:

A. Pencarian buta (*blind search*) : tidak ada informasi awal yang digunakan dalam proses pencarian
    1. Pencarian melebar pertama (*Breadth-First Search*)
    2. Pencarian mendalam pertama (*Depth-First Search*)
B. Pencarian terbimbing (*heuristic search*) : adanya informasi awal yang digunakan dalam proses pencarian
    1. Pendakian Bukit (*Hill Climbing*)
    2. Pencarian Terbaik Pertama (*Best First Search*)

2.4.2 Pengertian *Depth First Search*

Depth First Search (pencarian mendalam pertama) ialah suatu proses pencarian yang dilakukan pada semua anaknya sebelum melakukan pencarian ke node-node yang selevel. Pencarian dimulai dari node akar ke level yang lebih tinggi. Proses ini diulangi terus hingga ditemukannya solusi (Kusumadewi, Sri.2003:26)

Depth First Search melakukan penelusuran kaidah secara mendalam dari simpul akar bergerak menurun ke tingkat dalam yang berurutan (Arhami, Muhammad. 2005:20).

Penelusuran (pencarian) *Depth First* berarti setiap kemungkinan path ke goal digali (eksplorasi) ke kesimpulannya sebelum path lainnya di coba. Pencarian *Depth First Search* adalah kemungkinan metode terbaik yang dapat di ikuti dimana *Heuristic* tidak digunakan. (Siswanto.2010:76).

Pada Graph Kasus: misalnya kita ingin pergi dari Jakarta ke Surabaya:

Percobaan Graph 1: Jarak 3000

Jakarta-Solo-Yogyakarta-Surabaya

Percobaan Graph 2: Jarak 5000

Jakarta-Solo-Yogyakarta-Kediri-Surabaya

Percobaan Graph 3: Jarak 2600

Jakarta-Purwokerto-Surabaya

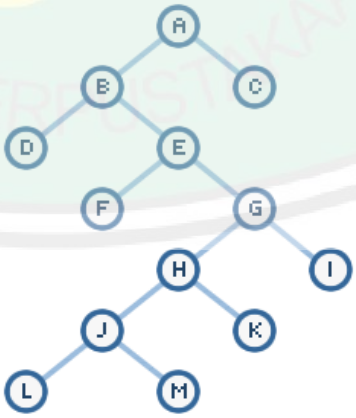
Evaluasi pencarian *Depth First*:

Percobaan 1: Penyelesaian yang baik karena tidak ada *back tracking*

Percobaan 2: Penyelesaian yang buruk

Percobaan 3: Penyelesaian Optimal

2.4.3 Prosedur *Depht First Search*



Gambar 2.2 Pohon *Depth First Search*
(Sumber : Siswanto.2010)

Sebagai contoh jika ingin mencari jalan dari A ke F dengan menggunakan *Depth First Search* adalah sebagai berikut :

1. Pertama dimulai dengan titik A, dengan menggunakan 2 istilah yaitu *open* dan *closed* yang artinya *open* menyatakan titik yang belum dilacak, sedangkan *closed* menyatakan titik yang sudah pernah dilacak.

Open : A

Closed : <kosong>

Gambar 2.3 Titik A
(Sumber : Siswanto.2010)

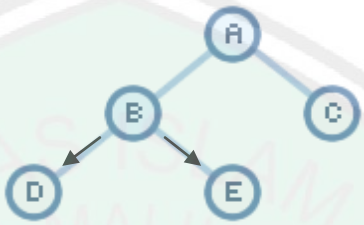
2. Awal penelusuran dengan mengecek titik A, terdapat 2 titik baru yang berhubungan dengan titik A. Titik tersebut adalah B dan C.

Gambar 2.4 *Open* Titik B dan C
(Sumber : Siswanto.2010)

Open : B, C

Closed : A

3. Penelusuran dimulai dengan titik B dan mengecek apakah B adalah F. Jika bukan maka dilakukan penelusuran terhadap titik B. Titik D dan E merupakan titik yang berhubungan dengan B.



Gambar 2.5 *Open* Titik D dan E
(Sumber : Siswanto.2010)

Open : D, E, C

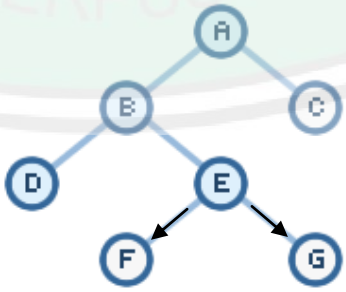
Closed : A, B

4. Kemudian dilakukan pengecekan terdapat titik D dan masih belum menemukan titik F.

Open : E, C

Closed : A, B, D

5. Pelacakan dilanjutkan dengan titik E. Titik E terdapat 2 cabang baru yaitu F dan G.

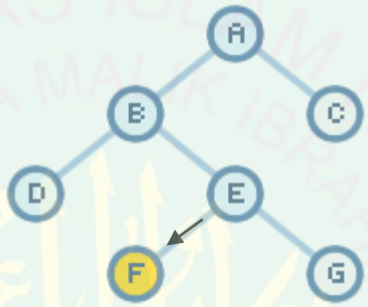


Gambar 2.6 *Open* Titik F dan G
(Sumber : Siswanto.2010)

Open : F, G, C

Closed : A, B, D, E

6. Pada penelusuran berikutnya ternyata terdapat titik F. Maka pencarian juga dihentikan.



Gambar 2.7 Titik F ditemukan
(Sumber : Siswanto.2010)

Open : G, C

Closed : A, B, D, E, F

Teknik ini dapat diimplementasi menggunakan *Tree Search* dengan antrian *last in first out* (LIFO) atau menggunakan fungsi rekursif. *Depth first search* memiliki persyaratan *memory* yang sederhana. Teknik ini hanya perlu menyimpan titik yang dijelajahi dan juga titik yang belum dijelajahi. Setelah sudah dijelajahi maka titik tersebut dihapus dari *memory*. Penelurusan titik pada *tree* tersebut dapat dimulai dari sebelah kanan ataupun sebelah kiri daripada *tree* tersebut. Pada umumnya untuk implementasi pada *mobile robot* digunakan prinsip pengambilan keputusan untuk selalu mengambil jalur atau titik di sebelah kiri ataupun selalu mengambil jalur atau titik di sebelah kanan sampai mencapai titik yang diinginkan.

Jika faktor percabangan dari sebuah *Tree* adalah b dan *Maximum Depth* adalah m, maka *Depth First Search* memerlukan besar penyimpanan b*m+ 1 titik. Kekurangan utama dari *Depth First Search* adalah pengambilan titik yang dapat menyebabkan buntu atau titik yang akan menuju titik yang tidak ada akhirnya sedangkan jika mengambil titik yang lain akan lebih cepat menemukan *goal*.

Pseudocode Depth First Search (Luger, 2002) :

```
function depthsearch (current_state)
begin
if current_state is a goal
then return SUCCESS;
add current_state to closed;
while current_state has unexamined children
begin
child = next unexamined child;
if child not member of closed
    then if depthsearch(child) = SUCCESS
then return SUCCESS
end;
return FAIL
    end
```

2.4.4 Keuntungan *Depth First Search*

Keuntungan metode *Depth first search* adalah:

1. Membutuhkan memori yang relatif kecil, karena hanya node-node pada lintasan yang aktif saja yang disimpan (Kusumadewi, Sri.2003:27).

2. Secara kebetulan (atau jika penanganan diambil dalam ututan state pengganti alternative). *Depth first search* dapat menemukan sebuah solusi tanpa uji coba beberapa penelusuran tempat pada keseluruhannya. Ini berbeda dengan *Breadth First Search*, semua bagian dari tree harus ujicoba pada level n sebelum beberapa node pada level n+1 dapat di uji coba.

   Ini secara khusus pasti beberapa solusi dapat diterima *Depth first search* dapat berhenti bila salah satu dari mereka ditentukan (Siswanto.2010)

2.4.5 Kelemahan *Depth First Search*

Kelemahan metode *Depth first search* adalah : (Kusumadewi, Sri.2003:27)

1. Memungkinkan tidak ditemukannya tujuan yang diharapkan
2. Hanya akan mendapatkan satu solusi pada setiap pencarian.

2.5. Platform Android

Android adalah sebuah sistem operasi untuk perangkat mobile berbasis *linux* yang mencakup sistem operasi, *middleware* dan aplikasi (Safaat, 2001:1). *Android* diakusisi oleh *Google* pada Juli 2005, dan baru dirilis perdana pada 5 November 2007. *Android* berlisensi di bawah GNU, *General Public Lisensi* Versi 2 (GPLv2), yang memperbolehkan pihak ketiga untuk mengembangkannya dengan menyertakan term yang sama. Pendistribusiannya di bawah *Lisensi Apache Software* (ASL/Apache2), yang memungkinkan untuk distribusi kedua dan seterusnya.

Perkembangan *Android* yang cepat telah merilis beberapa versi. *Android* versi 4.0 (ICS: *Ice Cream Sandwich*) merupakan versi terbaru dari *Android* saat ini yang diumumkan pada tanggal 19 Oktober 2011. Versi terbaru ini membawa fitur versi sebelumnya yaitu *Honeycomb* untuk *smart phone* dan menambahkan fitur baru termasuk membuka kunci dengan pengenalan wajah, jaringan data pemantauan penggunaan dan kontrol, terpadu kontak jaringan sosial, perangkat tambahan fotografi, mencari *email* secara *offline*, dan berbagi informasi dengan NFC (Riyadli Abrar, 2012:20-24).

Beberapa keunggulan *Platform Android* adalah sebagai berikut (Safaat, 2001:3):

1) Lengkap (*Complete Platform*). Para desainer dapat melakukan pendekatan yang komprehenshif ketika sedang mengembangkan *Platform Android*. *Android* menyediakan banyak *tools* dalam membangun *software* dan merupakan sistem operasi yang aman.
2) Terbuka (*Open Source Platform*). *Platform Android* disediakan melalui lisensi *Open Source*.
3) Bebas (*Free Platform*). *Android* merupakan *Platform* / aplikasi yang bebas untuk dikembangkan. Tidak ada lisensi atau biaya royalti untuk dikembangkan pada *Platform Android*.

BAB III

ANALISA DAN PERANCANGAN

3.1. Analisa dan Perancangan Sistem

Pada penelitian ini akan dibangun sebuah permainan Sudoku yang berbasis *mufrodat* (kata kerja) pada *Sharaf*. *Sudoku* merupakan permainan atau logika berbasis kombinatorial penempatan angka. Tujuan dari permainan Sudoku adalah menempatkan angka 1 sampai 9 ke dalam kotak-kotak sehingga kolom, baris dan area berisi semua angka 1 sampai 9. Pada penelitian ini, permainan Sudoku yang menggunakan angka akan diganti menggunakan *mufrodat* (kata kerja) untuk ditempatkan dalam kotak-kotak kosong dalam permainan sudoku.Skenario *game* akan dibuat untuk 1 karakter saja karena *game* yang akan dibuat merupakan *gameSudoku* yang hanya mempunyai karakter sebagai pemain (*player*).

Permainan Sudoku ini dikembangkan pada *platform Android* yang merupakan sistem operasi berbasis *mobile*. Hal ini bertujuan untuk mempermudah pemain dalam bermain Sudoku yang dapat dimainkan kapan saja dan dimana saja karena bersifat *mobile*. Permainan sudoku yang merupakan permainan logika terkadang menghabiskan waktu yang lama untuk memecahkan salah satu *puzzle*, sehingga pengembangan permainan sudoku menggunakan sistem yang berbasis *mobile* dapat membantunya.

Dalam permainan ini terdapat unsur pendidikan formal dan informal. Formal dalam artian bahwa pemain dikenalkan dengan perubahan kata kerja

dalam bahasa arab yang digunakan dalam permainan ini, sehingga dibutuhkan pengetahuan tentang kata kerja dan perubahannya, baik dari bentuk dan macam-macamnya. Sedang unsur pendidikan informal yang dimaksud adalah pendidikan untuk mengasah logika pemain dalam menyelesaikan sebuah *puzzle* dalam permainan ini. Penggunaan logika sangat penting untuk menyelesaikan permainan ini.

Untuk membangun game ini menggunakan algoritma *Linear Congruential Generator* sebagai pembangkit *puzzle* game sudoku dan *Depht First Search* Sebagai penyelesaian (*solver*)untuk membuat *puzzle sudoku* yang dinamis pada setiap level permainan Sudoku, sehingga pemain tidak merasa bosan dengan puzzle yang sama.

3.2. Perancangan Sistem Game Sudoku Sharaf

Perancangan adalah langkah awal untuk memetakan dan mengkonsep permainan *sudoku sharaf* yang akan dikembngkan. Ada beberapa perancangan yang akan dilakukan penulis, yakni perancangan menu system, perancangan antar muka system, perancangan level permainan, perancangan alur permainan dan perancangan penerapan algoritma *Linear Congruential Generator* dan algoritma *Depht First Search.*

3.2.1 Perancangan Menu Sistem

Menu adalah serangkain pilihan-pilihan yang dapat diklik untuk melakukan tugas tertentu. Rancangan menu dari game Sharaf yang akan dibuat adalah sebagai berikut:

Tabel 3.1 Rancangan menu *game* Sudoku Sharaf

No	Nama Menu	Keterangan
1	Tasrif	Menu ini bertujuan untuk memberikan gamabaran macam-macam dan perubahan bentuk kata kerja (*mufradat*) dalam bahasa arab
2	Cara Main	Menu ini bertujuan untuk memberikan panduan kepada user menggunakan aplikasi game Sharaf
3	Mainkan	Menu ini adalah menu inti dari aplikasi game Sharaf, menu ini digunakan untuk bermain game Sharaf
4	Keluar	Menu ini digunakan untuk keluar dari game Sharaf

Dari ke-lima menu tersebut, hanya 1 menu yang mempunyai sub menu. Yakni menu mainkan, yang mempunyai sub menu level, *mudah*, *sedang*, *sulit*. Menu ini digunkan untuk menetukan tingkat kesulitan permainan Sudoku.

