# Spesifikasi Tugas Besar


## Deskripsi Game

Bright Souls adalah permainan dengan *genre* **text-based RPG** yang dimainkan dengan cara memasukkan perintah melalui *command line interface*. Tujuan utama dari permainan ini adalah menjelajahi peta, dan mengalahkan *final boss* yang berada di suatu tempat. Permainan ini terdiri dari 2 bagian utama, yaitu penjelajahan peta dan pertarungan. Pada mode penjelajahan peta, pemain dapat berkeliling peta, dan bertemu dengan musuh. Ketika bertemu dengan musuh, pemain harus mengalahkan musuh tersebut untuk melanjutkan permainan.

## Deskripsi Soal

Dari deskripsi game di atas, anda perlu membuat sebuah program yang dapat mensimulasikan *game* tersebut menggunakan bahasa C. Interaksi dari pengguna terhadap program tersebut perlu diimplementasikan menggunakan *Command Line Interface* (CLI) dengan *command* yang akan dijelaskan pada masing-masing fitur game.

## Game Flow

1. Awalnya, perlu ada *main menu* yang memiliki New Game, Start Game, Load Game (bonus), dan Exit.
    a. Ketika memilih new game, perlu ada input nama dari user
    b. Ketika memilih start game, program perlu mengecek apakah user sudah memiliki nama, kemudian player akan ke langkah nomor 2.
    c. Ketika memilih load game, program perlu mengakses *file* eksternal yang menyimpan *state* dari *game* sebelumnya.
    d. Ketika memilih exit, program akan berhenti
2. Program akan membentuk beberapa area berukuran N*M dengan membaca dari teks file. Area-area ini akan kemudian dihubungkan membentuk peta besar. Lihat Gambar 1

agar lebih mudah dipahami.
Pada setiap awal permainan, peta besar dikonstruksi dari kumpulan area..

3. Program menampilkan peta pada terminal (Akan kemudian dispesifikasikan)
4. Program menunggu interaksi dari player. Untuk memudahkan *input* dari *player*, perintah yang diterima hanyalah satu kata. Berikut adalah perintah/*command* yang diterima
    a. GU
    *Command* ini adalah singkatan dari 'Go Up', sehingga posisi player berpindah ke atas.
    b. GD
    *Command* ini adalah singkatan dari 'Go Down', sehingga posisi player berpindah ke bawah.
    c. GL
    *Command* ini adalah singkatan dari 'Go Left', sehingga posisi player berpindah ke kiri.
    d. GR
    *Command* ini adalah singkatan dari 'Go Right', sehingga posisi player berpindah kekanan.
    e. SKILL
    *Command* ini digunakan untuk menampilkan skill yang sudah diambil, dan memilih skill baru yang akan diambil. Skill hanya mempengaruhi status pemain. Mekanisme dan tampilan skill dibebaskan, namun harus berbentuk pohon.
    f. SAVE (bonus)
    Command ini digunakan untuk menyimpan state permainan saat ini agar dapat dilanjutkan kemudian. Metode interaksi bebas.
    g. LOAD (bonus)
    *Command* ini digunakan untuk melanjutkan permainan sesuai dengan state yang ada pada text file. Metode interaksi bebas.
    h. EXIT
    *Command* ini digunakan untuk keluar dari program.
5. Musuh direpresentasikan dengan simbol E pada peta. Apabila ketika player bergerak dan bertemu dengan musuh, akan masuk ke fase bertarung.
6. Obat direpresentasikan dengan simbol M pada peta. Apabila player bertemu dengan obat, HP Player akan kembali menjadi penuh.
7. Terdapat beberapa jenis musuh yang memiliki nama dan status (HP, STR, DEF) yang berbeda. Selain itu setiap musuh juga memiliki experience point yang berbeda-beda bila berhasil dikalahkan. Seluruh data musuh disimpan di file eksternal.
8. Musuh memiliki 10 set aksi yang tersimpan di file eksternal. Di awal pertarungan, aksi musuh dibaca dari file, dan dimasukkan ke dalam stack secara random. Fase ini terdiri dari 10 ronde (20 ronde untuk *final boss*). Setiap ronde terbagi menjadi beberapa bagian:
    a. Di setiap awal ronde, pop dilakukan pada stack aksi musuh. Hasil pop kemudian ditampilkan di layar, dengan representasi yang sudah dijelaskan pada bagian

ADT Queue. Dua aksi akan disamarkan dengan karakter '**#**' untuk mempersulit permainan. Aksi yang disamarkan ditentukan secara **acak**.

