Penerapan Algoritma Greedy untuk Memaksimalkan Kemenangan pada *Minigame* Platoon

Pada Permainan Ni No Kuni: *Wrath Of The White Witch*

Reihan Andhika Putra 13519043
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
13519043@std.stei.itb.ac.id

***Abstrak*—Algoritma *Greedy* merupakan algoritma yang sering digunakan untuk menyelesaikan *game* karena strategi yang digunakan bersifat heuristik sehingga tidak memerlukan pembuktian yang kompleks namun menuntut pemain untuk menciptakan strategi yang pintar. Pada *game* yang populer selalu disediakan *minigame* untuk menghibur pemain. Salah satu *minigame* yang menarik pada *game* Ni No Kuni adalah Platoon. Dalam *minigame* Platoon, dua pemain (satu pemain dan satu bot) akan diberikan 10 kartu dan harus menyusun kartu tersebut dalam lima tumpukan. Secara bergantian, setiap pemain harus memilih satu tumpukan untuk ditandingkan dengan tumpukan pemain lainnya. Tumpukan dikatakan menang apabila nilainya lebih besar dari tumpukan lain atau hadir kartu khusus yang memberikan efek tertentu. Pemain dikatakan menang dalam satu ronde Platoon apabila minimal tiga tumpukan berhasil menang saat ditandingkan dengan tumpukan lain. Pada makalah ini akan dibahas penggunaan algoritma *greedy* untuk memaksimalkan kemenangan pada *minigame* Platoon. Makalah ini tidak membahas jumlah taruhan yang harus diambil supaya keuntungan dari *minigame* Platoon maksimal. Algoritma *greedy* dipilih karena dibutuhkan teknik dalam memilih kombinasi kartu pada suatu tumpukan dan memilih komposisi tumpukan yang paling optimal untuk memaksimalkan kemenangan pada *minigame* Platoon. Hasilnya, algoritma *Greedy* cukup efektif untuk memenangkan *minigame* ini.**

***Keywords—Greedy*, Ni no Kuni, Platoon , Kartu**

I. PENDAHULUAN

Tidak dapat dipungkiri bahwa perkembangan industri *game* sangat pesat pada zaman ini. Dari yang hanya bisa dimainkan oleh satu orang hingga dapat dimainkan oleh banyak orang secara *realtime*. Dari yang grafisnya hanya berupa *pixel* dengan ukuran tidak mencapai 10 MB hingga sekarang grafisnya yang mampu menyerupai manusia dengan ukuran hingga 100 GB. Perkembangan industri *game* diikuti dengan perkembangan teknologi yang pesat dan kebutuhan masyarakat untuk memenuhi kesenangan mereka. Berdasarkan analisis yang dilakukan *website* techjury.net, dari data yang bersumber dari We PC, keuntungan yang diraup dari industri *game* mencapai 90 miliar dolar *USD* [1]. Hal tersebut disebabkan rata-rata *game* jaman sekarang memiliki grafis dan *gameplay* yang menarik serta menawarkan waktu bermain hingga puluhan jam sehingga pemain lebih bisa menikmati *game* dan tidak perlu khawatir *game* akan cepat tamat. Game dengan waktu bermain hingga puluhan jam tersebut biasanya adalah *game offline* dengan *platform* PC yang cukup populer. Tidak jarang muncul *minigame* pada *game* tersebut untuk membuat pemain tidak jenuh dalam menyelesaikan cerita utama dalam *game* tersebut. Salah satu *game* yang populer tersebut adalah Ni No Kuni: *Wrath of The White Witch*.

Ni No Kuni: *Wrath of The White Witch* adalah *game* yang sempat dirilis tahun 2011 di Jepang dan 2013 di seluruh dunia dengan *platform* PlayStation 3 yang diproduksi oleh studio animasi Ghibli dan studio Level-5 sebagai developer. Game ini sangat kental dengan imajinasi anak-anak dan dongeng yang bisa dinikmati oleh semua umur. Pada tahun 2019 versi *remastered* dari game ini yang dikembangkan oleh studio Qloc dirilis dan mendapat *feedback* yang positif dari banyak pemain di seluruh dunia. Game ini merupakan adaptasi campuran sistem *turn-based* dengan *action* dimana kamu akan mampu memasukkan *command*, namun sembari berlari meninggalkan musuh demi menghindari serangannya. Dilansir dari situs howlongtobeat.com, waktu untuk menyelesaikan Ni No Kuni mencapai 92 jam [2].

