

IMPLEMENTASI ALGORITMA *FISHER-YATES SHUFFLE* PADA APLIKASI MULTIMEDIA INTERAKTIF UNTUK PEMBELAJARAN TENSES BAHASA INGGRIS

Beki Subaeki¹⁾, Dicky Ardiansyah²⁾

Program Studi Teknik Informatika¹⁾²⁾

Universitas Sangga Buana YPKP¹⁾, UIN Sunan Gunung Djati²⁾

Beki807@gmail.com¹⁾, dicky.ardiansyah55@gmail.com²⁾

ABSTRAK

Bahasa Inggris adalah Bahasa Internasional sekaligus bahasa universal yang digunakan oleh berbagai negara di dunia. Hal ini membuat Bahasa Inggris juga menjadi bahasa asing yang sering kita pelajari dari jenjang sekolah dasar hingga perguruan tinggi. Dalam Bahasa Inggris, *Time* (waktu) sangat erat hubungannya dengan *Tense* (perubahan kata kerja). Sehingga mempelajari *tenses*, juga merupakan modal dasar bagi seseorang untuk menguasai konteks kalimat Bahasa Inggris. Dalam skripsi ini media penyampaian informasi mengenai materi *tenses* dibuat dalam bentuk PBK (Pembelajaran Berbantuan Komputer), yakni Multimedia Interaktif. Multimedia Interaktif dapat memberikan sebuah interaktifitas antara pengguna dengan aplikasi. Interaktifitas yang dibuat dalam aplikasi pembelajaran *tenses* ini berupa simulasi soal untuk mengasah tingkat pemahaman materi *tenses* pengguna. Soal uji pemahaman dibuat ke dalam dua (2) kategori, yakni soal latihan dan soal tes evaluasi menggunakan format penyimpanan data standar berbasis *.XML* (*eXtensible Markup Language*). Implementasi algoritma acak (*shuffling*) pada pembuatan aplikasi digunakan untuk mengacak urutan soal uji pemahaman. Algoritma acak (*shuffling*) yang dipakai adalah algoritma *Fisher-Yates Shuffle*. Algoritma *Fisher-Yates Shuffle* dapat mengacak urutan data bank soal *.xml* yang dimasukkan kedalam *array* dan dipanggil menggunakan bahasa pemrograman *ActionScript 3.0*, sehingga soal yang diacak akan menghasilkan permutasi yang berbeda dan memiliki hasil bias.

Kata kunci : *Tenses*, PBK (Pembelajaran Berbantuan Komputer), Multimedia Interaktif, Soal Uji Pemahaman, *.XML* (*eXtensible Markup Language*), *ActionScript 3.0*, Algoritma *Fisher-Yates Shuffle*.

I. PENDAHULUAN

Bahasa Inggris adalah Bahasa Internasional sekaligus Bahasa *Universal* yang digunakan oleh berbagai Negara di dunia. Dalam Bahasa Inggris, *Time* (waktu) sangat erat kaitannya dengan *Tense* (perubahan kata kerja). Sehingga mempelajari *Tenses*, juga merupakan modal dasar bagi seseorang yang ingin dapat mengerti atau menguasai konteks kalimat dalam Bahasa Inggris.

Orang yang belajar atau menuntut ilmu, mendapat kehormatan di sisi Allah dan Rosulnya. Sebagaimana firman Allah S.W.T:

يَرْفَعُ اللَّهُ الَّذِينَ آمَنُوا مِنْكُمْ وَالَّذِينَ أُوتُوا الْعِلْمَ دَرَجَاتٍ وَاللَّهُ بِمَا تَعْمَلُونَ خَبِيرٌ

Artinya:

“Allah akan meninggikan orang-orang

yang beriman di antaramu dan orang-orang yang diberi ilmu pengetahuan beberapa derajat” (Q.s. al-Mujadalah : 11)

Serta sabda Nabi Muhammad s.a.w:

(أَطْلَبُ الْعِلْمَ مِنَ الْمَخْدِ إِلَى اللَّهِّدِ) (رواه مسلم)

Artinya:

“Carilah ilmu dari buaian sampai liang lahat”(HR. Muslim)

Ilmu pengetahuan yang semakin berkembang dan Teknologi Komputer yang semakin maju ikut serta menghantarkan penyampaian Informasi melalui berbagai media, termasuk media Pembelajaran Berbantuan Komputer (PBK). Multimedia sebagai salah satu jenis Pembelajaran Berbantuan Komputer yang mampu memberikan *simulation* (simulasi) dan *drill and practic* (praktik dan latihan), menjadikan

multimedia sebagai media penyampaian informasi yang dapat memberikan interaktifitas ke pengguna.