Representasi aksi adalah sebagai berikut:

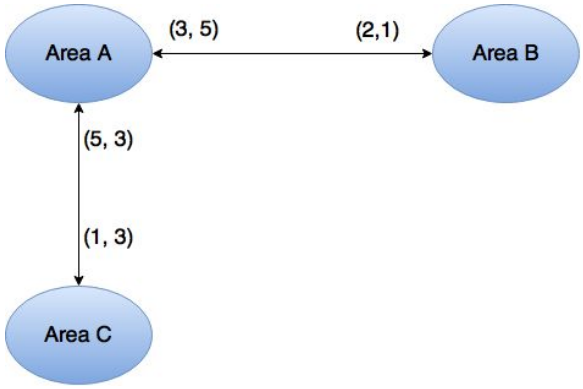
1. Attack : **A**
2. Block : **B**
3. Flank: **F**

b. Pemain memasukkan 4 aksi yang diinginkan ke dalam queue, setelah melihat aksi yang akan dilakukan oleh musuh. Pemain memasukkan input sesuai dengan representasi aksi. Input dimasukkan satu per satu. Untuk menghapus input yang pernah dimasukkan, pemain dapat memasukkan '**E**'.
c. Setelah semua perintah dimasukkan, perbandingan aksi pemain dan musuh dilakukan dengan cara dequeue untuk setiap elemen dari kedua queue. Elemen yang bersesuaian dibandingkan, dan dicari pemenangnya. dengan cara yang mirip dengan permainan kertas-gunting-batu.
    1. **Attack** kalah oleh **Block**
    2. **Block** kalah oleh **Flank**
    3. **Flank** kalah oleh **Attack**

d. Setelah perbandingan selesai, kalkulasi damage dilakukan. Aksi yang bila menang dapat menghasilkan damage. Hanyalah Attack dan Flank. Metode kalkulasi damage dibebaskan. Damage yang dihasilkan kemudian digunakan untuk mengurangi HP / nyawa dari pihak yang kalah. Hasil setiap aksi harus ditampilkan, serta hasil kalkulasinya. Setelah kalkulasi, ronde pun berakhir, dan ronde berikutnya dimulai.
e. Apabila jumlah ronde yang dijalani sudah mencapai 10 ronde (20 ronde untuk *final boss*), pertarungan berakhir tanpa pemenang. Pemain akan mundur ke petak sebelum bertemu musuh. HP pemain sesuai dengan hasil akhir pertarungan, namun HP musuh kembali penuh.
f. Apabila menang, player dapat memperoleh experience (EXP). Bila mencapai nilai EXP tertentu, player akan naik level. Mekanisme level up dibebaskan.

9. Bila pemain bertambah levelnya, status pengguna juga akan berubah. Metode pengubahan status dibebaskan. Selain itu, setiap bertambah level, skill pengguna juga akan bertambah.
10. Permainan berlanjut kembali, hingga pemain bertemu dengan *final boss*, yang berupa musuh dengan status yang jauh lebih tinggi dari biasanya. Apabila pemain berhasil menang melawan *boss*, permainan berakhir (Winner).
11. Permainan juga akan berakhir ketika HP pengguna telah habis (Game Over).
12. Setelah permainan berakhir, muncul tampilan *credit* pembuat game.

## Peta dan Transisi Area




Gambar 1. Contoh Peta Besar yang Terbentuk

Poin-poin yang ada pada *edge* graf artinya end point dari suatu area.
Misalkan area-area tersebut didefinisikan sebagai kumpulan matriks sebagai berikut

Area A

| | | | | |
|---|---|---|---|---|
| # | # | # | # | # |
| # | # | # | # | # |
| # | - | - | - | - |
| # | # | - | # | # |
| # | # | - | # | # |

Area B

| | | | | |
|---|---|---|---|---|
| # | # | # | # | # |
| - | - | # | # | # |
| # | - | - | - | # |
| # | # | - | # | # |
| # | # | # | # | # |

Area C

| | | | | |
|---|---|---|---|---|
| # | # | - | # | # |
| # | # | - | # | # |
| # | # | - | - | # |
| # | # | - | # | # |
| # | # | # | # | # |

dan Player berada pada titik (3,5) pada Area A. Ketika pengguna melakukan command sehingga Player yang seharusnya berada pada titik (3,6), akan transisi ke Area B dengan titik awal (2,1) pada Area B. Hal ini berlaku juga sebaliknya (dari Area B ke Area A).

## Tampilan Program

Catatan: yang di-*bold* merupakan input dari pemain.

### 1. Mode penjelajahan

| <NAME> | LVL:1 | HP: 20 | STR: 5 | DEF: 3 | EXP: 50 | |
|---|---|---|---|---|---|---|
| | # | # | - | # | # | |
| | # | E | - | M | # | |
| | # | # | P | - | # | |
| | # | # | - | # | # | |
| | # | # | # | # | # | |
| Got a medicine! HP +10 | | | | | | |
| Command: **GU** | | | | | | |

Pada mode ini, terdapat beberapa simbol, yaitu:

a. # : menandakan daerah tidak bisa dilewati
b. E : menandakan musuh
c. P : menandakan posisi pemain
d. M : menandakan ada obat yang dapat digunakan
e. <Name> : username
f. LVL : level player
g. HP : health dari player. Max health dihitung dari level player.
h. STR : Strength dari player, berpengaruh terhadap damage yang dihasilkan oleh player.
i. DEF : Defense dari player, berpengaruh terhadap damage yang didapatkan oleh player.
j. EXP : Experience point yang harus dicapai untuk naik level.

Ketika pemain sudah melewati batas yang ditetapkan, area otomatis berpindah.