Gambar 1. Cover *Game* Ni no Kuni: *Wrath of the White Witch Remastered*
Sumber: https://www.mobygames.com/game/ni-no-kuni-wrath-of-the-white-witch-remastered/cover-art/gameCoverId,592468/

Disamping waktu bermainnya yang cukup lama, Ni No Kuni juga menghadirkan beberapa *minigame* yang menarik sehingga pemain tidak akan jenuh dalam menyelesaikan. Beberapa *minigame* yang disediakan oleh Ni No Kuni yaitu:

Slot Machine, Blackjack, *Double Cross*, dan Platoon. *Minigame* ini dapat diakses di *Crypt Casino* yang terletak di salah satu peta pada *game* Ni No Kuni. *Crypt Casino* menyediakan banyak hadiah yang menarik namun dengan harga yang tidak masuk akal. Karena hal ini, banyak pemain yang berusaha untuk mendapatkan banyak uang lewat *minigame* yang ada di *Crypt Casino*. Dari keempat *minigame* yang ditawarkan, Platoon adalah *minigame* favorit para pemain karena pada *minigame ini*, kemenangan tidak hanya ditentukan murni oleh keberuntungan namun juga ditentukan oleh kemampuan dari pemain. Meskipun begitu, masih banyak pemain yang masih kesusahan dalam memenangkan Platoon karena kerakusan dalam menyusun kartu pada suatu tumpukan atau tidak mempunyai pedoman dalam penyusunan kartu pada suatu tumpukan (akan dibahas pada bab selanjutnya). Untuk itu diperlukan suatu algoritma agar kemenangan tumpukan pada *minigame* Platoon bisa dioptimalkan. Algoritma yang dibutuhkan adalah algoritma yang cukup sederhana, tanpa pembuktian matematis namun efektif pada *minigame* Platoon ini.

Gambar 2. Ilustrasi *Minigame* Platoon
Sumber: Dokumen Pribadi

Diambil dari buku Algoritma dan Struktur Data Menggunakan Bahasa Pemrograman C++, menurut Donald E. Knuth algoritma dalam pengertian modern mempunyai kemiripan dengan istilah resep, proses, metode, teknik, prosedur, rutin. Algoritma adalah sekumpulan aturan-aturan berhingga yang memberikan sederetan operasi-operasi untuk menyelesaikan suatu jenis masalah yang khusus [3]. Algoritma yang digunakan untuk memenangkan *minigame* Platoon pada makalah ini adalah algoritma *Greedy* karena diperlukan teknik dalam penyusunan kartu pada suatu tumpukan dan pemilihan komposisi tumpukan yang paling optimal untuk memaksimalkan kemenangan.

II. Dasar Teori

A. Algoritma Greedy

Algoritma Greedy merupakan salah satu algoritma yang populer dan paling sering sering digunakan untuk memecahkan persoalan terutama persoalan optimasi karena algoritmanya yang cukup sederhana dan *straightforward*. Persoalan optimasi dapat diartikan sebagai persoalan pencarian solusi yang optimal pada suatu kasus. Terdapat dua macam persoalan optimasi, yaitu maksimasi (*maximization*) dan minimasi (*minimization*).

Algoritma ini menggunakan pendekatan penyelesaian masalah dengan mencari nilai terbaik (optimal) dari setiap langkahnya (*step-by-step*) tanpa memperhatikan konsekuensi yang akan diterima pada langkah selanjutnya. Algoritma ini berusaha untuk mencari solusi optimal lokal dengan harapan kumpulan solusi optimal lokal tersebut akan membawa solusi yang optimal global [4]. Pada pemilihan kandidat optimal lokal di setiap langkah diperlukan pengecekan persyaratan yang bersifat heuristik. Cara kerja *greedy* yang rakus ini sesuai dengan prinsip dari algoritmanya, yaitu "*take what you can get now"*.

Pada implementasinya, nilai setiap solusi optimum lokal yang diperoleh dari tidak selalu membawa solusi yang optimum global. Hal ini disebabkan karena algoritma ini tidak melakukan pencarian yang penuh seperti algoritma *brute force*. Meskipun begitu, algoritma *greedy* memiliki kompleksitas waktu cukup rendah dibanding algoritma *brute force*. Algoritma ini biasa digunakan apabila kita tidak membutuhkan solusi yang dekat dengan solusi sempurna.

Terdapat lima buah elemen dalam algoritma *greedy*:

1. Himpunan kandidat, yaitu himpunan yang berisi kandidat yang akan dipilih pada setiap langkah untuk menjadi bagian dari solusi yang dipilih.
2. Himpunan solusi, yaitu himpunan yang berisi kandidat-kandidat yang telah dipilih.
3. Fungsi solusi, yaitu fungsi yang menentukan apakah himpunan kandidat yang dipilih sudah menghasilkan solusi.
4. Fungsi seleksi (*selection function*), yaitu fungsi yang digunakan untuk memilih kandidat berdasarkan strategy *greedy* tertentu. Strategi *greedy* yang digunakan bersifat heuristik.
5. Fungsi kelayakan (*feasible*), yaitu fungsi yang bertugas memeriksa apakah solusi yang dipilih tidak melanggar batasan atau *constraint* yang didefinisikan.
6. Fungsi objektif adalah fungsi yang menyatakan tujuan dari persoalan yang ingin diselesaikan.