Soal uji pemahaman adalah bentuk dari *simulation* juga *drill and practic*, yang dapat dibuat dalam bentuk aplikasi Multimedia Interaktif. Karena soal uji pemahaman ini merupakan poin penting bagi pengguna aplikasi pembelajaran, agar dapat mengukur seberapa jauh tingkat pemahaman terhadap materi yang sedang dipelajarinya. Diperlukan cara atau jalan untuk mengubah urutan soal uji pemahaman, yakni dengan menggunakan algoritma. Algoritma ini harus dapat mengubah atau mengacak urutan soal uji pemahaman yang disajikan.

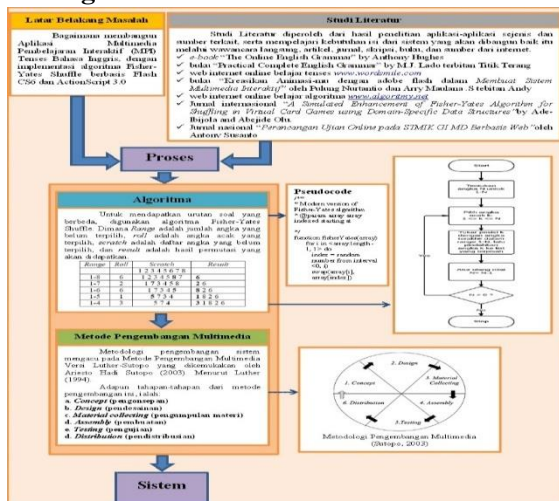
Algoritma acak (*shuffling*) *Fisher-Yates Shuffle*, dapat mengacak urutan suatu data yang dimasukkan kedalam *array*. Algoritma ini banyak digunakan dalam aplikasi permainan yang membutuhkan metode pengacakan, seperti permainan puzzle dan kartu. Karena algoritma ini bersifat bias (kecil kemungkinan tampil dengan urutan/posisi sama).

Peneliti: Ade-Ibijola, Abejide Olu. *A Simulated Enhancement of Fisher-Yates Algorithm for Shuffling in Virtual Card Games using Domain-Specific Data Structures*, International Journal of Computer Applications, 2012.

“Memberikan informasi mengenai keberhasilan dari proses algoritma acak *Fisher-Yates Shuffle* dalam mengacak urutan data dan menghasilkan permutasi berbeda, yang disimulasikan ke dalam *game* kartu virtual”.

(Sumber: <https://www.academia.edu>, diakses 22/10/2013, 05:55)

Kerangka Pemikiran:



Gambar 1. Kerangka Pemikiran

II. DASAR TEORI

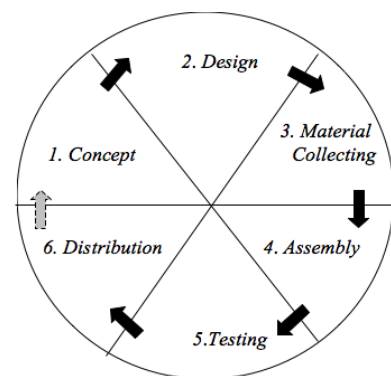
Metode Pengembangan Multimedia

Selanjutnya untuk metodologi pengembangan sistem mengacu pada Metode Pengembangan Multimedia Versi Luther-Sutopo yang dikemukakan oleh Ariesto Hadi Sutopo (2003). Menurut Luther (1994), metodologi pengembangan multimedia terdiri dari enam tahap, yaitu:

1. Pengonsepan (*Concept*)
2. Pendesainan (*Design*)
3. Pengumpulan Materi (*Material collecting*)
4. Pembuatan (*Assembly*)
5. Pengujian (*Testing*)
6. Pendistribusian (*Distribution*)

Keenam tahap ini tidak harus berurutan dalam praktiknya, tahap-tahap tersebut dapat saling bertukar posisi. Meskipun begitu, tahap *concept* memang harus menjadi hal yang pertama kali dikerjakan.

Sutopo (2003) mengadopsi metodologi Luther dengan modifikasi, seperti yang terlihat pada Gambar 2



Gambar 2. Metodologi Pengembangan Multimedia (Sutopo, 2003)

1. Pengonsepan (*Concept*)

Tahap Pengonsepan (*Concept*) adalah tahap untuk menentukan tujuan dan siapa pengguna program (identifikasi *audience*). Selain itu menentukan macam aplikasi (presentasi, interaktif, dll) dan tujuan aplikasi (hiburan, pelatihan, pembelajaran, dll).

2. Pendesainan (*Design*)

Tahap Pendesainan (*Design*) adalah tahap membuat spesifikasi mengenai arsitektur program, gaya, tampilan dan kebutuhan material/bahan untuk program.

