

LAPORAN TUGAS BESAR
IF2111 Algoritma dan Struktur Data

BNMO


Dipersiapkan oleh:

Kelompok 13

18219048 Dwiky Hared Darmawan
18221162 Ceavin Rufus De Prayer Purba
18221046 Vincent Winarta
18221084 Rei Arriel Clyfton
18221118 Ardhan Nur Urfan
18221128 Fadhlan Nazhif Azizy

Sekolah Teknik Elektro dan Informatika - Institut Teknologi Bandung

Jl. Ganesha 10, Bandung 40132

	Sekolah Teknik Elektro dan Informatika ITB	Nomor Dokumen		Halaman
		IF2111-TB2-13		40
		Revisi	-	2 Desember 2022

Daftar Isi

1 Ringkasan	3
2 Penjelasan Tambahan Spesifikasi Tugas	4
1.1 Bonus Game (Amogus Fight)	4
1.2 Bonus Game (Tower of Hanoi)	5
1.3 Fitur Tambahan Game Hangman	5
3 Struktur Data (ADT)	5
2.1 ADT Stack dan Stackhistory	5
2.2 ADT Binary Tree	5
2.3 ADT List Double Pointer	6
2.4 ADT List Score	6
2.5 ADT List Linear	6
2.6 ADT Map	6
2.7 ADT Point	6
4 Program Utama	6
5 Algoritma-Algoritma Menarik	8
4.1 Fungsi SearchByChild	8
6 Data Test	8
5.1 Scoreboard	8
5.2 Reset Scoreboard	10
5.3 History <n>	12
5.4 Reset History	13
5.5 Hangman	13
5.6 Tower of Hanoi	17
5.7 Snake on Meteor	19
5.8 Amogus Fight	21
7 Test Script	27
8 Pembagian Kerja dalam Kelompok	30
9 Lampiran	31
8.1 Deskripsi Tugas Besar 2	31
1. About the System	31
2. Main Menu	31

3. Command	32
a. Command Dasar	32
b. SCOREBOARD	32
c. RESET SCOREBOARD	32
d. HISTORY <n>	32
e. RESET HISTORY	33
Spesifikasi Game	33
1. RNG	33
2. Diner Dash	33
3. Hangman	33
4. Tower of Hanoi	33
5. Snake on Meteor	34
8.2 Notulen Rapat	36
8.3 Log Activity Anggota Kelompok	36
8.4 MoM Asistensi Tugas Besar	37

1 Ringkasan

Setelah berhasil membantu Indra dan Doni dalam memperbaiki sistem robot *video game console* milik mereka, yaitu BNMO. Kami sebagai programmer handal yang dicari oleh Indra dan Doni berinisiatif untuk memprogram game tambahan untuk BNMO tersebut dengan cara membuat sebuah permainan berbasis CLI (*Command Line Interface*), serta tampilan scoreboard dan history.

Dalam laporan ini berisi penjelasan mengenai program yang telah dibuat, antara lain spesifikasi program, yang mencakup tipe data bentukan (ADT) yang digunakan pada program ini, algoritma-algoritma khusus yang terdapat pada laporan, data tes program, dan *test script* saat kami melakukan testing. Pada program ini tentunya terdapat program utama atau *main program* yang digunakan sebagai wadah sinkronisasi antar program *command* sehingga program BNMO dapat dijalankan secara utuh dan selaras. Hal - hal mengenai program utama juga akan dijelaskan secara lebih lanjut pada laporan ini. Dalam laporan juga terlampir pembagian kerja dalam kelompok, notulen rapat, deskripsi tugas besar, dan *log activity* anggota kelompok.

Program ini melanjutkan dari tugas pertama 1, menggunakan bahasa pemrograman C serta menggunakan ADT *stack*, *stackhistory*, *binary tree*, *list double pointer*, *list score*, *lists linier*, *point*, dan *map*. Beberapa fitur tambahan pada fungsi utama yang dikerjakan pada tugas besar kedua kali ini adalah scoreboard (menampilkan skor), reset scoreboard (menghapus data scoreboard), history (riwayat permainan), serta reset history (menghapus riwayat). Selain itu, terdapat game Tower of Hanoi yang disesuaikan dengan spesifikasi tugas besar 2, Hangman, Snake on Meteor, dan Amogus fight.

2 Penjelasan Tambahan Spesifikasi Tugas

2.1 Bonus Game (Amogus Fight)

Pada bonus *game*, kami membuat *game* yang mengimplementasikan ADT *binary tree*, yaitu Amogus Fight. Objektif dari *game* ini adalah *player* yang memiliki HP dan *damage* tertentu harus bisa *survive* dalam melawan *impostor-impostor* yang semakin kuat setiap *impostor* sebelumnya dikalahkan. Setiap kali *player* mengalahkan *impostor*, *player* mendapatkan *elixir* dan *score* sebesar HP *impostor* yang dikalahkan. *Player* dapat meracik atau membeli ramuan yang dapat digunakan untuk meningkatkan status (HP dan *damage*) dari *player*. *Elixir* dipakai untuk membeli ramuan atau bahan-bahan untuk membuat ramuan. Terdapat beberapa pilihan aksi yang dapat dilakukan oleh *player*. Jika *player* melakukan aksi BREW, USE, atau SKIP, akan dihitung sebagai satu putaran.

Implementasi ADT *binary tree* terdapat pada fitur meracik ramuan. Untuk membuat sebuah ramuan, tentunya terdapat bahan yang diperlukan. ADT *binary tree* di sini berperan untuk menggambarkan resep yang diperlukan untuk membuat sebuah ramuan. *Child* dari sebuah *node* menggambarkan bahan-bahan yang perlu dicampur untuk membuat ramuan, sedangkan *parent* dari kedua *child* tersebut menggambarkan hasil dari racikan kedua bahan tersebut, yang dapat berupa bahan baru, maupun ramuan yang sudah siap digunakan.

2.2 *Bonus Game (Tower of Hanoi)*

Bonus *game* kedua yang dibuat oleh kelompok kami adalah permainan Tower of Hanoi. Di dalam permainan ini ada 2 prosedur untuk mencetak *state* dari setiap *tower*, yaitu prosedur *DisplayState* dan *PrintDisk*. Selain itu, terdapat prosedur *towerofhanoi* untuk memulai permainan. Permainan akan berakhir ketika semua disk pada tower 1 sudah dipindahkan ke tower 3 dengan ketentuan ukuran disk terurut membesar dimulai dari elemen top.

2.3 *Fitur Tambahan Game Hangman*

Dalam game hangman yang dibuat oleh kelompok kami, telah ditambahkan fitur tambahan sesuai dengan spesifikasi bonus. Di dalam game hangman kami, terdapat dua pilihan menu saat pengguna melakukan play game, yaitu *PLAY* dan *TAMBAH KATA*. Menu *PLAY* adalah menu untuk bermain langsung dan menu *TAMBAH KATA* adalah menu untuk menambah kata-kata baru ke dalam kamus daftar kata. Selain itu, kamus daftar kata yang kelompok kami gunakan dibaca langsung dari sebuah file bernama “kata.txt” yang pada awalnya berisi 125 kata random dari bahasa Indonesia.

3 Struktur Data (ADT)

3.1 *ADT Stack dan Stackhistory*

ADT stack merupakan ADT yang merepresentasikan sebuah tumpukan. Dalam tumpukan berlaku LIFO (*last in First out*). Dalam hal ini apabila terjadi penambahan suatu elemen akan dilakukan dengan cara menambahkan elemen pada urutan paling atas, begitu juga dengan penghapusan suatu elemen, penghapusan akan dilakukan pada urutan paling atas. ADT stack ini akan digunakan pada *game* Tower of Hanoi yang merupakan permainan yang sangat berkaitan dengan tumpukan ini.

Untuk mengakomodasi kebutuhan pada game Tower of Hanoi, ADT stack yang digunakan di sini kami modifikasi pada bagian pop dan push. Pada kedua prosedur ditambahkan parameter baru, yaitu *succeed*, untuk mengetahui apakah pop atau push bekerja dengan baik. Pada bagian pop, initial state dari stack boleh kosong, tetapi jika dilakukan pop pada stack yang kosong prosedur hanya akan melakukan pengubahan pada nilai *succeed* menjadi false. Pada bagian push, terjadi sedikit perubahan pada aturan penambahan elemen, yaitu elemen yang ingin ditambahkan dalam suatu stack harus lebih kecil dari elemen TOP pada stack tersebut. Jika memenuhi syarat tersebut, prosedur akan mengubah nilai *succeed* menjadi false.

