Aplikasi Algoritma Greedy dalam Pengumpulan *Flower Events* pada Permainan *Harvest Moon: The Tale of Two Towns*

Karina Iswara 13517031
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung. Jl. Ganesha 10 Bandung 40132, Indonesia
karinaiswara.ki@gmail.com

***Abstract* — Algoritma greedy merupakan salah satu algoritma yang umum digunakan dalam pencarian solusi optimum. algoritma greedy mengambil pilihan paling optimal di setiap langkahnya dalam pemecahan persoalan optimasi. Algoritma greedy tidak selau menghasilkan jawaban yang paling optimum melainkan *sub-optimum* atau *pseudo-optimum*.**

Permainan *Harvest Moon: The Tale of Two Towns* merupakan sebuah permainan video simulasi dalam kehidupan bertani. Namun, selain bertani, dalam permainan video ini kita memiliki alur cerita yang dapat berkembang seiring berjalannya permainan. Perkembangan alur cerita dalam permainan video ini bergantung pada kedekatan pemain dengan beberapa karakter di dalam permainan ini yang ditentukan dengan *friendship points(fp)*. *Friendship points* ini lah yang dapat memacu terjadi perkembangan alur cerita yang pada Permainan Harvest Moon: The Tale of Two Towns disebut *Flower Event,* atau kejadian yang terjadi dengan syarat-syarat tertentu.

Makalah ini akan membahas mengenai aplikasi algoritma greedy untuk mendapatkan *flower event* terbanyak dalam perjalanan pemain dari suatu tempat menuju beberapa tempat lainnya dalam menjalankan misi-misi utama dalam permainan.

***Kata kunci* — Algoritma Greedy, *Harvest Moon: The Tale of Two Towns, flower event, friendship points(fp)*.**

I. PENDAHULUAN

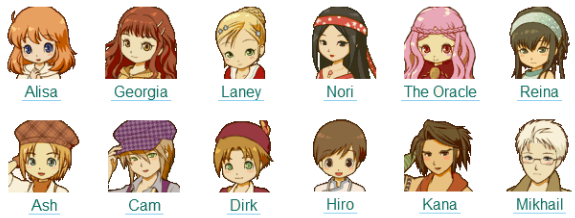
Permainan *Harvest Moon: The Tale of Two Towns* merupakan salah satu permainan dari serian permainan *Harvest Moon. P*ermainan ini merupakan permainan berjenis simulasi yang dikembangkan oleh Marvelous Entertainent pada platform Nintendo DS, Nintendo 3DS. Permainan ini menyimulasikan kehidupan bertani kepada para pemain.

Meskipun permainan ini terkesan simpel, dan mudah untuk dimainkan, namun *Harvest Moon: The Tale of Two Towns* tidak sesimpel yang dikira. permainan ini selain membutuhkan pengaturan waktu yang tepat, ia juga membutuhkan juga membutuhkan kreativitas dan kecerdikan dalam melakukan berbagai hal, seperti diantaranya ialah dalam berinteraksi dengan para karakter dari dalam permainan. Interaksi pemain dengan karakter tertentu dari dalam permainan dapat mengubah *friendship points* yang dimiliki pemain dengan karakter tersebut.. *Friendship points* dalam game ini seolah-olaha melambangkan kedekatan pemain dengan karakter tersebut. Jika dalam salah satu interaksi pemain tidak beinteraksi dengan tepat maka *friendship points* pemain dengn karakter tersebut justru bisa berkurang meskipun pemain berinteraksi dengan karakter tersebut dalam kuantitas yang banyak. Karakter-karakter dalam *Harvest Moon: The Tale of Two Towns* memiliki karakteristiknya masing-masing. Mereka memiliki kesukaan mereka masing-masing.

Dalam permainan yang menyimulasikan kehidupan bertani ini, terdapat juga faktor lain kehidupan lain selain bertani yang dapat didapatkan oleh para pemain, diantaranya ialah menikah. Pemain dapat menikah dengan karakter khusus memang sudah disediakan dalam permainan. Dalam permainan *Harvest Moon: The Tale of Two* Towns terdapat 6 karakter perempuan (jika pemain memilih untuk menjadi petani laki-laki) atau 6 karakter laki-laki (jika pemain memilih untuk menjadi petani perempuan) yang dapat dipilih oleh pemain untuk dinikahi. Namun untuk menikahi salah satu dari mereka tidaklah semudah itu. Seperti halnya orang menikah dalam kehidupan nyata yakni orang menikah dengan mereka yang dianggap cocok dan tentunya sudah tergolong dekat dengan diri mereka, permainan ini mengimplementasikan hal tersebut ke dalam permainan. Pemain harus sudah tergolong dekat dan cocok pada karakter yang ingin mereka nikahi. Untuk membuktikan kedekatan pemain dengan karakter, pemain membutuhkan *flower events* dengan karakter tersebut dan untuk membuat *flower events* terjadi, pemain membutuhkan *friendship points* yang sesuai dengan syarat-syarat *flower events*.