Contoh pengaplikasian strategi *greedy* pada persoalan *integer knapsack problem* adalah sebagai berikut:

1. *greedy by profit*, pada setiap langkah memilih barang yang memiliki keuntungan paling banyak,
2. *greedy by weight*, pada setiap langkah memilih barang dengan berat paling kecil, atau
3. *greedy by density*, pada setiap langkah selalu memilih barang dengan densitas *profit per weight* paling besar.

Selain itu, algoritma *greedy* juga memiliki dua tipe paradigma:

1. *Subset paradigm*, algoritma *greedy* digunakan untuk memilih kandidat-kandidat yang membentuk *subset* paling optimal.

2. *Ordering paradigm*, algoritma *greedy* digunakan untuk membuat urutan memilih kandidat yang akan menghasilkan solusi optimum.

B. *Minigame* Platoon

Platoon adalah salah satu *minigame* yang dapat ditemukan di dalam *Crypt Casino* pada *game* Ni No Kuni: *Wrath of The White Witch*. *Crypt Casino* dapat diakses setelah pemain menyelesaikan 50% cerita utama. *Crypt Casino* terletak di dalam *dungeon Tombstone Trail*. Untuk pergi ke *dungeon* tersebut, pemain bisa melakukan *teleport* atau menjelajahi peta. Setelah masuk ke *Tombstone Trail* pemain cukup bergerak lurus hingga menemukan *memory stone* lalu bergerak sedikit ke kanan. Disitu akan tampak pintu masuk *Crypt Casino.*

Gambar 3. Pintu Masuk *Crypt Casino*
Sumber: Dokumen Pribadi

Setelah masuk ke dalam *Crypt Casino,* pemain tinggal bergerak lurus sedikit lalu di samping kanan akan tampak perempuan berambut abu-abu yang berperan sebagai *dealer* Platoon. Pemain dapat berinteraksi dengan perempuan tersebut untuk memainkan Platoon. Platoon adalah *minigame* yang dimainkan oleh dua pemain (diri sendiri dan *bot*). Pada *minigame* Platoon, tiap pemain *akan* diberikan 10 kartu secara acak yang kemudian disusun menjadi lima tumpukan berbeda. Tiap tumpukan akan dipertandingkan dengan tumpukan dari pemain lain. Pemain dikatakan menang dalam satu ronde Platoon apabila tumpukan dari pemain berhasil menang minimal tiga kali melawan tumpukan yang dimiliki oleh lawan. Platoon adalah mini*game* yang mengasah *strategi* dan kemampuan analisis pemain karena pemain harus bisa menciptakan kombinasi kartu yang optimal dari kartu yang didapat secara acak. Tata cara permainan dan peraturan yang lebih jelas akan dibahas di bawah.

Gambar 4. *Dealer* Platoon
Sumber: Dokumen Pribadi

Ada beberapa kartu khusus dengan efek tertentu yang bisa merubah alur pertandingan antara dua tumpukan. Ada lima jenis kartu pada *minigame* Platoon. Penjelasan tentang kartu tersebut dapat dilihat pada tabel 1.

Tabel 1. Kartu yang Ada dalam *Minigame* Platoon

Foto	Nama	Penjelasan
 Gambar 5. *Spawn Card* *Sumber: Dokumen Pribadi*	***Pawn card***	**Deskripsi:** *Pawn card* adalah kartu dengan nomor dari dua hingga sepuluh.
		Nilai: Kartu ini mempunyai nilai sesuai dengan nomor yang ada di kartunya.
		Efek: Kartu ini tidak mempunyai efek khusus apapun.
 Gambar 6. *Jack/Queen* *Sumber: Dokumen Pribadi*	***Jack / Queen***	**Deskripsi:** *Jack / Queen* adalah kartu dengan tulisan huruf J (*Jack*) dan Q (*Queen*).
		Nilai: Kartu ini mempunyai nilai 10.
		Efek: Kartu ini tidak mempunyai efek khusus apapun.
 Gambar 7. *King* *Sumber: Dokumen Pribadi*	***King***	**Deskripsi:** *King* adalah kartu dengan gambar mahkota dan tulisan huruf K.
		Nilai: Kartu ini mempunyai nilai 10.
		Efek: Kartu ini akan selalu menang melawan tumpukan dengan kartu selain bishop. Apabila tumpukan musuh ada *bishop* maka king langsung kalah.
 Gambar 8. *Bishop* *Sumber:*	***Bishop***	**Deskripsi:** *Bishop* adalah kartu dengan gambar topi pendeta dan tulisan huruf A. Pada permainan kartu remi biasa, *bishop* dikenal dengan sebutan kartu “As”.
		Nilai: Kartu ini mempunyai nilai satu.