Pada ADT stackhistory, memiliki cara kerja yang sama seperti stack. Namun, perbedaannya adalah tipe element pada list tersebut merupakan word, bukan integer.

3.2 *ADT Binary Tree*

ADT binary tree merupakan tipe data bentukan untuk mengimplemtasi struktur tree yang digunakan pada game amogusfight. Pada kata bentukan ini, terdapat *nodeinfo*type yang

merupakan info dari akar tree tersebut, dan addrNode left dan right yang merupakan pointer ke address node untuk cabang kiri dan kanan dari akar.

3.3 ADT List Double Pointer

ADT list double pointer merupakan tipe data bentukan berupa linked list dengan double pointer, digunakan untuk game snake on meteor. Pada data bentukan tersebut, terdapat info list yang berupa integer, dan sebuah pointer ke address yang mengarah ke elemen list sebelum dan sesudah. Selain itu, terdapat pointer yang akan menunjuk ke elemen pertama list dan last. Selain itu, terdapat POINT yang mendata posisi dari bagian ular nya.

3.4 ADT List Score

ADT list score merupakan tipe data bentukan yang terdiri dari data bentukan ElTypeScore dan Neff (jumlah efektif elemen yang ada pada list). ElTypeScore sendiri merupakan data bentukan yang terdiri dari GameTitle berjenis word dan MapScoreboard berjenis adt Map. ADT ini digunakan dalam function scoreboard.

3.5 ADT List Linear

ADT list linear merupakan tipe data bentukan list berkait yang hanya mempunyai single pointer ke elemen next. Terdapat info elemen yang berupa binary tree, dan pointer ke elemen berikutnya. Lalu terdapat pointer yang menunjuk ke elemen pertama pada list.

3.6 ADT Map

ADT map merupakan tipe data bentukan yang berisi element yang bertipe data infotypeMap (berisi key (bertipe word) dan value (integer)) dan count yang merupakan jumlah element map. ADT map sendiri digunakan untuk data scoreboard.

3.7 ADT Point

ADT point merupakan tipe data bentukan yang menyimpan informasi berupa X dan Y dengan tipe data integer yang merepresentasikan sebuah posisi pada suatu sistem koordinat. ADT ini digunakan pada ADT list double pointer untuk menyimpan informasi posisi sebuah elemen list atau badan dari snake pada game snake on meteor.

4 Program Utama

Program utama, merupakan lanjutan dari tugas besar pertama, berfungsi sebagai driver program lain dengan memasukkan fungsi yang telah dibuat yaitu, string.h, arraydin.h, queue.h, word.h, mesinkata.h, dan mesin karakter.h. Pada program utama juga terdapat fungsi main menu yang menampilkan opsi START dan LOAD. Ketika program dijalankan, program akan meminta pemain untuk memasukkan START atau LOAD. Ketika pemain memasukkan START atau LOAD yang valid maka program akan dimulai dan menampilkan perintah-perintah yang dapat dimasukkan oleh pemain. Jika perintah tidak valid maka akan menampilkan pesan masukkan tidak valid dan pemain diminta memasukkan perintah lagi.

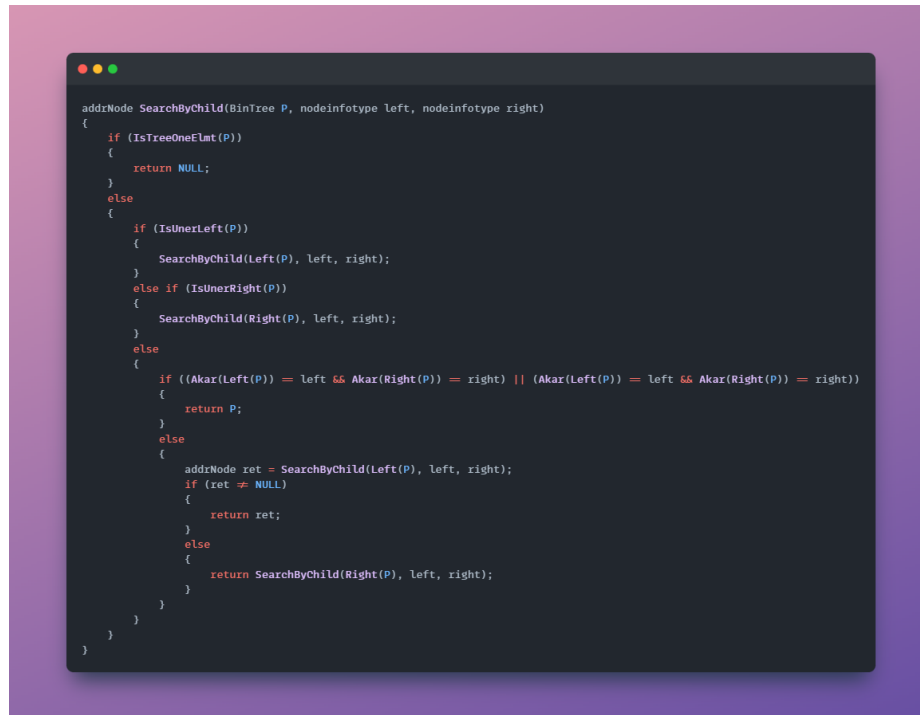
Pada control loop pemain akan diminta untuk memasukkan perintah yang valid. SAVE untuk menyimpan file berisi state game dari pemain. Terdapat tambahan function dari tugas besar pertama, yaitu :

- SCOREBOARD : pada command ini, program akan menampilkan nilai-nilai setiap orang pada masing-masing game
- RESET SCOREBOARD : pada command, ini program akan menghapus nilai setiap orang pada game tertentu, yang dipilih oleh pengguna.
- HISTORY <n> : command ini berfungsi untuk menampilkan riwayat permainan apa saja yang telah dimainkan sejumlah n game terakhir. Jika nilai input n lebih besar dari jumlah game yang sudah dimainkan, maka akan menampilkan semua game.
- RESET HISTORY : command reset history akan menghapus semua riwayat permainan yang telah dimainkan, dan akan dimulai kembali dari 0.

Setelah pengguna selesai menggunakan program tentu pengguna akan melakukan command QUIT untuk keluar dari program. Pada saat menginput command QUIT, maka akan ada dua pilihan yaitu untuk menyimpan file yang sudah diproses saat menggunakan program atau tidak menyimpan file.

5 Algoritma-Algoritma Menarik

5.1 Fungsi SearchByChild



Gambar 5.1.1

Gambar di atas merupakan algoritma untuk mencari node yang memiliki child left dan right yang sesuai dengan parameter input dari fungsi tersebut. Basis dari fungsi ini adalah basis 1 karena dipastikan pemanggilan fungsi ini akan menerima tree tidak kosong. Algoritma ini unik karena fungsi ini merupakan pengembangan lebih lanjut dari primitif-primitif yang sudah ada pada ADT tree.

6 Data Test

6.1 Scoreboard

Setelah kita memasukan command START pada program awal, masukan command “SCOREBOARD” pada main function untuk menampilkan scoreboard sementara yang ada


```

**** SCOREBOARD GAME RNG ****
| NAMA      | SKOR      |
|-----|
| Vincent   | 96        |
|-----|

**** SCOREBOARD GAME Diner DASH ****
| NAMA      | SKOR      |
|-----|
|----- SCOREBOARD KOSONG -----|

**** SCOREBOARD GAME HANGMAN ****
| NAMA      | SKOR      |
|-----|
| vINCENT   | 0         |
|-----|

**** SCOREBOARD GAME TOWER OF HANOI ****
| NAMA      | SKOR      |
|-----|
|----- SCOREBOARD KOSONG -----|

**** SCOREBOARD GAME SNAKE ON METEOR ****
| NAMA      | SKOR      |
|-----|
|----- SCOREBOARD KOSONG -----|

**** SCOREBOARD GAME TICTACTOE ****
| NAMA      | SKOR      |
|-----|
| Cepus     | 5         |
|-----|

**** SCOREBOARD GAME AMOGUS FIGHT ****
| NAMA      | SKOR      |
|-----|
|----- SCOREBOARD KOSONG -----|

Enter to back to main menu...