Gambar 1.1Tampilan *Harvest Moon: The Tale of Two Towns*

(sumber: https://www.nintendo.co.uk/Games/Nintendo-3DS/Harvest-Moon-The-Tale-of-Two-Towns-527624.html)



Gambar 1.2 Karakter dalam *Harvest Moon: The Tale of Two Towns* yang dapat dinikahi oleh pemain

(sumber: http://lang8088.blogspot.com/2014/03/harvest-moon-tale-of-two-towns-menikah.html)

Pada makalah ini, penulis ingin membuat algoritma sederhana yang menggunakan algoritma greedy untuk mendapatkan *flower events* terbanyak dalam rute pemain menjalankan misi-misi utama di simulasi bertani dalam permainan *Harvest Moon: The Tale of Two Towns*.

II. Landasan Teori

A. Algoritma Greedy

Algoritma greedy merupakan salah satu algoitma yang paling popular untuk dipakai dalam persoalan optimasi, yakni persoalan yang mencari solusi optimum. Terdapat dua macam persoalan optimasi, yaitu:

1. Maksimasi (*maximization),* yaitu mencari angka maksimum dalam suatu persoalan.
2. Minimasi (*minimization),* yaitu mencari angka minimum dalam suatu persoalan.

Algoritma greedy membentuk solusinya langkah per langkah (*step by step*) dan pada setiap langkah terdapat banyak pilihan yang perlu dievaluasi. Dengan prinsip: *take what you can get now*, dalam setiap langkah algoritma greedy, harus dibuat keputusan terbaik dalam menentukan pilihan untuk langkah selanjutnya,, yakni pada setiap langkah akan dipilih sebuat optimum lokal (*local optimum).* Dari pemilihan optimum lokal pada setiap langkahnya, diharapkan bahwa pada akhirnya langkah-langkah dengan optimum lokal tersebut akan menghasilkan sebuah optimum global.

Dalam pembuatan algoritma greedy, terdapat beberapa elemen yang diperlukan, yakni:

1. Himpunan kandidat (C).
2. Himpunan solusi (S).
3. Fungsi seleksi (*selection function*).
4. Fungsi kelayakan (*feasible*).
5. Fungsi obyektif .

Dari kelima elemen di atas, dapat dikatakan bahwa algoritma greedy mencari sebuah himpunan bagian (solusi, S) dari himpunan kandidat solusi (C), dengan himpunan bagian (S) harus memenuhi syarat – syarat yang sudah ditentukan terlebih dahulu (fungsi seleksi dan fungsi kelayakan) dan dioptimasi oleh fungsi obyektif.

Meskipun hasil dari algoritma greedy merupakan optimum global, namun optimum global tidaklah sama dengan solusi optimum (terbaik), tetapi optimum global hanyalah *sub-optimum* atau *pseudo-optimum* yang dalam beberapa persoalan dapat menjadi sebuah solusi optimum. Hal ini menyatakan bahwa hasil dari algoritma greedy tidak selalu merupakan solusi optimum dari persoalan yang dikerjakan, ini dikarenakan beberapa alasan, yakni:

1. Algoritma greedy tidak mencoba seluruh alternatif solusi yang ada dari persoalan (sebagaimana pada metode exhaustive search).
2. Fungsi SELEKSI yang berbeda dapat menghasilkan solusi yang berbeda pula, maka fungsi seleksi yang dipiliih harus merupakan fungsi seleksi yang tepat jika ingin algoritma menghasilkan solusi optiamal.

