

LAPORAN TUGAS BESAR 2

IF2111 Algoritma dan Struktur Data

<BNMO>

Dipersiapkan oleh:

Kelompok 12

Ghaylan Muhammad Fatih 18221042

Gracia Theophilia 18221078


Fikri Naufal Hamdi 18221096

Reinhart Wisely Lim 18221154

Esther Regina 18221086

Sekolah Teknik Elektro dan Informatika - Institut Teknologi Bandung

Jl. Ganesha 10, Bandung 40132

| | | | | |
|---|---|----------------------|----------|------------------------|
|  | Sekolah Teknik Elektro dan Informatika ITB | Nomor Dokumen | | Halaman |
| | | <i>IF2111-TB2-12</i> | | <i>40</i> |
| | | <i>Revisi</i> | <i>1</i> | <i>2 Desember 2022</i> |

Daftar Isi

| | |
|---|-----------|
| 1 Ringkasan | 3 |
| 2 Penjelasan Tambahan Spesifikasi Tugas | 4 |
| 2.1 Hangman | 5 |
| 2.2 Scoreboard | 5 |
| 3 Struktur Data (ADT) | 5 |
| 3.1 ADT Stack | 5 |
| 3.2 ADT StackInt | 6 |
| 3.3 ADT Map | 7 |
| 3.4 ADT Array of Map | 8 |
| 3.5 ADT Point | 9 |
| 3.6 ADT Linked List | 10 |
| 4 Program Utama | 12 |
| 5 Algoritma-Algoritma Menarik | 13 |
| 5.1 Random String dalam Permainan Hangman | 13 |
| 5.2 Linked List Bertipe Point dalam Permainan Snake on Meteor | 14 |
| 6 Data Test | 15 |
| 6.1 Data Test SCOREBOARD | 15 |
| 6.2 Data Test RESET SCOREBOARD | 16 |
| 6.3 Data Test HISTORY <n> | 18 |
| 6.4 Data Test RESET HISTORY | 19 |
| 6.5 Data Test HANGMAN | 19 |
| 6.6 Data Test TOWER OF HANOI | 23 |
| 6.7 Data Test SNAKE ON METEOR | 24 |
| 7 Test Script | 31 |
| 8 Pembagian Kerja dalam Kelompok | 34 |
| 9 Lampiran | 34 |
| 9.1 Deskripsi Tugas Besar 2 | 34 |
| 9.2 Notulen Rapat | 34 |
| 9.3 Log Activity Anggota Kelompok | 35 |
| 9.4 Lain-Lain | 36 |

| | |
|------------------------|----|
| 9.4.1 Form Asistensi 1 | 36 |
| 9.4.2 Form Asistensi 2 | 38 |

1 Ringkasan

BNMO adalah sebuah *robot video game console* berbasis CLI (*command-line interface*) yang dapat menjalankan permainan. BNMO memiliki beberapa fitur utama, yaitu memainkan game, menambahkan game, menghapus game, mengurutkan game yang akan dimainkan, menampilkan game yang akan dimainkan, dan menampilkan scoreboard game. Pada setiap kali permainan, pemain dapat memasukkan berbagai command. Selain command dasar (START, LOAD <filename>, SAVE <filename>, CREATEGAME, LISTGAME, DELETEGAME, QUEUEGAME, PLAYGAME, SKIPGAME <n>, QUIT, dan HELP), pemain juga dapat memasukkan command SCOREBOARD, RESET SCOREBOARD, HISTORY <n>, dan RESET HISTORY. Program tidak akan mengenali command jika pemain memasukkan command selain beberapa command yang telah disebutkan dan program akan meminta pemain memasukkan command yang valid.

Permainan BNMO dibuat dengan bahasa C dengan memanfaatkan struktur data yang sudah dipelajari dalam mata kuliah Algoritma dan Struktur Data. Library yang boleh digunakan pada tugas besar kali ini adalah stdio.h, stdlib.h, time.h, dan math.h. Berdasarkan spesifikasi permainan yang ada, berikut beberapa ADT yang dapat digunakan:

1. ADT Array
2. ADT Mesin Karakter
3. ADT Mesin Kata
4. ADT Queue
5. ADT Stack
6. ADT Set & Map
7. ADT Linked List

Penjelasan mengenai ADT yang digunakan dan modifikasi yang kami lakukan pada ADT dalam Tugas Besar 2 ini akan dijelaskan pada Bagian 3 Struktur Data pada laporan ini.

Program ini terdiri dari beberapa game, yaitu RNG, Diner Dash, dan tiga game tambahan, yaitu Hangman, Tower of Hanoi, dan Snake on Meteor. Hangman adalah permainan yang mengharuskan pemain untuk menebak kata asal. Kemudian, Tower of Hanoi merupakan permainan yang mengharuskan pemain untuk memindahkan piringan ke tiang C dengan posisi piringan paling besar di paling bawah dan posisi piringan paling kecil di paling atas. Selanjutnya, Snake on Meteor adalah permainan yang mirip dengan *game snake* dalam berbagai konsol lama. Namun, dalam permainan ini, akan ada meteor yang dapat mengenai *snake* tersebut dan jika *snake* terkena serangan meteor, panjang *snake* akan berkurang sebanyak 1 unit.

Secara umum, laporan ini berisi ringkasan mengenai program yang kami buat, penjelasan tambahan mengenai spesifikasi tugas, struktur data yang digunakan dalam membuat permainan BNMO, program utama, algoritma-algoritma menarik dalam program, data test, test script, pembagian kerja dalam kelompok, dan lampiran-lampiran yang berisi notulen rapat, log activity anggota kelompok, dan sebagainya.

2 Penjelasan Tambahan Spesifikasi Tugas

Dalam membuat dan menjalankan program, terdapat beberapa fitur atau command yang memerlukan spesifikasi tambahan agar tidak terjadi miskonsepsi terhadap program tersebut. Berikut penjelasan tambahan terkait beberapa spesifikasi fitur maupun command.

2.1 Hangman

Pada permainan Hangman, pemain diasumsikan tidak dapat hanya melakukan penambahan kata ke dalam list tebakan karena pemain juga harus melanjutkannya dengan bermain permainan hangman. Kemudian, jika pemain menebak huruf yang sama, akan keluar output “Kamu sudah pernah menebak huruf ini!”. Jika tebakan huruf yang berulang tersebut tidak ada pada kata yang diminta untuk ditebak, kesempatan pemain dalam menebak tetap akan dikurangi 1. Namun, jika tebakan huruf yang berulang tersebut ada pada kata yang diminta untuk ditebak, kesempatan pemain dalam menebak tidak akan dikurangi. Pada bagian “Tebakan sebelumnya”, program akan menuliskannya sesuai dengan input huruf tebakan pengguna (kapital atau tidaknya). Selain itu, file untuk menyimpan daftar kata tebakan bernama “daftarkata.txt”.