```

Gambar 6.1.3

6.2 *Reset Scoreboard*

Command “RESET SCOREBOARD” digunakan untuk menghapus data scoreboard yang ada.

```

1. SAVE <nama_file>
2. CREATE GAME
3. LIST GAME
4. DELETE GAME
5. QUEUE GAME
6. PLAY GAME
7. SKIP GAME <n>
8. SCOREBOARD
9. RESET SCOREBOARD
10. HISTORY <n>
11. RESET HISTORY
12. QUIT
13. HELP

ENTER COMMAND: RESET SCOREBOARD

```

Gambar 6.2.1

Setelah itu, program akan meminta masukan scoreboard untuk game yang diinginkan untuk dihapus oleh pengguna. Pada kesempatan kali ini, kita akan menghapus scoreboard untuk game HANGMAN yang tertera pada nomor 3

```

DAFTAR SCOREBOARD:
0. ALL
1. RNG
2. Diner DASH
3. HANGMAN
4. TOWER OF HANOI
5. SNAKE ON METEOR
6. TICTACTOE
7. AMOGUS FIGHT

SCOREBOARD YANG INGIN DIHAPUS: 3

APAKAH KAMU YAKIN INGIN MELAKUKAN RESET SCOREBOARD HANGMAN (YA/TIDAK)? YA

Scoreboard berhasil di-reset.

Enter to back to main menu...

```

Gambar 6.2.2

Dengan begitu, scoreboard untuk game hangman akan kembali kosong

```

**** SCOREBOARD GAME RNG ****
| NAMA      | SKOR      |
|-----|
| Vincent   | 96        |
|-----|

**** SCOREBOARD GAME Diner DASH ****
| NAMA      | SKOR      |
|-----|
|-----|
|-----| SCOREBOARD KOSONG |
|-----|

**** SCOREBOARD GAME HANGMAN ****
| NAMA      | SKOR      |
|-----|
|-----|
|-----| SCOREBOARD KOSONG |
|-----|

**** SCOREBOARD GAME TOWER OF HANOI ****
| NAMA      | SKOR      |
|-----|
|-----|
|-----| SCOREBOARD KOSONG |
|-----|

**** SCOREBOARD GAME SNAKE ON METEOR ****
| NAMA      | SKOR      |
|-----|
|-----|
|-----| SCOREBOARD KOSONG |
|-----|

**** SCOREBOARD GAME TICTACTOE ****
| NAMA      | SKOR      |
|-----|
|-----|
| Cepus     | 5         |
|-----|

**** SCOREBOARD GAME AMOGUS FIGHT ****
| NAMA      | SKOR      |
|-----|
|-----|
|-----| SCOREBOARD KOSONG |
|-----|

Enter to back to main menu...

```

Gambar 6.2.3

Selanjutnya, pengguna akan menghapus scoreboard untuk semua game, dengan kembali memanggil command “RESET SCOREBOARD” dan memilih angka 0

```

DAFTAR SCOREBOARD:
0. ALL
1. RNG
2. Diner DASH
3. HANGMAN
4. TOWER OF HANOI
5. SNAKE ON METEOR
6. TICTACTOE
7. AMOGUS FIGHT

SCOREBOARD YANG INGIN DIHAPUS: 0

APAKAH KAMU YAKIN INGIN MELAKUKAN RESET SCOREBOARD ALL (YA/TIDAK)? YA

Scoreboard berhasil di-reset.

Enter to back to main menu...

```

Gambar 6.2.4

Dengan begitu, scoreboard untuk semua game akan kembali kosong

```

**** SCOREBOARD GAME RNG ****
| NAMA          | SKOR          |
|-----|-----|
| SCOREBOARD KOSONG |
|-----|-----|

**** SCOREBOARD GAME Diner DASH ****
| NAMA          | SKOR          |
|-----|-----|
| SCOREBOARD KOSONG |
|-----|-----|

**** SCOREBOARD GAME HANGMAN ****
| NAMA          | SKOR          |
|-----|-----|
| SCOREBOARD KOSONG |
|-----|-----|

**** SCOREBOARD GAME TOWER OF HANOI ****
| NAMA          | SKOR          |
|-----|-----|
| SCOREBOARD KOSONG |
|-----|-----|

**** SCOREBOARD GAME SNAKE ON METEOR ****
| NAMA          | SKOR          |
|-----|-----|
| SCOREBOARD KOSONG |
|-----|-----|

**** SCOREBOARD GAME TICTACTOE ****
| NAMA          | SKOR          |
|-----|-----|
| SCOREBOARD KOSONG |
|-----|-----|

**** SCOREBOARD GAME AMOGUS FIGHT ****
| NAMA          | SKOR          |
|-----|-----|
| SCOREBOARD KOSONG |
|-----|-----|

Enter to back to main menu...

```

Gambar 6.2.5

6.3 *History <n>*

History adalah command yang akan menampilkan game yang sudah dimainkan sejumlah n terakhir. Pada testing pertama, kita akan melihat 2 game paling terakhir yang telah dimainkan.

```

Berikut adalah daftar Game yang telah dimainkan
1. TICTACTOE
2. HANGMAN

Enter to back to main menu...

```

Gambar 6.3.1

Maka, data yang muncul adalah game TICTACTOE dan HANGMAN sebagai 2 game terakhir yang dimainkan. Lalu, testing akan mencoba memasukan nilai n yang lebih besar dari jumlah game yang sudah dimainkan (3), dengan nilai n adalah 8.

```

Berikut adalah daftar Game yang telah dimainkan
1. TICTACTOE
2. HANGMAN
3. RNG

Enter to back to main menu...

```

Gambar 6.3.2

Untuk kasus ini, program akan menampilkan semua game yang telah dimainkan pengguna

6.4 *Reset History*

Memasukan command ini pada program utama, akan mengakibatkan semua riwayat permainan akan hilang. Ketika dimasukan, akan ada pertanyaan sebagai konfirmasi apakah benar ingin menghapus riwayat.

```

APAKAH KAMU YAKIN INGIN MELAKUKAN RESET HISTORY? TIDAK

History tidak jadi di-reset. Berikut adalah daftar Game yang telah dimainkan
1. TICTACTOE
2. HANGMAN
3. RNG

Enter to back to main menu...

```

Gambar 6.4.1

Pada kesempatan ini, testing dilakukan dengan menjawab “TIDAK” sehingga tidak menghapus riwayat permainan.

```

APAKAH KAMU YAKIN INGIN MELAKUKAN RESET HISTORY? ya
Masukan tidak valid coba lagi.

APAKAH KAMU YAKIN INGIN MELAKUKAN RESET HISTORY? YA

History berhasil di-reset.

Enter to back to main menu...

```

Gambar 6.4.2

Jika pada pertanyaan konfirmasi, pengguna memasukan “YA” secara valid (dengan *caps lock*), maka riwayat akan terhapus. Sehingga, saat memanggil command history, tidak ada data yang ditampilkan.

```

Berikut adalah daftar Game yang telah dimainkan

===== Daftar Kosong =====

Enter to back to main menu...

```

Gambar 6.4.3

6.5 *Hangman*

Saat melakukan memasukan command “PLAY GAME” ketika urutan queue game pada game hangman, maka game tersebut akan di run. Lalu, akan muncul apakah pengguna ingin langsung bermain atau menambahkan kata pada jawaban di game hangman tersebut.

```

===== SELAMAT DATANG DI HANGMAN =====
=====
***** MENU *****
PLAY      : Bermain game
TAMBAH KATA : Menambah kata dalam game

Masukkan pilihan menu: TAMBAH KATA

```

Gambar 6.5.1

Setelah itu, program akan meminta masukan untuk kata yang mau ditambahkan ke dalam file kata.txt sebagai kata yang harus ditebak pada game hangman.

```

Masukkan kata baru (DALAM HURUF BESAR): ITB

Kata berhasil ditambahkan.
Enter to continue...