Berikut terlampir fungsi algoritma greedy secara umum, dengan himpunanKandidat merupakan struktur data untuk himpunan kandidat. Fungsi akan mengembalikan himpunan solusi jika ada dan himpunan kosong jika fungsi tidak memiliki solusi apa pun.

```
1.  function algoritmaGreedy (input C:
    himpunanKandidat) -> himpunanKandidat
2.  Deklarasi
3.  Var
4.      x: kandidat
5.      S: himpunanKandidat

7.  Algoritma
8.      S <- {}
9.      while (not Solusi(S) and ( C ≠ {}))
10.         x <- Seleksi(C)
11.         C <- C - {x}
12.         if  (Layak ( S + {x})) then
13.             S <- S + {x}
14.         endif
15.     endwhile
16.     -> S
```

Persoalan yang tidak memerlukan jawaban terbaik secara mutlak, umumnya menggunakan algoritma greedy karena algoritma greedy sangat berguna untuk menghasilkan solusi yang berupa hampiran (*approximation*), ketimbang menggunakan algoritma yang lebih rumit dan lebih mahal untuk menghasilkan jawaban terbaik yang mutlak. Algoritma greedy yang optimum dapat dibuktikan keoptimalannya secara matematis. Sebagai contoh algoritma Dijkstra (*graph shortest path algorithm*) yang kebenaran greedynya dapat dibuktikan dengan induksi matematika (sumber: https://web.engr.oregonstate.edu/~glencora/wiki/uploads/dijkstra-proof.pdf).

B. Harvest Moon: The Tale of Two Towns

Dalam permainan simulasi kehidupan bertani *Harvest Moon: The Tale of Two Towns* terdapat banyak hal yang dapat pemain lakukan, tidak hanya bertani mengenai berbagai tumbuhan mau pun memelihara berbagai hewan ternak, Salah satu aktivitasnya ialah pemain dapat bertemu dengan hewan liar yang tentunya akan menyerang pemain, kecuali pemain dengan

kreatif menghindari dengan cara berbicara dan memberi mereka makan sesuai makanan yang mereka suka agar mereka dapat berteman dengan pemain,

Serian *Harvest Moon* yang satu ini memiliki latar cerita yang cukup unik, yakni terdapat 2 kota bernama Bluebell dan Konohana yang bertetangga dan kabarnya ratusan merupakan 2 kota yang saling berteman dengan ramah. Kedua kota tersebut terhubung dengan sebuah terowongan yang melalui bukit yang cukup besar. Namun setelah beberapa lama, terdapat argument besar mengenai kota mana yang memiliki masakan lebih enak. Suatu saat, di salah satu konfrontasi yang terjadi antara kedua kota tersebut, Dewa Tani pun lelah akan pertengkaran mereka dan menghancurkan terowongan yang menghubungkan kedua kota. Berabad-abad setelah kejadian tersebut, kedua kota masih tidak akur, dan hanya berinteraksi melalui festival masak mingguan. Permainan dimulai dengan pemain yang sedang menaiki kuda di bukit tempat terowongan dibangung. Saat pemain sedang berada di bukit tersebut, terdapat seekor rubah yang menyerang pemain sehingga pemain jatuh dan pingsan. Lalu pemain pun ditemukan oleh karakter bernama Rutger, walikota Bluebell, dan Ina, walikota Konohana. Saat karakter pemain siuman, karakter pemain tidak dapat mengingat dari kota mana ia berasal dan kedua walikota bersikeras bahwa pemain berasal dari kota mereka masing-masing, namun keputusan asal kota pemain sepenuhnya ada di tangan pemain. Player akan diminta untuk memilih jenis kelamin dan kota asal mereka saat permainan benar-benar dimulai. Kota yang mereka pilih berpengaruh kepada hal yang akan lebih mereka fokuskan, yakni Kota Bluebell akan lebih focus ke hewan ternak dan Kota Konohana akan lebih focus ke tumbuh-tumbuhan, namun kota pilihan pemain tidak akan membuat pemain tidak dapat pergi ke kota satunya. Jenis kelamin player akan berpengaruh kepada karakter dalam game yang bisa dinikahi oleh pemain.