Dokumen Pribadi		**Efek:** Kartu ini akan selalu kalah melawan tumpukan apapun namun justru menang apabila tumpukan musuh ada *king*.
 Gambar 9. *Wizard* *Sumber: Dokumen Pribadi*	**Wizard**	**Deskripsi:** Wizard adalah kartu dengan gambar topi penyihir dan tulisan "JOKER". Pada permainan kartu remi biasa, *bishop* dikenal dengan sebutan kartu "Joker".
		Nilai: Kartu ini mempunyai nilai nol.
		Efek: Kartu ini akan menyebabkan pemain untuk menukar tumpukannya dengan tumpukan musuh sebelum menentukan pemenang dari pertandingan tumpukan.

Adapun peraturan terkait penentuan tumpukan pemenang dan aturan pembentukan tumpukan dalam *minigame* Platoon adalah sebagai berikut:

1. Jika tidak ada tumpukan yang mengandung *King* atau *Bishop* maka tumpukan dengan total nilai lebih tinggi yang menang.
2. Jika kedua tumpukan mengandung *King* maka tumpukan dengan total nilai yang lebih besar yang menang.
3. Jika kedua tumpukan mengandung *Bishop* maka tumpukan dengan total nilai yang lebih besar yang menang.
4. Jika kedua tumpukan mengandung *Wizard* maka tumpukan dituka dua kali sehingga setiap pemain tetap mempertahankan tumpukan awal.
5. *Wizard* tidak boleh diletakkan sendirian dalam satu tumpukan.
6. *King* dan *Bishop* tidak boleh diletakkan bersamaan pada satu tumpukan.

Secara garis besar, alur dari *minigame* Platoon adalah sebagai berikut:

1. Pada awal permainan, pemain dapat menaikkan atau menurunkan jumlah taruhan awal. Jumlah taruhan awal minimal 200 *chip*.
2. Setelah menentukan jumlah taruhan, pemain akan diberikan 10 kartu secara acak. Pemain harus menyusun 10 kartu tersebut menjadi lima tumpukan yang berbeda. Pemain dan *bot* musuh tidak bisa melihat isi dari tumpukan yang bukan miliknya sendiri. Setelah selesai menyusun dan tidak ditemukan pelanggaran pada penyusunan tumpukan, pemain dapat lanjut ke fase pertandingan tumpukan.
3. Pada awal fase pertandingan, dua pemain (diri sendiri dan *bot* musuh) harus menandai tumpukan yang diyakini akan memenangkan pertandingan. Tanda ini akan terlihat oleh kedua pemain.
4. Setelah itu, *dealer* akan mengkocok kartu untuk dua pemain. Pemain yang mendapatkan kartu dengan nilai lebih tinggi dapat menyerang tumpukan musuh terlebih dahulu.
5. Pada setiap giliran, pemain akan secara bergantian memilih tumpukan yang akan digunakan untuk menyerang dan tumpukan pemain lain yang akan diserang. Tumpukan yang sudah digunakan dalam pertandingan akan dibuka dan tidak bisa digunakan untuk bertanding lagi.
6. Apabila pemain berhasil menang minimal tiga kali dalam pertandingan tumpukan maka pemain berhasil memenangkan satu ronde Platoon. Jika tidak, maka pemain dianggap kalah.
7. Setelah pemain menang maka pemain mendapatkan beberapa *chip* tambahan dan dapat melanjutkan Platoon ke ronde selanjutnya dengan taruhan yang lebih besar atau selesai bermain dan mendapatkan *chip* yang sudah terkumpul.

III. Persiapan

Pada bagian ini, akan ditentukan faktor-faktor yang menjadi pertimbangan dalam menyusun kombinasi kartu pada suatu tumpukan dan menyusun kombinasi tumpukan untuk mendapatkan hasil yang optimal dengan menggunakan algoritma *greedy*. Pada bagian ini juga akan dibahas asumsi dan batasan yang akan digunakan untuk menyederhanakan faktor-faktor pengambilan keputusan.