```

Gambar 6.5.2

Tambahan kata yang dimasukan, harus diisi dengan semua huruf besar. Selanjutnya, pengguna akan mencoba untuk memainkan dengan memasukan command ‘PLAY’.

```

Tebakan sebelumnya: -
Kata: _____
Kesempatan: 10

Masukkan tebakan:

```

Gambar 6.5.3

Tampilan awal permainan hangman, dimana terdapat tebakan sebelumnya, berapa huruf dalam kata, serta kesempatan yang tersisa.

```

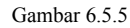
Tebakan sebelumnya: a
Kata: _____
Kesempatan: 9

=====
Masukkan tebakan: I

```

Gambar 6.5.4

Pada tebakan sebelumnya, pengguna menebak huruf “A” yang ternyata salah, sehingga total kesempatan akan berkurang 1. Selanjutnya, pemain akan menebak huruf “I”.



```
Tebakan sebelumnya: aiejodxb
Kata: __B__
Kesempatan: 3

+---+
|   |
o   |
/|\  |
    |
=====
Masukkan tebakannya:
```

Gambar 6.5.6

```
Tebakan sebelumnya: ajiejodxbcg  
Kata: G_B_  
Kesempatan: 2
```

```
+---+  
|   |  
o   |  
/\  |  
/   |  
    |  
=====
```

Masukkan tebakan: u

Gambar 6.5.7

```
Tebakan sebelumnya: aiejodxbcgu
Kata: GUBU
Kesempatan: 2

+---+
|   |
| o  |
| /|\ |
| /   |
|   |
=====
Masukkan tebakkan: k
```

Gambar 6.5.8

Pemain benar lagi dalam menebak huruf “u”, sehingga tersisa 1 huruf yang belum berhasil ditebak. Berikutnya, pemain akan memilih huruf “k”.

```
Tebakan sebelumnya: aiejodxbcg
Kata: GUBU_
Kesempatan: 2

+---+
|   |
o   |
/|\  |
/   |
|   |
=====
Masukkan tebakan: k

Berhasil menebak kata GUBUK! Kamu mendapatkan 5 poin!

Enter to continue...
```

Gambar 6.5.9

Pemain berhasil menebak kata “GUBUK” dan karena masih terdapat 2 kesempatan tersisa, pemain akan menebak kata berikutnya.

```
Tebakan sebelumnya: a
Kata: A_A_____
Kesempatan: 2

+---+
|   |
o   |
/|\  |
/   |
|   |
=====
Masukkan tebakan: X
```

Gambar 6.5.10

Pada kata yang kedua, sisa kesempatan dan gambar hangman masih sama dari pemilihan kata yang pertama. Lalu, pemain benar menebak untuk huruf “A”, dan akan mencoba menebak huruf “x”.

```
Tebakan sebelumnya: ax
Kata: A_A_____
Kesempatan: 1

+---+
|   |
o   |
/|\  |
/   |
|   |
=====
Masukkan tebakan: z

Kesempatan kamu untuk menebak sudah habis. Selamat kamu berhasil mengumpulkan 5 poin!

Masukan Nama:
```

Gambar 6.5.11

Ternyata huruf “x” kurang tepat, dan memasukan huruf “z”. Sebagai kesempatan terakhir, huruf “z” juga tidak benar, maka pemain akan berhenti bermain. Pemain diminta untuk memasukan nama untuk nama dan poin pemain

6.6 Tower of Hanoi

Saat melakukan memasukan command “PLAY GAME” ketika urutan queue game pada game Tower of Hanoi, maka game tersebut akan di run. Lalu, akan muncul berapa tumpukan piringan pada game tersebut, pilihan antara 3-9.

Banyak piringan (3-9):

Gambar 6.6.1

Pada keperluan testing kali ini, pengguna memilih 3 piringan

```
Moves: 0
Minimum moves to solve: 7

  |           |           |
 ***         |           |
*****       |           |
*****       |           |
  A           B           C

Tiang asal: A
Tiang tujuan: C
```

Gambar 6.6.2

Setelah memilih, maka akan muncul tampilan seperti gambar 6.6.2. Akan tertera sudah berapa kali pemain melakukan gerakan, minimum gerak untuk menyelesaikan game, serta input untuk memindahkan piringan dari tiang asal ke tujuan.

```
Moves: 1
Minimum moves to solve: 7
```

 ***** ***** A	 B	 *** C
------------------------------	---------------------------	------------------------

Tiang asal:

Gambar 6.6.3

Tampilan akan menunjukan bahwa piringan teratas pada tiang A akan dipindahkan ke tiang C.

```
Moves: 1
Minimum moves to solve: 7
```

 ***** ***** A	 B	 *** C
-------------------------	---------------------------	------------------------------

Tiang asal:

Gambar 6.6.4

Pemain kembali ingin memindahkan dari tiang asal A ke tujuan C. Namun, akan menampilkan gambar yang sama seperti 6.6.3, karena hal tersebut tidak valid. Karena, piringan yang ada di A memiliki ukuran lebih besar dibandingkan tiang C.

```
Tiang asal: C
Tiang tujuan: C
Input tidak valid!
```

Gambar 6.6.5

Selanjutnya, pemain ingin memindahkan dari tiang asal C ke tujuan C. Tentunya hal ini juga invalid dikarenakan asal dan tujuan pada tiang yang sama.

```
Moves: 3
Minimum moves to solve: 7
```

```
      |           |           |
      |           |           |
      |           *   *   *   |
***** *   *   *   |
A       B             C
```

Tiang asal:

Gambar 6.6.6

Pemain akhirnya memindahkan dari C ke B, sehingga moves akan bertambah. Saat ini sudah terjadi 3 pergerakan.

```
Moves: 4 •
Minimum moves to solve: 7
```

 	 *** *****	 *****
A	B	C

Tiang asal: B
Tiang tujuan: A

Gambar 6.6.7

Selanjutnya, pemain memindahkan piringan A ke C. Pada kasus ini, piringan terbesar sudah berada pada paling dasar di tiang C. Lalu, pemain akan memindahkan dari tiang B ke A.

Moves: 5
Minimum moves to solve: 7

***	****	*****
A	B	C

Tiang asal: B
Tiang tujuan: C

Gambar 6.6.8

Setelah itu, pemain akan memindahkan dari tiang B ke tiang C.

```
Moves: 7
Minimum moves to solve: 7
```


A	B	C

•

Berhasil! Kamu menyelesaikan dalam 7 langkah.
Skor: 6
Masukan Nama:

Gambar 6.6.9

Terakhir, piringan di A akan dipindahkan ke C, dengan begitu game telah selesai dan pemain berhasil menyelesaikan permainan tersebut. Akan diminta masukan nama untuk nantinya tertera pada scoreboard berisi nama dan skor yang didapat.

6.7 *Snake on Meteor*

Setelah berhasil menjalankan game, maka kita akan mendapat tampilan awal map dari game yang terdiri dari snake, makanan, dan obstacle.

```
Selamat datang di snake on meteor!
Mengenerate peta, snake dan makanan...
Berhasil digenerate!
-----

Enter to continue...
Berikut merupakan peta permainan:

+---+---+---+---+---+
|   | O |   |   |   |
+---+---+---+---+---+
| # |   |   |   |   |
+---+---+---+---+---+
|   |   |   |   |   |
+---+---+---+---+---+
|   |   |   |   |   |
+---+---+---+---+---+
| H |   |   | 2 | 1 |
+---+---+---+---+---+
```

Gambar 6.7.1

Kemudian, untuk memainkan game, input yang valid menggunakan w/a/s/d. setelah turn 1, maka meteor akan muncul

```
TURN 1:
Silahkan masukkan command anda: d

Berhasil bergerak!
Berikut merupakan peta permainan:
Anda terkena meteor!

+---+---+---+---+---+
|   | O |   |   |   |
+---+---+---+---+---+
| # |   |   |   |   |
+---+---+---+---+---+
|   |   |   |   |   |
+---+---+---+---+---+
|   |   |   |   |   |
+---+---+---+---+---+
| 1 | H |   |   | M |
+---+---+---+---+---+
Silahkan lanjutkan permainan
```

Gambar 6.7.2

Salah satu bagian dari ular terkena meteor, maka akan muncul pesan anda terkena meteor.

```
TURN 2:
Silahkan masukkan command anda: s

Berhasil bergerak!
Berikut merupakan peta permainan:

+---+---+---+---+
|   H   |   |   |
+---+---+---+---+
| #   |   |   | O |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
|   M   |   |   |
+---+---+---+---+
| 2 | 1 |   |   |
+---+---+---+---+

Anda beruntung tidak terkena meteor! Silahkan lanjutkan permainan
TURN 3:
Silahkan masukkan command anda: 
```