3. Pengumpulan Materi (*Material collecting*)

Tahap Pengumpulan Materi (*Material collecting*) adalah tahap dimana pengumpulan bahan yang sesuai dengan kebutuhan dilakukan. Tahap ini dapat dikerjakan paralel dengan tahap *assembly*. Pada beberapa kasus, tahap *Material Collecting* dan tahap *Assembly* akan dikerjakan secara linear tidak paralel.

4. Pembuatan (*Assembly*)

Tahap *assembly* (pembuatan) adalah tahap dimana semua objek atau bahan multimedia dibuat. Pembuatan aplikasi didasarkan pada tahap *design*.

5. Pengujian (*Testing*)

Dilakukan setelah selesai tahap pembuatan (*assembly*) dengan menjalankan aplikasi/program dan dilihat apakah ada kesalahan atau tidak. Tahap ini disebut juga sebagai tahap pengujian *alpha* (*alpha test*) dimana pengujian dilakukan oleh pembuat atau lingkungan pembuatnya sendiri.

6. Pendistribusian (*Distribution*)

Tahapan dimana aplikasi disimpan dalam suatu media penyimpanan. Pada tahap ini jika media penyimpanan tidak cukup untuk menampung aplikasinya, maka dilakukan kompresi terhadap aplikasi tersebut.

III. ICONCEPT DAN PERANCANGAN

Algoritma Fisher-Yates Shuffle

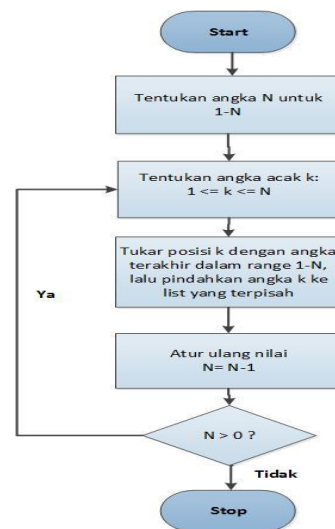
Analisis algoritma *Fisher-Yates Shuffle*, dimaksudkan untuk memberikan gambaran konsep dari algoritma acak (*shuffling*) jika diterapkan pada Aplikasi Multimedia *Tenses* Bahasa Inggris. Analisis Algoritma *Fisher-Yates Shuffle* menggunakan array, dapat dilihat pada Gambar 3

| Kondisi Array | | | | | Satu data acak yang diambil | Keterangan |
|---------------|---|---|---|---|-----------------------------|---|
| A | B | C | D | E | C | Semua data di sebelah kanan C bergeser satu kali ke kiri. |
| 0 | 1 | 2 | 3 | 4 | 2 | |
| A | B | D | E | | | Isikan data selain yang sudah diambil. Berikutnya, nomor indeks dapat disesuaikan lagi. |
| 0 | 1 | 2 | 3 | | | |

Gambar 3. Analisis algoritma menggunakan *array*

Dari Gambar 3, diketahui bahwa *array* pertama berukuran 5 (berisi 5 buah data), diberi indeks atau nomor mulai dari 0 sampai 4. Berikutnya buat array baru dengan menimpa *array* lama, ukurannya N-1.

Dalam proses pengacakan, Algoritma *Fisher-Yates Shuffle* akan dijelaskan menggunakan *Flowchart*. Akan dijelaskan pada Gambar 4.



Gambar 4. Flowchart Algoritma Fisher-Yates Shuffle

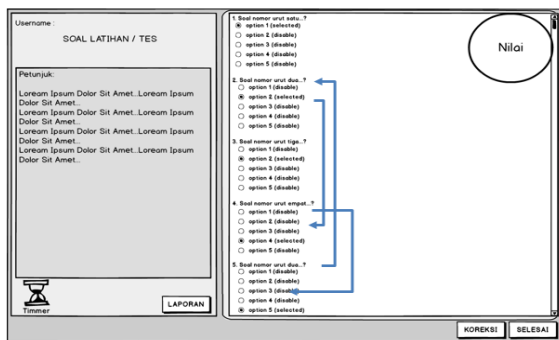
Dari Gambar 4, diketahui bahwa algoritma *Fisher-Yates Shuffle* dalam mengacak urutan data diawali dengan menentukan angka N untuk range 1-N. Setelah itu tentukan angka acak k dari range 1-N, tidak boleh 0 dan melebihi nilai N. Tukar posisi k dengan angka terakhir range 1-N ditempat terpisah. Atur ulang N, N=N-1. Jika N bernilai 0, maka hentikan proses pengacakan.