2.2 Scoreboard

Pada prosedur CREATEGAME dan SCOREBOARD, jika ingin menambahkan game, pemain dapat memasukkan game tersebut secara uppercase maupun lowercase dan keduanya akan dihitung sama. Ketika game over, pemain akan diminta nama player. Nama player yang valid adalah nama player yang hanya terdiri dari satu kata saja.

3 Struktur Data (ADT)

Pada program ini, kami menggunakan 3 buah ADT utama dan 3 ADT tambahan. Berikut daftar dari beberapa ADT yang kami gunakan dalam program ini.

3.1 ADT Stack

ADT Stack terdiri dari struktur Stack yang terdiri atas array of char dan TOP (integer). Selain itu, ADT Stack mendefinisikan UNDEF (indeks Stack dengan elemen kosong) adalah -1 dan MaxStackEl (maksimum elemen dalam stack) adalah 100. Konstruktor dari ADT Stack adalah CreateEmptyStack. Selain itu, terdapat juga dua predikat untuk *test* keadaan koleksi, yaitu IsStackEmpty dan IsStackFull. ADT Stack juga terdiri dari dua operasi, yaitu Push (operasi penambahan elemen ke Stack) dan Pop (operasi penghapusan elemen dari Stack). Tidak hanya itu, terdapat juga prosedur ReverseStack untuk membalikkan urutan Stack.

ADT Stack digunakan pada prosedur HISTORY dan RESETHISTORY. Pada prosedur HISTORY dan RESETHISTORY, ADT Stack digunakan untuk merepresentasikan urutan dari permainan yang telah dimainkan. TOP pada Stack merepresentasikan permainan yang terakhir dimainkan.

Sketsa pada ADT Stack terdiri dari:

- void CreateEmptyStack(Stack *S)

Digunakan untuk membuat sebuah Stack kosong berkapasitas MaxStackEl. Pada program yang kami buat, prosedur CreateEmptyStack digunakan dalam prosedur RESETHISTORY untuk membuat daftar kosong permainan yang telah dimainkan agar HISTORY permainan dapat di-*reset*.

- boolean IsStackEmpty(Stack S)

Digunakan untuk mengetahui apakah sebuah Stack kosong. Pada program yang kami buat, fungsi IsStackEmpty digunakan untuk memeriksa apakah daftar permainan yang telah dimainkan kosong atau tidak.

- **boolean IsStackFull(Stack S)**
Digunakan untuk mengetahui apakah sebuah Stack sudah penuh (mencapai kapasitas maksimal, yaitu 100).
- **void Push(Stack * S, stackElmt X)**
Digunakan untuk menambahkan sebuah elemen ke Stack. Pada program yang kami buat, prosedur Push digunakan dalam prosedur LOADBNMO, DELETEGAME, dan SKIPGAME untuk menambahkan suatu elemen ke daftar permainan yang telah dimainkan (GamesHistory).
- **void Pop(Stack * S, stackElmt* X)**
Digunakan untuk menghapus sebuah elemen dari Stack. Pada program yang kami buat, prosedur Pop digunakan dalam prosedur SAVEBNMO, DELETEGAME, dan HISTORY untuk menghapus elemen TOP dari daftar permainan yang telah dimainkan (GamesHistory).
- **void ReverseStack(Stack *S)**
Digunakan untuk membalikkan urutan Stack. Pada program yang kami buat, prosedur ReverseStack digunakan dalam prosedur LOADBNMO dan DELETEGAME untuk membalikkan urutan daftar permainan yang telah dimainkan (GamesHistory) dan menghapus permainan dari GamesHistory.

3.2 ADT StackInt

Secara singkat, ADT StackInt merupakan ADT Stack yang dimodifikasi menjadi array of integer. ADT StackInt terdiri dari struktur StackInt yang terdiri atas array of integer dan TOP (integer). Selain itu, ADT StackInt mendefinisikan NilSInt (indeks Stack dengan elemen kosong) adalah -1 dan MaxElSInt (maksimum elemen dalam Stack) adalah 100. Konstruktor dari ADT StackInt adalah CreateEmpty_SInt. Selain itu, terdapat juga tiga predikat untuk *test* keadaan koleksi, yaitu IsEmpty_SInt, IsFull_SInt, dan lengthSInt. ADT Stack juga terdiri dari dua operasi, yaitu Push_SInt (operasi penambahan elemen ke Stack) dan Pop_SInt (operasi penghapusan elemen dari Stack).

ADT StackInt digunakan dalam permainan Tower of Hanoi. Pada permainan Tower of Hanoi, ADT StackInt digunakan untuk merepresentasikan urutan tumpukan tiang A, B, dan C.

Sketsa pada ADT StackInt terdiri dari:

- **void CreateEmpty_SInt(StackInt *S)**
Digunakan untuk membuat sebuah Stack kosong berkapasitas MaxElSInt. Pada program yang kami buat, prosedur CreateEmpty_SInt digunakan untuk membuat tumpukan tiang kosong, baik tiang A, B, maupun C.
- **boolean IsEmpty_SInt(StackInt S)**
Digunakan untuk mengetahui apakah sebuah StackInt kosong. Pada program yang kami buat, fungsi IsEmpty_SInt digunakan untuk memeriksa apakah tumpukan tiang kosong atau tidak.
- **boolean IsFull_SInt(StackInt S)**
Digunakan untuk mengetahui apakah sebuah StackInt sudah penuh (mencapai kapasitas maksimal, yaitu 100).
- **void Push_SInt(StackInt * S, int X)**

Digunakan untuk menambahkan sebuah elemen ke StackInt. Pada program yang kami buat, prosedur Push_SInt digunakan untuk menambahkan suatu elemen ke tumpukan tiang A, B, maupun C.

- `void Pop_SInt(StackInt * S, int* X)`

Digunakan untuk menghapus sebuah elemen dari StackInt. Pada program yang kami buat, prosedur Pop_SInt digunakan untuk mengurangi suatu elemen dari tumpukan tiang A, B, maupun C.

- `int lengthSInt(StackInt S)`

Digunakan untuk mengirimkan ukuran StackInt. Pada program yang kami buat, fungsi lengthSInt digunakan sebagai kondisi berhenti dalam suatu loop.

3.3 ADT Map

ADT Map terdiri dari struktur Map yang terdiri atas array (Elements[MaxEl] yang terdiri atas array of char berupa Key dan array of int berupa Value), MapName (string), dan Count (integer). ADT Map mendefinisikan Nil adalah 0, MaxEl adalah 20, dan Undefined adalah -999. Konstruktor dari ADT Map adalah CreateScoreboard. Selain itu, terdapat juga dua predikat untuk keadaan koleksi, yaitu IsEmptySB dan IsFullSB. ADT Map juga terdiri dari beberapa operasi dasar, yaitu Value, InsertSB, DeleteSB, IsMemberSB, InsertSortedDesc, PrintScoreBoard, MaxNameLength, dan SortSB.

ADT Map digunakan pada prosedur SCOREBOARD dan RESETSCOREBOARD. Secara umum, ADT Map digunakan untuk menyimpan skor setiap nama pemain. Key dalam Map berperan dalam mencatat nama pemain, sedangkan Value berperan dalam mencatat jumlah skor tiap pemain.