Gambar 6.7.3

pada kondisi turn 2, input s dianggap valid karena ular dapat menembus dinding dan muncul pada bagian dinding lainnya. Karena pada titik head bergerak terdapat makanan, maka ular tumbuh satu segmen di sebelah kiri posisi ekor yang seharusnya menghasilkan ekor baru.

```
TURN 3:
Silahkan masukkan command anda: d

Berhasil bergerak!
Berikut merupakan peta permainan:
Anda terkena meteor!

+---+---+---+---+
|   1   | M |   |   |
+---+---+---+---+
| #   |   |   | O |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
|   2   |   |   |
+---+---+---+---+

Kepala snake terkena meteor.
Game berakhir. skor : 4
Masukan Nama: 
```

Gambar 6.7.4

apabila game dilanjutkan dan ular berhasil bergerak, maka segmen ular akan menempati posisi sebelumnya dari segmen di depannya. pada turn 3, tempat head bergerak dijatuhi meteor sehingga ular akan mati dan game berakhir. skor yang dihitung adalah bagian badan ular tanpa kepala. satu segmen mendapat dua poin, karena tersisa 2 segmen sehingga poin yang didapat player adalah 4.

Setelah berhasil menjalankan game, maka kita akan mendapat tampilan awal map dari game yang terdiri dari snake, makanan, dan obstacle.

6.8 Amogus Fight

Tampilan awal dari game ini saat dijalankan adalah menampilkan command list yang dapat digunakan oleh user seperti yang terdapat pada gambar di bawah.

```
===== COMMAND LIST =====
RESEP           : Melihat resep untuk membuat potion
BUY             : Membeli bahan/potion
BREW <ID potion 1> <ID potion 2> : Membuat potion
USE <ID potion> : Menggunakan potion
SKIP           : Melewatkan 1 turn tanpa melakukan action
USED          : Melihat potion yang sedang digunakan
STATUS        : Melihat status dari Amogus dan Impostor
INFO          : Melihat info dari potion-potion pada game
HELP          : Melihat list command yang dapat digunakan

Enter to continue...|
```

Gambar 6.8.1

Setelah menekan enter, program akan menampilkan state dari player dan enemy. Pertama, akan dilakukan testing pada command HELP. Jika memanggil command HELP, maka program akan menampilkan seperti yang terdapat pada gambar 6.8.1.

```
Elixir: @
[=====]

               .eeeeeee
              ee.....ee
             e .....##e(ee.
            eC.. @#,,, ,ee
           eeeee./..@###, .....e
          e ./e/...e#####ee
         e///e/.....e
        e///e/.....e
       e///e/C.....ee
      ee///e///C.....//e
     ee////e e/////e
    eeeeeee eeeeeee

                AMOGUS

                IMPOSTOR

===== INVENTORY =====
ID      | Nama                  | Durasi
-----|-----|-----
1      | Water Bottle         | 1
2      | Nether Wart           | 1
9      | Awkward Potion       | 1

Masukkan perintah!
>> HELP
```

Gambar 6.8.2

Kemudian, akan dilakukan testing pada command yang seharusnya menimbulkan pesan eror. Di sini terdapat 2 contoh, yaitu BREW dan USE. Jika command yang diinput tidak sesuai format, maka akan ditampilkan pesan eror seperti yang terdapat pada gambar 6.8.3.

```
===== INVENTORY =====
ID      | Nama                  | Durasi
-----|-----|-----
1       | Water Bottle         | 1
2       | Nether Wart          | 1
9       | Awkward Potion       | 1

Input tidak valid!
```

Gambar 6.8.3

Kemudian akan dilakukan testing pada command RESEP. Saat memanggil command RESEP, program akan menampilkan daftar resep yang valid.

```

===== DAFTAR RESEP =====
Awkward Potion      : Water Bottle + Nether Wart
Potion of Weakness   : Fermented Spider Eye + Awkward Potion
Potion of Strength   : Blaze Powder + Awkward Potion
Potion of Defense    : Magma Cream + Awkward Potion
Potion of Poison     : Spider Eye + Awkward Potion
Potion of Harming    : Fermented Spider Eye + Potion of Poison
Potion of Regen      : Ghost Tear + Awkward Potion
Potion of Healing    : Glistening Melon + Awkward Potion
Potion of Power      : Potion of Strength + Glowstone
Potion of Curing     : Potion of Regen + Glowstone
Potion of Weakness II : Potion of Weakness + Redstone
Potion of Strength II : Potion of Strength + Redstone
Potion of Defense II  : Potion of Defense + Redstone
Potion of Poison II   : Potion of Poison + Redstone
Potion of Regen II    : Potion of Regen + Redstone

Enter to continue...|

```

Gambar 6.8.4

Kemudian akan dilakukan testing command BREW 1 2. Karena bahan 1 dan 2 di-*brew*, maka kedua bahan tersebut akan menghilang dan muncul awkward potion berdasarkan resep yang ada.

```

===== INVENTORY =====
ID      | Nama                | Durasi
-----|-----|-----
2       | Nether Wart         | 1
9       | Awkward Potion      | 1
9       | Awkward Potion      | 1

```

Gambar 6.8.5

Kemudian akan dilakukan testing command INFO. Saat memanggil command ini, maka akan ditampilkan efek dari setiap ramuan yang ada.

```

===== INFO RAMUAN =====
Potion of Weakness   : Reduce Amogus damage by 10% of base damage for 5 turns
Potion of Strength   : Increase Amogus damage by 50% of base damage for 3 turns
Potion of Defense    : Reduce Impostor damage by 50% of base damage for 3 turns
Potion of Poison     : Reduce Amogus HP by 10% of base HP for 3 turns
Potion of Regen      : Increase Amogus HP by 10% of base HP for 5 turns
Potion of Harming    : Reduce Amogus HP by 50% of base HP
Potion of Healing    : Increase Amogus base HP by 25%
Potion of Power      : Increase Amogus base damage by 25%
Potion of Curing     : Restore Amogus HP
Potion of Weakness II : Reduce Amogus damage by 10% of base damage for 10 turns
Potion of Strength II : Increase Amogus damage by 50% of base damage for 6 turns
Potion of Defense II  : Reduce Impostor damage by 50% of base damage for 6 turns
Potion of Poison II   : Reduce Amogus HP by 10% of base HP for 6 turns
Potion of Regen II    : Increase Amogus HP by 10% of base HP for 10 turns

Enter to continue...|

```

Gambar 6.8.6

Jika memanggil command SKIP, state akan otomatis berubah tanpa melakukan apa apa. Jika musuh berhasil dikalahkan, akan muncul musuh baru, skor dan elixir bertambah, dan muncul bahan/ramuan baru.


```

===== INVENTORY =====
ID      | Nama                | Durasi
-----|-----
1       | Water Bottle        | 1
2       | Nether Wart         | 1
2       | Nether Wart         | 1
2       | Nether Wart         | 1
9       | Awkward Potion      | 1
9       | Awkward Potion      | 1
13      | Potion of Poison     | 3

Masukkan perintah!
>> USE 13

```

Gambar 6.8.10

Untuk mencoba command USE, maka akan dipanggil USE 13 (Menggunakan Potion of Poison). Kemudian akan terlihat bahwa HP dari player akan berkurang sebanyak 10%. Lalu untuk melihat potion yang sedang digunakan, dapat menggunakan command USED, sehingga akan muncul tabel seperti pada gambar di bawah.

```

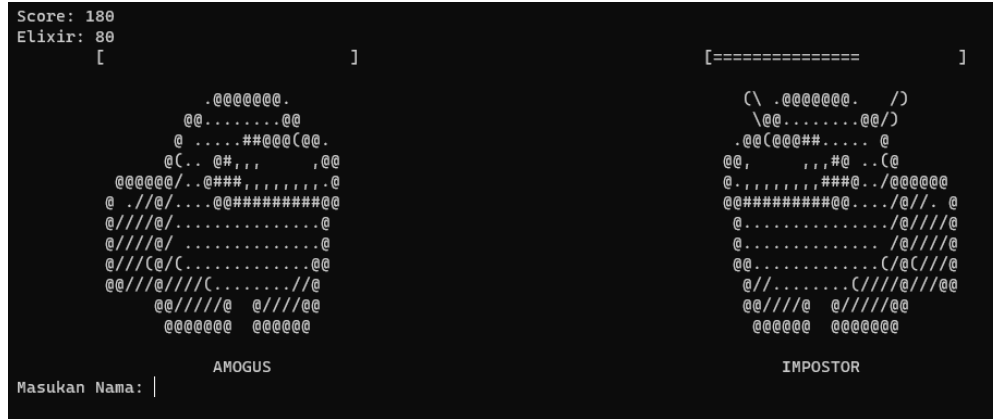
===== INVENTORY =====
ID      | Nama                | Durasi
-----|-----
13      | Potion of Poison     | 2

Enter to continue...