Berikut penjelasan dalam bentuk tabel urutan soal. Terdapat 5 buah soal dalam bank soal XML, dimana urutannya ter-urut dari 1-5. Akan di *suffling* (acak), sehingga urutan yang ditampilkan akan berbeda. *Range* adalah jumlah angka yang belum terpilih, *Roll* adalah angka acak yang terpilih, *Scratch* adalah daftar angka yang belum terpilih, dan *Result* adalah hasil permutasi yang didapatkan. Untuk mengetahui analisis pengacakan urutan soal, Perhatikan Tabel 1:

Tabel 1. Analisis pengacakan urutan soal

| Range | Roll | Scracth | Result |
|-------|------|-----------|---------|
| | | 1 2 3 4 5 | |
| 1-5 | 4 | 1 2 3 5 | 4 |
| 1-4 | 2 | 1 5 3 | 2 4 |
| 1-3 | 3 | 1 5 | 3 2 4 |
| 1-2 | 2 | 1 | 5 3 2 4 |

Dari tabel di atas, maka didapatkan hasil permutasi 1 5 3 2 4. Sehingga walaupun urutan soal dibank soal XML berurut, namun akan ditampilkan berbeda oleh program pada halaman Soal Latihan dan Tes. Untuk lebih jelas dalam memahami proses mengacak (*shuffling*) urutan pada form soal. Perhatikan Gambar 5:



Gambar 4. Halaman Soal Latihan/Tes

Dengan melihat alur algoritma menggunakan *flowchart* dan tabel konsep pengacakan urutan soal, berikut adalah analisis *pseudocode* dari algoritma *Fisher-Yates Shuffle*. Perhatikan Gambar 6:

```

1  Algoritma Fisher-Yates Shuffle
2  {Algoritma untuk mengacak (shuffling) data. Digunakan untuk
3  mengubah urutan soal latihan dan tes, sehingga pertanyaan
4  yang ditampilkan urutannya akan berbeda}
5
6  function fish(a):Array
7  {tentukan banyaknya a}
8
9  DEKLARASI
10     i:integer
11     j:integer;
12     t:Number;
13     temp:Array
14
15  ALGORITMA:
16  i:integer ← a.length {masukkan banyaknya anggota a}
17  temp:Array ← new Array() {tempat penampungan sementara nilai
18  masukan}
19
20     while (--i)
21     {mulai pengacakan}
22         t ← a[i]
23         j ← Math.random() * (i + 1) {masukan nilai random}
24         a[i] ← a[j]
25         a[j] ← t
26     endwhile
27     {kondisi berhenti pengacakan}
28
29     return a
30
    
```

Gambar 6 Pseudocode Algoritma Fisher-Yates Shuffle

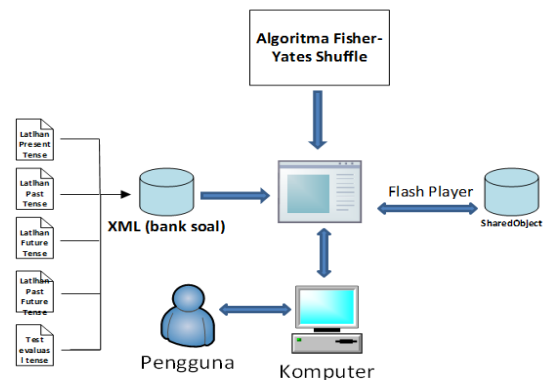
Fungsi *fish(a)* bertipe *array* dengan parameter *a* sebagai nilai masukkan yang akan diacak. Algoritma di atas dilanjutkan oleh deklarasi yaitu *i* sebagai *integer* (untuk memasukkan nilai banyaknya anggota *a*), *j* sebagai *integer*, *t* sebagai *number*, dan *temp* sebagai Array sementara untuk menampung nilai masukkan.

Kemudian dilanjutkan dengan alur algoritma yang melakukan pengulangan *while* dengan kondisi jika *--i* adalah benar. Pada algoritma tersebut, dilakukan penukaran antara *t*, *j* dan *a*. *t* dimasukkan anggota *a* ke-*i*, *j* dimasukkan nilai random, nilai *a[i]* dimasukkan nilai *a[j]*, sedangkan *a[j]* dimasukkan nilai *t*.

Pengulangan *while* tersebut berakhir ketika *--i* bernilai *false* (bernilai kurang dari 0). Setelah nilai teracak, maka akan dikembalikan melalui *return a*.

Arsitektur Sistem

Arsitektur sistem adalah istilah untuk menyatakan bagaimana mendefinisikan konsep dari komponen-komponen yang lebih spesifik secara terstruktur dalam menggambarkan alur sistem dan pengguna. Arsitektur sistem dari Aplikasi Multimedia Interaktif Pembelajaran *Tenses* Bahasa Inggris ini, dapat dilihat pada Gambar 7:



Gambar 7. Arsitektur Sistem

Arsitektur sistem di atas dapat dijelaskan sebagai berikut:

1. Algoritma *Fisher-Yates Shuffle* diimplementasikan untuk

- mengacak (*shuffling*) soal dari XML.
2. XML berfungsi sebagai format penyimpanan data standar untuk menyimpan bank soal latihan dan tes evaluasi.
 3. *SharedObject* dibuat untuk *save* dan *update username* username dari form Pilih User serta skor nilai dari Form Soal Latihan dan Tes.
 4. Aplikasi Multimedia Pembelajaran Interaktif *Tenses Bahasa Inggris* ditampilkan oleh *Flash Player* yang terinstal di komputer.
 5. Pengguna berinteraksi langsung dengan komputer secara *stand alone* yang berarti aplikasi tidak terhubung ke jaringan ataupun *server*, karena Aplikasi Multimedia ini bersifat *SharedObject*.