Gambar 2.1 Contoh peta Kota Bluebell

(sumber: https://harvestmoon.fandom.com/wiki/Bluebell_(TToTT))

Gambar 2.2 Contoh peta Kota Konohana

(sumber: https://harvestmoon.fandom.com/wiki/Konohana_(TToTT))

Pemain dapat menikah dengan karakter dalam game, dengan syarat pemain sudah punya rumah dengan ranjang besar dan sudah dekat dengan karakter yang ingin dinikahi. Kedekatan ini ditandai dengan *friendship points* yang dimiliki oleh pemain dengan karakter tersebut. Pemain dapat menikahi karakter dalam permainan jika pemain sudah memiliki *friendship points* minimal 60.000 . *Friendship points* dapat diperoleh dengan berinteraksi dengan karakter tersebut, termasuk memberi hadiah. Jika hal-hal yang dilakukan bertepatan dengan syarat-syarat tertentu, maka akan terjadi *flower events* yang jika dijawab dengan benar akan menambah *friendship points* namun jika salah malah akan mengurangi *friendship points*. *Flower events* ditandai dengan adanya rangkaian bunga pada sepanjang sisi kotak dialog percakapan. Warna dari bunga itu akan berubah jika *friendship points* dari pemain dengan karakter tersebut berubah.Warga biasa akan memiliki bunga berwarna putih, sedangkan warga yang dapat dinikahi dapat memiliki bunga dengan warna lain. Warna dari bunga- bunga akan mengindikasi perasaan calon pernikahan (kedekatan). Ada 7 ragam warna bunga, putih, ungu, biru, hijau, oranye, pink,,dan merah dengan urutan *friendship points* membesar.

Gambar 2.3 Bunga kotak dialog dan nilai *friendships points.*

(sumber: http://wino-blog.blogspot.com/2013/05/tingkat-pertemanan-atau-friendship.html)

Gambar 2.4 Contoh *flower events*

(sumber: https://harvestmoon.fandom.com/wiki/Jealousy_Points_(TToT))

III. IMPLEMENTATIONS

Banyak dari pemain *Harvest Moon: The Tale of Two Towns* mengejar *side story* dari permainan ini, yakni mereka mengejar untuk mendapatkan sebanyak mungkin *flower events* yang mungkin mereka dapatkan. Dan persoalan ini dapat diselesaikan dengan berbagai cara, diantaranya dengan algoritma greedy. Ada beberapa hal yang perlu diasumsikan sebelum implementasi dilakukan. Asumsi-asumsi tersebut ialah sebagai berikut:

1. Source code dari game tidak dibuka oleh developer, sehingga penulis mengasumsikan peta kota sebagai data graf tidak berarah.
2. Simpul pada graf merupakan tempat pada kota yang dapat dikunjungi oleh pemain.
3. Sisi pada graf menghubungkan tempat yang terletak bersebelahan (bersebelahan dengan arti tidak perlu melangkahi 1 tempat untuk mencapai tempat tersebut).
4. Pemain sedang dalam proses mengantarkan barang – barang ke semua lokasi dalam peta
5. Pemain lebih mementingkan *flower events* ketimbang total jarak yang ditempuh

Berikut dilampirkan beberapa fungsi yang nantinya dapat membantu algoritma greedy.

Fungsi seleksiFE menerima P sebagai status dari pemain yang mengandung lokasi player dan *friendship points* pemain dan C sebagai himpunan semua *flower events* yang terdapat dalam peta. Fungsi seleksiFE akan mengembalikan himpunan *flower events* yang syaratnya terpenuhi oleh staus dari pemain

```
function seleksiFE(P:statusPlayer,C:flowerEvents
) -> flowerEvents
Deklarasi
Var
    X: status
    S: flowerEvents
    i: integer
Algoritma
    X <- P.status
      S <- {}
      i <- 0

      while (i < C.length)
          if (Memenuhi(X,Cᵢ)) then
              S <- S + Cᵢ
          endif
          i <- i +1
      endwhile
      -> S
```

Fungsi countFE menerima X yang merupakan sebuah penanda lokasi dan C yang berisi himpunan *flower events* yang syaratnya sudah terpenuhi. Fungsi countFE akan mengembalikan jumlah *flower events* yang mungkin didapakan dari lokasi X (*flower events* yang lokasinya bertetanggaan dengan X, termasuk lokasi X itu sendiri)

```
function countFE(X: locate, C: flowerEvents
) -> integer
Deklarasi
Var
    i: integer
    count: integer
Algoritma
    i <- 0
    count <- 0
    while (i < S.length)
        if (Si.locate = X) then
            count <- count + 1
        endif
        i <- i +1
    Endwhile
    if (X ∈ S.locate) then
        count <- count + 1
    endif
    -> count
```