3.2.2 Perancangan Antar Muka Sistem

Sub menu ini memberikan gambaran rancangan dari tampilan *Game sharaf*, Gambar 3.1 adalah rancangan tampilan awal *Game Sudoku Sharaf*. Pada rancangan tampilan pada gambar 3.1 terdapat 4 menu pilihan. Yakni Mainkan, Cara main, *Tashrif* dan keluar, menu-menu tersebut mempunyai fungsi sebagaimana yang dijelaskan pada table 3.1.

Gambar 3.1 Rancangan Tampilan halaman utama.

Pada menu mainkan akan menghubungkan user ke menu selanjutnya, yakni menu pilih level. Rancangan menu pilih level sebagaimana pada gambar 3.2

Gambar 3.2 Rancangan Tampilan Pilih level

Ada 3 level yang akan dibuat pada Game Sudoku Sharaf, yakni level mudah, sedang dan sulit. Untuk penjelasan dari masing-masing level sebagaimana pada tabel 3.2

Gambar 3.3 Rancangan halaman *game*

Menu pilih level merupakan menu penghubung player ke halaman game, rancangan halaman game Sudoku Sharaf sebagaimana pada gambar 3.3.

3.3.3 Perancangan Level game Sudoku Sharaf

Level pada game ini dirancang menjadi 3 level, yakni level mudah, sedang dan sulit. Perbedaan pada setiap level didasari pada banyaknya kotak kosong yang harus diisi oleh pemian. Semakin tinggi levelnya, maka semakin banyak kotak kosong yang harus diisi. Masing-masing level mempunyai ketentuan sebagai berikut :

a. Level Mudah

Player harus mengisi 36 kotak kosong, dengan masing-masing *grid* berisi 4 kotak kosong yang harus diisi oleh player.

b. Level Sedang

Player harus mengisi 45 kotak kosong, dengan masing-masing *grid* berisi 5 kotak kosong yang harus diisi oleh player.

c. Level Sulit

Player harus mengisi 54 kotak kosong, dengan masing-masing *grid* berisi 6 kotak kosong yang harus diisi oleh player.

Rekapitulasi dalam bentuk table untuk level permainan game *Sudoku Sharaf*, sebagaimana pada table 3.2

Table 3.2 Level permainan game Sharaf Sudoku

No	Nama Level	Jumlah Kotak Kosong
1	Mudah	36
2	Sedang	45
3	Sulit	54

Untuk level mudah, kotak kosong yang harus diisi *shighah* ada 27. Pada level sedang ada 36 kotak kosong, sedangkan pada level sulit terdapat 45 kotak kosong yang harus diisi.

Selain pengaturan jumlah kotak kosong padah setiap level, permainan ini juga mengharuskan pemain untuk memilih *wazn* yang akan dimainkan. Dengan memilih *wazn* terlebih dahulu, maka shighah yang akan dipilih untuk mengisi kotak kosong juga akan disesuaikan berdasarkan *wazn*-nya.

Tabel 3.3 *Tashrif* 6 *Wazn* dari *Tsulasi Mujarrad*
Karya *Assyaikh* Muhammad Ma'shum bin Ali

رقم	وزن	التصريفالاصطلاحى
١	فَعَلَ- يَفْعُلُ	فَعَلَ- يَفْعُلُ- فَعْلاً - مَفْعَلاً - فَاعِلٌ - مَفْعُوْلٌ - اُفْعُلْ - لاَتَفْعُلْ- مَفْعَلٌ- مَفْعَلٌ- مِفْعَلٌ
٢	فَعَلَ- يَفْعِلُ	فَعَلَ- يَفْعِلُ- فَعْلاً - مَفْعَلاً - فَاعِلٌ - مَفْعُوْلٌ - اِفْعِلْ - لاَتَفْعِلْ- مَفْعِلٌ- مَفْعِلٌ- مِفْعَلٌ
٣	فَعَلَ- يَفْعَلُ	فَعَلَ- يَفْعَلُ- فَعْلاً - مَفْعَلاً - فَاعِلٌ - مَفْعُوْلٌ - اِفْعَلْ - لاَتَفْعَلْ- مَفْعَلٌ- مَفْعَلٌ- مِفْعَالٌ
٤	فَعِلَ- يَفْعَلُ	فَعِلَ- يَفْعَلُ- فَعْلاً -مَفْعَلاً - فَاعِلٌ - مَفْعُوْلٌ - اِفْعَلْ - لاَتَفْعَلْ- مَفْعَلٌ- مَفْعَلٌ
٥	فَعُلَ- يَفْعُلُ	فَعُلَ- يَفْعُلُ- فُعْلاً -مَفْعَلاً - فَعَلٌ - اُفْعُلْ - لاَتَفْعُلْ- مَفْعَلٌ- مَفْعَلٌ
٦	فَعِلَ- يَفْعِلُ	فَعِلَ- يَفْعِلُ- فُعْلاَنًا -مَفْعَلاً - فَاعِلٌ - مَفْعُوْلٌ - اِفْعِلْ - لاَتَفْعِلْ- مَفْعِلٌ- مَفْعِلٌ

Pada tabel 3.3 adalah wazn dari *fi'il tsulasi mujarrad* yang berjumlah 6 *wazn*. *fi'il tsulasi mujarrad* memiliki 10 *shighah* dan 9 *shighah* pada *wazn*فَعُلَ- يَفْعُلُ. Masing-masing *shighah* akan memberikan arti yang berbeda sesuai bentuk *shighat*nya. Secara berurutan dari kanan ke kiri bentuk-bentuk nama *shighah* dan penjelasanya adalah sebagai berikut.

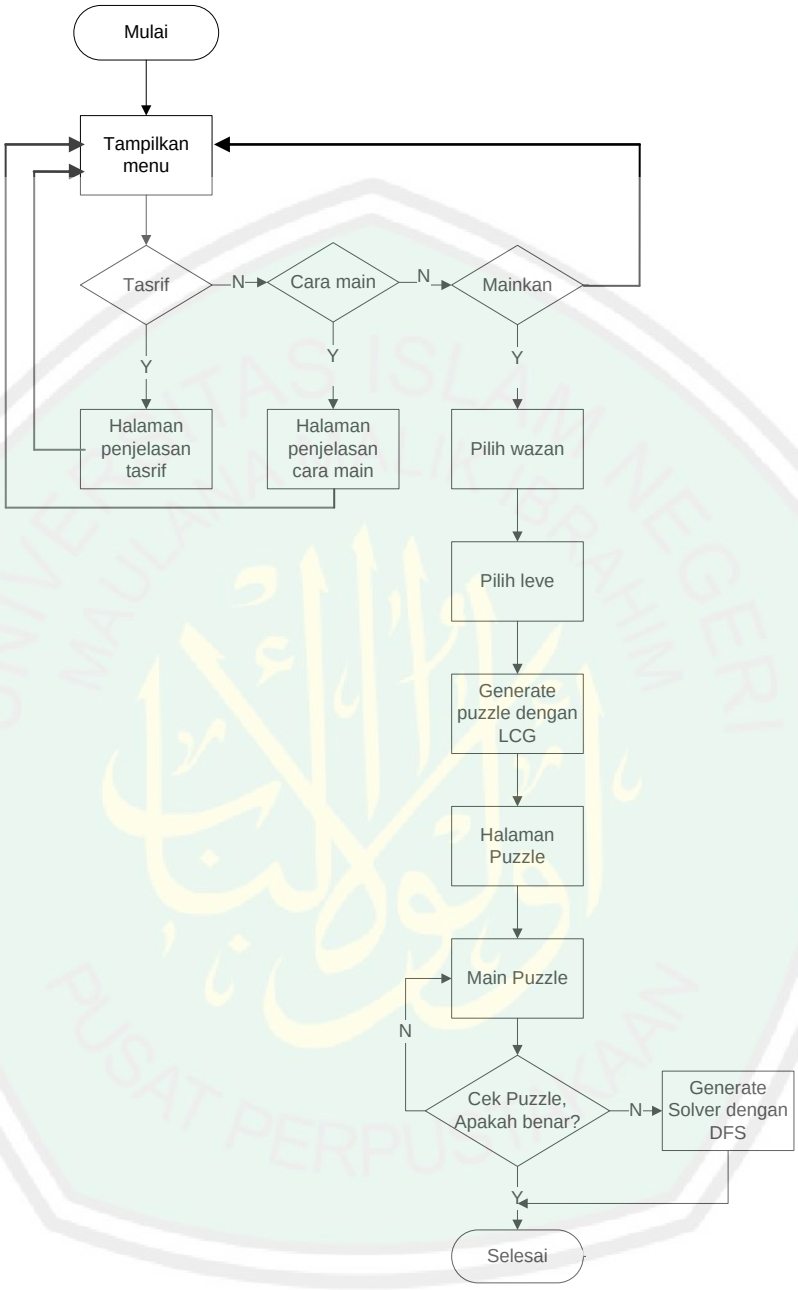
1. *SighatMasdhar*, yaitu bentuk kata dasar
2. *Mashdar* Mim, yaitu mashdar yang mendapat tambahan mim
3. *IsimFa'il*, yaitu kata benda yang menunjukkan pengertian pelaku
4. *Shighat Mutasyabihah biismil Fa'il*, yaitu kata sifat yang disamakan dengan *isim fa'il*
5. *Sighat Mubalaghah*, yaitu bentuk penyangatan
6. *Isim Tafdhil*, yaitu kata benda yang mengandung pengertian lebih
7. *Isim Maf'ul*, yaitu kata benda yang menunjukkan pengertian penderita
8. *Isim Makan*, yaitu kata benda yangmenunjukkan pengertian tempat
9. *Isim Zaman*, yaitu kata benda yang menunjukkan pengertian waktu

10. *Isim Alat*, yaitu kata benda yang menunjukkan pengertian alat.

Pada *wazn* فَعِلَ- يَفْعَلُ,فَعُلَ- يَفْعُلُفَعِلَ- يَفْعِلُmemiliki *shighah* isim *makan* dan isim *zaman* yang sama. Pada *wazn* فَعِلَ- يَفْعِلُ dan فَعِلَ- يَفْعَلُpermainan Sudoku bisa dijalankan karena *shighah* yang berbeda masih berjumlah 9, jadi *user* tidak perlu memilih *shighah*. System hanya menampilkan dari *shighah* yang akan dimainkan, sedangkan pada *wazn* فَعُلَ- يَفْعُلُyang hanya memiliki 9 *shighah* dan mempunyai kesamaan pada *isim zaman* dan *isim makan*, maka *shighah* yang berbeda pada *wazn* ini kurang 1. Supaya permainan tetap sesuai dengan kaidah *Sudoku* bahwa antar baris, antar kolom, antar area tidak boleh ada *shighah* yang sama. Maka, peneliti menggunakan garis bawah berwarna pada *shighah isim makan* dan *isim zaman* untuk membedakannya.

3.3. Perancangan Alur Sistem

Sub bab ini menjelaskan perancangan alur sistem yang digambarkan dengan menggunakan *flow chart*. Halaman utama *game* Sudoku Sharaf berupa menu yang harus dipilih salah satu oleh *user*.



Gambar 3.4 *Flowchart* Game Sudoku Sharaf

Jika user memilih menu "tashrif" maka akan muncul halaman penjelasan tentang tasrif, jika user memilih menu "cara main" maka akan muncul halaman penjelasan cara bermain game sudoku Sharaf. Jika user meilih menu "keluar", maka akan kelaur dari game sudoku Sharaf. Jika user meilih menu

"mainkan", maka akan muncul menu "pilih wazan", kemudian muncul menu "pilih level" yang akan menentukan sulitnya permainan.

Setelah itu, system akan menggenerate isi *puzzle* dengan algoritma *Linear Congruential Generator* (LCG) dan mengosongkan kotak puzzle sesuai dengan level permainan yang telah user pilih. Jika, user telah mengisi semua *puzzle* yang kosong user bisa mengecek kebenaran isian *puzzle*. Jika, semua isian *puzzle* benar permainan selesai. Jika, jawaban *user* masih ada yang salah, user harus bermain lagi dan mengecek kebenaran *puzzle* secara berulang-ulang. Jika *user* bosan dengan permainan karena terus gagal atau salah mengisikan *puzzle* yang kosong, maka user bisa menggunakan fasilitas *solver*. Yakni dengan menggunakan algoritma *Depth Firs Search (DFS)*.

3.4. Perancangan Algoritma

Pada penelitian yang akan dilakukan menggunakan 2 algoritma, yakni algoritma *Linear Congruential Generator* (LCG) dan *Depht First Search (DFS)*.

3.5.1 Perancangan Algoritma *Linear Congruential Generator (LCG)*

Dalam permainan sudoku, pembangkit bilangan secara acak yang digunakan untuk menampilkan soal sangatlah penting untuk menentukan tingkat kesulitan permainan tesebut. Oleh karena itu dibutuhkan sebuah *generator* (pembangkit) yang mampu membangkitkan bilangan acak.Menurut Riani L. (2010:3), bilangan acak adalah bilangan yang tidak

dapat diprediksi kemunculannya. bilangan acak diperoleh dengan cara membentuk bilangan acak secara numerik/ aritmatik (menggunakan komputer), dan disebut "*Pseudo Random Number*".