Sketsa pada ADT Map terdiri dari:

- `void CreateScoreBoard(Map *M)`

Digunakan untuk membuat sebuah Map kosong berkapasitas MaxEl. Pada program yang kami buat, prosedur CreateScoreBoard digunakan dalam prosedur STARTBNMO, LOADBNMO, CREATEGAME, SCOREBOARD, dan RESETSCOREBOARD untuk membuat scoreboard permainan.

- `boolean IsEmptySB(Map M)`

Digunakan untuk mengetahui apakah sebuah Map kosong. Pada program yang kami buat, fungsi IsEmptySB digunakan untuk memeriksa apakah scoreboard permainan kosong atau tidak.

- `boolean IsFullSB(Map M)`

Digunakan untuk mengetahui apakah sebuah Map sudah penuh (mencapai kapasitas maksimal, yaitu 20).

- `valuetype Value(Map M, keytype k)`

Digunakan untuk mengembalikan nilai value dengan key dari Map dan jika tidak ada key pada M, program akan mengembalikan Undefined.

- `void InsertSB(Map *M, keytype k, valuetype v)`

Digunakan untuk menambahkan suatu elemen sebagai elemen Map. Pada program yang kami buat, prosedur InsertSB digunakan dalam prosedur LOADBNMO untuk menambahkan nama pemain yang baru bermain *game* dan jumlah skornya.

- `void DeleteSB(Map *M, keytype k)`

- Digunakan untuk menghapus suatu elemen dari Map.
- **boolean IsMemberSB(Map M, keytype k)**
Digunakan untuk mengetahui apakah suatu Key ada dalam suatu Map. Jika iya, program akan mengembalikan nilai *true*.
- **void InsertSortedDesc(Map *M, keytype k , valuetype v)**
Digunakan untuk memasukkan suatu Key dan Value dalam suatu Map secara berurut berdasarkan Value. Pada program yang kami buat, prosedur InsertSortedDesc digunakan untuk memasukkan scoreboard per nama pemain beserta skornya yang diurutkan berdasarkan skornya (dari skor terbesar hingga skor terkecil).
- **void PrintScoreBoard(Map M)**
Digunakan untuk melakukan *print* atau menampilkan Map. Pada program yang kami buat, prosedur PrintScoreBoard digunakan dalam prosedur SCOREBOARD untuk menampilkan scoreboard.
- **int MaxNameLength(Map M)**
Digunakan untuk mengirimkan panjang maksimal key dari suatu Map. Pada program yang kami buat, fungsi MaxNameLength digunakan untuk mengirimkan panjang maksimal nama pemain yang bermain BNMO.
- **Map SortSB(Map M)**
Digunakan untuk mengirim map yang telah di-*sort* berdasarkan Value elemennya.

3.4 ADT Array of Map

Secara singkat, ADT Array of Map merupakan ADT Array yang dimodifikasi menjadi array yang bertipe map. ADT Array of Map terdiri dari struktur ArrMap yang terdiri atas array of map dan Neff (integer). ADT Array of Map mendefinisikan IdxMax adalah 100, IdxMin adalah 0, dan IdxUndef adalah -1. Konstruktor dari ADT Array of Map adalah MakeEmptyArrMap. Selain itu, terdapat juga beberapa selektor seperti NbElmtArrMap, MaxNbElArrMap, GetFirstIdxArrMap, GetLastIdxArrMap, GetElmtArrMap, dan SetElArrMap. Pada ADT Array of Map, terdapat juga operator untuk melakukan *test* penuh atau kosong, yaitu IsEmptyArrMap dan IsFullArrMap.

ADT Array of Map digunakan pada prosedur SCOREBOARD dan RESETSCOREBOARD. Pada prosedur SCOREBOARD dan RESETSCOREBOARD, ADT Array of Map digunakan sebagai representasi kumpulan scoreboard dari semua permainan. Kumpulan scoreboard tersebut terdiri atas scoreboard dari beberapa permainan yang berisi nama pemain dan skor tiap pemain, sedangkan Neff merepresentasikan jumlah scoreboard permainan yang digunakan.

Sketsa pada ADT Array of Map terdiri dari:

- **void MakeEmptyArrMap(ArrMap *A)**
Digunakan untuk membuat sebuah Array of Map kosong dengan Neff bernilai 0.
- **int NbElmtArrMap(ArrMap A)**
Digunakan untuk mengirimkan banyaknya elemen dalam Array of Map. Pada program yang kami buat, fungsi NbElmtArrMap digunakan dalam prosedur SAVEBNMO, CREATEGAME, SCOREBOARD, dan RESETSCOREBOARD untuk mengirimkan banyaknya kumpulan scoreboard permainan sekaligus sebagai kondisi berhenti dalam suatu loop.

- `int MaxNbElArrMap(ArrMap A)`
Digunakan untuk mengirimkan maksimal elemen dalam Array of Map.
- `IdxType GetFirstIdxArrMap(ArrMap A)`
Digunakan untuk mengirimkan indeks pertama dari Array of Map.
- `IdxType GetLastIdxArrMap(ArrMap A)`
Digunakan untuk mengirimkan indeks terakhir dari Array of Map.
- `ArrType GetElmtArrMap(ArrMap A, IdxType i)`
Digunakan untuk mengirimkan elemen berindeks *i* dari Array of Map. Pada program yang kami buat, fungsi `GetElmtArrMap` digunakan dalam prosedur `SAVEBNMO`, `SCOREBOARD`, dan `RESETSCOREBOARD` untuk mengirimkan scoreboard permainan berindeks *i* dari daftar kumpulan scoreboard permainan.
- `void SetElArrMap(ArrMap *A, IdxType i, ArrType SB)`
Digunakan untuk menjadikan Map sebagai elemen di indeks ke-*i* dari Array of Map. Pada program yang kami buat, fungsi `SetElArrMap` digunakan dalam prosedur `STARTBNMO`, `LOADBNMO`, `CREATEGAME`, dan `RESETSCOREBOARD` untuk menjadikan scoreboard suatu permainan sebagai elemen di indeks tertentu dari daftar kumpulan scoreboard.
- `boolean IsEmptyArrMap(ArrMap A)`
Digunakan untuk mengetahui apakah sebuah Array of Map kosong. Pada program yang kami buat, fungsi `IsEmptyArrMap` digunakan dalam prosedur `LOADBNMO` untuk memeriksa apakah daftar kumpulan scoreboard permainan kosong atau tidak.
- `boolean IsFullArrMap(ArrMap A)`
Digunakan untuk mengetahui apakah sebuah Array of Map sudah penuh (mencapai kapasitas maksimal).

3.5 ADT Point

ADT Point terdiri dari struktur Point yang terdiri atas *X* (integer) dan *Y* (integer). ADT Point mendefinisikan `POINT_UNDEF` sebagai -1. Konstruktor dari ADT Point adalah `CreatePoint`. Selain itu, terdapat prosedur yang merupakan kelompok interaksi dengan I/O *device*, yaitu `PrintPoint`. Operasi-operasi lainnya dari ADT Point berupa `NextX`, `NextY`, `PlusDelta`, `shiftPoint`, dan `EQ`.