A. Asumsi dan Batasan

Berdasarkan penjelasan pada bab sebelumnya, kita tahu bahwa untuk memulai Platoon kita harus memasang nilai taruhan awal dan apabila kita menang maka kita dapat melanjutkan permainan dengan taruhan yang lebih tinggi. Algoritma *greedy* yang akan dibuat pada makalah ini tidak membahas cara untuk memaksimalkan jumlah keuntungan yang bisa didapat dari taruhan. Kita juga tahu bahwa setelah berhasil membentuk lima tumpukan yang tidak melanggar aturan maka pemain secara bergantian memilih tumpukan yang akan bertanding dan memilih satu tumpukan musuh yang akan menjadi lawannya. Algoritma *greedy* yang akan dibuat pada makalah ini tidak membahas cara untuk memilih musuh yang akan ditandingkan karena komposisi tumpukan musuh sangatlah bervariasi dan tidak bisa diidentifikasi. Cara musuh meletakkan tumpukannya-pun cenderung acak sehingga terlalu besar untuk dianalisis. Algoritma *greedy* pada makalah ini

dibuat untuk memaksimalkan jumlah tumpukan yang bisa menang dalam satu ronde Platoon.

B. Implementasi Algoritma Greedy

Untuk dapat mengoptimalkan jumlah tumpukan yang bisa menang tentu kita harus memformulasikan susunan kartu dalam tiap tumpukan agar memiliki peluang menang yang tinggi. Namun karena kartu yang didapat adalah kartu hasil acakan, tentu kita tidak bisa berekspektasi bahwa kita selalu mendapatkan kartu dengan nilai yang tinggi atau kartu khusus yang kuat. Untuk itu penulis mengklasifikan kombinasi kartu menjadi tiga jenis yaitu kombinasi berpeluang menang tinggi, kombinasi berpeluang menang sedang, dan kombinasi berpeluang kalah [5]. Kombinasi kartu yang berpeluang menang tinggi antara lain:

1) *Wizard* yang digabung dengan kartu dengan nilai yang rendah (sekitar 1 - 5),
2) *King* yang diletakkan sendirian,
3) *Wizard* yang digabung dengan *bishop, dan*
4) Kombinasi kartu dengan nilai tinggi (diatas 18 dan dibawah 26).

Kombinasi kartu yang berpeluang menang sedang antara lain:

1. Kombinasi kartu dengan nilai sedang (sekitar 6-18) dan
2. *Bishop* yang diletakkan sendirian

Kombinasi kartu yang berpeluang kalah antara lain:

1. *Bishop* yang diletakkan sendirian,
2. Kartu/kumpulan kartu dengan nilai rendah (sekitar 1-4), dan
3. Kemungkinan kombinasi kartu lain.

Urutan ketiga kombinasi kartu diatas juga menentukan prioritas kombinasi mana yang akan dibentuk terlebih dahulu. Dengan menggunakan penjelasan diatas, elemen algoritma *greedy* yang diterapkan dalam permainan ini adalah sebagai berikut:

1. Himpunan Kandidat

   Himpunan kandidatnya adalah 10 kartu yang didapatkan pemain secara acak.

2. Himpunan Solusi

   Himpunan solusinya adalah kombinasi kartu yang ditaruh di lima tumpukan yang berbeda. Kombinasi kartu dapat berupa kombinasi dengan peluang menang tinggi, peluang menang sedang, atau peluang kalah.

3. Fungsi Solusi

   Fungsi solusinya adalah fungsi yang mengecek apakah tumpukan yang diisi sudah berjumlah lima buah atau belum.

4. Fungsi Seleksi (*selection function*)

   Fungsi seleksinya adalah fungsi untuk membentuk kombinasi kartu dari kartu yang dimiliki oleh pemain. Kombinasi kartu yang dibentuk diprioritaskan dari kombinasi dengan peluang menang yang tinggi, kombinasi dengan peluang menang sedang, lalu kombinasi dengan peluang kalah.

5. Fungsi Kelayakan

   Fungsi kelayakannya adalah fungsi yang mengecek apakah tumpukan yang di hasilkan dari fungsi seleksi adalah tumpukan yang tidak kosong dan tidak melanggar aturan pembentukan tumpukan.

6. Fungsi Objektif

   Fungsi objektifnya adalah membuat tumpukan dengan peluang menang tinggi sebanyak mungkin.

Secara umum, algoritma *greedy* yang penulis buat adalah algoritma untuk memaksimalkan jumlah tumpukan dengan peluang menang tinggi. Pada setiap langkah, algoritma akan mencoba untuk membuat tumpukan dengan peluang menang tinggi (optimum lokal), apabila tidak bisa maka algoritma akan mencoba membuat tumpukan dengan peluang menang sedang atau peluang menang kalah. Dengan memaksimalkan jumlah tumpukan peluang dengan menang tinggi, harapannya kita dapat memenangkan satu ronde platoon (optimum global).