Rumus *Linear Congruential Generator* :

$$Z_i = (aZ_{i-1} + c) \bmod m$$

Dimana:

Z_i = bilangan acak ke-*i* dari deretnya

Z_{i-1} = bilangan acak sebelumnya

a = factor pengali

c = *increment*

m = *modulus*

Kunci pembangkit adalah Z_0 yang disebut umpan**.**Untuk menghasilkan output dari algoritma tersebut, maka pemberian nilai variabel a, c, m dan Z_0 yang merupakan umpan dari algoritma *Linear Congruential Generator* harus dilakukan secara *trial and error*. Penulis telah mencoba beberapa kali pemberian nilai pada variabel a, c, m, Z_0dan akhirnya penulis mendapatkan angka-angka yang cocok. Berikut ini hasil dari uji coba:

Input:

a= 7, c=6, m=11 dan $Z_{0=}$ 1

$$Z_i = (aZ_{i-1} + c) \bmod m$$

$$1 = (7.1_{1-1} + 6) \bmod 11$$

Output= 2

Pada penelitian yang akan dilakukan, Algoritma *Linear Congruential Generator (LCG)* dgunakan untuk:

a. Mengisi semua kotak kosong dengan *shighah*menggunakan algoritma *Linear Congruential Generator*.

Pada proses ini, output dari LCG hanya berjumlah 1 angka yang digunakan untuk menentukan secara *random* 10 soal yang telah dibuat oleh peneliti. Soal tersebut disimpan di dalam sebuah system dalam bentuk *array list*, masing-masing *array list* memiliki 81 indeks yang berisi bilangan integer 0-80. Jumlah indeks tersebut sesuai dengan jumlah kotak Sudoku.

```
String mudah []=
{"862459371914673852573218469638597124497126538
125834796783916245249785361651342987"
"452871369963425187178693452248713596759246318 3
619857246271849355316928748495372 16"
"976485321385721649241693875293517648417836592 5
689241378527691349741532683164827 59"
"572391648936248751841675923265934817874512369 1
397862541594267834271836953685974 12"
"961247853372586491485193672632498715149257863 7
583162491293845767386159245649278 31"
"671842953439165872852793614764329518258741693 1
395682474972851365863149273219764 85"
"159423768382657194647198352271345986569821473 8
349762155376948122167389454895217 63"
"397586421852431976461972538164935278587624193 2
938176548497536123152687497261493 85"
"234615798976382145158497623453972186867451293 9
123865745618293477386145292497358 61"
"534691872786423195921587346963458217247961358 8
157326943261897455196748324782531 69"};
```

Gambar 3.5 Soal *Sudoku* level mudah pada bentuk *array*

LCG akan melakukan random *output*untuk menentukan soal yang akan ditampilkan, sebagaimana perhitungan berikut:

Input:

a= 7, c=6, m=11 dan $Z_{0=}$ 1

$$Z_i = (aZ_{i-1} + c) \text{mod } m$$

$$1 = (7.0 + 6) \text{mod } 11$$

Output = 2

Maka system akan menampilkan soal nomer 2, dan sistem akan membaca isi dari *array list* soal nomer 2 dan mengisi penuh kotak Sudoku sesuai dengan *inisialisasi* soal nomer 2 pada area 3x3 dan ditunjukkan pada contoh 3.6

فاعل	فعلا	مفعلا	مفعول	أفعل	لاتفعل	مفعل	فعل	يفعل
مفعول	أفعل	يفعل	فعل	مفعل	فاعل	فعلا	مفعلا	لاتفعل
فعل	مفعل	لاتفعل	فعلا	مفعلا	يفعل	فاعل	مفعول	أفعل
لاتفعل	فاعل	مفعل	أفعل	مفعول	فعل	مفعلا	يفعل	فعلا
مفعلا	يفعل	مفعول	لاتفعل	فاعل	فعلا	أفعل	مفعل	فعل
أفعل	فعل	فعلا	مفعل	يفعل	مفعل	لاتفعل	فاعل	مفعول
مفعل	مفعول	فعل	فاعل	فعلا	أفعل	يفعل	لاتفعل	مفعلا
يفعل	لاتفعل	أفعل	مفعول	فعل	مفعل	مفعول	فعلا	فاعل
فعلا	مفعلا	فاعل	يفعل	لاتفعل	مفعول	فعل	أفعل	مفعل

Gambar 3.6 Contoh *Puzzle* setelah Terisi.

Proses pada pengisian *puzzle* kosong pada soal sudoku dilakukan perulangan untuk mengisi array kotak dengan output LCG untuk mengisi *puzzle*. Jika dijelaskan dengan *flowchart* sebagaimana gambar 3.7

Gambar 3.7 *Flowchart* Pengisian *Puzzle* Kosong

b. Menghilangkan beberapa kotak *puzzle* berdasarkan level permainan.

Semua kotak yang telah terisi *shighah* sebagaimana pada gambar 3.6 akan dihilangkan beberapa isinya dengan menggunakan algoritma *Linear Congruential Generator*.Penghilangan kotak soal dilakukan per area 3x3. Output LCG ditentukan sesuai dengan level permainan. untuk level mudah kotak soal per area 3x3 sebanyak 4 atau 36 angka, level sulit 5 kotak atau 45 angka dan level sulit 6 kotak soal atau 54 angka output algoritma *Linear Congruential Generator* .Penulis akan melanjutkan contoh penghilangan kotak soal dengan melanjutkan contoh penggunaan LCG pada tahap sebelumnya (tahap pengacakan soal). Untuk memudahkan pemahaman penulis gambarkan pengindeksan Sudoku sebagaimana tabel 3.4.

Tabel 3.4 Pengindeksan sudoku

1.1	2.1	3.1	4.1	5.1	6.1	7.1	8.1	9.1
1.2	2.2	3.2	4.2	5.2	6.2	7.2	8.2	9.2
1.3	2.3	3.3	4.3	5.3	6.3	7.3	8.3	9.3
1.4	2.4	3.4	4.4	5.4	6.4	7.4	8.4	9.4
1.5	2.5	3.5	4.5	5.5	6.5	7.5	8.5	9.5
1.6	1.6	3.6	4.6	5.6	6.6	7.6	8.6	9.6
1.7	2.7	3.7	4.7	5.7	6.7	7.7	8.7	9.7
1.8	2.8	3.8	4.8	5.8	6.8	7.8	8.8	9.8
1.9	2.9	3.9	4.9	5.9	6.9	7.9	8.9	9.9

Algoritma LCG akan melakukan random untuk menghasilkan output 36 angka. Perhitungan manual random algoritma LCG sebagaimana pada tabel 3.5 berikut :

Tabel 3.5 Perhitungan Manual Algoritma LCG pada level Mudah

Random Ke	$Z_i = (aZ_{i-1} + c) \bmod m$	Hasil Random	Indeks Pada Sudoku	Shighah yang dihilangkan
1	(7.1+6) mod 11	2	2.1	فعلا
2	(7.2-1 +6) mod 11	8	2.3	مفعل
3	(7.8-1 +6) mod 11	6	3.2	يفعل
4	(7.6-1+6)mod 11	3	3.1	مفعلا
5	(7.3-1 +6) mod 11	4	4.2	فعل
6	(7.4-1 +6) mod 11	0	-	
7	(7.0-1 +6) mod 11	5	5.2	مفعل
8	(7.5-1 +6) mod 11	7	4.3	فعلا
9	(7.7-1 +6) mod 11	10	4.1	مفعول
10	(7.10-1 +6) mod 11	9	9.3	أفعل
11	(7.9-1 +6) mod 11	2	8.1	فعل
12	(7.2-1 +6) mod 11	8	8.3	مفعول
13	(7.8-1 +6) mod 11	6	9.2	لاتفعل
14	(7.6-1+6)mod 11	3	3.4	مفعل
15	(7.3-1 +6) mod 11	4	1.5	مفعلا
16	(7.4-1 +6) mod 11	0	-	
17	(7.0-1 +6) mod 11	5	2.5	يفعل
18	(7.5-1 +6) mod 11	7	2.6	فعل
19	(7.7-1 +6) mod 11	10	4.4	أفعل
20	(7.10-1 +6) mod 11	9	6.6	مفعل
21	(7.9-1 +6) mod 11	2	5.4	مفعول
22	(7.2-1 +6) mod 11	8	5.6	يفعل
23	(7.8-1 +6) mod 11	6	9.5	فعل
24	(7.6-1+6)mod 11	3	9.4	فعلا
25	(7.3-1 +6) mod 11	4	7.5	أفعل
26	(7.4-1 +6) mod 11	0	-	
27	(7.0-1 +6) mod 11	5	8.5	مفعل
28	(7.5-1 +6) mod 11	7	1.9	فعلا
29	(7.7-1 +6) mod 11	10	1.7	مفعل
30	(7.10-1 +6) mod 11	9	3.9	فا عل
31	(7.9-1 +6) mod 11	2	2.7	مفعول
32	(7.2-1 +6) mod 11	8	5.9	يفعل

33	(7.8-1 +6) mod 11	6	6.8	مفعل
34	(7.6-1+6)mod 11	3	6.7	أفعل
35	(7.3-1 +6) mod 11	4	4.8	مفعول
36	(7.4-1 +6) mod 11	0	-	
37	(7.0-1 +6) mod 11	5	8.8	فعلا
38	(7.5-1 +6) mod 11	7	7.9	فعل
39	(7.7-1 +6) mod 11	10	7.7	يفعل
40	(7.10-1 +6) mod 11	9	9.9	مفعل

Proses penghilangan kota puzzle Sudoku dilakukan per-area 3x3 dimulai dari ujung atas sebelah kiri. Dari proses algoritma *Linear Congruential Generator* menghasilkan output 4 pertama berupa bilangan2,8,6,3. Angka tersebut menjadi acuan penghilangan indeks kotak soal pada areaa 3x3 yang pertama. Kemudian angka tersebut disimpan pada *arraylist* dan *disorting* dari urutan yang terkecil. Tujuanya untuk memudahkan proses penghilangan beberapa kotak yang akan menjadi soal. Maka akan menjadi 2,3,6 dan 8.

Telah kita ketahui pada soal nomer 2 bahwa isi kotak 3x3 yang pertama adalah 4,5,2,8,7,1,3,6,9. Maka system akan merubah angka pada indeks 2,3,6 dan 8 dengan angka 0 sehingga menjadi 4,0,0,8,7,0,3,0,9.

\

فاعل	X	X	مفعول	أفعل	لاتفعل	مفعل	فعل	يفعل
مفعول	أفعل	X	فعل	مفعل	فاعل	فعلا	مفعلا	لاتفعل
فعل	X	لاتفعل	فعلا	مفعلا	يفعل	فاعل	مفعول	أفعل
لاتفعل	فاعل	مفعل	أفعل	مفعول	فعل	مفعلا	يفعل	فعلا
مفعلا	يفعل	مفعول	لاتفعل	فاعل	فعلا	أفعل	مفعل	فعل
أفعل	فعل	فعلا	مفعل	يفعل	مفعل	لاتفعل	فاعل	مفعول
مفعل	مفعول	فعل	فاعل	فعلا	أفعل	يفعل	لاتفعل	مفعلا
يفعل	لاتفعل	أفعل	مفعول	فعل	مفعل	مفعول	فعلا	فاعل
فعلا	مفعلا	فاعل	يفعل	لاتفعل	مفعول	فعل	أفعل	مفعل

Gambar 3.8 Penghilangan Kotak Sudoku pada area 3x3

Proses ini penghilangan kotak *puzzle* sebagaimana pada gambar 3.7 dilakukan per-area kotak Sudoku dan diulang-ulang sebanyak 9 kali. Inisialisasi letak kotak Sudoku sebagaimana gambar 3.8.

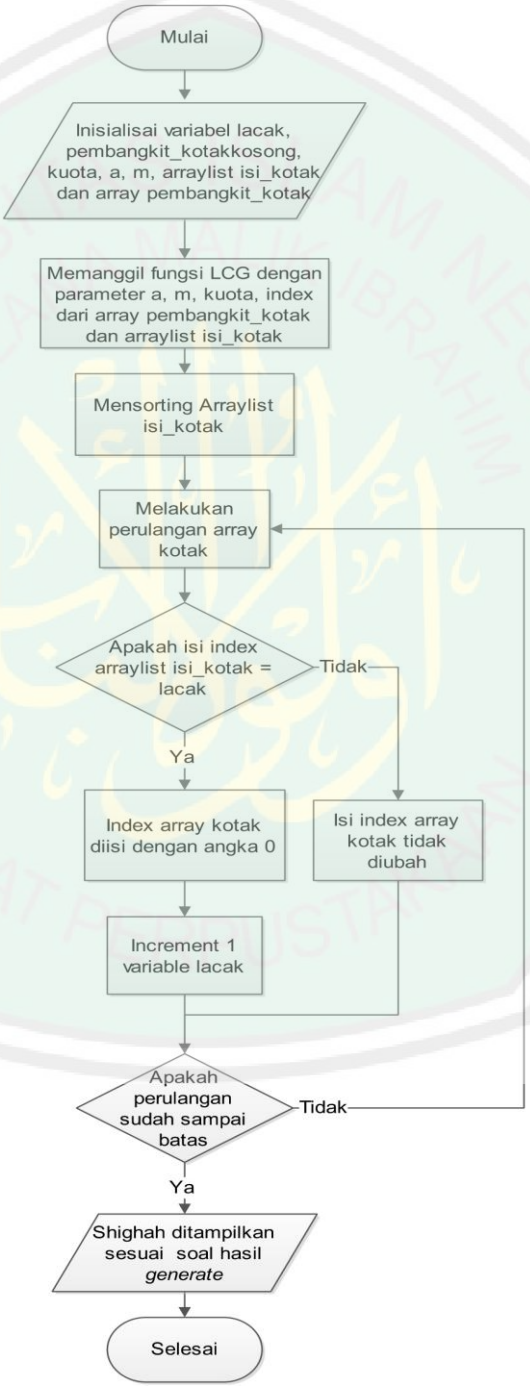
1	2	3
4	5	6
7	8	9

Gambar 3.9 Inisialisasi kotak Sudoku per area 3x3

Proses penghilangan kotak *puzzle* Sudoku untuk dijadikan soal terangkum sebagaimana pada gambar 3.9.