ADT Point digunakan pada ADT Linked List sekaligus permainan Snake on Meteor. Pada ADT Linked List, ADT Point digunakan sebagai tipe dari position dalam `ElmtList`. Pada permainan Snake on Meteor, ADT Point digunakan untuk merepresentasikan posisi dari kepala ular, posisi dari ekor ular, posisi dari makanan, posisi dari meteor, dan posisi dari obstacle.

Sketsa pada ADT Point terdiri dari:

- `void CreatePoint (Point * P, int X, int Y)`
Digunakan untuk membentuk sebuah point dari komponen-komponennya, yaitu *X* dan *Y*. Dalam program yang kami buat, `CreatePoint` digunakan untuk me-*random* titik *X* dan *Y* yang nantinya akan menjadi head Snake/meteor/obstacle/makanan.
- `void PrintPoint (Point P)`
Digunakan untuk menulis nilai *P* ke layar dengan format “(*X*, *Y*)”.
- `Point NextX (Point P)`
Digunakan untuk mengirim salinan *P* dengan absis ditambah satu.

- **Point NextY (Point P)**
Digunakan untuk mengirim salinan P dengan ordinat ditambah satu.
- **Point PlusDelta (Point P, int deltaX, int deltaY)**
Digunakan untuk mengirim salinan P yang absisnya adalah Absis(P)+deltaX dan ordinatnya adalah Ordinat(P)+deltaY
- **void shiftPoint (Point *P, int deltaX, int deltaY)**
Digunakan untuk menggeser absis sebanyak deltaX dan menggeser ordinat sebanyak deltaY.
- **boolean EQ (Point P1, Point P2)**
Digunakan untuk mengetahui apakah kedua point sama ($P1=P2$) atau tidak ($P1 \neq P2$). Pada program snake on meteor, fungsi EQ digunakan untuk mengecek apakah Food telah dimakan oleh snake yang ditandai dengan posisi first dari Snake sama dengan posisi first dari Food.

3.6 ADT Linked List

ADT Linked List terdiri dari struktur List yang terdiri atas First (alamat *node*), Last (alamat *node*), dan Length (integer). ADT Linked List terdiri dari struktur tElmtList yang terdiri atas Point Position, idxtype info, addressN next, dan addressN prev dan struktur List yang terdiri atas addressN first, addressN Last, dan int length. ADT Linked List memiliki beberapa primitif yaitu IsEmpty_LDP, CreateEmpty_LDP, Alokasi_LDP, Dealokasi_LDP, Search_LDP, SearchPos_LDP, InsVFirst_LDP, InsVLast_LDP, DelVFirst_LDP, DelVLast_LDP, InsertFirst_LDP, InsertLast_LDP, InsertAfter_LDP, DelFirst_LDP, DelLast_LDP, DelP_LDP, DelAfter_LDP, DelBefore_LDP, PrintForward_LDP, PrintBackward_LDP.

ADT Linked List digunakan bersamaan dengan ADT Point dalam game Snake On Meteor yang mana ADT ini digunakan untuk pergerakan snake, bertambah panjangnya snake saat berhasil memakan makanan, penghapusan bagian badan snake jika terkena meteor.

Sketsa dari ADT Linked List terdiri dari :

- **boolean IsEmpty_LDP (List L)**
Digunakan untuk mengecek apakah suatu linked list kosong/tidak
- **void CreateEmpty_LDP (List *L)**
Digunakan untuk membuat linked list yang kosong. Pada program yang kami buat, prosedur CreateEmpty_LDP digunakan untuk membuat List Snake, List Meteor, List Food, dan List Obstacle menjadi kosong pada awal program. Lalu, CreateEmpty_LDP digunakan juga untuk mengosongkan List Food jika makanan sudah berhasil dimakan dan digunakan juga untuk mengosongkan List Meteor setiap kali turn.
- **addressN Alokasi_LDP (idxtype X, Point Pos)**
Digunakan untuk mengirimkan addressN hasil alokasi sebuah elemen. Alokasi_LDP digunakan pada implementasi InsVFirst_LDP dan InsVLast_LDP untuk mengalokasi infotype X dan Point Pos.
- **addressN Dealokasi_LDP (addressN P)**
Digunakan untuk melakukan pengembalian addressN. Dealokasi_LDP digunakan pada implementasi DelVFirst_LDP dan DelVLast_LDP untuk mengembalikan address yang ingin dihapus.
- **addressN search_LDP (List L, idxtype X)**

- Digunakan untuk mencari apakah ada elemen list dengan Info(P) yang bernilai X.
- **addressN searchPos_LDP (List L, Point Pos)**
Digunakan untuk mencari apakah ada elemen list dengan posisi Point Pos. Pada program snake on meteor, searchPos_LDP digunakan dalam prosedur PrintMap yang digunakan untuk mencari titik pada list snake, food, meteor, dan obstacle, lalu menampilkannya pada petak. Selain itu, searchPos_LDP digunakan pada fungsi isMeteorHitSnake untuk mengecek apakah terdapat meteor pada list snake.
 - **void InsVFirst_LDP (List *L, idxtype X, Point Pos)**
Digunakan untuk menambahkan elemen list di awal.
 - **void InsVLast_LDP (List *L, idxtype X, Point Pos)**
Digunakan untuk menambahkan elemen list di akhir. Pada program Snake on Meteor, InsVLast_LDP digunakan untuk menambahkan elemen snake jika berhasil memakan makanan, menambahkan elemen pada list food, dan menambahkan elemen pada list meteor.\
 - **void DelVFirst_LDP (List *L, idxtype *X)**
Digunakan untuk menghapus elemen pertama pada list
 - **void DelVLast_LDP (List *L, Point *Pos)**
Digunakan untuk menghapus elemen terakhir pada list.
 - **void InsertFirst_LDP (List *L, addressN P)**
Digunakan pada implementasi InsVFirst_LDP
 - **void InsertLast_LDP (List *L, addressN P)**
Digunakan pada implementasi InsVLast_LDP
 - **void InsertAfter_LDP(List *L, AddressN P, addressN Prec)**
Digunakan untuk menambah elemen sesudah elemen beralamat Prec
 - **void InsertBefore_LDP (List *L, AddressN P, addressN Succ)**
Digunakan untuk menambahkan elemen sebelum elemen beralamat Succ
 - **void DelFirst_LDP (List *L, AddressN *P)**
Digunakan untuk menghapus elemen pertama dan menggantikannya dengan suksesor elemen pertama yang lama
 - **void DelLast_LDP (List *L, AddressN *P)**
Digunakan untuk menghapus elemen terakhir dan menjadikan elemen sebelum terakhirnya sebagai elemen terakhir
 - **void DelP_LDP (List *L, idxtype X, Point Pos)**
Digunakan untuk menghapus elemen pada titik tertentu. Pada program Snake On Meteor, DelP_LDP digunakan untuk menghapus bagian tubuh snake jika terkena meteor.
 - **void DelAfter_LDP (List *L, addressN *Pdel, addressN Prec)**
Digunakan untuk menghapus elemen sesudah Prec. DelAfter_LDP digunakan pada implementasi DelP_LDP.
 - **void DelBefore_LDP (List *L, addressN *Pdel, addressN Succ)**
Digunakan untuk menghapus elemen sebelum Succ.
 - **void PrintForward_LDP (List L)**
Digunakan untuk menampilkan elemen dari elemen pertama hingga terakhir
 - **void PrintBackward_LDP (List L)**
Digunakan untuk menampilkan elemen dari elemen terakhir ke elemen pertama.