Apabila dalam pembentukan tumpukan dengan algoritma *greedy* didapatkan hasil berupa jumlah tumpukan yang kurang dari lima maka program akan membentuk tumpukan baru dengan menggunakan satu kartu dengan nilai paling kecil yang ada di tumpukan sebelumnya. Apabila didapatkan hasil berupa jumlah tumpukan yang lebih dari lima, maka program akan menggabungkan dua tumpukan yang paling lemah menjadi satu tumpukan. Algoritma greedy yang penulis gunakan dalam bentuk *pseudocode* adalah sebagai berikut

```
Definisi Tipe Data
Card: Card mendefinisikan semua kartu yang ada
pada minigame Platoon. Untuk mengacu pada kartu
tertentu cukup gunakan nama kartu dengan semua
huruf dikapitalkan
Contoh
bishop : BISHOP
king   : KING
spawn nomor 01: SPAWN_01
Tumpukan : Tumpukan adalah List of Card

Function
drawCard()
{Melakukan draw sejumlah 10 kartu secara acak}

mergeLowestPile(input/output A: List of
Tumpukan)
{Menggabungkan dua tumpukan yang paling lemah
menjadi satu tumpukan}

getWeakestCardInPile(A: List of Tumpukan)→ Card
{Mendapatkan satu kartu terlemah di dalam
kumpulan tumpukan, kartu yang didapatkan akan
dihilangkan dari tumpukan }
```

```
addPile (input/output A: List of Tumpukan,
input B: List of Card)
{Menambahkan tumpukan baru yang berisikan
kumpulan kartu di B ke dalam A}
{addPile akan otomatis mengacak urutan tumpukan
di dalam A setelah ditambahkan tumpukan baru}

containCard(A: List of Card, B: Lisf of Card)
→ Boolean
{Mengembalikan True apabila semua kartu di B
terdapat di dalam A}

removeCard(input/output A: List of Card, input
B: List of Card)
{prekondisi: A mengandung semua kartu di B}
{Mengurangi kartu B di dalam A}
addCard(input/output A: List of Card, input B:
List of Card)
{Menambahkan semua kartu B di dalam A}

getValuedCard(A: List of Card, up,down :
Integer) → List Of Card
{Mendapatkan sekumpulan kartu bernilai total
sejumlah down hingga up, dipastikan jeni kartu
yang didapatkan hanya kartu biasa (spawn card
dan Queen/Jack)}
{return list kosong apabila tidak memenuhi}

isValidPile(A: Tumpukan)→ Boolean
{Return True apabila tumpukan A adalah tumpukan
yang valid (tidak kosong dan tidak melanggar
aturan)}

size(A : List) → Integer
{Mengembalikan ukuran List A}

Kamus Data:
Algoritma:
myCard ← drawCard()
allPile ← []
while (size(myCard) ≠ 0 ) do
  {Fungsi Seleksi + Fungsi Objektif}
  newPile ← []
  strongCard ← getValuedCard(myCard,19,26)
  mediumCard ← getValuedCard(myCard,6,18)
  weakCard ← getValuedCard(myCard,1,5)
  {seleksi kartu berpeluang menang tinggi}
  {mempunyai king}
  if contain(myCard, [KING]) then
    addCard(newPile, [KING])
  {mempunyai bishop dan kartu nilai rendah}
  elif contain(myCard, [BISHOP]+weakCard) then
    addCard(newPile, [BISHOP]+weakCard)
  {mempunyai wizard dan bishop}
  elif contain (myCard, [WIZARD, BISHOP]) then
    addCard(newPile, [WIZARD, BISHOP])
 {mempunyai kartu nilai tinggi}
  elif contain(myCard, strongCard) then
    addCard(newPile, strongCard)
 {seleksi kartu berpeluang menang sedang}
 {mempunyai kartu nilai sedang}
  elif contain(myCard, mediumCard) then
    addCard(newPile, mediumCard)
 {seleksi kartu berpeluang kalah}
 {mempunyai bishop}
  elif contain(myCard, [BISHOP]) then
    addCard(newPile, [BISHOP])
  {mempunyai kartu nilai rendah}
  elif contain(myCard, weakCard) then
    addCard(newPile, weakCard)

 {kombinasi tidak dikenali (edge case)}
 {taruh kartu tersisa kedalam satu tumpukan}
 else then
    addCard(newPile, myCard)
    removeCard(myCard, myCard)
 endif
 {Fungsi Kelayakan}
 if (isValidPile(newPile)) then
   removeCard(myCard, newPile)
   addPile(allPile,newPile)
 endif
endwhile

{Fungsi Solusi}
while (size(allPile)≠5) do
  if (size(allPile)>5) then
    mergeLowestPile(allPile)
  elif (size(allPile)<5) then
    newPile ← []
    weakestCard ← getWeakestCardInPile(allPile)
    addCard(newPile, myCard)
    addPile(allPile,newPile)
  endif
endwhile
if (size(allPile)=5)then
  output("Tumpukan siap bertanding")
else
  output("Unexpected error")
endif
```

IV. Eksperimen

Pada bagian ini, akan dilakukan eksperimen dengan memainkan beberapa ronde Platoon pada *game* Ni No Kuni. Komposisi kartu dalam tiap tumpukan yang digunakan pada setiap ronde adalah hasil dari algoritma *greedy* yang dibahas pada bab sebelumnya. Pemilihan tumpukan musuh akan dilakukan secara acak.