Dalam proses penginisialisasian variabel acak dalam pembangkit kotak kosong rumusan a, c, m dimasukkan pada *array list* untuk mengisi kotak dan adapun array sebagai pembangkit kotak pada soal sudoku. Sehingga memanggil fungsi LCG dengan parameter a, m, kuota, index dari array pembangkit kotak soal dan

array list untuk mengisi kotak soal sudoku secara disorting dan melakukan perulangan array pada kotak soal sudoku. Proses untuk menentukan *puzzle* soal dijelaskan pada gambar 3.1



Gambar 3.10 *Flowchart* Menentukan *Puzzle* Soal

3.5.2 Perancangan Algoritma *Depht First Search (DFS)*

Pertama-tama akan diberikan indeks pada setiap kotak pada sudoku agar mudah untuk menunjuk posisi kotak tertentu. Pengindeks-an kotak yang lengkap ditunjukkan pada tabel 3.6.

Tabel 3.6 Pengindeks-an sudoku

1.1	2.1	3.1	4.1	5.1	6.1	7.1	8.1	9.1
1.2	2.2	3.2	4.2	5.2	6.2	7.2	8.2	9.2
1.3	2.3	3.3	4.3	5.3	6.3	7.3	8.3	9.3
1.4	2.4	3.4	4.4	5.4	6.4	7.4	8.4	9.4
1.5	2.5	3.5	4.5	5.5	6.5	7.5	8.5	9.5
1.6	1.6	3.6	4.6	5.6	6.6	7.6	8.6	9.6
1.7	2.7	3.7	4.7	5.7	6.7	7.7	8.7	9.7
1.8	2.8	3.8	4.8	5.8	6.8	7.8	8.8	9.8
1.9	2.9	3.9	4.9	5.9	6.9	7.9	8.9	9.9

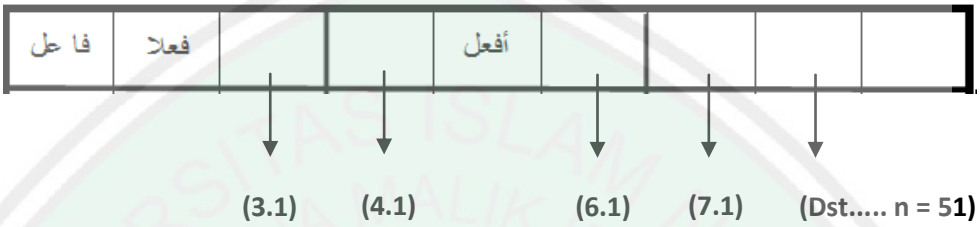
Angka yang ada di dalam matriks merupakan indeks posisi kotak, indeks dimulai dari ujung kiri-atas yaitu (1,1) sampai ujung kanan-bawah yaitu (9,9). Hal ini akan bermanfaat untuk proses implementasi selanjutnya.

Contoh kasus sudoku Sharaf pada tashrif istilahi dan menyelesaikan dengan menggunakan pohon status DFS dan *backtracking*. Misalkan diberikan kasus sudoku seperti gambar 3.11.

				أفعل			فعلا	فا عل
			فعل	مفعل	فا عل			مفعول
	مفعول						مفعل	لاتفعل
فعلا				مفعول				لاتفعل
فعل			لاتفعل		فعلا			مفعلا
مفعول				يفعل				أفعل
	لاتفعل	يفعل					مفعول	
فا عل			مفعلا	فعل	مفعل			
مفعل	أفعل			لاتفعل				

Gambar 3.11 Desain contoh kasus *Sudoku Sharaf* pada *Tashrif Istilahi* (perubahan kata yang paling dasar)

Langkah pertama adalah kita akan melakukan pencarian solusi pada kotak paling kiri-atas yang kosong, dalam hal ini adalah kotak (3.1). Selanjutnya kotak (4.1) , (6.1), dan seterusnya.



Gambar 3.12 Penjelasan dalam pengindeks-an

Pada contoh kasus diatas ada beberapa langkah dalam peyelesaian pada kolom yang kosong sebagai soal dalam permainan *sudoku Sharaf* dengan aturan permainan tersebut diantaranya adalah :

Langkah 1 : Pada kolom (3.1) yang memenuhi fungsi kelayakan adalah

1. فعل
2. يفعل
3. مفعلا

Langkah ٢ : Pada kolom (٤.1) yang memenuhi fungsi kelayakan adalah

1. يفعل
2. مفعول
3. مفعل

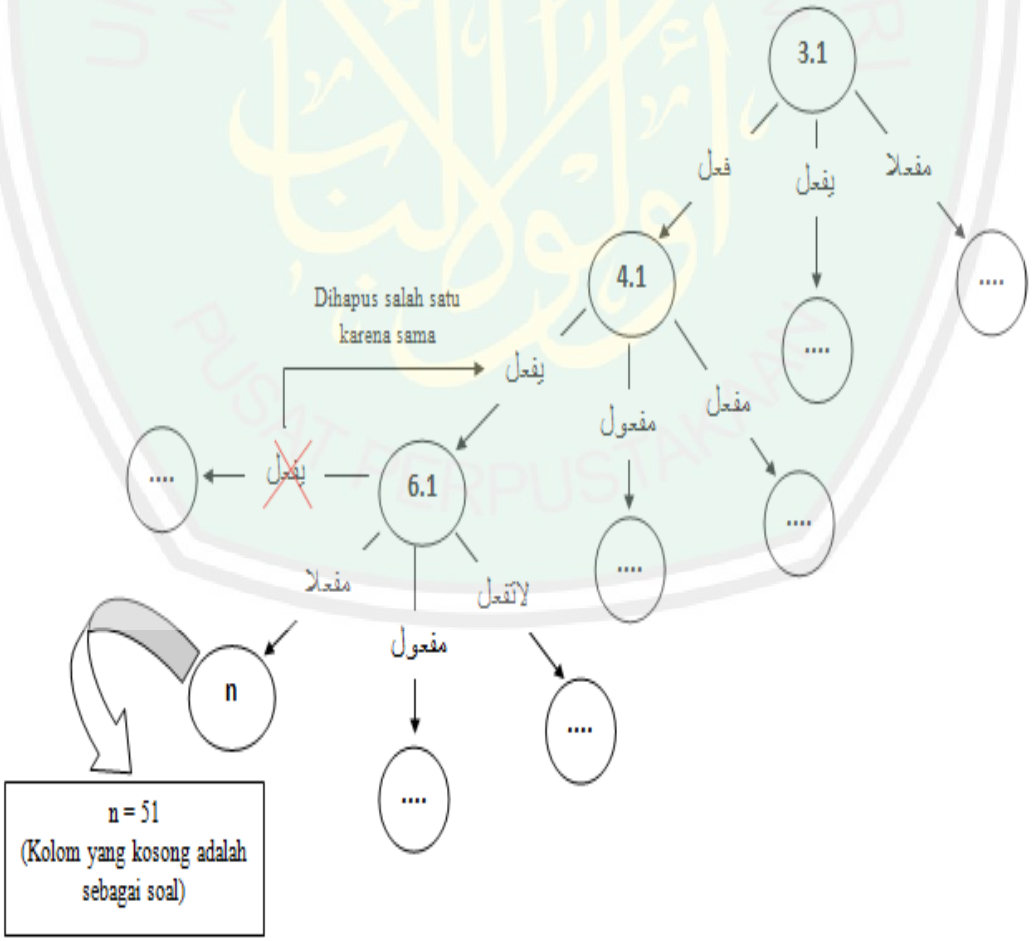
Langkah ٣ : Pada kolom (٦.1) yang memenuhi fungsi kelayakan adalah

1. يفعل
2. مفعلا
3. مفعول

4. لاتفعل

Langkah 4 dan seterusnya dilakukan sama dengan langkah sebelumnya mencari fungsi kelayakan sebagai jawaban pada kolom yang kosong yang dijadikan soal dalam permaian sudoku tersebut.

Adapun implementasi langkah diatas pada pohon DFS yang digunakan untuk memenuhi fungsi kelayakan pada kolom yang kosong dan menggambarkan tentang proses fungsi kelayakan dalam penyelesaian permainan sudoku ditunjukkan pada gambar 3.13.



Gambar 3.13 Proses Pohon DFS dalam penyelesaian permainan

3.5. Kebutuhan Sistem

Pada bagian kebutuhan sistem ini diulas tentang kebutuhan sistem perangkat keras maupun lunak yang mendukung dalam pembuatan dan uji coba aplikasi *Sudoku Sharaf*.

a. Kebutuhan Perangkat Keras (*hardware*)

   Perangkat keras yang dibutuhkan untuk mendukung proses pengembangan sistem aplikasi Sudoku Sharaf adalah:

   1) PC / Laptop dengan spesifikasi minimal : Processor Core2Duo 2.00 GHz dan RAM 1 GB, digunakan untuk pembuatan aplikasi.
   2) HandPhone / Perangkat Mobile yang berbasis Android, dibutuhkan untuk melakukan uji coba.

b. Kebutuhan Perangkat Lunak

   Beberapa perangkat lunak yang dibutuhkan untuk mendukung proses pengembangan sistem aplikasi Sudoku Sharaf adalah:

   1) Android SDK (*Software Development Kit*), yang diperlukan sebagai alat bantu dan API dalam mengembangkan aplikasi Android menggunakan bsahasa java.
   2) Java, digunakan untuk dapat melakukan kompilasi aplikasi Android. Versi yang digunakan Sun Java SE versi 1.5 atau 1.6 atau versi di atasnya.
   3) Software Eclipse. Merupakan software yang dibutuhkan untuk melakukan coding aplikasi Android. Eclipse yang digunakan

adalah versi 3.5 (*Eclipse Galileo*) yang support dengan *Android Development Tools* (ADT).

3.6. Perancangan Uji Coba Game Sudoku Sharaf

3.6.1 Pengujian keberhasilan dari algoritma *solver*

Pengujian kebenaran algoritma *solver* bertujuan untuk memastikan bahwasanya solusi pemecahan soal yang diberikan oleh algoritma Depth First search sesuai dengan ketentuan permainan Sudoku. Yakni, dalam satu grid, kolom dan baris tidak boleh ada *shighat* yang sama.

Pengujian dilakukan sebanyak 36 kali. Untuk menghitung keberhasilan dan kegagalan dari algoritma solver adalah sebagai berikut :

$$\text{Prosentase keberhasilan} = \frac{Jumlah\ Keberhasilan}{Banyaknya\ Uji\ Coba} \text{ x } 100\ \%$$

$$\text{Prosentase kegagalan} = \frac{Jumlah\ Kegagalan}{Banyaknya\ Uji\ Coba} \text{ x } 100\ \%$$

3.6.2 Pengujian keberhasilan algoritma solver dengan penelitian lain yang berkaitan

Pengujian ini bertujuan untuk melihat *performance* algoritma *Depth First Search* dibandingkan dengan algoritma lain. Algoritma lain yang dibandingkan merujuk pada beberapa penelitian yang dilakukan oleh mahasiswa Universitas Islam Negeri (UIN) Maulana Malik Ibrahim Malang dengan topik yang sama akan tetapi dengan menggunakan algoritma yang berbeda. Penelitian tersebut adalah Penelitian yang dilakukan oleh Ahmad Baihaqi (2013) dengan menggunakan algoritma

solver *Branch And Bound.*Pengujian dilakukan dengan membandingkan waktu dari masing-masing algoritma dalam menyelesaikan *puzzle* game sudoku yang kosong pada 3 level yang berbeda.

3.6.3 Pengujian User

Pengujian ini dilakukan untuk mengetahui kegunaan dan keberhasilan *game* Sudoku Sharaf untuk pembelajaran gramatikal (*sharaf)* bahasa arab. Pengujian dilakukan dengan memberikan angket (pertanyaan) kepada siswa-siswi di MAN Kota Blitar pada Jurusan Agama dan IPA kelas XII sebagai pembelajaran bahasa Arab khususnya pada Ilmu Sharaf. Dan adapun juga sebagai perbandingan pemahaman antara kelas Agama yang dimana mayoritas siswa/i lebih banyak mengetahui tetang Ilmu Sharaf dan kelas IPA siswa/i-nya lebih sedikit yang memahami Ilmu Sharaf tersebut.

Isi dari angket tersebut untuk mengetahui keberhasilan program tersebut dari sisi *user*, yakni terkait tampilan *game* sharaf (*user interface*), kemudahan penggunan (*user friendly*) dan kegunaan bagi user (*usability*).

BAB IV

HASIL DAN PEMBAHASAN

4.1 Implementasi

Game Sudoku Sharaf dibangun dengan menggunakan IDE *eclipse Android Developer Tools* versi 21.0.1.201212060302 yang bersifat *open source*. pada IDE *eclipse* versi ini, sudah ter*package* semua kebutuhan untuk membangun *android apps*, sehingga penulis langsung bisa melakukan *development game sudoku sharaf* dan melihat hasilnya dengan cara melakukan *running*pada *emulator* android.