4 Program Utama

Pada `console.c` terdapat prosedur untuk menampilkan scoreboard, melakukan reset terhadap scoreboard, menampilkan history, dan melakukan reset history. Scoreboard dibuat dengan memanfaatkan ADT array of map dan map. Program penampilan scoreboard akan memasuki loop. Di dalam loop tersebut, akan diciptakan sebuah map yang kosong. Program akan mengakses elemen array map dari 0 hingga `nbMap-1`. Lalu elemen array akan ditampilkan namun akan diurutkan terlebih dahulu. Reset scoreboard dibuat dengan memanfaatkan ADT Array of Map. Program akan menampilkan semua daftar game dan pemain akan diminta nomor scoreboard yang ingin dihapus. Jika input yang dimasukkan user adalah 0, program akan meminta validasi dari pemain apakah akan melakukan reset semua scoreboard. Jika Ya, program akan melakukan reset terhadap scoreboard. Namun, jika tidak, program akan menampilkan gagal direset. Selanjutnya, jika pemain memberikan input yang tidak valid, program akan meminta input yang valid dari pengguna. Jika pemain memasukkan nomor scoreboard game tertentu yang ingin direset, maka program hanya akan mereset scoreboard untuk game tersebut. Scoreboard dan reset scoreboard akan dipanggil di program utama saat pemain memasukkan command “SCOREBOARD” atau “RESET SCOREBOARD”.

Selanjutnya, history dibuat dengan memanfaatkan ADT Stack. Jika stack GamesHistory kosong, maka program akan menampilkan bahwa belum ada game yang dimainkan. Jika tidak kosong, program akan menampilkan GamesHistory sebanyak inputan pengguna. Reset History akan membuat stack yang bernama GamesHistory menjadi kosong jika inputan pemain “Ya”. History dan reset history akan dipanggil di program utama saat pemain memasukkan command “RESET HISTORY” atau “HISTORY <n>”.

Program Hangman dibuat pada folder Games yang di dalamnya terdapat folder Hangman. Program ini meng-include `stdio.h`, `stdlib.h`, dan `Hangman.h`. Pada saat program dijalankan, program utama akan menampilkan menu hangman yang mengandung dua pilihan, yaitu langsung bermain Hangman dan menambahkan kata ke dalam list tebakan. Jika pemain memberikan input “1”, permainan Hangman akan langsung berjalan. Jika pemain memberikan input “2”, program akan meminta pemain untuk menuliskan kata yang ingin ditambahkan dan program akan menuliskannya dalam file `daftarkata.txt`. Kemudian, program akan kembali menampilkan menu hangman. Program akan selalu melakukan looping jika pemain memberikan input “2”. Jika pemain memasukkan input di luar itu, input dianggap tidak valid sehingga pemain akan diminta untuk memasukkan input yang valid dan menu hangman kembali ditampilkan. Dalam permainan hangman, program akan membaca file `daftarkata.txt` yang berisi daftar kata tebakan. Kemudian, program akan menentukan suatu angka acak sebagai *indeks* string atau kata acak yang akan terpilih. Setelah menentukan angka acak, program akan menyimpan string acak yang indeksinya sesuai dengan angka acak yang terpilih. Kemudian, program akan melakukan looping untuk meminta input berupa tebakan huruf pemain selama kesempatan pemain masih ada. Jika pemain berhasil menebak satu kata dan kesempatannya masih ada, program akan kembali menentukan suatu kata acak dan kembali melakukan looping input tebakan pemain. Program akan menyimpan skor pemain yang dihitung berdasarkan panjang kata yang berhasil ditebak. Kemudian, program akan berhenti jika kesempatan pemain habis.

Tower of Hanoi dibuat pada folder Games yang di dalamnya terdapat folder Tower_of_Hanoi. Program meng-include stdio.h, stdlib.h, dan ToH.h. Pada saat program dijalankan, program utama akan masuk ke loop. Program akan menampilkan tiang dan akan meminta command tiang asal dan tiang tujuan dari pemain. Masukan pemain akan divalidasi. Input yang valid adalah piringan yang hendak dipindahkan dari tiang asal harus lebih kecil dari piringan paling atas pada tiang tujuan. Jika input tidak valid, pemain akan diminta untuk memasukkan command yang valid. Program akan terhenti/game over jika semua piringan berhasil dipindahkan ke tiang C. Jika seluruh piringan sudah dipindah ke tiang C, maka skor pemain akan ditampilkan dengan rumus $310/\text{steps}$ dan pemain akan diminta nama player untuk dimasukkan ke scoreboard.

Program Snake on Meteor dibuat pada folder Games yang di dalamnya terdapat folder Snake_on_Meteor. Program meng-include stdio.h, stdlib.h, time.h, dan SoM.h. Pada saat program dijalankan, program utama akan membuat list Snake, list Meteor, list Obstacle, dan list Food yang kosong. Program akan me-random kepala snake, posisi obstacle, dan posisi makanan. Setelah itu, pada putaran kedua dan seterusnya program akan merandom meteor. Sepanjang permainan tidak game over, program akan secara terus menerus me-random Meteor. Food juga akan dirandom ulang jika ular berhasil memakan makanan. Program akan memasuki loop pertama selama tidak Game over. Lalu, program akan menampilkan petak permainan Snake on Meteor. Di dalam loop tersebut, terdapat pengecekan kondisi apakah makanan sudah dimakan, pengecekan posisi head dari snake dan first dari meteor, pengecekan posisi head dari snake dan first dari Obstacle, dan pengecekan apakah terdapat meteor pada badan snake. Permainan akan selesai jika meteor/obstacle mengenai Head dari Snake, snake tidak dapat bergerak lagi karena titik sekitarnya sudah terhalangi oleh badannya, dan tail tidak dapat ditambah. Jika permainan sudah game over, maka skor pemain akan ditampilkan dengan rumus panjang badan snake dikalikan dua dan pemain akan diminta nama player untuk dimasukkan ke scoreboard.

5 Algoritma-Algoritma Menarik

Dalam pengerjaan tugas besar 2 ini, kami menemukan beberapa algoritma menarik yang telah kami buat. Berikut penjelasan mengenai berbagai algoritma menarik yang kami temukan.