Fungsi greedyFE menerima P sebagai status player, A sebagai himpunan seluruh *flower events* pada peta, G sebagai graf berbobot dengan jarak antar simpul sebagai bobotnya. Fungsi greedy akan mengembalikan himpunan lokasi secara terurut. Pertama-tama fungsi akan menyeleksi dulu A (himpunan *flower events*), lalu dari himpunan hasil seleksi tersebut, akan dicari titik yang memungkinkan terjadinya *flower events* terbanyak yang belum ada dalam himpunan solusi. Jika ada 2 yang totanya sama, maka akan dipilih yang memiliki jarak terdekat.Jarak terdekat dapat diperoleh dengan berbagai cara, diantaranya perhitungan 2 titik (Tarik garis lurus dari titik 1 ke titik 2) atau mungkin perhitungan dengan algoritma dijkstra agar lebih efektif.

```
function greedyFE
(P:statusPlayer, A:flowerEvents, G:
weightedGraph) -> setOfLocate
Deklarasi
Var
    C: flowerEvents
    X: locate
    S: setOfLocate
    i: integer
    count: integer
```

```
Algoritma
    S <- {}
    C <- seleksiFE(P,A)
    while (S.length ≠ G.length)
        C <- seleksiFE(P,C)
        X <- G1.locate
        count <- countFE(G1.locate)
        for i<-1 to G.length
            if (Gi.locate ⊄ S) and (count < countFE(Gi.locate,C))
                X <- Gi.locate
                count <- countFE(Gi.locate,C)
            else if (Gi.locate ⊄ S) and (count = countFE(Gi.locate,C))
                if (Gi.cost < X.cost)
                 X <- Gi.locate
                 Count <-countFE(Gi.locate,C)
                endif
            endif
        endfor
        P.locate <-X
        P.time <- P.time + 1
        C <- C – {FE(X)}
        S <- S + {X}
    endwhile
    -> S
```

Seperti algoritma greedy pada umumnya, algoritma greedy untuk perhitungan jalan dengan *flower events* di atas memiliki lima elemen di dalamnya. Kelima elemen tersebut dapat didefinisikan sebagai berikut:x

1. Himpunan kandidat (C)

   Himpunan yang didapat dari fungsi seleksiFE yang seiring berubahnya status pemain, akan ikut juga berubah

2. Himpunan solusi (S).

   Himpunan yang dengan jalan terurut agar dapat menjalankan *flower events* terbanyak, merupakan hasil dari fungsi greedyFE.

3. Fungsi seleksi (*selection function*).

   Diimplementasikan dengan fungsi seleksiFE, dimana himpunan *flower events* diseleksi berdasarkan syarat *flower events* dan status dari pemain itu sendiri

4. Fungsi kelayakan (*feasible*)

   Semua pilihan titik akan layak karena tidak ada batas waktu maupun batas biaya dalam algoritma

5. Fungsi obyektif .

   Melewati seluruh tempat dalam peta tepat sekali dengan mendapat *flower events* terbanyak, dalam satu kali perjalanan.

Contoh keberjalanan program secara runtut pada contoh kasus seperti dibawah dijelaskan sebagai berikut:

Gambar 3 Contoh peta Kota Konohana dengan graf

(sumber: https://harvestmoon.fandom.com/wiki/Konohana_(TToTT))

Dari gambar 3 diatas, terlihat cara membuat graf dari peta Kota Konohana, dengan titik 0 sebagai titik tempat player berada. Misalkan pada peta diatas terdapat 3 *flower events* dengan keterangan:

1. Pada simpul 1 terdapat *flower events* dengan syarat pemain memiliki *friendship points* 10.000 dan dapat dilakukan pada jam sebelum jam 1.
2. Pada simpul 2 terdapat *flower events* dengan syarat pemain memiliki *friendship points* diatas 20.000 dan dapat dilakukan pada jam sebelum jam 1.
3. Pada simpul 4 terdapat *flower events* dengan syarat pemain memiliki *friendship points* diatas 10.000 dan dapat dilakukan pada jam sebelum jam 2.