4.2 Hasil Implementasi

Pada bab III sudah dijelaskan halaman utama *game sudoku sharaf* akan memiliki 4 empat menu utama sesuai fungsi masing-masing, yakni menu *mainkan, sharaf, tutorial* dan keluar. Menu *mainkan* adalah menu inti pada aplikasi permainan *sudoku sharaf*, karena menu *mainkan* adalah menu digunakan untuk memulai permainan. Sedangkan menu *sharaf* dan tutorial adalah menu pelengkap sebagai *knowledge base* bagi *player* yang belum pernah mengenal ilmu *sharaf* dan belum pernah memainkan permainan *sudoku sharaf*. Gambar 4.1 adalah gambar halaman utama permainan *sudoku sharaf*.

Gambar 4.1 Halaman Utama

Untuk mengetahui sekilas tentang ilmu *Sharaf*, pengguna dapat menekan tombol *Sharaf*. Dalam halaman *Sharaf* akan dijelaskan definisi dari ilmu *Sharaf* dan pembagian *wazn* dari *fi'il tsulatsi mujarrad* yang terdiri dari 6 *wazn*. Sedangkan menu keluar digunakan jika pengguna tidak ingin memainkan *gameSudokusharaf*.

Untuk memulai *game Sudoku Sharaf* yang menjadi inti dari aplikasi ini, pengguna dapat menekan *button* mainkan dan akan tampil halaman pilih *level*. Pada halaman pilih level pengguna diharuskan memilih level dari 3 level yang telah disediakan, yakni level mudah, sedang dan sulit. Perbedaan dari ketiga level tersebut sebagaimana pada perancangan pada bab III, yakni sebagai berikut:

a. Level Mudah,Player harus mengisi 36 kotak kosong, dengan masing-masing *grid* berisi 4 kotak kosong yang harus diisi oleh *player*.
b. Level Sedang, Player harus mengisi 45 kotak kosong, dengan masing-masing *grid* berisi 5 kotak kosong yang harus diisi oleh *player*.

c. Level Sulit, Player harus mengisi 54 kotak kosong, dengan masing-masing *grid* berisi 6 kotak kosong yang harus diisi oleh *player*.

Gambar 4.2 Pilih level

Setelah tampil halaman pilih level, akan tampil halaman pilih *wazn*. Pada halaman ini pengguna diharuskan memilih 1 *wazn* dari 6 *wazn* yang ditampilkan.*Wazn* dalam ilmu *sharaf* berarti patokan yang harus diikuti oleh kata kerja (*fi'il*) lain. Sedangkan kata fi'il yang bentuknya mengikuti *wazn*disebut *mauzun.Wazn* yang digunakan dalam game *Sudoku sharaf* ini adalah*wazn* dari *fi'il tsulatsi mujarrad* yang terdiri dari 6 *wazn.* Halaman pilih *wazn* sebagaimana pada gambar 4.3

Gambar 4.3 Pilih *Wazn*

Setelah memilih *wazn*, pengguna akan memilih 1 *fi'il* dari 2 *fi'il* yang ditampilkan. Fi'il pada langkah ini bisa diartikan sebagai *mauzun*, yakni kata kerja yang akan mengikuti *wazn* yang dipilih pada langkah sebelumnya. Gambar 4.3 adalah halaman pilih *fi'il*.

Gambar 4.4 Halaman Pilih Fi'il

Setelah itu *player* diharuskan memilih *shighah*, *shighah* bisa diartikan sebagai bentuk dari isim, dalam ilmu bahasa arab biasanya setiap *fi'il* memiliki 10, 9 atau 8*shighah*, , *shighah-sighah* tersebut adalah sebagai berikut:

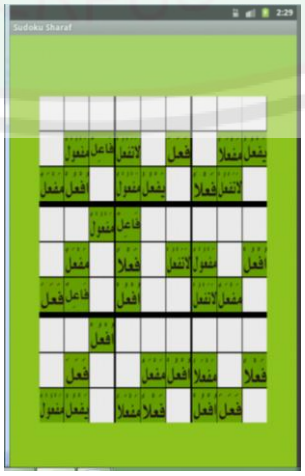
1. *Sighat Masdhar*, yaitu bentuk kata dasar
2. *Mashda* Mim, yaitu mashdar yang mendapat tambahan mim
3. *IsimFa'il*, yaitu kata benda yang menunjukkan pengertian pelaku
4. *Shighat Mutasyabihah biismil Fa'il*, yaitu kata sifat yang disamakan dengan *isim fa'il*
5. *Sighat Mubalaghah*, yaitu bentuk penyangatan
6. *Isim Tafdhil*, yaitu kata benda yang mengandung pengertian lebih
7. *Isim Maf'ul*, yaitu kata benda yang menunjukkan pengertian penderita
8. *Isim Makan*, yaitu kata benda yangmenunjukkan pengertian tempat
9. *Isim Zaman*, yaitu kata benda yang menunjukkan pengertian waktu
10. *Isim Alat*, yaitu kata benda yang menunjukkan pengertian alat.

Biasanya *fi'il* memiliki 10 *shighah*, akan tetapi ada yang kurang dari 10 *shighah*, yang digunakan dalam penelitian ini adalah fi'il yang memilki 10,9 dan *shighah* saja. Jadi, ketika pengguna memilih *fi'il* yang memiliki lebih dari 9 *shighah*, maka halaman pilih *shighah* akan ditampilkan. Akan tetapi, jika pengguna memlih *fi'il* yang hanya memiliki 9 *shighah*, maka *shighah-shighah* tersebut hanya ditampilkan, tidak ada proses pemilihan *shighah*. Sedangkan jika pemain memilih *wazn* yang jumlah *shighah*nya berjumlah 8, maka akan ada *shighah* kembar pada permainan *sudoku sharaf*. Akan tetapi, peneliti memberikan

tanda garis berwarna pada *shighah* kembar tersebut.Gambar 4.5 adalah halaman pilih *shighah*.

Gambar 4.5 Halaman Pilih *shighah*

Pada gambar 4.5 adalah halaman pilih *shighah* pada *fi'il* yang memiliki 10 *shighah*, *player* harus memperhatikan *shighah* apa yang dipilih dan *shighah* apa yang tidak dipilih, game shudoku *sharaf* hanya bisa dimainkan jika *player* sudah memilih 9 *shighah*, tidak kurang dan tidak lebih. Setelah menekan *button* permainan akan terlihat papan *puzzle* sudoku sebagaimana gambar 4.6



Gambar 4.6 *Puzzle* soal sudoku pada level mudah

4.3 PembahasanAlgoritma

Pada penelitian ini menggunakan 2 algoritma, yakni algoritma *Linear Congruential Generator* dan Algoritma *Depth First Search*, kegunaan dan fungsi masing-masing algoritma akan dijelaskan sebagai berikut:

4.3.1 Algoritma *Linear Congruential Generator*

Algoritma*Linear Congruential Generator* diimplementasikan menjadi sebuah fungsi random dalam sebuah class java yang diberi nama*"generator"*.Nama dari fungsi *random* tersebut adalah LCG dengan parameter berupa range, zo, a, m, c dan sebuah *arraylist* yang digunakan untuk menampung output dari fungsi tersebut.Semua parameter dari fungsi LCG bernilai integer.

```
public class generator {
      public static int lcg(int range,int zo,int a,int
m,int ci,ArrayList<String>isi_random) {
fo  r (int i = 0; i < range; i++) {
zo = (a * zo + ci) % m;
            isi_random.add(String.valueOf(zo));
          }
return zo;
   }}
```

Gambar 4.7 *Source Code Linear Congruential Generator*

Pada penelitian algortima LCG digunakan pada 2 tahap yakni:

a. Pembangkitan *Shighah*

   Pada tahap ini fungsi LCG digunakan untuk mengisi semua kotak *sudoku sharaf* yang berjumlah 81 kotak. Ke-81 kotak tersebut diisi secara

random.Hasil *random* LCG menghasilkan 81 data dengan type *integer*.Sehingga kotak sudoku sebanyak 81 kotak terisi semua.

```
ArrayList<String>isi_soal=new ArrayList<String>();
ArrayList<String>isi_kotak=new ArrayList<String>();
//mengisi semua kotak kosong
pembangkit_pilihan=(int)(Math.random()*10);
l.lcg(1,pembangkit[pembangkit_pilihan], a, m,c
isi_soal);
indeks=isi_soal.get(0);
soal=level[(Integer.parseInt(indeks)-1)];
//Memindahkan data arraylist ke array 4 dimensi
int batas=0;
for(int aa=0;aa<3;aa++){
      for(int b=0;b<3;b++){
            for(int ci=0;c<3;c++){
      for(int d=0;d<3;d++){
                        kotak[aa][b][ci][d]=Integer.par
seInt(String.valueOf(soal.charAt(batas)));
batas++;
          }}}}
//menampilkan shighah ke semua kotak puzzle
for(int aa=0;aa<3;aa++){
for(int b=0;b<3;b++){
for(int ci=0;ci<3;ci++){
for(int d=0;d<3;d++){
      puzzle[aa][b][ci][d].setBackgroundResource(gamb
ar[shighah_pil[(kotak[aa][b][ci][d])-1]]);
      }
    }}}
```

Gambar 4.8 Pembangkitan *Shighah*

b. Pembuatan Soal

```
//menentukan kotak soal
for(int kotak=0;kotak<9;kotak++){
   pembangkit_kotakkosong=(int)(Math.random()*4);
   generator.lcg(kuota,pembangkit_kotak[pembangkit_kotakk
   osong], a, m,c, isi_kotak);
}
//Sorting array isi_kotak
int pola=kuota;
int ulang=isi_kotak.size()/kuota;
for(int i=0;i<ulang;i++){
int start=i*pola;
int max=(i+1)*pola;
for(int j=start;j<max;j++){
for(int k=start;k<max;k++){

if(Integer.parseInt(isi_kotak.get(j))<Integer.parseInt(is
i_kotak.get(k))){

int temp=Integer.parseInt(isi_kotak.get(j));
    isi_kotak.set(j,isi_kotak.get(k));
    isi_kotak.set(k,String.valueOf(temp));
}}}}
//  Genarate soal
int ambil=0;
int lacak=0;
for(int aa=0;aa<3;aa++){
for(int b=0;b<3;b++){
lacak=0;
for(int ca=0;c<3;c++){
for(int d=0;d<3;d++){
      lacak++;
if(lacak==Integer.parseInt(isi_kotak.get(ambil))){
kotak_cek[aa][b][ca][d]=kotak[aa][b][ca][d];
kotak[aa][b][ca][d]=0;
puzzle[aa][b][ca][d].setBackgroundResource(R.drawable.ala
sjadi);
puzzle[aa][b][ca][d].setOnClickListener(this);
ambil++;
}
```

Gambar 4.9 *Source Code* Pembuatan Soal

Pada langkah sebelumnya, kotak *shudoku* terisi semua. Fungsi LCG kembali digunakan untuk membuat soal (menutup kotak) sesuai level yang dipilih. Jika Player memilih level sedang, maka tiap area fungsi LCG akan melakukan penutupan sebanyak 4 kotak atau 36 kotak pada semua kotak shudoku. Outputnya ditampung dalam array isi_kotak dan kemudian dilakukan penyortingan secara *ascending*. Setelah array isi_kotak disorting, maka langkah selanjutnya adalah melakukan perulangan *array* kotak dan setiap *indeksarray* kotak akan diberikan nilai 0 apabila nilai variabel acak sama dengan isi *indeksarray* isi_kotak. Hal tersebut diimplementasikan sebagaimana gambar 4.9

Tahapan implementasi LCG dapat diilustrasikan dengan gambar 4.10 dan gambar 4.11 sebagai berikut:

لاتفعل	فعل	فعلا
مفعلا	فا عل	مفعل
يفعل	أفعل	مفعول

Gambar 4.10 Hasil *Generate* LCG pada kotak 3x3 pada LCG tahap 1

لاتفعل	x	x
مفعلا	فا عل	مفعل
x	أفعل	x

Gambar 4.11 Hasil *Generate* LCG pada kotak 3x3 pada LCG tahap II

4.3.2 Algoritma*Depth First Search (DFS)*

Pada penelitian ini, algoritma *Depth First Search* digunakan sebagai *solver*permainan *Sudoku sharaf*, ada beberapa langkah yang dilakukan, yaitu:

a. Pencarian solusi Jawaban

Seperti yang dijelaskan sebelumnya bahwasanya *shighah-shighah*di*inisialisasi*kan dengan menggunakan angka 1-9, seperti pada permainan shudoku angka.Pencarian kemungkinan jawaban dilakukan per-area dari kotak shudoku.*Inisialisasi* dilakukan sebagai berikut:

Tabel 4.11 Inisialisasi *Shighah*

Shighah Pada sharaf	Inisialisasi Angka
فعل	1
يفعل	2
فعلا	3
مفعلا	4
فا عل	5
مفعول	6
أفعل	7
لاتفعل	8
مفعل	9

Semisal pada kotak area sudoku muncul soal sebagai berikut:

لاتفعل		
مفعلا	فا عل	مفعل
	أفعل	

Gambar 4.12 Contoh Soal

Maka, Jawaban yang akan mengisi kotak kosong sesuai dengan *inisialisasi* adalah 1,2,3,6.

```
//analisa kotak yang kosong >> analisa angka yg tidak ada
dalam kotak 3x3 kecil
public String [] CariKemungkinanAngka(){
int jumlah=9;
          String [] Random = new String[jumlah];
for(int i=0;i<Random.length;i++){
            String db="123456789";
int start=i*9;int loop=(i*9)+9;
            String soalArea=soal.substring(start, loop);
soalArea=soalArea.replaceAll("0", "");
for(int j=0;j<soalArea.length();j++){
db=db.replaceAll(String.valueOf(soalArea.charAt(j)), "");
            }
            Random[i]=db;
            //System.out.println(Random[i]);
}return Random;
     }
```