5.1 Random String dalam Permainan Hangman

Dalam pembuatan Hangman, terdapat fungsi randomstr yang akan melakukan *return* terhadap suatu string yang tercatat dalam file yang berisi daftar kata. Dalam pembuatan fungsi randomstr, digunakan library time.h yang merupakan file header yang didefinisikan dalam C Standard Library. Fungsi rand akan digunakan dalam implementasi dari fungsi randomstr, tetapi fungsi rand akan selalu menghasilkan angka acak yang sama. Oleh karena itu, perlu digunakan fungsi srand untuk memberikan *seed* pada fungsi rand. Selain itu, perlu juga digunakan fungsi time untuk melakukan *seed* pada randomizer sehingga dapat menghasilkan angka acak yang berbeda.

Hal yang menarik dari algoritma ini adalah program dapat mengirimkan suatu string acak, tidak hanya sebuah nomor acak seperti pada permainan RNG. Pada permainan HANGMAN, *return* terhadap suatu string yang acak dilakukan dengan melakukan pengacakan nomor yang merepresentasikan indeks string dalam file daftar kata. Oleh karena itu, sebelum dilakukan pengacakan nomor, program harus membaca file daftar kata terlebih dahulu untuk

menentukan banyaknya kata yang dapat diacak. Setelah itu, program akan melakukan pengacakan nomor indeks kata tersebut dengan skala 0 hingga banyaknya kata-1. Setelah didapatkan nomor indeks acak, program akan mengirimkan string atau kata dari file daftar kata sesuai dengan indeks acak yang didapatkan tersebut.

Berikut algoritma fungsi randomstr tersebut.

```
char* randomstr(char* filename, int maxLength, int maxKata, char
kata[maxKata][maxLength]) {
    srand(time(NULL));
    char path[100];
    char input[maxLength];
    int totalKata;
    totalKata = 0;
    FILE *fp;
    stringConcat("./data/", filename, path);
    fp = fopen(path, "r");
    if (fp==NULL) {
        printf("File gagal dibuka!\n");
    }
    while (fgets(input, maxLength-1, fp)) {
        sscanf(input, "%s", kata[totalKata]);
        totalKata++;
    }
    int idx;
    idx = rand()%totalKata;
    fclose(fp);
    return(kata[idx]);
}
```

5.2 Linked List Bertipe Point dalam Permainan Snake on Meteor

Dalam pembuatan game Snake on Meteor, linked list menyimpan tipe bentukan Point. Point ini berguna untuk menyimpan nilai x dan y untuk menentukan koordinat dalam peta permainan. Algoritma ini menjadi menarik karena biasanya kita hanya menyimpan info, prev, next dari sebuah address. Namun, sekarang kita menyimpan informasi tambahan yaitu Point sebagai posisi.

Berikut adalah structnya:

```
typedef struct tElmtlist {
    Point position;
    idxtype info;
    addressN next;
    addressN prev;
} ElmtList;
```

6 Data Test

Dalam program yang telah kami buat, yaitu BNMO, terdapat beberapa fitur atau command yang dijalankan oleh pemain. Untuk memastikan berbagai fitur dan command dapat dijalankan dengan baik dan benar, perlu dilakukan *testing* terhadap beberapa kondisi. Berikut *testing* dari beberapa fitur dan command dalam program kami, Selain itu, terdapat juga penjelasan data test, input, serta output yang akan diberikan program dalam tiap kondisi.

6.1 Data Test SCOREBOARD

Scoreboard adalah suatu command untuk melihat nama dan skor dari setiap permainan. Scoreboard akan menampilkan skor mengurut dari yang terbesar hingga ke terkecil untuk setiap permainan yang terdapat pada list game.

```
ENTER COMMAND: SCOREBOARD

**** SCOREBOARD GAME RING ****
| NAMA | SCORE |
|-----|-----|
| BNMO | 99    |
| Finn | 80    |
| YAHU | 30    |
| YIHA | 15    |
| YADU | 10    |
| YUHU | 5     |

**** SCOREBOARD GAME Diner DASH ****
| NAMA | SCORE |
|-----|-----|
| YO   | 44    |
| YU   | 30    |
| YI   | 21    |
| YA   | 20    |

**** SCOREBOARD GAME HANGMAN ****
| NAMA | SCORE |
|-----|-----|
| Jake | 58    |
| Finn | 31    |
| Marcelline | 30 |
| Mor  | 28    |
| Kil  | 12    |
| Dip  | 10    |

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

**** SCOREBOARD GAME SNAKE ON METEOR ****
| NAMA | SCORE |
|-----|-----|
| Marshall | 77 |

**** SCOREBOARD GAME GAME ASAL ****
| NAMA | SCORE |
|-----|-----|
|-----SCOREBOARD KOSONG-----

**** SCOREBOARD GAME TEST ****
| NAMA | SCORE |
|-----|-----|
|-----SCOREBOARD KOSONG-----

**** SCOREBOARD GAME Pokemon ****
| NAMA | SCORE |
|-----|-----|
|-----SCOREBOARD KOSONG-----
```

Gambar 6.1.1 Tampilan Scoreboard sebelum menambahkan skor pemain baru dari permainan yang baru dimainkan

```
Skor anda: 5
Masukan nama player : Mark

ENTER COMMAND: SCOREBOARD

**** SCOREBOARD GAME RING ****
| NAMA | SCORE |
|-----|-----|
| BNMO | 99    |
| Finn | 80    |
| YAHU | 30    |
| YIHA | 15    |
| YADU | 10    |
| YUHU | 5     |

**** SCOREBOARD GAME Diner DASH ****
| NAMA | SCORE |
|-----|-----|
| YO   | 44    |
| YU   | 30    |
| YI   | 21    |
| YA   | 20    |

**** SCOREBOARD GAME HANGMAN ****
| NAMA | SCORE |
|-----|-----|
| Jake | 58    |
| Finn | 31    |
| Marcelline | 30 |
| Mor  | 28    |
| Kil  | 12    |
| Dip  | 10    |

**** SCOREBOARD GAME TOWER OF HANOI ****
| NAMA | SCORE |
|-----|-----|
| Mark | 5     |

**** SCOREBOARD GAME SNAKE ON METEOR ****
| NAMA | SCORE |
|-----|-----|
| Marshall | 77 |

**** SCOREBOARD GAME GAME ASAL ****
| NAMA | SCORE |
|-----|-----|
|-----SCOREBOARD KOSONG-----

**** SCOREBOARD GAME TEST ****
| NAMA | SCORE |
|-----|-----|
|-----SCOREBOARD KOSONG-----

**** SCOREBOARD GAME Pokemon ****
| NAMA | SCORE |
|-----|-----|
|-----SCOREBOARD KOSONG-----
```

Gambar 6.1.2 Tampilan Scoreboard setelah ada tambahan skor pemain baru setelah bermain permainan Tower of Hanoi

```

data > ≡ save2.txt
33  YA 20
34  6
35  Jake 58
36  Finn 31
37  Marcelline 30
38  Mor 28
39  Kil 12
40  Dip 10
41  1
42  Mark 5
43  1
44  Marshall 77
45  0
46  0
47  0