### 2. Mode pertarungan: Input Command

| <NAME> | LVL:1 | HP: 20 | STR: 5 | DEF: 3 | Round 1 |
|---|---|---|---|---|---|
| <ENEMY NAME> | HP: 50 | Command: S # B # | | | |

| <ENEMY NAME> Attacked!<br>Please input your command | |
|---|---|
| Inserted Commands: | _ _ _ _ |
| Command: | **B** |

Hasil setelah command dimasukkan:

| <NAME> | LVL:1 | HP: 20 | STR: 5 | DEF: 3 | Round 1 |
|---|---|---|---|---|---|
| <ENEMY NAME> | HP: 50 | Command: S # B # | | | |
| <ENEMY NAME> Attacked!<br>Please input your command | | | | | |
| Inserted Commands: | B _ _ _ | | | | |
| Command: | **E** | | | | |

Setelah command ‘E’ dimasukkan:

| <NAME> | LVL:1 | HP: 20 | STR: 5 | DEF: 3 | Round 1 |
|---|---|---|---|---|---|
| <ENEMY NAME> | HP: 50 | Command: S # B # | | | |
| <ENEMY NAME> Attacked!<br>Please input your command | | | | | |
| Inserted Commands: | _ _ _ _ | | | | |
| Command: | **B** | | | | |

Setelah seluruh command dimasukkan, command diproses:

| <NAME> | LVL:1 | HP: 20 | STR: 5 | DEF: 3 | Round 1 |
|---|---|---|---|---|---|
| <ENEMY NAME> | HP: 50 | Command: >A # B # | | | |
| <ENEMY NAME> Attacked!<br><ENEMY NAME> attack <NAME>, but it’s blocked! | | | | | |

| | |
|---|---|
| | |
| Inserted Command: | >B A F A |

| | | | | | |
|---|---|---|---|---|---|
| <NAME> | LVL:1 | HP: 20 | STR: 5 | DEF: 3 | Round 1 |
| <ENEMY NAME> | HP: 40 | Command: A >F B # | | | |
| <ENEMY NAME> Attacked!<br><ENEMY NAME> attacks <NAME>, but it's blocked!<br><NAME> attacks <ENEMY NAME>! <ENEMY NAME> -10 HP | | | | | |
| Inserted Command: | B >A F A | | | | |

Perhatikan bahwa command musuh yang sebelumnya tersembunyi, kini muncul.

Setelah keempat command diproses, ronde berikutnya dimulai. Proses berulang kembali hingga ada yang kalah / 10 ronde sudah terlewati.

## Bonus

1. Peta dikonstruksi secara random (dengan beberapa *constraint*) dari area-area yang ada, sehingga setiap permainan akan memiliki peta yang berbeda.
2. Skill yang dibuat berupa skill aktif yang dapat digunakan ketika pertarungan. Silakan asumsikan sendiri mekanismenya.
3. Tampilan yang menarik (warna-warni, dll)
4. Save dan Load game
5. Ada narasi dan cerita, seiring perjalanan.

# Daftar Abstract Data Type (ADT) yang Digunakan

Dapat pula menggunakan ADT lain, cantumkan analisis alasan penggunaan ADT tersebut pada laporan.

## 1. ADT Jam

Digunakan untuk menyimpan waktu ketika melakukan *save*.

## 2. ADT Point

ADT ini digunakan untuk menunjukkan posisi *player* pada setiap waktu. Integer

## 3. ADT Matriks

ADT ini digunakan sebagai representasi sebagian dari area yang ada pada game.

## 4. ADT Mesin Karakter + Mesin Kata

ADT ini digunakan untuk:
- membaca peta dan informasi aksi musuh dari file eksternal,
- membaca *command* dari interaksi user terhadap program, dan
- membaca *state* dari *game* yang sudah pernah disimpan.

## 5. ADT Queue

ADT ini digunakan pada saat *player* bertarung dengan musuh.

## 6. ADT Stack

ADT ini digunakan pada saat *player* bertarung dengan musuh. ADT ini berfungsi sebagai tempat penyimpanan seluruh queue gerakan musuh yang akan dilakukan.

## 7. ADT List Linier (Dummy element/circular/double pointer/rekursif)

ADT ini akan digunakan untuk implementasi Graf.

## 8. ADT Tree (Binary Tree)

ADT ini digunakan untuk menyimpan *skill list* yang dimiliki oleh suatu player. Setiap *skill* direpresentasikan oleh sebuah *node*. Suatu skill dapat diambil apabila *parent* dari *node* tersebut sudah diambil. *Skill list* dibebaskan ke mahasiswa (dengan tinggi pohon minimum 3). *Skill* yang disimpan bersifat pasif, sehingga hanya berpengaruh pada *status* pemain. Pembuatan skill aktif (Dapat digunakan sebagai aksi dalam pertarungan) merupakan bonus.

## 9. ADT Graf (variasi multilist)

ADT ini digunakan untuk menghubungkan masing-masing bagian dari peta agar terbentuk suatu peta besar.

## 10. ADT Lain

ADT ini dibuat dan digunakan untuk abstraksi beberapa hal seperti Player dan Enemy. Mahasiswa dipersilakan untuk mendefinisikan sendiri.