Gambar 4.13 *Source Code* Pencarian Jawaban Soal

b. Membuat Kemungkinan susunan solusi jawaban

Sistem sudah mengetahui, bahwasanya jawaban yang mungkin adalah 1,2,3,6. Kemudian system akan mencari kemungkinan susunan solusi jawaban yang paling benar dengan menggunakan rumus permutasi. Yakni n! dengan n adalah jumlah kotak kosong. Jadi kemungkinan jawaban pada soal tersebut 4!=4x3x2x1 =24

```
public  int faktorial(int angka){
int c=1;
for(int i=angka;i>0;i--){
c*=i;
}return c;}
publicint [][] bikinSeluruhKombinasi(){
int [][] arr = newint[level][faktorial(level)];
        //isi default array
for(int i=0;i<arr.length;i++){
for(int j=0;j<arr[0].length;j++){
arr[i][j]=-8;
            }}
for(int i=0;i<level-2;i++){
int fk=faktorial(level-1-i);
      int isi=0;int j=0;int id=0;
while(true){
for(int k=0;k<fk;k++){
id=(j*fk)+k;
while(true){
boolean kembar=false;
for(int l=0;l<arr.length-1;l++) {
if(isi==arr[l][id]){
kembar=true;
break} }
if(kembar) {
if(isi==level-1) isi=0;
else isi++;
}else {
arr[i][id]=isi;
break;}}}
for(int i=0;i<arr.length;i++){
for(int j=0;j<arr[0].length;j++){
arr2[j][i]=arr[i][arr[0].length-j-1];
            }}
```

```
if(isi==level-1) isi=0;
else isi++;
j++;
if(id==faktorial(level)-1) break;
            }
        }
//ubahstruktur array
int [][] arr2 = newint[faktorial(level)][level];
//two last index
boolean balik=true;
for(int i=0;i<arr2.length;i++){
            //ngisi database
            ArrayList<String> db = new ArrayList<String>();
for(int q=0;q<level;q++) db.add(String.valueOf(q));
for(int j=0;j<arr2[0].length;j++){
db.remove(String.valueOf(arr2[i][j]));
            }
if(balik){
arr2[i][arr2[0].length-2]=Integer.parseInt(db.get(0));
arr2[i][arr2[0].length-1]=Integer.parseInt(db.get(1));
balik=false;
}else{
arr2[i][arr2[0].length-2]=Integer.parseInt(db.get(1));
arr2[i][arr2[0].length-1]=Integer.parseInt(db.get(0));
balik=true;
            }
        }
return arr2;
    }
```

Gambar 4.14 *Source Code* Pencarian Jawaban Soal

c. *Filterisasi*

Proses *filterisasi* dilakukan untuk mempercepat proses percobaan kombinasi kemungkinan jawaban terbaik. Proses *filterisasi* dilakukan dengan cara membuang awalan kombinasi yang sudah tidak mungkin dijadikan sebagai solusi jawaban terbaik.

```
//filtering komposisi awal
public String [] komposisi(){
String [] kompo = new String[9]; //seluruh komposisi 9
kotak besar
String [] ab = new String[9];
int id=0;
for(int i=0;i<3;i++) for(int j=0;j<3;j++) {
ab[id]=i+""+j;
id++;}
        //identifikasi soal yg masih kosong
        //perulangan komposisi 24,120,720
for(int i=0;i<kemungkinanAngka9x9.length;i++){
//array penampung buat menghapus komposisi
String [] index = new String[faktorial(level)];
for(int q=0;q<index.length;q++) index[q]="0";
//convert komposisi combo ke array list biar bisa
dihapus dinamis
ArrayList<String> cbi = new ArrayList<String>();
for(int k=0;k<combo.length;k++){
                String abc="";
for(int j=0;j<combo[0].length;j++){
abc+=String.valueOf(combo[k][j]);
                }
cbi.add(abc);}
//menentukan a dan b sekarang
int a=Integer.parseInt(String.valueOf(ab[i].charAt(0)));
int b=Integer.parseInt(String.valueOf(ab[i].charAt(1)));
for(int j=0;j<cbi.size();j++){ // per komposisi
                //untuk menetukan komposisi pada char.at
```

```
int ambil=0;
                //perulangan kotak 3x3 kecil
for(int c=0;c<3;c++){
for(int d=0;d<3;d++){
//cek apakah soal memiliki value 0 yg berarti kosong
if(Soal4Dimensi[a][b][c][d]==0){
//tentukan index komposisi (isi komposisi)
int idx
=Integer.parseInt(String.valueOf(cbi.get(j).charAt(ambil)))
//angka sebenarnya dari komposisi
int angka
=Integer.parseInt(String.valueOf(kemungkinanAngka9x9[i].char
At(idx)));
//cek baris dan kolom
boolean cek=cekInputan(a, b, c, d, angka, Soal4Dimensi);
//jika false ato ada yang angka yg sama pada baris dan kolom
yg sama
if(!cek){
//perulangan kompsisi 24, 120, 720
for(int z=cbi.size()-1;z>=0;z--){
//jika ada kesamaan angka dan letak
indexif(Integer.parseInt(String.valueOf(cbi.get(z).charAt(am
bil)))==idx) {
//tampung index yang akan dihapus
index[z]="ada";
}} }
ambil++;
     }}
 //remove komposisi
for(int q=index.length-1;q>=0;q--) {
if(index[q].equals("ada")){
cbi.remove(q);}}
            //gabung isi komposisi terfilter
            String filteran="";
for(int q=0;q<cbi.size();q++) filteran+=cbi.get(q);
//tambahkan pada komposisi keseluruhan 9x9}
return kompo;}}
```

Gambar 4.15 *Source Code Filterisasi*

d. Pengecekan *Deadlock*

```
//dead lock proses
public String deadLock(String [][] arr /*kombinasi hasil
filter*/ ){
int [] index = new int[arr.length];
index[0]=0; String hasil=arr[0][0];
int a=1; //kolomint next=0; //baris
while(true){
            String ceker = "";
while(true){
if(next==maxLengthFilterKompo[a]) {
index[a]=0;
                    a--;
index[a]=index[a]+1;
hasil="";
if(a==0) {
hasil=arr[0][index[a]];
                        a=1;
                    }
else{
for(int i=0;i<a;i++){
hasil+=arr[i][index[i]];
                        }}
next=index[a];}
else{break;}}
ceker=solver(arr, a, hasil, next);
int posisi= Integer.parseInt(ceker.substring(0,
ceker.length()-1)); //valid
Intb=Integer.parseInt(String.valueOf(ceker.charAt(ceker.leng
th()-1))); //posisi berikutnya
if(a<b){
index[a]=posisi;
hasil+=arr[a][index[a]];
                a=b;next=0;
            }
```

```
else if(a>b){ //deadlock
index[b]=index[b]+1;
if(b==0) {
                    a=1;hasil=arr[b][index[b]];
next=0;
                }
else {
                    a=b;hasil="";
for(int i=0;i<a;i++){
hasil+=arr[i][index[i]];
                    }
next=index[a];
                }}
if(a==arr.length) break;}
return hasil;
    }
```

Gambar 4.16 *Source Code* Pengecekan *deadlock*

Setelah proses *filterisasi* selesai, semua area (area) kotak shudoku memiliki kemungkinan jawaban terbaik. Untuk memastikan bahwasanya antara kotak satu pada area kolom dan baris tidak ada *shighah* yang sama, maka diperlukan pengecekan terjadinya *deadlock* atau tidak pada kolom dan baris kotak sudoku. Jika tidak terjadi *deadlock* system akan menampilkan hasil penyelesaian, kalau terjadi *deadlock* system akan mengulangi proses tersebut sampai tidak terjadi *deadlock* antar baris dan kolom pada kotak sudoku.

Jawaban terbaik yang telah dilakukan pengecekan *deadlock*dalam bentuk type data string, kemudian disimpan dalam sebuah array dan *diparsing* ke type data int. kemudian system akan menampilkan *shighah* sesuai dengan *inisialisasi* pada proses diatas.

```
public String [] solverJadi(String solver){
        String [] arr = new String[kemungkinanAngka9x9.length];
//convert solver
        String itsSolve ="";
int id=0;
int itungan=0;
for(int i=0;i<solver.length();i++){
int isi = Integer.parseInt(String.valueOf(solver.charAt(i)));
itsSolve+=String.valueOf(kemungkinanAngka9x9[id].charAt(isi));
itungan++;
if(itungan==level){
itungan=0;
id++;
            }
        }
```

Gambar 4.17 *Source Code* Untuk Meng-*convert* hasil *solver*

4.4 Uji Coba *Game* Sudoku Sharaf

Pengujian terhadap penerapan Algoritma *Linear Congruential Generator* dan *Depth First Search* pada aplikasi *SudokuSharaf* dilakukan dengan perangkat keras Axioo Picopad 7 GGM dengan spesifikasi sebagai berikut:

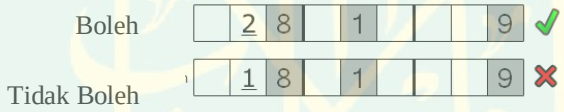
1. Prosesor : 1,2 GHz *multicore*
2. Internal Memory : 1GB DDR3
3. Resolusi : 1024×600 Pixels
4. OS : Android ICS 4.0.3 (Ice Cream Sandwich)

Pengujian dilakukan dengan menyelesaikan *game Sudoku Sharaf*. Waktu yang digunakan telah terprogram pada aplikasi yang akan diuji. Pengujian

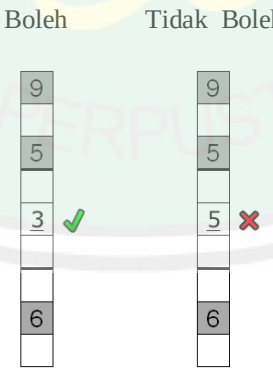
dilakukan pada setiap levelnya, yaitu: level mudah, sedang dan sulit. Perbedaan level terdapat pada banyaknya kotak kososng yang harus diisi.

4.4.1 Uji Coba Kebenaran Game Sudoku Sharaf berdasarkan permainan Sudoku

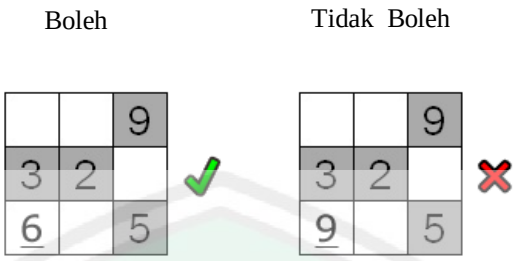
Uji coba ini dimaksudkan untuk mengetahui apakah aplikasi yang dibuat sesuai dengan peraturan (*rule*) permainan sudoku yang menggunakan angka pada umumnya, secara ringkas peraturan permainan sudoku angka adalah sebuah angka harus muncul sekali di setiap baris, kolom dan area. Penjelasanya adalah sebagai berikut:



Gambar. 4.18 Aturan Baris Permainan Sudoku
(Sumber :http://www.sudoku.name)



Gambar 4.19 Aturan Kolom Permainan Sudoku
(Sumber :http://www.sudoku.name)



Gambar 4.20 Aturan Area Permainan Sudoku
(Sumber :http://www.sudoku.name)

Jika peraturan tersebut diterjemahkan ke dalam *sudoku sharaf* yang dibuat oleh peneliti, maka peraturanya adalah tidak boleh ada shighah yang sama disetiap baris, kolom dan area. Uji coba ini dimaksudkan apakah aplikasi mampu memberikan warning, jika *shighah* yang dipilih oleh *player* telah ada pada baris, kolom atau area *puzzle* sudoku.

Gambar 4.21 *Shighah* Pilihan

Gambar 4.22 *Warning* Kesalahan Penempatan *Shighah*

Dari hasil percobaan yang dilakukan *game* yang telah dibuat mampu memberikan pesan kesalahan jika penempatan *shighah*tidak sesuai dengan peraturan *sudoku*.

Percobaan kedua, dilakukan dengan menguji kebenaran *solver* dengan menggunakan algoritma *Depth First Search* dari *game* yang telah dibuat oleh peneliti. Uji coba dilakukan dengan cara melakukan pengamatan langsung pada kotak sudoku hasil *solver* dari *system*. Pengamatan tersebut dengan memperhatikan apakah ada shighah yang sama pada setiap area, kolom dan baris. Uji coba dilakukan sebanyak 36 kali pada setiap *level*, *wazn* dan *fi'il* yang berbeda-beda.

Gambar 4.23 Hasil *Solver* dengan menggunakan DFS

Untuk menghitung keberhasilan dan kegagalan dari algoritma solver adalah sebagai berikut :

$$\text{Prosentase keberhasilan} = \frac{Jumlah\ \ Keberhasilan}{Banyaknya\ Uji\ Coba} \text{ x } 100$$

$$\text{Prosentase kegagalan} = \frac{Jumlah\ \ Kegagalan}{Banyaknya\ Uji\ Coba} \text{ x } 100$$

Dari 36 kali uji coba menghasilkan prosentase keberhasilan dan prosentase kegagalan sebagai berikut:

$$\text{Prosentase keberhasilan} = \frac{36}{36} \text{ x } 100 = 100\%$$

$$\text{Prosentase kegagalan} = \frac{0}{36} \text{ x } 100 = 0\%$$

4.4.2 Perbandingan Algoritma *Solver* (*Depth First Search*) dengan Algoritma *Branch and Bound.*

Pengujian ini bertujuan untuk melihat *performance* algoritma *Depth First Search* dibandingkan dengan algoritma lain. Algoritma lain yang dibandingkan merujuk pada beberapa penelitian yang dilakukan oleh

mahasiswa Universitas Islam Negeri (UIN) Maulana Malik Ibrahim Malang dengan topik yang sama akan tetapi dengan menggunakan algoritma yang berbeda. Penelitian tersebut adalah Penelitian yang dilakukan oleh Ahmad Baihaqi (2013) dengan menggunakan algoritma *solverBranch And Bound.*Pengujian dilakukan dengan membandingkan waktu dari masing-masing algoritma dalam menyelesaikan *puzzlegame sudoku* yang kosong pada 3 level yang berbeda.