```

Gambar 6.1.3 Tampilan file setelah terdapat penambahan skor pemain baru dan pemain melakukan prosedur SAVE

```

Masukan nama player : Jake
Nama tidak valid! Silahkan input ulang nama player.
Masukan nama player : 

```

Gambar 6.1.4 Tampilan jika pemain memasukkan nama player yang sudah ada

6.2 Data Test RESET SCOREBOARD

Reset scoreboard adalah command untuk membuat scoreboard game tertentu/semua scoreboard menjadi kosong. Pemain akan diminta masukan scoreboard game yang hendak di-reset dan program akan menanyakan ulang apakah pengguna benar benar ingin melakukan reset terhadap scoreboard. Jika ya, program akan me-reset scoreboard. Jika tidak , program tidak akan me-reset scoreboard.

```

ENTER COMMAND: RESET SCOREBOARD
DAFTAR SCOREBOARD:
0. All
1. RING
2. Diner DASH
3. HANGMAN
4. TOWER OF HANOI
5. SNAKE ON METEOR
6. GAME ASAL
7. TEST
8. Pokemon
SCOREBOARD YANG INGIN DIHAPUS: 4
APAKAH KAMU YAKIN INGIN MELAKUKAN RESET SCOREBOARD ALL (YA/TIDAK)? YA
Scoreboard berhasil di-reset.

```



```

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

```

Gambar 6.2.1 Tampilan Reset Scoreboard dan tampilan Scoreboard setelah pemain menghapus Scoreboard permainan Tower of Hanoi

```

ENTER COMMAND: RESET SCOREBOARD
DAFTAR SCOREBOARD:
0. All
1. RNG
2. Diner DASH
3. HANGMAN
4. TOWER OF HANOI
5. SNAKE ON METEOR
6. GAME ASAL
7. TEST
8. Pokemon
SCOREBOARD YANG INGIN DIHAPUS: 0
APAKAH KAMU YAKIN INGIN MELAKUKAN RESET SCOREBOARD ALL (YA/TIDAK)? YA
Scoreboard berhasil di-reset.

```

```

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

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

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

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

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

```

Gambar 6.2.2 Tampilan Reset Scoreboard dan tampilan Scoreboard setelah pemain menghapus semua Scoreboard permainan

```

data > save2.txt
1 8
2 RNG
3 Diner DASH
4 HANGMAN
5 TOWER OF HANOI
6 SNAKE ON METEOR
7 GAME ASAL
8 TEST
9 Pokemon
10 11
11 TOWER OF HANOI
12 TEST
13 GAME ASAL
14 GAME ASAL
15 TEST
16 TEST
17 Pokemon
18 Pokemon
19 HANGMAN
20 TOWER OF HANOI
21 SNAKE ON METEOR
22 0
23 0
24 0
25 0
26 0
27 0
28 0
29 0

```

Gambar 6.2.3 Tampilan file setelah pemain melakukan RESET SCOREBOARD terhadap semua permainan

6.3 Data Test HISTORY <n>

History <n> adalah command yang menampilkan permainan sebanyak n yang telah dimainkan dimana urutan teratas adalah permainan yang terakhir dimainkan. Jika n yang dimasukkan lebih besar dari jumlah game yang telah dimainkan, maka akan ditampilkan seluruh game yang telah dimainkan.

```

ENTER COMMAND: HISTORY 2
Berikut adalah daftar Game yang telah dimainkan
1. Diner DASH
2. HANGMAN

```

Gambar 6.3.1 Tampilan History jika n yang dimasukkan lebih kecil dari jumlah game yang telah dimainkan

```

ENTER COMMAND: HISTORY 5
Berikut adalah daftar Game yang telah dimainkan
1. Diner DASH
2. HANGMAN
3. RNG
4. HANGMAN

```

Gambar 6.3.2 Tampilan History jika n yang dimasukkan lebih besar dari jumlah game yang telah dimainkan maka akan menampilkan semua game yang dimainkan

6.4 Data Test RESET HISTORY

Reset History adalah command untuk menghapus semua permainan yang pernah dimainkan. Kemudian, program akan menanyakan ulang apakah pengguna benar-benar ingin menghapus seluruh history. Jika Ya, program akan menghapus seluruh history permainan. Jika tidak, reset history gagal dilakukan.

```
ENTER COMMAND: RESET HISTORY
APAKAH KAMU YAKIN INGIN MELAKUKAN RESET HISTORY? (YA/TIDAK) TIDAK

History tidak jadi di-reset. Berikut adalah daftar Game yang telah dimainkan
1. Diner DASH
2. HANGMAN
3. RNG
4. HANGMAN
```

Gambar 6.4.1 Tampilan Reset History jika pemain tidak jadi melakukan reset history

```
ENTER COMMAND: RESET HISTORY
APAKAH KAMU YAKIN INGIN MELAKUKAN RESET HISTORY? (YA/TIDAK) YA

History berhasil di-reset.

ENTER COMMAND: HISTORY 3
Berikut adalah daftar Game yang telah dimainkan
Belum ada game yang dimainkan.
```

Gambar 6.4.2 Tampilan reset history jika pemain melakukan reset history

6.5 Data Test HANGMAN

Hangman adalah permainan yang mengharuskan pemain untuk menebak sebuah kata. Dalam permainan ini, pemain diberikan 10 kesempatan untuk menebak satu huruf yang terdapat pada kata tersebut. Jika pemain berhasil menebak satu huruf yang terdapat pada kata, maka huruf yang sudah tertebak akan ditampilkan pada layar. Sebaliknya, jika pemain gagal menebak satu huruf yang terdapat pada kata, kesempatan pemain akan berkurang 1. Pemain tidak boleh menebak huruf yang sama sehingga program akan mengeluarkan output “Kamu sudah pernah menebak huruf ini!” jika pemain menebak huruf yang sama. Jika pemain berhasil menebak suatu kata, pemain tersebut akan mendapatkan skor permainan sesuai dengan panjang kata yang berhasil ditebak dan program akan memberikan kata baru untuk ditebak. Permainan akan terus berlanjut hingga kesempatan pemain habis.

```

Loading HANGMAN ...
Selamat datang dalam permainan HANGMAN!

----- MENU HANGMAN -----
/ 1. Langsung bermain HANGMAN /
/ 2. Menambahkan kata ke dalam list tebakan /
-----

Pilihlah nomor yang ingin kamu lakukan (1/2): █

```

Gambar 6.5.1 Tampilan awal menu Hangman

```

----- MENU HANGMAN -----
/ 1. Langsung bermain HANGMAN /
/ 2. Menambahkan kata ke dalam list tebakan /
-----

Pilihlah nomor yang ingin kamu lakukan (1/2): 3
Input tidak valid! Masukkan nomor 1 atau 2 sesuai MENU HANGMAN.

----- MENU HANGMAN -----
/ 1. Langsung bermain HANGMAN /
/ 2. Menambahkan kata ke dalam list tebakan /
-----

Pilihlah nomor yang ingin kamu lakukan (1/2): █