A. Ronde Pertama

Kartu yang Didapat: *spawn* no 2 *1, *spawn* no 3 *1, *spawn* no 5 * 1, *spawn* no 7 * 2, *spawn* no 8 * 1, *bishop* *1, *wizard*1.

Tumpukan yang Terbentuk:

1. *Bishop *1* (peluang kalah)
2. *Wizard *1 + spawn* no *2 *1* (peluang menang tinggi)
3. *Wizard*1 + spawn* no *3 *1* (peluang menang tinggi)
4. *Spawn* no *7 * 1 + spawn* no *5 *1* (peluang menang sedang)
5. *Spawn* no *8 * 1 + spawn* no *7 *1 + spawn* no *3 *1* (peluang menang tinggi)

Pertandingan:

1. *Wizard *1 + spawn* no *2 *1 VS spawn* no *4 *1 + spawn* no *6 *1 + spawn* no *8 *1* (19 vs 2) (menang)
2. *Bishop *1* vs *king* *1 (1 vs 10) (menang)
3. *Spawn* no *8 * 1 + spawn* no *7 *1 + spawn* no *3 *1 vs Spawn no 2 *1* (18 vs 2) (menang)

Hasil Akhir: Menang 3-0

Dokumentasi:

Kartu yang didapat

Tumpukan yang dibentuk

Hasil pertandingan

Gambar 10. Dokumentasi Ronde Pertama
Sumber: Dokumen Pribadi

B. Ronde Kedua

Kartu yang Didapat: *spawn* no 3 *1, *spawn* no 4 *1, *spawn* no 7 *1, *spawn* no 9 *1, *queen* *2, *bishop* *2, *wizard* *1, *king* *1.

Tumpukan yang Terbentuk:

1. *King *1* (peluang menang tinggi)
2. *Queen *1 + spawn* no *9 *1 + spawn* no *3 *1* (peluang menang tinggi)
3. *Queen *1 + spawn* no *7 *1 + spawn* no *4 *1* (peluang menang tinggi)
4. *Bishop *1* (peluang kalah)
5. *Bishop *1+ Wizard *1* (peluang kalah)

Pertandingan:

1. *Queen *1 + spawn* no *9 *1 + spawn* no *3 *1 vs spawn* no *7 *1* (22 vs 7) (menang)
2. *King *1* vs *spawn* no *2 *1 + spawn* no *10 *1 + wizard *1* (12 vs 10) (kalah)
3. *Bishop * 1 + vs spawn no 5 *1 + spawn no 9 *1* (1 vs 14) (kalah)
4. *Queen *1 + spawn* no *7 *1 + spawn* no *4 *1 vs spawn* no *2 *1 + spawn* no *10 *1 (21 vs 12)* (menang)
5. *Wizard *1 + bishop *1 vs spawn no 4 *1 + spawn no 8 *1* (12 vs 1) (menang)

Hasil Akhir: Menang 3-2

Dokumentasi:

Kartu yang didapat

Tumpukan yang dibentuk

Hasil pertandingan

Gambar 11. Dokumentasi Ronde Kedua
Sumber: Dokumen Pribadi

C. Ronde Ketiga

Kartu yang Didapat: *spawn* no 2 *1, *spawn* no 5 *2, *spawn* no 10 *1, *jack*2, *queen* *2, *king* *1, *wizard*1.

Tumpukan yang Terbentuk:

1. *Wizard*1 + spawn* no *2 *1* (peluang menang tinggi)
2. *King *1* (peluang menang tinggi)
3. *Queen *2 + spawn* no *5 *1* (peluang menang tinggi)
4. *Jack *2 + spawn* no *5 *1* (peluang menang tinggi)
5. *Spawn* no *10 * 1* (peluang menang sedang)

Pertandingan:

1. *Spawn* no *10 * 1 vs Spawn* no *7 *1 + Spawn* no *8 *2 + Spawn* no *9 *1* (10 vs 32) (kalah)
2. *King *1* vs *bishop* *1 + *wizard *1* (1 vs 10) (menang)
3. *Jack *2 + spawn* no *5 *1 vs spawn* no *3 *1* (25 vs 4) (menang)
4. *Wizard*1 + spawn* no *2 *1 vs spawn* no *6 *1* (6 vs 2) (menang)

Hasil Akhir: Menang 4-1

Dokumentasi:

Kartu yang didapat

Tumpukan yang dibentuk

Hasil pertandingan

Gambar 12. Dokumentasi Ronde Ketiga
Sumber: Dokumen Pribadi

V. KESIMPULAN

Algoritma greedy dapat digunakan untuk membantu menyelesaikan *minigame* Platoon. Dalam *minigame* Platoon, algoritma ini digunakan untuk memilih komposisi kartu pada suatu tumpukan untuk memaksimalkan jumlah tumpukan dengan peluang menang yang tinggi sehingga meningkatkan kemenangan dalam *minigame* Platoon. Meskipun algoritma greedy belum tentu memberikan solusi yang optimal, akan tetapi algoritma ini cocok digunakan dalam *minigame* Platoon karena dibutuhkan teknik dalam memilih kombinasi kartu pada suatu tumpukan agar jumlah tumpukan berpeluang menang tinggi maksimal.

Berdasarkan eksperimen yang dilakukan pada bagian sebelumnya, dapat ditarik kesimpulan bahwa algoritma *greedy* efektif untuk meningkatkan kemenangan pada *minigame* Platoon. Dari tiga percobaan yang dilakukan, tiga-tiganya berhasil dimenangkan. Dari 12 pertandingan yang dilakukan, pemain dapat memenagkan sembilan pertandingan. Berdasarkan statistik tersebut *winrate* bisa dikatakan cukup tinggi sehingga bisa memenangkan ketiga ronde Platoon. Dapat disimpulkan bahwa algoritma *greedy* cocok untuk memaksimalkan kemenangan pada *minigame* Platoon.

PRANALA VIDEO YOUTUBE

Untuk memberi gambaran yang lebih jelas tentang penelitian ini, telah dibuat sebuah video penjelasan yang dapat diakses melalui tautan: https://youtu.be/4xnCBIn14i4

REFLEKSI DAN APRESIASI

Penulis mengucapkan puji syukur kepada Allah SWT karena berkat kehendak-Nya penulis dapat menyelesaikan tugas pembuatan makalah ini dengan tepat waktu. Terima kasih juga kepada dosen pengajar mata kuliah IF2211 Strategi Algoritma K-01, Ir. Rila Mandala, M.Eng., Ph.D. Terakhir, penulis ingin berterima kasih kepada semua pihak yang telah berkontribusi pada pembuatan makalah ini terutama kepada seluruh pemilik referensi yang telah penulis cantumkan karena pembuatan makalah ini tidak mungkin selesai tanpa adanya bantuan dari referensi tersebut.

Penulis sadar bahwa dalam penyusunan makalah ini terdapat banyak kekurangan seperti banyak penyederhanaan kasus, kurang rincinya percobaan, kurang kompleksnya penyususan fungsi seleksi sehingga penggunaan algoritma *greedy* menjadi kurang maksimal. Penulis memohon maaf jika ada kesalahan baik dalam penulisan karena penulis masih dalam proses belajar untuk membuat makalah dengan baik dan benar. Akhir kata, penulis berharap pembuatan makalah ini dapat dikembangkan sehingga dapat bermanfaat untuk banyak orang.

DAFTAR PUSTAKA

[1] Dobrilova, T. (2021, Maret 19). How much is the gaming industry worth in 2021? Diakses pada tanggal Mei 02, 2021, dari https://techjury.net/blog/gaming-industry-worth/#gref

[2] H. (n.d.). How long is Ni no KUNI: Wrath of the White Witch? Diakses pada Mei 02, 2021, dari https://howlongtobeat.com/game?id=6585

[3] Nugroho, Andi. 2011. Algoritma dan Struktur Data Menggunakan Bahasa Pemrograman C++. Yogyakarta: Penerbit Andi Diakses pada Mei 02, 2021

[4] Munir, R. (2021, Januari 27). Greedy Bagian 1, 2, dan 3. Diakses pada Mei 01, 2021, dari https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/stima20-21.htm

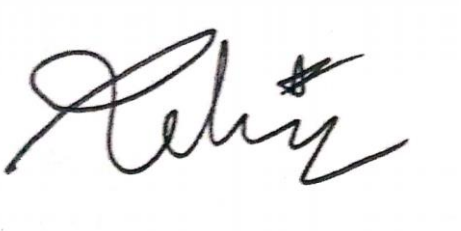
[5] J-Mill, J. (2013, April 06). Ni no Kuni: Platoon Strategy. Diakses pada Mei 02, 2021, dari https://letsgametalk.wordpress.com/2013/04/06/ni-no-kuni-platoon-strategy/

[6] SoopaSte13. (2013, April 03). Ni no Kuni: Wrath of the White Witch – Platoon Guide Diakses pada Mei 02, 2021, dari https://gamefaqs.gamespot.com/ps3/998014-ni-no-kuni-wrath-of-the-white-witch/faqs/65908

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, 03 Mei 2021



Reihan Andhika Putra 13519043