Sebelum membandingkan waktu penyelesaian dengan penelitian yang lain, maka peneliti terlebih dahulu melakukan uji coba waktu terhadap game shudoku yang telah dibuat oleh peneliti. Skenario uji coba sama persis dengan apa yang dilakukan oleh peneliti yang akan dibandingkan. Baik skenario level pengujian, banyaknya pengujian dan *device* yang digunakan.

Uji coba ini dilakukan pada tanggal 20 Agustus 2013 jam 06.30 di tempat tinggal peneliti. Hasil uji coba dipaparkan sebagaimana table 4.2, 4.3 dan 4.4.

Tabel 4.2 Tabel Uji Coba Waktu Penyelesaian Pada Level Mudah

No	Percobaan	Waktu (detik)
1	Pertama	7,763
2	Kedua	6,254
3	Ketiga	8,107
4	Keempat	8,348
5	Kelima	6,76
6	Keenam	6,44
7	Ketujuh	5,574
8	Kedelapan	5,696
9	Kesembilan	5,895
10	Kesepuluh	6,296
Rata-Rata Waktu		7.3124

Tabel 4.2 menunjukkan hasil uji coba algoritma *Depth First Search (DFS)* untuk menyelesaikan permainan pada level mudah. Dari 10 uji coba yang dilakukan menunjukkan bahwasanya rata-rata waktu penyelesaian pada level mudah selama 7.3124 detik.

Tabel 4.3 Tabel Uji Coba Waktu Penyelesaian Pada Level Sedang

No	Percobaan	Waktu (detik)
1	Pertama	63,314
2	Kedua	26,75
3	Ketiga	26,54
4	Keempat	45,599
5	Kelima	24,903
6	Keenam	26,534
7	Ketujuh	26,968
8	Kedelapan	25,594
9	Kesembilan	28,098
10	Kesepuluh	41,072
Rata-Rata Waktu		33.5372

Tabel 4.3 menunjukkan hasil uji coba algoritma *Depth First Search* (DFS) untuk menyelesaikan permainan pada level sedang. Dari 10 uji coba yang dilakukan menunjukkan bahwasanya rata-rata waktu penyelesaian pada level sedang selama 33.5372 detik.

Tabel 4.4 Tabel Uji Coba Waktu Penyelesaian Pada Level Sulit

No	Percobaan	Waktu (detik)
1	Pertama	811,133
2	Kedua	666,996
3	Ketiga	634,472
4	Keempat	603,289
5	Kelima	751,987
6	Keenam	832,123
7	Ketujuh	764,23
8	Kedelapan	645,965
9	Kesembilan	765,865
10	Kesepuluh	673,327
Rata-Rata Waktu		714.9387

Tabel 4.4 menunjukkan hasil uji coba algoritma *Depth First Search (DFS)* untuk menyelesaikan permainan pada level sulit. Dari 10 uji coba yang dilakukan menunjukkan bahwasanya rata-rata waktu penyelesaian pada level sulit selama 714.9387detik.

Dari hasil penelitian tersebut dapat diketahui bahwa algoritma *Depth First Search (DFS)* mampu menyelesaikan permainan *sudoku sharaf* baik pada level mudah, sedang dan sulit. Akan tetapi, terdapat perbedaan waktu penyelesaian pada tiap-tiap level. Semakin mudah level permainan semakin sedikit waktu yang digunakan algoritma DFS untuk menyelesaikan permainan *sudoku sharaf*.

Waktu penyelesaian algoritma DFS pada permainan *sudoku sharaf* dibandingkan dengan algoritma *branch and bound* yang diteliti oleh Ahmad Baihaqi, Jurusan Teknik Informatika, Universitas Islam Negeri Maulana Malik Ibrahim Malang.Hasil penelitian yang dilakukan oleh Ahmad Baihaqi (2013) sebagaimana pada table 4.5,4.6 dan 4.7.

Tabel 4.5 Tabel Uji Coba Waktu Penyelesaian Pada Level Mudah
(Sumber: Baihaqi,2013)

No	Percobaan	Waktu (detik)
1	Pertama	0,659
2	Kedua	0,564
3	Ketiga	0,512
4	Keempat	0,365
5	Kelima	0,899
6	Keenam	0,566
7	Ketujuh	0,401
8	Kedelapan	0,439
9	Kesembilan	0,365
10	Kesepuluh	0,371
Rata-Rata Waktu		0.5141

Rata-rata waktu penyelesaian algoritma *branch and bound* pada level mudah selama 0.5141. lebih cepat bila dibandingkan dengan menggunakan algoritma DFS.

Tabel 4.6 Tabel Uji Coba Waktu Penyelesaian Pada Level Sedang
(Sumber: Baihaqi,2013)

No	Percobaan	Waktu (detik)
1	Pertama	2,193
2	Kedua	4,346
3	Ketiga	2,065
4	Keempat	3,214
5	Kelima	2,416
6	Keenam	2,278
7	Ketujuh	2,278
8	Kedelapan	2,223
9	Kesembilan	4,195
10	Kesepuluh	5,367
Rata-Rata Waktu		3.0575

Rata-rata waktu penyelesaian algoritma *branch and bound* pada level sedang selama 3.0575. lebih cepat bila dibandingkan dengan menggunakan algoritma DFS.

Tabel 4.7 Tabel Uji Coba Waktu Penyelesaian Pada Level Sulit
(Sumber: Baihaqi,2013)

No	Percobaan	Waktu (detik)
1	Pertama	813,111
2	Kedua	67,113
3	Ketiga	59,191
4	Keempat	54,341
5	Kelima	365,269
6	Keenam	59,191
7	Ketujuh	54,341
8	Kedelapan	58,741
9	Kesembilan	62,183
10	Kesepuluh	99,938
Rata-Rata Waktu		169.3419

Rata-rata waktu penyelesaian algoritma *branch and bound* pada levelsulit selama 169.3419. lebih cepat bila dibandingkan dengan menggunakan algoritma DFS.

4.4.3 Pengujian User

Pengujian ini dilakukan untuk mengetahui kegunaan dan keberhasilan *game Sudoku sharaf* untuk pembelajaran gramatikal (*sharaf)* bahasa arab. Pengujian dilakukan dengan memberikan angket (pertanyaan) kepada siswa-siswi di MAN Kota Blitar pada Jurusan bahasa arab kelas XII dan Jurusan IPA.Isi dari angket tersebut untuk mengetahui keberhasilan program tersebut dari sisi *user.*

Pengujian aplikasi *SudokuSharaf* dilakukan pada responden dari Madrasah Aliyah Negeri (MAN) Kota Blitar. Pengujian dilakukan pada tanggal 06 September 2013 pukul 09.00 WIB dan pukul 11.00 WIB. Terdapat 35 responden dari kelas XII-IPA 1 dan 33 responden dari jurusan Agama kelas XII. Para responden diminta untuk melakukan uji coba (bermain) *game sudokusharaf* kemudian responden diminta untuk mengisi angket. Hasil rekapitulasi dan angket pertanyaan sebagaimana pada table 4.7

Tabel 4.8 Rekapitulasi Hasil Uji Coba pada kelas XII Agama

No	Pertanyaan	Sangat Setuju	Setuju	Netral	Tidak Setuju	Sangat Tidak Setuju
1	Mengetahui Kegunaan setiap menu pada menu utama	96,96% (32 responden)	3,04% (1 responden)	-	-	-

2	Menu Bantuan dapat membantu untuk memahami permainan *shudoku sharaf*	68,75% (22 responden)	31,25% (11 responden)	-	-	-
3	Menu *Wazn*dapat membantu memahami tentang *wazn*	100% (33 responden)	-	-	-	-
4	Menu *sharaf* dapat membantu memahami tentang *sharaf*	100% (33 responden)	-	-	-	-
5	Cara bermain permainan *shudoku sharaf* mudah dimainkan dan dipahami	62,5 % (20 responden)	33,33 % (11 responden)	6,62% (2 responden)	-	-
6	Mengetahui kegunaan menu **selesaikan**	93,94% (31 responden)	6.06% (2 responden)	-	-	-
7	Menu **selesaikan** sangat berguna membantu permainan	78,79% (26 responden)	2,12 % (7 responden)	-	-	-
8	*Game shudoku sharaf* menarik dan menyenangkan	27,27 5 (9 responden)	42,42% (14)	9,09% (3 responden)	21,21% (7 responden)	-
9	*Game shudoku sharaf* membantu menghafal *wazn*	87,88% (29 responden)	12,12% (4 responden)	-	-	-
10	*Game shudoku sharaf* membantu menghafal *sighah*	90,91% (30 responden)	6,06% (2 responden)	3,03% (1 responden)	-	-

Tabel 4.9 Rekapitulasi Hasil Uji Coba pada kelas XII IPA

No	Pertanyaan	Sangat Setuju	Setuju	Netral	Tidak Setuju	Sangat Tidak Setuju
1	Mengetahui Kegunaan setiap menu pada menu utama	82,86% (29 respond en)	17,14% (6 respond en)	-	-	-
2	Menu Bantuan membantu untuk memahami permainan *shudoku sharaf*	82,85% (29 respond en)	14,28% (5 respond en)	2,85(1 respond en)	-	-
3	Menu *Wazn*cukup membantu memahami tentang *wazn*	8,57% (3 respond en)	11,43% (4 respond en)	-	57,14% (20 respond en)	14,29% (5 respond en)
4	Menu *sharaf* cukup membantu memahami tentang *sharaf*	14,29% (5 respond en)	20% (7 respond en)	-	57,14% (20 respond en)	8,57% (3 respond en)
5	Cara bermain permainan *shudoku sharaf* mudah dimainkan dan dipahami	45,71% (16 respond en)	14,29% (5 respond en)	2,86% (1 respond en)	22,86 (8 respond en)	14,29% (5 respond en)
6	Mengetahui kegunaan menu **selesaikan**	88,58% (31 respond en)	11,43 % (4 respond en)	-	-	-
7	Menu **selesaikan** sangat berguna membantu permainan	91,43% (32 respond en)	8,57 (3 respond en)	-	-	-
8	*Game shudoku sharaf* menarik dan menyenangkan	11,43% (4 respond en)	22,86% (8 respond en)	14,29% (5 respond en)	20% (7 respond en)	31,43% (11 respond en)
9	*Game shudoku sharaf* membantu menghafal *wazn*	97,14% (34 respond en)	2,86% (1 respond en)	-	-	-
10	*Game shudoku sharaf*membantu	91,43% (32	5,71% (2	2,86% (1	-	-

	menghafal *sighah*	respond en)	respond en)	respond en)		

Dari uji coba yang dilakukan pada kelas XII jurusan Agama dan kelas XII jurusan IPA Madrasah Aliyah Negeri (MAN) Kota Blitar sebagaimana ditunjukkan pada tabel 4.6 dan 4.7 dapat disimpulkan bahwa hampir seluruh responden dari jurusan bahasa maupun responden dari jurusan IPAsangat setuju bahwa mereka mengetahui kegunaan dan fungsi menu utama. begitu juga penilaian, menu bantuan dapat membantu untuk memahami alur permainan *sudoku sharaf*hampir semua responden sangat setuju dan setuju. Hanya 2,85% responden dari jurusan IPA yang tidak berpendapat (netral).

Responden dari jurusan bahasa 100 % sangat setuju kalau menu *wazn*dan *sharaf* dapat membantu pemahaman mereka terhadap *wazn*dan *sharaf*. Akan tetapi, berbeda pada responden jurusan IPA, hanya 8,57% yang sangat setuju, 11,43% setuju, 57,14% tidak setuju dan 14,29% sangat tidak setuju permainan *sudoku sharaf* membantu memahami tentang *wazn*. Begitu juga dengan fungsi menu *sharaf* permainan *sudoku sharaf* untuk membantu memahami tentang *sharaf*, hanya 14,29% responden yang menyatakan sangat setuju, 20% setuju, 57,14% tidak setuju dan 8,57% sangat tidak setuju. Hampir seluruh responden dari jurusan Agama dan jurusan IPA menyatakan sangat setuju dan setuju kalau permainan *shudoku sharaf* mudah dimainkan dan dipahami, hanya 6,62 % responden dari jurusan IPA yang netral.

Bahkan semua responden dari jurusan Agama dan IPA menyatakan sangat setuju dan setuju mengetahui menu selesaikan dan berpendapat menu selesaikan

sangat berguna membantu permainan *sudoku sharaf*. Hal tersebut terbukti, pada responden jurusan 93,94% responden mengetahui kegunaan menu tersebut dan dari 93,94% tersebut berpendapat bahwa menu selesaikan 78,79 sangat berguna membantu permainan *sudoku sharaf*. Sedangkan pada responden IPA berturut-turut 88,58% dan 91,43%.

Sedangkan kegunaan permainan *sudoku sharaf* untuk hafalan (pembelajaran) *sharaf* hampir seluruh responden dari jurusan IPA maupun jurusan Agama berpendapat sangat setuju dan setuju bahwa permainan *sudoku sharaf* membantu menghafal *wazn* dan *shighah*, 87,88% responden dari jurusan bahasa sangat setuju permainan *shudoku sharaf* membantu menghafal *wazn* dan 90,91% sangat setuju membantu menghafal *shigha*. Hanya 2,86% dari jurusan IPA yang berpendapat netral.