```

Gambar 6.8.11

Saat digunakan, potion akan otomatis digunakan untuk 1 turn, sehingga yang seharusnya efek potion of poison bertahan selama 3 turn, otomatis digunakan 1 turn sehingga yang ditampilkan pada tabel tersisa 2 turn.



Gambar 6.8.15

7 Test Script

No.	Fitur yang Dites	Tujuan Testing	Langkah-Langkah Testing	Input Data Test	Hasil yang Diharapkan	Hasil yang Keluar
1	SCOREBOARD	Untuk menguji SCOREBOARD berjalan dengan baik	Menjalankan program, memasukkan perintah START, memasukkan perintah SCOREBOARD	Data test 6.1	Scoreboard berhasil ditampilkan tanpa adanya kesalahan	Sesuai yang diharapkan
2	RESET SCOREBOARD	Untuk menguji RESET SCOREBOARD berjalan dengan baik	Menjalankan program, memasukkan perintah START, memasukkan perintah SCOREBOARD untuk mengecek isi SCOREBOARD, memasukkan perintah RESET SCOREBOARD, memasukkan perintah SCOREBOARD untuk mengecek apakah SCOREBOARD sudah di reset	Data test 6.2	Scoreboard berhasil di reset	Sesuai yang diharapkan
3	HISTORY	Untuk menguji apakah HISTORY berjalan dengan baik	Menjalankan program, memasukkan perintah START, memasukkan	Data test 6.3	History permainan dengan jumlah yang dimasukkan	Sesuai yang diharapkan

			perintah HISTORY <n>		berhasil ditampilkan	
4	RESET HISTORY	Untuk menguji apakah RESET HISTORY berjalan dengan baik	Menjalankan program, memasukkan perintah START, memasukkan perintah RESET HISTORY, memasukkan perintah TIDAK, memasukkan perintah RESET HISTORY, memasukkan perintah ya	Data test 6.4	History permainan tidak di reset ketika memasukkan perintah TIDAK dan history permainan di reset ketika memasukkan perintah YA	Sesuai yang diharapkan
6	HANGMAN	Untuk menguji apakah game HANGMAN berjalan dengan baik	Menjalankan program, memasukkan perintah START, memasukkan HANGMAN dengan perintah QUEUE GAME dan 3, memulai permainan dengan perintah PLAY GAME, memasukkan perintah TAMBAH KATA untuk menambah kata, memasukkan kata yang ditambah dengan huruf besar, memasukkan perintah PLAY GAME, memasukkan huruf a sampai z untuk menebak kata, setelah selesai pemain memasukkan nama dan poin yang didapat	Data test 6.5	Kata tebakan berhasil ditambahkan, permainan berjalan dengan baik dengan hasil pemain kalah	Sesuai yang diharapkan
6	TOWER OF HANOI	Untuk menguji apakah game TOWER OF HANOI berjalan dengan baik	Menjalankan program, memasukkan perintah START, memasukkan HANGMAN dengan perintah	Data test 5.6	Berhasil memindahkan piringan dari A ke C sesuai dengan urutan dengan jumlah gerakan	Sesuai yang diharapkan

			QUEUE GAME dan 4, memulai permainan dengan perintah PLAY GAME, memasukan huruf A,B, dan C sebagai perintah untuk memindahkan piringan dari tiang ke tiang lainnya. Setelah berhasil, akan diminta masukan nama untuk dicatat di scoreboard berupa nama dan skor yang didapat.		sebanyak 7 dan mendapat skor 6.	
7	SNAKE ON METEOR	Untuk menguji apakah game SNAKE ON METEOR berjalan dengan baik	Menjalankan program, memasukkan perintah START, memasukkan SNAKE ON METEOR dengan perintah QUEUE GAME, memulai permainan dengan perintah PLAY GAME, memasukkan perintah w/a/s/d untuk menggerakkan ular, setelah permainan selesai pemain memasukkan nama dan skor yang didapat.	Data test 6.7	Permainan berjalan dengan baik dengan hasil pemain mendapatkan skor 4	Sesuai yang diharapkan
8	AMOGUS FIGHT	Untuk menguji apakah permainan AMOGUS FIGHT berjalan dengan baik	Menjalankan program, memasukkan perintah START, memasukkan AMOGUS FIGHT dengan perintah QUEUE GAME, memulai permainan dengan perintah PLAY GAME, Memasukan command HELP untuk melihat	Data test 6.8	Permainan berjalan dengan baik dengan hasil pemain mendapatkan skor 180	Sesuai yang diharapkan

			command list, BREW dengan diikuti 2 parameter ID, USE diikuti dengan sebuah ID, BUY kemudian memasukkan ID yang ingin dibeli, RESEP untuk melihat list resep untuk membuat potion, INFO untuk melihat efek setiap potion, SKIP untuk melewati 1 turn tanpa melakukan tindakan, STATUS untuk melihat status dari pemain dan musuh			
--	--	--	--	--	--	--

8 Pembagian Kerja dalam Kelompok

NIM	Nama	Pembagian Kerja
18221162	Ceavin Rufus De Prayer Purba	<ul style="list-style-type: none"> - Membuat game Amogus Fight (Game yang mengimplementasikan ADT tree) - Membuat ADT point, tree, dan list linear serta driver masing-masing ADT tersebut - Membuat laporan bagian algoritma menarik, tambahan spesifikasi bagian 2.1, data test bagian 6.8 dan test script nomor 8
18221118	Ardhan Nur Urfan	<ul style="list-style-type: none"> - Membuat function scoreboard dan history - Membuat ADT stackhistory, listscore, dan map serta driver masing-masing ADT tersebut
18221046	Vincent Winarta	<ul style="list-style-type: none"> - Melakukan testing pada program utama, game hangman, snake on meteor, dan tower of hanoi - Membuat MoM form asistensi II

		<ul style="list-style-type: none"> - Membuat laporan pada ringkasan, ADT, data testing, dan test script - Membantu finalisasi laporan akhir
18219048	Dwiky Hared Darmawan	<ul style="list-style-type: none"> - Membuat testscrip
18221084	Rei Arriel Clyfton	<ul style="list-style-type: none"> - Membuat game hangman - Membuat laporan pada bagian algoritma menarik
18221128	Fadhlan Nazhif Azizy	<ul style="list-style-type: none"> - Membuat game snake on meteor - Membuat data testing pada snake on meteor - Membuat ADT listdp (double pointer) serta driver

9 Lampiran

9.1 Deskripsi Tugas Besar 2

BNMO (dibaca: Binomo) adalah sebuah robot video game console yang dimiliki oleh Indra dan Doni. Dua bulan yang lalu, ia mengalami kerusakan dan telah berhasil diperbaiki. Sayangnya, setelah diperbaiki ia justru mendapatkan lebih banyak bug dalam sistemnya. Oleh karena itu, Indra dan Doni mencari programmer lain yang lebih handal untuk ulang memprogram robot video game console kesayangannya.

1. About the System

BNMO merupakan suatu robot game console yang dapat menjalankan permainan. BNMO memiliki beberapa fitur utama, yaitu:

1. Memainkan game
2. Menambahkan game
3. Menghapus game
4. Mengurutkan game yang akan dimainkan
5. Menampilkan game yang telah dimainkan
6. Menampilkan scoreboard game

2. Main Menu

Ketika program pertama kali dijalankan, BNMO akan memperlihatkan main menu yang berisi welcome page dan beberapa menu pilihan yaitu START dan LOAD. Setelah itu, main menu akan menerima input commands yang akan dijelaskan pada bagian berikutnya.