Bank Soal

Sedangkan untuk kebutuhan data pada aplikasi multimedia ini, akan dibuat bank soal yang disimpan dalam bentuk file .xml yang merupakan format penyimpanan data standar. Perhatikan Gambar 8:

| No. | Simple Present Tense | |
|-----|---|---|
| | Soal Latihan | Soal Tes |
| 1 | 1. _____grandfather drink coffee every morning? a. Have b. Had c. Did d. Does e. Would Jawaban: d. Does | 1. My father washes his car every week. (ubah ke dalam kalimat negatif) a. My father was not washing his car every week. b. My father will not washing his car every week. c. My father doesn't wash his car every week. d. My father Hadn't wash his car every week. e. My father didn't wash his car every week. |
| 2 | 2. Do you speak English? a. Yes I do b. No, I does c. Yes, I can d. No, I will e. Yes, I have Jawaban: a. Yes I do | 2. Does my mother cook meal in the morning? a. Yes, she did b. No, she didn't c. Yes, she was d. No, she hadn't e. No, she doesn't Jawaban: e. No, she doesn't |
| 3 | 3. My grandpa _____a newspaper everyday. (read) a. hadn't read b. wasn't read c. didn't read d. don't read e. doesn't read | 3. My brother Murdoch _____ fishing everyday.(go) a. goes b. gone c. will gone d. had gone e. going |
| 4 | 4. _____Farida go to school everyday? a. Did b. Does c. Do d. Had e. Will Jawaban: b. Does | 4. The sun shines brightly. (ubah ke dalam kalimat tanya?) a. Has the sun shine brightly? b. Have the sun shine brightly? c. Do the sun shine brightly? d. Does the sun shine brightly? e. Did the sun shine brightly? |
| 5 | 5. Do we clean our classroom everyday? a. yes, we did b. yes, we does c. yes, we do d. no, we doesn't e. no, we didn't Jawaban: c. yes, we do | 5. Does the mother kiss both boys every morning? (ubah ke dalam kalimat negatif) a. The Mother will not kiss both boys every morning b. The Mother had not kiss both boys every morning c. The Mother didn't kiss both boys every morning d. The Mother don't kiss both boys every morning e. The Mother doesn't kiss both boys every morning Jawaban: e. The Mother doesn't kiss both boys every |

Gambar 8. Bank Soal

Akan dibuatkan sebanyak delapan puluh (80) soal latihan yang akan dibagi menjadi empat (4) sub soal latihan tenses berdasarkan

materi yang dipelajari. Sedangkan untuk soal tes berjumlah delapan puluh (80). Jadi total keseluruhan bank soal Aplikasi Multimedia Tenses ini, adalah seratus enam puluh (160) soal.

IV. IMPLEMENTASI DAN PENGUJIAN

Pengujian Antarmuka (Interface)

Pengujian antarmuka (*interface*) dilakukan untuk mengetahui, apakah *interface* yang dirancang pada tahap *Design* (pendisainan) dan *Material Collecting* (pengumpulan materi) sudah sesuai dengan tujuan pembuatan perangkat lunak.

a. *Interface* Main Menu

Tampilan dimana saat *user* menjalankan aplikasi, maka *user* akan langsung diarahkan ke halaman Main Menu. Untuk melihat *Interface* Main Menu, perhatikan Gambar 4.1:



Gambar 8. Tampilan main menu

Dari Gambar 8, diketahui *interface* Main Menu menyajikan tombol menu kelola *user* (pilih *user*), pilih materi, tes (*test*), lihat nilai (*info*), lihat pembuat (*about*), panduan (*how to use*), kelola sistem (*setting*), dan tombol *exit*. Serta beberapa konten grafis tahapan *material collecting* yang dimasukkan kedalam halaman.

b. *Interface* Kelola User (Pilih User)

Tampilan dimana saat *user*, menekan tombol menu Pilih User untuk membuat dan menghapus *username*. *Interface* dari menu Kelola User dapat dilihat pada Gambar 9:



Gambar 9. Tampilan pilih user

c. Interface Materi Tenses

Tampilan dimana user menekan tombol materi tenses yang dipilihnya. Interface dari halaman materi tenses, dapat dilihat pada Gambar 10:



Gambar 10 Tampilan materi tenses

d. Interface Lihat Nilai (Info)

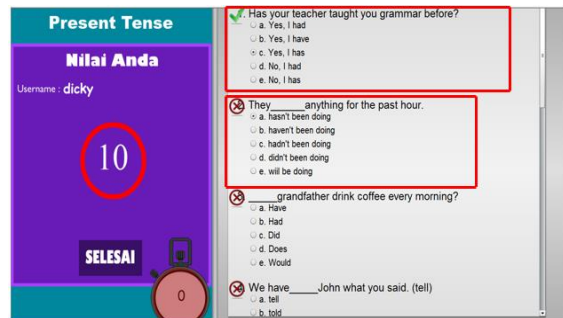
Tampilan dimana user ingin melihat histori nilai soal latihan dan tes, dengan menekan tombol Menu Info. Interface lihat nilai dapat dilihat, pada Gambar 11:



Gambar 11. Tampilan lihat nilai

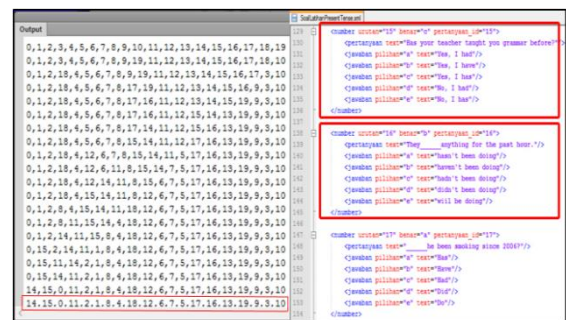
Pengujian Algoritma Fisher-Yates Shuffle

Pengujian Algoritma Fisher-Yates Shuffle dilakukan, untuk melihat kinerja dari algoritma acak (shuffling) dalam mengacak array soal yang dipanggil dari bank soal .xml. Pengujian dilakukan dengan simulasi pengerjaan soal, perhatikan Gambar 12:



Gambar 12. Simulasi pengerjaan soal

Dari Gambar 12, telah dilakukan pengujian untuk melihat hasil kinerja dari algoritma Fisher-Yates Shuffle dengan melihat output hasil compile Aplikasi pada saat proses run. Urutan yang ditampilkan lembar soal disesuaikan dengan urutan array pada layar output dan urutan bank soal xml. Perhatikan Gambar 13:



Gambar 13 Tampilan Output Array Aplikasi

Dari Gambar 13, setelah melakukan proses acak (shuffling) array sebanyak range 20 (jumlah soal latihan). Dapat dilihat posisi array pada output, memiliki kesamaan dengan posisi soal pada bank soal .xml (n+1). Urutan array terakhir pada output inilah yang ditampilkan pada form soal (lihat Gambar 4.5).

Telah dilakukan pengujian lain, untuk melihat perpindahan step by step array yang berpindah posisi. Namun masih tetap dengan file .xml yang memiliki jumlah soal sebanyak 20 buah. Perhatikan Gambar 14:

| | |
|----|---|
| 1 | 0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19 |
| 2 | 0,1,2,3,4,5,6,7,19,9,10,11,12,13,14,15,16,17,18,8 |
| 3 | 0,1,2,3,4,5,6,7,19,9,10,18,12,13,14,15,16,17,11,8 |
| 4 | 0,1,2,3,4,5,6,7,19,9,10,18,12,17,14,15,16,13,11,8 |
| 5 | 0,1,2,3,4,5,6,7,19,9,10,18,12,16,14,15,17,13,11,8 |
| 6 | 0,1,2,3,4,5,6,7,19,9,10,18,12,15,14,16,17,13,11,8 |
| 7 | 0,1,2,14,4,5,6,7,19,9,10,18,12,15,3,16,17,13,11,8 |
| 8 | 0,1,2,14,4,5,6,15,19,9,10,18,12,7,3,16,17,13,11,8 |
| 9 | 0,1,12,14,4,5,6,15,19,9,10,18,2,7,3,16,17,13,11,8 |
| 10 | 0,1,12,14,4,5,6,18,19,9,10,15,2,7,3,16,17,13,11,8 |
| 11 | 0,10,12,14,9,5,6,18,19,4,1,15,2,7,3,16,17,13,11,8 |
| 12 | 0,10,12,14,9,5,6,18,19,4,1,15,2,7,3,16,17,13,11,8 |
| 13 | 0,10,12,14,9,5,6,19,18,4,1,15,2,7,3,16,17,13,11,8 |
| 14 | 19,10,12,14,9,5,6,0,18,4,1,15,2,7,3,16,17,13,11,8 |
| 15 | 19,6,12,14,9,5,10,0,18,4,1,15,2,7,3,16,17,13,11,8 |
| 16 | 19,5,12,14,9,6,10,0,18,4,1,15,2,7,3,16,17,13,11,8 |
| 17 | 9,5,12,14,19,6,10,0,18,4,1,15,2,7,3,16,17,13,11,8 |
| 18 | 14,5,12,9,19,6,10,0,18,4,1,15,2,7,3,16,17,13,11,8 |
| 19 | 14,12,5,9,19,6,10,0,18,4,1,15,2,7,3,16,17,13,11,8 |
| 20 | 12,14,5,9,19,6,10,0,18,4,1,15,2,7,3,16,17,13,11,8 |