Sebagian besar responden dari jurusan Agama sangat setuju dan setuju bahwa permainan *sudoku sharaf* menarik dan menyenangkan, dengan prosentase 27,27% sangat setuju, 42,42% setuju, 9,09 netral, 21,21% tidak setujandinu. Hal ini berbanding terbalik dari responden jurusan IPA yang sebagian besar menganggap permainan *sudoku sharaf* tidak menarik dan tidak menyenangkan. dengan 11,43% Sangat setuju, 22,86% setuju, 14,29% netral, 20% tidak setuju dan 31,43% sangat tidak setuju.

Terjadi perbedaan hasil uji coba antara responden jurusan Agama dan responden jurusan IPA.Perbedaan tersebut dikarenakan *knowledge base*responden yang melakukan uji coba. Responden jurusan Agama sering mempelajari bahasa arab, khususnya ilmu *sharaf*daripada responden jurusan

IPA. Sehingga responden dari jurusan bahasa lebih mudah menerima permainan *sudoku sharaf*daripada responden jurusan IPA. Sehingga dari uji coba tersebut dapat disimpulkan bahwasanya permainan *sudoku sharaf*lebih mudah dipahami dan dimengerti oleh orang yang pernah bermain *sudoku* dan pernah belajar ilmu *sharaf*.

4.5 Integrasi Islam

Bahasa arab adalah bahasa al-qur'an, bahasa surga dan bahasa nabi Muhammad SAW. Dan beliau sendiri menyuruh umat islam mencintai (mengeri) bahasa arab.

أَحِبُّوا الْعَرَبَ لِثَلَاثٍ: لِأَنِّي عَرَبِيٌّ وَالْقُرْآنَ عَرَبِيٌّ وَكَلَامَ أَهْلِ الْجَنَّةِ عَرَبِيٌّ

"Cintailah bahasa Arab karena tiga hal, yaitu bahwa aku adalah orang Arab, bahwa al-Quran adalah bahasa Arab, dan bahasa penghuni syurga di dalam syurga adalah bahasa Arab." (HR. at-Thabrani).

Hadits tersebut diriwatkan oleh At-Thabrani pada kitab *Majma' Al-Bayan Fi Tafsir Al-Qur'an* halaman206.Hadist tersebut dipertegas dengan firman Allah SWT pada surat Az-Zukhruf ayat 3 yang berbunyi.

إِنَّا جَعَلْنَاهُ قُرْءَٰنًا عَرَبِيًّا لَّعَلَّكُمْ تَعْقِلُونَ ﴿٣﴾

"Sesungguhnya Kami menjadikan Al Quran dalam bahasa Arab supaya kamu memahami(nya)"(QS. Az-Zukhruf:3).

Al-qur'an adalah pedoman bagi orang islam dan bahasa Al-qur'an adalah bahasa arab. Sehingga sebagai seorang muslim sudah seharusnya memahami pedoman agamanya. Untuk memahami al-qur'an maka perlu mempelajari bahasa arab, baik dari segi *nahwu* dan *sharaf*.

Dalam mempelajari al-qur'an ilmu *sharaf* berguna untuk mengetahui perubahan bentuk kata yang mengakibatkan perubahan arti pada kata tersebut. Jadi, selain mengerti arti dasar kata, untuk mengetahui arti kata dalam al-qur'an harus mengetahui juga *shighah*nya. Hal ini dikarenakan tiap-tiap *shighah* dari satu kata memiliki arti yang berbeda.

Permainan *sudoku sharaf* bisa digunakan sebagai media pembelajaran dan penghafalan perubahan bentuk kata (*shighah*). Dalam permainan tersebut *user* secara tidak langsung disuruh berfikir kelengkapan *shighah-shighah* dari *fi'il* yang dimainkan menurut *wazn* yang telah disajikan pada menu pilih *wazn*. Sehingga ketika *user* sering memainkan permainan *sudoku sharaf* secara berulang-ulang secara tidak langsung ada proses pembelajaran (menghafal) perubahan kata dalam bahasa Arab (*sharaf*).

BAB V

PENUTUP

5.1 Kesimpulan

Dari hasil penelitian yang dilakukan menunjukkan bahwasanya algoritma *Linear Congruential Generator* mampu melakukan pengacakan soal *game Sudoku* sharaf dan algoritma *Depth First Search* (DFS) berhasil digunakan sebagai *solver* dengan tingkat keberhasilan 100%. Dari hasil penelitian menunjukkan bahwasanya semakin mudah level permainan semakin sedikit waktu yang digunakan untuk menyelesaikan permainan *sudoku*. Rata - rata waktu yang dibutuhkan untuk level mudah selama 7.3124 detik, sedangkan untuk rata - rata waktu sedang sebesar 33.5372 dan rata-rata penyelesaian level sulit selama 714.9387 detik. Penelitian tersebut mengindikasikan bahwasanya algoritma *Depth First Search* (DFS) dari sudut pandang waktu penyelesaian kurang baik digunakan sebagai *solver game Sudoku Sharaf* dibandingkan dengan *algoritma Branch and Bounch* pada penelitian baihaqi (2013).

Dari sisi pembelajaran *game Sudoku Sharaf* sangat membantu untuk mempelajari dan menghafal perubahan kata (*sharaf*) dalam bahasa arab hal ini terbukti dari penelitian yang dilakukan 90,91% responden pada jurusan Agama dan 91,43% pada jurusan IPA menyatakan bahwa *game Sudoku sharaf* membantu proses belajar dan menghafal *shighah*.

5.2 Saran

Masih banyak kekurangan dalam penelitian ini, oleh karena itu, penulis menyarankan beberapa hal sebagai berikut :

1. Membandingkan algoritma *solver Depth First Search* (DFS) dengan algoritma – algoritma lain, sehingga didapatkan algoritma terbaik untuk melakukan *solver* permainan *Sudoku Sharaf*.
2. Menambahkan *fi'il* yang dimainkan lebih banyak lagi dan disimpan di dalam system *database*.
3. Mengembangkan permainan *Sudoku Sharaf* lebih menarik, baik desain maupun fasilitas aplikasinya.
4. Mengembangkan permainan *Sudoku Sharaf* diberbagai *platform mobile* seperti Symbian, Windows Phone, BlackBerry, iOS dan lainya.

DAFTAR PUSTAKA

Abrar, Riyadli. 2012. *Aplikasi Permainan Arabic Sudoku Menggunakan Metode Harmony Search sebagai Pembangkit dan Penyelesaian Permainan*. (*Skripsi*. Jurusan Teknik Informatika Fakultas Sains dan Teknologi UIN Maulana Malik Ibrahim. Malang)

Al-Mahalli, Jalaluddin Muhammad dan As-Suyuthi, Jalaluddin Abdurrahman. 2010. *Tafsir Jalalain*. Surabaya: Pustaka elBa.

Baihaqi, Ahmad.2013. *Aplikasi Permainan Shudoku Sharaf Menggunakan Metode LCG sebagai Generator dan Branch and Bound sebagai solver. (Skripsi,* Jurusan Teknik Informatika Fakultas Sains dan Teknologi UIN Maulana Malik Ibrahim. Malang)

Dian, Arif.2010. *Pengembangan aplikasi permainan sudoku kata dengan Solusi algoritma runut-balik pada platform j2me.* Yogyakarta : AMIKOM

Farida, Luluk. 2008. *Mengenal Kata Dalam Bahasa Arab*. Yogyakarta: Aziziyah.

Grace, Lindsay. 2005. *Game Type and Game Genre*. diakses 30 April 2013.

Hasan, Ahmad Bin. 2010. *Kitab Tasrif Juz Pertama*. Bangil: Raihan.

H S, Simon. 2007. *Pencarian Solusi Sudoku dengan Algoritma Berbasis Branch and Bound*. Departemen Teknik Informatika Institut Teknologi Bandung. Bandung.

Husnul. 2011. *Metode pembelajaran Ilmu Nahwu dan Shorof pada Siswa Siswi Madrasah Aliyah*. Diakses pada 21 juli 2013

Idris, Muhammad Jauhari. 2012. *Al-Qawaa'id Ash-Sharfiyyah*. Sumenep: Al-Amien Press

Ismail, Andang. 2006. *Education Games.* Yogyakarta: Pilar Media.

Kusumadewi, Sri. 2003. *Artificial Intelligent (Teknik dan Aplikasinya).* Yogyakarta:Graha Ilmu

L, Riani. 2010.*Pembangkit Bilangan Acak*. Mata Kuliah Pemodelan & Simulasi. Jurusan Teknik Informatika Universitas Komputer Indonesia. Bandung.

Mahtarami, Affan dan Ifansyah, Noor , M. 2010. *Pengembangan Game Pembelajaran Otomata Finit*. Seminar Nasional Informatika. Jurusan Teknik Informatika. Yogyakarta: Universitas Islam Indonesia.

Ma'shum, Muhammad. 1965. *Al-Amtsilatut Tashrifiyyah.* Semarang: Maktabah Alawiyah.

Mufarakhah, Eva Rizqiyatul. 2011. *Rancang Bangun Game Dasar-Dasar Kitab Al-Amtsilatut Tashrifiyyah*. Jurusan Teknik Informatika Fakultas Sains dan Teknologi UIN Maulana Malik Ibrahim. Malang.

Omimura, Satsuko. 2009. *222 Puzzle Sudoku Untuk Melejitkan Potensi Kreatif Otak Anda*. Jakarta: Prestasi Pustakaraya.

Ramadhan, Andresta. Tanpa Tahun. *Perbandingan Algoritma Linear Congruential Generators, BlumBlumShub, dan MersenneTwister Untuk Membangkitkan Bilangan Acak Semu*. Program Studi Teknik Informatika Institut Teknologi Bandung. Bandung.

Safaat, Nazruddin. 2011. *Android, Pemrograman Aplikasi Mobile Smartphone dan Tablet PC Berbasis Android*. Bandung: Informatika.

Siswanto. 2010. *Kecerdasan Tiruan*. Yogyakarta: Graha Ilmu

Shihab, Quraish. 2009. *Tafsir Al-Mishbah.* Jakarta: Lentera Hati.

Sidabutar, Juandi. 2008. *Pengembangan Algoritma Pencarian Untuk Menyelesaikan Problema Program Taklinier Integer Campuran.* Sekolah Pasca Sarjana Universitas Sumatera Utara.Medan.

Sumitro, Yoel Krisnanda. 2007. *Penyusunan Jarkom HMIF ITB dengan Menggunakan Algoritma Branch and Bound*. Departemen Teknik Informatika Institut Teknologi Bandung. Bandung.

Syam, Umi Fadilah Wardati. 2007. *Perumusan Sudoku Dengan Solusi Unik Menggunakan Algoritma Percabangan dan Pembatasan (Branch and Bound).* Program Studi Teknik Informatika Sekolah Teknik Elektro dan Informatika Institut Teknologi Bandung. Bandung.

Thabrasi, Aminuddin Abu Ali Al-Fadhl ibn Hasan ibn Al-Fadhl. 1997. *Majma' Al-Bayan Fi Tafsir Al-Qur'an Jilid-2.* Beirut: Dar Al Kutub Al-Ilmiyah.

Tim EMS. 2012. *Panduan Cepat Pemrograman Android*. Jakarta: Elex Media Komputindo.

Zaman, Saiful. Dyah R Helmi. 2009. *12 Permainan Untuk Meningkatkan Inteligensi Anak.* Jakarta: Visimedia.

LAMPIRAN – LAMPIRAN

KEMENTERIAN AGAMA
MADRASAH ALIYAH NEGERI KOTA BLITAR
NSM : 131135720001 NPSN : 20535102
Office: Jl. Jati No.78 Blitar Telp/Fax : (0342) 801041 // Website : www.mankotablitar.sch.id
E-mail : mankotablitar@yahoo.co.id

SURAT KETERANGAN OBSERVASI

Nomor : Ma.15.89/TL.01.1/589 /2013

Yang bertanda tangan dibawah ini :

Nama	: **Drs. H. KHUSNUL KHULUK, M.Pd**
NIP	: 196602011992031002
Jabatan	: Kepala MAN Kota Blitar
Golongan / Ruang	: Pembina Tk. I / IVb

Menerangkan dengan sebenarnya bahwa :

Nama	: **HADIROTUL MUFIDA**
NIM	: 08650072
Tempat/Tgl. Lahir	: Blitar, 9 Mei 1990
Alamat	: Jl. Ciliwung Kel. Bendo - Kec. Kepanjen Kidul - Kota Blitar
Keterangan	:

Yang bersangkutan telah melaksanakan penelitian di MAN Kota Blitar pada tanggal 6 September 2013 untuk menyelesaikan Skripsi yang berjudul: **APLIKASI SHOROF SUDOKU MENGGUNAKAN METODE LINEAR CONGRUENTIAL GENERATOR SEBAGAI PEMBANGKIT DAN DEPHT FIRST SEARCH SEBAGAI PENYELESAIAN PERMAINAN BERBASIS ANDROID.**

Demikian Surat Keterangan ini kami buat untuk dapat digunakan sebagaimana mestinya.

Blitar, 7 September 2013
Kepala Madrasah,

Drs. H. KHUSNUL KHULUK, M.Pd
NIP. 196602011992031002

Uji coba user di MAN Kota Blitar pada tanggal 06 September 2013

Siswa – Siswi MAN Kota Blitar kelas XII IPA

Uji coba game Sudoku Sharaf

Siswa – Siswi MAN Kota Blitar kelas XII Agama

Uji coba dengan memainkan game Sudoku sharaf