3. Command

Terdapat beberapa aturan umum command yang digunakan:

1. Semua command yang valid harus berupa **huruf kapital**. Mahasiswa dibebaskan apabila ingin melakukan handling terhadap huruf kecil.
2. Command yang tidak sesuai dengan ketentuan yang disebutkan pada bagian dibawah dianggap tidak valid. Handling command tidak valid dibebaskan kepada mahasiswa (boleh meminta ulang command yang sama atau meminta command baru)

Pada setiap giliran, pemain dapat memasukkan command-command berikut:

a. Command Dasar

Semua command yang terdapat pada

 Spesifikasi Tugas Besar 1 IF2111 2022/2023

b. SCOREBOARD

- Di setiap keadaan *game over* atau menang sebuah game, program meminta nama pemain.
- Nama pemain yang valid adalah **nama yang belum terpakai di scoreboard game yang sedang dimainkan**. Kemudian program menyimpan nama dan skor *game* tersebut di ADT Map.
- Perintah SCOREBOARD merupakan command yang digunakan untuk melihat nama dan skor untuk semua game.
- Urutan *scoreboard game* yang ditampilkan mengikuti **urutan pada command LIST GAME**.
- Urutan nama pada *scoreboard* diurutkan berdasarkan skor. Skor tertinggi berada di urutan pertama dan yang terendah berada di urutan terakhir. Jika ada skor yang sama, skor yang lebih dulu dimasukkan ke *scoreboard* ditampilkan duluan.

c. RESET SCOREBOARD

RESET SCOREBOARD merupakan command yang digunakan untuk melakukan reset terhadap scoreboard permainan. Reset dapat dilakukan untuk menghapus semua informasi pada setiap permainan maupun memilih salah satu permainan untuk di-*reset*.

d. HISTORY <n>

HISTORY <n> merupakan command yang digunakan untuk melihat permainan apa saja yang telah dimainkan dari data yang sudah ada dari file konfigurasi (**Jika LOAD**) dan dari mulai Start Game juga, dengan <n> adalah jumlah permainan yang telah dimainkan yang ingin ditampilkan. Urutan teratas merupakan

permainan terakhir yang dimainkan. Jika <n> lebih besar dari jumlah permainan yang telah dimainkan, akan menampilkan seluruh permainan yang telah dimainkan.

e. **RESET HISTORY**

RESET HISTORY merupakan command yang digunakan untuk menghapus semua history permainan yang dimainkan

Spesifikasi Game

1. RNG

Dijelaskan pada [Spesifikasi Tugas Besar 1 IF2111 2022/2023](#).

2. Diner Dash

Dijelaskan pada di [Spesifikasi Tugas Besar 1 IF2111 2022/2023](#).

3. Hangman

BNMO suka dengan *game* yang melibatkan tebak-tebakan kata sehingga ia membuat game yang terinspirasi dari game [Hangman](#). Berikut adalah spesifikasi game tersebut:

- Pada awal permainan program menentukan sebuah kata yang harus ditebak oleh pemain. **List kata dibebaskan kepada mahasiswa**. Boleh ditentukan di *code*. Kemudian, pemain diberikan 10 kesempatan untuk melakukan penebakan huruf yang tidak terdapat dalam kata.
- Pada setiap giliran, pemain menebak satu huruf yang terdapat pada kata tersebut. Apabila huruf tebakan terdapat dalam kata, maka huruf yang sudah tertebak akan ditampilkan pada layar. Apabila salah, maka pemain kehilangan satu kesempatan. Pemain tidak boleh menebak huruf yang sudah ditebak sebelumnya pada kata yang sama.
- Apabila pemain berhasil menebak suatu kata, maka pemain tersebut diberikan **poin sesuai dengan panjang kata yang berhasil ditebak**, kemudian program memberikan kata baru yang harus ditebak oleh pemain dengan jumlah kesempatan yang tersisa.
- Permainan akan berlanjut hingga pemain kehabisan kesempatan untuk menebak huruf yang salah.

4. Tower of Hanoi

BNMO melihat Finn dan Jake sedang bermain Tower of Hanoi secara langsung, dengan adanya 3 tiang, dapat disebutkan sebagai tiang A, tiang B, dan tiang C dan posisinya terurut dari kiri ke kanan. Pada tiang A, terdapat 5 piringan, dengan piringan **paling**

bawah merupakan piringan yang **paling besar** dan piringan **paling atas** merupakan piringan yang **paling kecil**. Cara bermainnya mudah, yaitu kelima piringan tersebut harus dipindahkan ke tiang C dengan posisi yang sama (piringan paling bawah merupakan piringan yang paling besar dan piringan paling atas merupakan piringan yang paling kecil), dengan peraturannya adalah piringan yang di bawah tidak boleh lebih kecil daripada piringan yang ada di atasnya. BNMO ingin membuat permainan ini dan agar lebih mengerti mengenai permainan ini, BNMO melihat panduannya di [The Enchiridion](#). Setelah itu, BNMO ingin melakukan modifikasi permainan ini:

- Jumlah piringan hanya 5 saja untuk permainan ini.
- Piringan direpresentasikan sebagai gambar piringan dengan *, dengan piringan paling besar adalah 9 dan balok silinder paling kecil adalah 1.
- Skor untuk permainan ini tergantung dengan seberapa optimal langkah dari pemain (dengan langkah paling optimalnya adalah 31) dan skor maksimalnya adalah 10 (Cara perhitungan skornya dibebaskan, yang terpenting adalah jika langkahnya 31, skornya adalah 10).

5. *Snake on Meteor*

BNMO memiliki sebuah *game* andalannya yang menjadi daya tarik bagi para pemainnya, yaitu *snake on meteor*. Singkatnya, game ini mirip dengan *game snake* yang ada pada berbagai konsol lama, tetapi dipersulit dengan adanya kehadiran meteor yang dapat mengenai *snake* tersebut. Dampak yang didapat oleh pemain apabila *snake* terkena serangan meteor tersebut adalah panjang *snake* akan berkurang sebanyak 1 unit.

Untuk pemahaman spek tubes, kosakata ini akan digunakan:

1. **Kepala:** Bagian pertama dari *snake* (*hint: head pada linked list*)
2. **Badan:** Bagian yang bukan pertama dan terakhir dari *snake* (*hint: bagian yang ditunjuk oleh head dan terus berkait hingga menunjuk pada tail linked list*)
3. **Ekor:** Bagian terakhir dari *snake* (*hint: tail pada linked list*)

Berikut ini merupakan spesifikasi yang lebih *detail* terkait game *snake on meteor*:

- **Dimensi Peta:** Dimensi peta adalah 5x5 unit dengan <0,0> merupakan sisi kiri atas dan <4,4> sisi kanan bawah. Sistem koordinat untuk penjelasan spek ini adalah:
- **Panjang snake:** Panjang awal dari *snake* adalah 3 unit. Kepala dari *snake* di-*random* pada sebuah titik dan 2 anggota badan yang berurut menurun dengan prioritas horizontal yang sama. Apabila badan menabrak dinding, maka akan

berurut menurun dengan prioritas vertical yang sama. **Maksud dari menurun adalah secara skalar (Cth: Dari angka 3 ke angka 2, dari angka 2 ke 1, dst)**

Makanan: Makanan akan diberikan secara random pada sebuah titik $\langle x,y \rangle$ (ditandai dengan huruf **o**). Lokasi munculnya makanan tidak mungkin berada di titik yang sama dengan komponen tubuh *snake*. Jika berhasil dimakan oleh *snake*, maka *snake* akan langsung bertambah panjang ekornya sebanyak 1 unit dan makanan baru akan di-random lagi pada sebuah titik $\langle x,y \rangle$

Proses pertambahan tail adalah sebagai berikut:

- a. **Secara umum**, ekor *snake* akan bertambah pada titik ordinat yang sama dengan tail, tetapi dengan titik koordinat satu sebelum tailnya (**Apabila tail berada pada titik $\langle x,y \rangle$, maka pertambahan tail akan dilakukan pada titik $\langle x-1,y \rangle$**).
- b. **Apabila kasus a tidak mungkin** (misalkan karena tail berada pada titik $\langle 0,1 \rangle$ dan tidak memungkinkan ada nilai titik $\langle -1,1 \rangle$), maka pertambahan akan dilakukan pada titik ordinat satu sebelum ekor *snake* sekarang, tetapi pada titik koordinat yang sama (**Apabila tail berada pada titik $\langle x,y \rangle$, maka pertambahan tail akan dilakukan pada titik $\langle x,y-1 \rangle$**).
- c. **Apabila kasus b tidak mungkin** (misalkan karena ekor berada pada titik $\langle 0,0 \rangle$ dan tidak memungkinkan adanya nilai $\langle -1,0 \rangle$ ataupun $\langle 0,-1 \rangle$), maka pertambahan akan dilakukan pada titik ordinat satu setelah ekor *snake* sekarang, tetapi pada titik koordinat yang sama (**Apabila tail berada pada titik $\langle x,y \rangle$, maka pertambahan tail akan dilakukan pada titik $\langle x,y+1 \rangle$**).
- d. **Apabila kasus a,b dan c tidak mungkin** (misalkan karena ekor berada pada titik $\langle 0,0 \rangle$ dan tidak memungkinkan adanya nilai $\langle -1,0 \rangle$ ataupun $\langle 0,-1 \rangle$ serta terdapat anggota tubuh pada titik $\langle 0,1 \rangle$), maka pertambahan akan dilakukan pada titik ordinat yang sama dengan ekor *snake* sekarang, tetapi pada titik koordinat yang bertambah satu (**Apabila tail berada pada titik $\langle x,y \rangle$, maka pertambahan tail akan dilakukan pada titik $\langle x+1,y \rangle$**).
- e. **Apabila kasus a,b,c,d tidak mungkin** (misalkan ekor berada pada titik $\langle 2,2 \rangle$ dan terdapat anggota tubuh di titik $\langle 1,2 \rangle, \langle 2,1 \rangle, \langle 2,3 \rangle$ dan $\langle 3,2 \rangle$) maka game akan berakhir

Cara bergerak: *Game* setiap putarannya akan meminta pemain untuk memasukkan huruf 'a', 'w', 's' atau 'd' untuk menggerakkan kepala *snake* (*game* akan meminta *re-input* apabila masukan selain huruf tersebut. Spesifikasi *lowercase* atau *uppercase* dibebaskan kepada kalian). Huruf 'a' untuk menggerakkan kepala ke kiri, 'w' untuk menggerakkan kepala ke atas, 's' untuk menggerakkan kepala ke bawah dan 'd' untuk menggerakkan kepala ke kanan. Setiap pergerakan akan menambahkan nilai koordinat/ordinat kepala *snake* (tergantung *input* yang diberikan)

sebanyak 1 unit. Posisi pergerakan anggota tubuh akan mengikuti posisi anggota tubuh sebelumnya. Kepala *snake* tidak mungkin bergerak ke anggota tubuhnya sendiri(game akan meminta input ulang apabila hal tersebut terjadi)

9.2 Notulen Rapat

Rapat 1	Catatan Rapat: Pembagian tugas: <ol style="list-style-type: none"> 1. Ceavin: Game implementasi adt tree 2. Rei: hangman 3. Ardhan, Vincent : scoreboard dan history 4. Alan : Snake on meteor Pembagian akan fleksibel, terutama untuk laporan. Masing-masing saling bantu jika ada kesulitan dan menginformasikan jika ada debugging.
Tanggal : 19 November 2022	
Tempat : LINE	
Kehadiran Anggota Kelompok: 18221162 Ceavin Rufus De Prayer Purba 18221084 Rei Arriel Clyfton 18221046 Vincent Winarta 18219048 Dwiky Hared Darmawan 18221118 Ardhan Nur Urfan 18221128 Fadhlan Nazhif Azizy	

9.3 Log Activity Anggota Kelompok

Keterangan lebih lanjut mengenai log activity saat membuat program dapat dilihat di [.Commit History](#)

Log Act	Anggota kelompok yang terlibat	Tanggal
Pembagian tugas	18219048 Dwiky Hared Darmawan 18221162 Ceavin Rufus De Prayer Purba 18221046 Vincent Winarta 18221084 Rei Arriel Clyfton 18221118 Ardhan Nur Urfan 18221128 Fadhlan Nazhif Azizy	19 November 2022
Membuat program scoreboard dan history	18221046 Vincent Winarta 18221118 Ardhan Nur Urfan	19 November 2022 - 24 November 2022
Mengubah program Tower of Hanoi (menyesuaikan dengan spesifikasi)	18221162 Ceavin Rufus De Prayer Purba	19 November 2022 - 20 November 2022
Membuat program games implementasi ADT Tree	18221162 Ceavin Rufus De Prayer Purba	20 November 2022 - 24 November 2022

Membuat program Hangman	18221084 Rei Arriel Clyfton 18221162 Ceavin Rufus De Prayer Purba	19 November 2022 - 22 November 2022
Membuat program Snake on Meteor	18221128 Fadhlan Nazhif Azizy 18221118 Ardhan Nur Urfan 18221162 Ceavin Rufus De Prayer Purba	20 November 2022 - 27 November 2022
Debugging & fixing	18221162 Ceavin Rufus De Prayer Purba 18221046 Vincent Winarta 18221118 Ardhan Nur Urfan 18221128 Fadhlan Nazhif Azizy	28 November 2022 - 2 Desember 2022
Finalisasi laporan	18221162 Ceavin Rufus De Prayer Purba 18221046 Vincent Winarta 18221084 Rei Arriel Clyfton 18219048 Dwiky Hared Darmawan	2 Desember 2022

9.4 *MoM Asistensi Tugas Besar*




MoM asistensi tugas besar terlampir pada bagian akhir laporan.





**Form Asistensi Tugas Besar
IF2110/Algoritma dan Struktur Data
Sem. 1 2022/2023**

No. Kelompok/Kelas : 13/K02
Nama Kelompok : 13INOMCUK
Anggota Kelompok (Nama/NIM) :
1. Dwiky Hared Darmawan/18219048
2. Ceavin Rufus De Prayer Purba / 18221162
3. Vincent Winarta / 18221046
4. Rei Arriel Clyfton / 18221084
5. Ardhan Nur Urfan / 1822118
6. Fadhlhan Nazhif Azizy / 18221128




Asisten Pembimbing : Aditya Bimawan /13519064




Asistensi I

Tanggal : 25 November 2022	Catatan Asistensi: Untuk spesifikasi bonus pada, dapat dibebaskan. Saat kepala kena badan, tidak kalah. Namun, bagaimana saat kondisi sudah tidak dapat bergerak kemana-mana, akan ditanya lagi kepada asisten yang lain dan dimasukkan ke dalam form FAQ. Sistem scoring dalam pada permainan bonus, diserahkan kepada kelompok. Disarankan untuk mengerjakan bonus meskipun sistem pembobotan belum fix. Jika tidak ada keterangan jenis list, maka menggunakan list biasa saja.
Tempat : Zoom Meeting	
Kehadiran Anggota Kelompok: 1 18221162 Ceavin Rufus De Prayer Purba 	
2 18221084 Rei Arriel Clyfton 	
3 18221046 Vincent Winarta 	
4 18219048 Dwiky Hared Darmawan	

 5 18221118 Ardhan Nur Urfan  6 18221128 Fadhlhan Nazhif Azizy 	
	Tanda Tangan Asisten: 

Asistensi II

Tanggal :	Catatan Asistensi: Comment atau output yang keluar dari game snake on meteor, tidak perlu 100% persis dengan spesifikasi. Saat kepala kena dinding, maka tail yang baru bisa tembus ke sisi lainnya Snake on meteor, tidak ada kondisi kalah saat tail tidak bisa nambah Tidak perlu di clear
Tempat : Zoom Meeting	
Kehadiran Anggota Kelompok:	
1 18221046 Vincent Winarta  2 18221162 Ceavin Rufus De Prayer Purba  3 18221084 Rei Arriel Clyfton 	

<p>4 18219048 Dwiky Hared Darmawan</p>  <p>5 18221118 Ardhan Nur Urfan</p>  <p>6 18221128 Fadhlan Nazhif Azizy</p> 	
	<p>Tanda Tangan Asisten:</p> 