Gambar 14. Output perpindahan posisi array

Dari Gambar 14, dapat dilihat perpindahan posisi array yang terjadi. Posisi array yang sudah ditukar dengan array lain akan terus tetap ditukar, dan tidak terpengaruh oleh posisi array lain yang belum ditukar sampai pertukaran selesai dilakukan sebanyak 20 kali (disesuaikan dengan banyaknya jumlah soal). Urutan array di posisi ke-20 sudah berbeda jauh dengan urutan awal sebelum di acak (*shuffling*).

Dari 20 array yang di *shuffling*, telah dilakukan pengujian sebanyak 10 kali percobaan untuk mengetahui permutasi yang dihasilkan. Pada tabel urutan soal, array akan ditambahkan 1 atau (n+1), agar mempermudah memahami pengacakan urutan soal. Perhatikan Gambar 15:

| | Permutasi | | | | | | | | | | |
|---|-----------|----|----|----|----|----|----|----|----|----|----|
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | |
| U | 1 | 13 | 19 | 10 | 16 | 2 | 9 | 5 | 13 | 6 | 7 |
| R | 2 | 15 | 16 | 2 | 11 | 1 | 8 | 12 | 2 | 20 | 12 |
| U | 3 | 6 | 11 | 4 | 13 | 6 | 19 | 20 | 20 | 16 | 13 |
| T | 4 | 10 | 8 | 18 | 14 | 3 | 2 | 19 | 18 | 15 | 6 |
| A | 5 | 20 | 12 | 16 | 5 | 14 | 10 | 11 | 10 | 14 | 3 |
| N | 6 | 7 | 7 | 20 | 19 | 12 | 12 | 2 | 6 | 7 | 19 |
| S | 7 | 11 | 4 | 15 | 12 | 17 | 15 | 14 | 3 | 17 | 9 |
| O | 8 | 1 | 9 | 11 | 1 | 13 | 16 | 18 | 16 | 19 | 15 |
| A | 9 | 19 | 5 | 3 | 9 | 18 | 11 | 13 | 15 | 12 | 5 |
| L | 10 | 5 | 13 | 12 | 17 | 5 | 20 | 4 | 8 | 11 | 2 |
| | 11 | 2 | 10 | 7 | 6 | 9 | 4 | 15 | 9 | 3 | 20 |
| | 12 | 16 | 20 | 8 | 8 | 11 | 1 | 17 | 12 | 4 | 4 |
| | 13 | 3 | 18 | 17 | 15 | 20 | 3 | 7 | 11 | 10 | 14 |
| | 14 | 8 | 1 | 9 | 4 | 10 | 17 | 10 | 1 | 2 | 18 |
| | 15 | 4 | 3 | 19 | 10 | 16 | 13 | 16 | 17 | 8 | 16 |
| | 16 | 17 | 15 | 13 | 7 | 15 | 7 | 1 | 14 | 18 | 10 |
| | 17 | 18 | 14 | 5 | 2 | 8 | 5 | 9 | 19 | 9 | 17 |
| | 18 | 14 | 2 | 14 | 20 | 4 | 14 | 3 | 5 | 5 | 11 |
| | 19 | 12 | 17 | 1 | 3 | 7 | 18 | 8 | 7 | 1 | 8 |
| | 20 | 9 | 6 | 6 | 18 | 19 | 6 | 6 | 4 | 13 | 1 |

Gambar 15. Hasil Pengujian acak (*shuffling*) Soal

Dari Gambar 15, dapat dilihat 10 permutasi acak (*shuffling*) hasil percobaan.

Algoritma *Fisher-Yates Shuffle* menghasilkan urutan soal teracak, dan dari hasil 10 kali *shuffling* tersebut tidak ada urutan soal yang sama.