Dengan asumsi jam pada saat dimainkan ialah jam 12, maka hasil penelusuran dengan fungsi greedyEF akan menghasilkan path (dari lokasi pemain): 2,4,3,5,6,1. Dengan rincian langkah:

1. Pada langkah pertama memilih simpul 2 karena simpul 2 memiliki hasil seleksiEF 3 sedangkan simpul lainnya hanya akan menghasilkan simpul countEF 2 atau 0. (jam menjadi jam 1).
2. Pada langkah kedua memilih titik 4 yang memiliki hasil seleksiEF 1 sedangkan hasil countEF untuk simpul lainnya ialah 0 karena tidak ada *flower events* yang dapat dijalankan kembali selain *flower events* pada simpul 4 (*flower events* pada simpul 1 sudah tidak lolos dari fungsi seleksiEF)
3. Untuk langkah ke tiga sampai seterusnya, hanya akan dipilih simpul dengan jarak terdekat karena semua simpul sudah tidak mempunyai *flower events* yang dapat dilakukan oleh pemain. Dari perkiraan pada gambar, jalan terdekat ialah ke simpul 3
4. Setelah dari simpul 3, diperkirakan simpul terdekat ialah simpul 5

5. Dari simpul 5, simpul yang diperkirakaan terdekat ialah simpul 6
6. Dari simpul 6 akan ke simpul 1 sebagai simpul terakhir yang belum dilalui oleh pemain.

Dari hasil jalannya algoritma greedyEF, maka pemain akan berhasil mendapat 2 buah *flower events* pada simpul 2 dan simpul 4, namun tidak mendapatkan *flower events* pada simpul 1 karena jam yang sudah tidak sesuai dengan syarat.

IV. K ESIMPULAN

Algoritma greedy merupakan salah satu algoritma yang cukup popular dalam menyelesaikan persoalan optimasi namun tidak membutuhkan jawaban yang mutlak dan berupa hampiran. Salah satu aplikasinya ialah pada pencarian *flower events* pada permainan *Harvest Moon: The Tale of Two Town.*

V. UCAPAN T ERIMA KASIH

Pertama-tama, penulis bersyukur kepada Tuhan Yang Maha Esa karena berkat-Nya sehingga penulis dapat menyelesaikan makalah ini dengan baik dan tepat waktu. Penulis turut mengucapkan terima kasih yang sebesar-besarnya kepada Ibu Nur Ulfa Maulidevi, selaku dosen dari mata kuliah Strategi Algoritma terutama dalam mengajarkan pokok bahasan algoritma greedy. Penulis juga mengucapkan terima kasih yang sebesar-besarnya kepada seluruh teman dan keluarga penulis yang selalu mendukung dalam proses pembuatan makalah ini. Penulis juga mengucapkan terimakasih kepada pihak-pihak yang membantu dalam proses pembuatan makalah ini baik secara langsung maupun tidak langsung.

REFERENCES

[1] Munir, Rinaldi, Matematika Diskrit, Bandung: Informatika Bandung, 2009.

[2] Chen, Mao, 2008, *A Greedy Algorithm with Forward-Looking Strategy*.

[3] http://lang8088.blogspot.com/2014/03/harvest-moon-tale-of-two-towns-menikah.html, tanggal akses terakhir: 26 April 2019, pukul 03.51

[4] https://brilliant.org/wiki/greedy-algorithm/, tanggal akses terakhir: 26 April, pukul 03.24

[5] https://harvestmoon.fandom.com/wiki/Wiki, tanggal akses terakhir: 26 April 2019, pukul 04.13

[6] https://www.nintendo.co.uk/Games/Nintendo-3DS/Harvest-Moon-The-Tale-of-Two-Towns-527624.html, tanggal akses terakhir: 26 April 2019, pukul 04.58

[7] https://fogu.com/hm10/, tanggal akses terakhir, 26 April 2019, pukul 05.18

[8] http://www.opendatastructures.org/ods-python-screen.pdf, tanggal akses terakhir: 26 April 2019, pukul 05.49

[9] https://web.engr.oregonstate.edu/~glencora/wiki/uploads/dijkstra-proof.pdf, tanggal akses terakhir: 26 April 2019, pukul 06.32

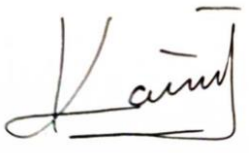
[10] http://wino-blog.blogspot.com/2013/05/tingkat-pertemanan-atau-friendship.html, tanggal akses terakhir: 26 April 2019, pukul 08.33

[11] http://masterharvestmoon.blogspot.com/2013/02/harvest-moon-tale-of-two-towns-flower.html, tanggal akses terakhir: 26 April 2019, pukul 09.57

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi

Bandung, 26 April 2019



Karina Iswara - 13517031