Pengujian Alpa (*Black Box Testing*)

Pengujian dilakukan terhadap aplikasi untuk memastikan bahwa aplikasi dapat berjalan dengan benar sesuai dengan kebutuhan dan tujuan yang diharapkan. Pengujian ini dilakukan secara *black box* yaitu pengujian yang berfokus pada persyaratan fungsional perangkat lunak. Akan dijelaskan pada Tabel 4.1:

Tabel 2. Uji Menu Aplikasi Multimedia

| NO | SKENARIO | HASIL | | KETERANGAN |
|----|---------------------|--------|-------|------------|
| | | SUKSES | GAGAL | |
| 1 | Main Menu | √ | | |
| 2 | Menu Pilih User | √ | | |
| 3 | Pilih Materi Tenses | √ | | |
| 4 | Soal Latihan | √ | | |
| 5 | Menu Test | √ | | |
| 6 | Menu Info | √ | | |
| 7 | Menu About | √ | | |
| 8 | Menu How to Use | √ | | |
| 9 | Menu Setting | √ | | |
| 10 | Exit | √ | | |

V. PENUTUP

Kesimpulan

Setelah melakukan penelitian dalam skripsi ini, penulis dapat mengambil kesimpulan sebagai berikut:

- Aplikasi Multimedia *Tenses* Bahasa Inggris ini, dapat dibangun menggunakan *Adobe Flash CS6* dengan bahasa pemrograman *ActionScript3.0*. Dapat dikoneksikan dengan bank soal, menggunakan format penyimpanan data standar berbasis *XML (eXtensible Markup Language)*. Untuk konten *Multimedia* yang disajikan seperti *audio*, gambar, dan animasi dapat dibuat dengan *software* pendukung *Adode Family (Photoshop CS6, Illustrator CS6, dan Audtion CS6)*.
- Algoritma *Fisher-Yates Shuffle*, dapat digunakan untuk mengacak (*shuffling*) data berupa soal dari file *XML*. Dengan mengacak urutan array yang dipanggil oleh bahasa pemrograman *ActionScript3.0* dari bank soal *.xml*. Array yang sudah dipindahkan pada setiap tahapan (*step*) berdasarkan jumlah *range*, disimpan ke *temporary array* sementara sampai *range array* yang ditentukan bernilai nol. Maka akan

terlihat urutan *array* di awal akan berbeda dengan urutan *array* di akhir. Meskipun menggunakan skala *range I-N*, serta nilai *k* ditentukan berdasarkan *range* (jumlah) soal. Namun urutan soal berikutnya tidak bisa ditebak, karena algoritma *Fisher-Yates Shuffle* menggunakan *math random* untuk mengubah urutan *array* yang dipanggil.

Saran

Berdasarkan hasil penelitian dapat diperoleh saran sebagai berikut:

- a. Dari segi Multimedia, aplikasi ini sudah memenuhi unsur multimedia sebagai *Content Production (teks, audio, graphics, animation, video dan interactivity)*. Namun untuk segi materi Bahasa Inggris, perlu dilengkapi menjadi *English Grammar* dan tidak sebatas materi *Tenses*.
- b. Untuk penelitian lebih lanjut, Aplikasi Multimedia Interaktif ini dapat dikembangkan lagi menjadi aplikasi pembelajaran materi lain (*Reusability*).

DAFTAR PUSTAKA

- [1] Ade-Ibijola, Abejide Olu. 2012. *A Simulated Enhancement of Fisher-Yates Algorithm for Shuffling in Virtual Card Games using Domain-Specific Data Structures*, International Journal of Computer Applications.
- [2] All English Tenses (Revision). http://www.anglo-link.com/index.php/?online_lessons/video_lessons (diakses tanggal 11 agustus 2014, 01:20)
- [3] Ambler, S.W., 2005, *The Elements of UML 2.0 Style*, Cambridge University Press, Cambridge.
- [4] Arsyad, Azhar. 2007. *Media Pembelajaran*. Jakarta: PT Rajagrafindo Persada.
- [5] Binanto, I., 2010, *Multimedia Digital Dasar Teori dan Pengembangannya*, Penerbit Andi, Yogyakarta.
- [6] Black, Paul E. (2005-12-19). "[Fisher-Yates shuffle](#)". *Dictionary of Algorithms and Data Structures*. [National Institute of Standards and Technology](#). Retrieved 2007-08-09. (diakses tanggal 22 september 2013, 22:52)
- [7] Durstenfeld, Richard (July 1964). "Algorithm 235: Random permutation". *Communications of the ACM* 7 (7): 420. (diakses tanggal 22 september 2013, 22:52)
- [8] Fisher, R.A.; Yates, F. (1948) [1938]. *Statistical tables for biological, agricultural and medical research* (3rd ed.). London: Oliver & Boyd. pp. 26–27. (diakses tanggal 22 september 2013, 22:52)
- [9] *Fisher-Yates Shuffle*. <http://en.algoritmy.net/article/43676/Fisher-Yates-shuffle> (diakses tanggal 22 september 2013, 22:52)
- [10] Herlawati, P.P.W., 2011, *Menggunakan UML*, Penerbit Informatika, Bandung.
- [11] Hughes, A., 2011, *The Online English Grammar*, <http://www.english4today.com/grammar/> (diakses tanggal 3 januari 2014, 10:30)