```

Gambar 6.5.2 Tampilan jika input awal menu Hangman tidak valid

```

----- MENU HANGMAN -----
/ 1. Langsung bermain HANGMAN /
/ 2. Menambahkan kata ke dalam list tebakan /
-----

Pilihlah nomor yang ingin kamu lakukan (1/2): 2

Masukkan kata yang ingin ditambahkan (Huruf kapital): KIWI
Kata berhasil ditambahkan ke dalam list tebakan!

----- MENU HANGMAN -----
/ 1. Langsung bermain HANGMAN /
/ 2. Menambahkan kata ke dalam list tebakan /
-----

Pilihlah nomor yang ingin kamu lakukan (1/2): █

```

Gambar 6.5.3 Tampilan jika pemain memilih untuk menambahkan kata ke dalam list tebakan

```
data > ≡ daftarkata.txt
1  MANGGA
2  ANGGUR
3  PISANG
4  STROBERI
5  DURIAN
6  LECI
7  JERUK
8  UAS
9  KIWI
```

Gambar 6.5.4 Tampilan kata berhasil ditambahkan ke dalam list kata

```
----- MENU HANGMAN -----
/ 1. Langsung bermain HANGMAN /
/ 2. Menambahkan kata ke dalam list tebakan /
-----

Pilihlah nomor yang ingin kamu lakukan (1/2): 1
Input tidak valid! Masukkan nomor 1 atau 2 sesuai MENU HANGMAN.

Selamat bermain HANGMAN!

Tebakan sebelumnya: -
Kata: _ _ _ _ _
Kesempatan: 10
Masukkan tebakan: █
```

Gambar 6.5.5 Tampilan jika pemain memilih untuk langsung bermain Hangman

```
Tebakan sebelumnya: -
Kata: _ _ _ _ _
Kesempatan: 10
Masukkan tebakan: G

Tebakan sebelumnya: G
Kata: _ _ _ GG_
Kesempatan: 10
Masukkan tebakan: █
```

Gambar 6.5.6 Tampilan jika suatu huruf dalam kata tebakan berhasil ditebak oleh pemain

```
Tebakan sebelumnya: G
Kata: _ _ _ GG_
Kesempatan: 10
Masukkan tebakan: I

Tebakan sebelumnya: GI
Kata: _ _ _ GG_
Kesempatan: 9
Masukkan tebakan: █
```

Gambar 6.5.7 Tampilan jika pemain gagal menebak suatu huruf dalam kata tebakan

```

Tebakan sebelumnya: GI
Kata: _ _ _ GG_
Kesempatan: 9
Masukkan tebakan: a

Tebakan sebelumnya: GIa
Kata: _ A_ GGA
Kesempatan: 9
Masukkan tebakan: 

```

Gambar 6.5.8 Tampilan jika pemain memasukkan huruf tebakan dengan huruf kecil (tetap terhitung benar)

```

Tebakan sebelumnya: GIa
Kata: _ A_ GGA
Kesempatan: 9
Masukkan tebakan: G
Kamu sudah pernah menebak huruf ini!

Tebakan sebelumnya: GIa
Kata: _ A_ GGA
Kesempatan: 9
Masukkan tebakan: 

```

Gambar 6.5.9 Tampilan jika pemain menebak huruf yang sudah pernah ditebak sebelumnya

```

Tebakan sebelumnya: GIaM
Kata: MA_ GGA
Kesempatan: 8
Masukkan tebakan: n

Berhasil menebak kata MANGGA! Kamu mendapatkan 6 poin!

Tebakan sebelumnya: -
Kata: _ _ _ _ _
Kesempatan: 8
Masukkan tebakan: 

```

Gambar 6.5.10 Tampilan jika pemain berhasil menebak sebuah kata dan kesempatannya belum habis

```

Tebakan sebelumnya: -
Kata: _ _ _ _ _
Kesempatan: 8
Masukkan tebakan: tebak
Input tidak valid! Masukan tebakan sebanyak 1 char.

Masukkan tebakan: 

```

Gambar 6.5.11 Tampilan jika input tebakan tidak valid



Gambar 6.6.2 Tampilan jika input yang dimasukkan pemain tidak valid. Input tidak valid karena piringan pada tiang asal lebih besar daripada piringan paling atas pada tiang tujuan



Gambar 6.6.3 Tampilan jika input yang dimasukkan pemain tidak valid. Input tidak valid karena tiang asal dan tiang tujuan adalah tiang yang sama



Gambar 6.6.4 Tampilan jika semua piringan berhasil dipindahkan ke tiang C

6.7 Data Test SNAKE ON METEOR

Snake on Meteor adalah permainan yang mengharuskan pemain untuk memakan makanan agar panjang badan snake bertambah. Pemain dikatakan menang jika panjang badan snake sepanjang 10. Pada game snake on meteor, nilai maksimum dari pemain adalah sebesar 20 poin dimana perhitungannya adalah 2 dikalikan dengan panjang snake. Namun, pemain harus berhati-hati karena terdapat serangan meteor yang dapat memberikan damage pada badan snake. Jika meteor mengenai kepala snake maka permainan akan game over. Jika meteor mengenai badan snake selain snake, maka bagian badan tersebut akan terhapus. Pemain juga harus berhati-hati dengan obstacle karena obstacle tidak dapat dilewati. Jika obstacle dilewati, maka pemain langsung akan kalah. Selain itu, meteor pada turn sebelumnya tidak dapat dilewati pada putaran berikutnya dan akan memberikan output bahwa “Meteor masih panas” dan pemain harus memberikan input lain.


```

ENTER COMMAND: PLAY GAME
Berikut adalah daftar Game-mu
1. SNAKE ON METEOR

Loading SNAKE ON METEOR ...

Selamat Datang di Snake on Meteor!
Men-generate peta, snake, dan makanan...
Berhasil digenerate!

Berikut merupakan peta permainan
+---+---+---+---+---+
| 1 | H |   |   |   |
+---+---+---+---+---+
| 2 |   |   |   |   |
+---+---+---+---+---+
|   |   |   |   |   |
+---+---+---+---+---+
|   |   | o |   |   |
+---+---+---+---+---+
| x |   |   |   |   |
+---+---+---+---+---+

TURN 1
Pilihan gerakan :
Atas (w) | Kiri (a) | Bawah (s) | Kanan (d)

Silahkan masukkan command anda: █

```

Gambar 6.7.1 Tampilan awal Snake on Meteor

```

TURN 1
Pilihan gerakan :
Atas (w) | Kiri (a) | Bawah (s) | Kanan (d)

Silahkan masukkan command anda: a
Anda tidak dapat bergerak ke tubuh anda sendiri!
Silahkan input command yang lain

+---+---+---+---+---+
| 1 | H |   |   |   |
+---+---+---+---+---+
| 2 |   |   |   |   |
+---+---+---+---+---+
|   |   |   |   |   |
+---+---+---+---+---+
|   |   | o |   |   |
+---+---+---+---+---+
| x |   |   |   |   |
+---+---+---+---+---+

TURN 1
Pilihan gerakan :
Atas (w) | Kiri (a) | Bawah (s) | Kanan (d)

Silahkan masukkan command anda: █

```

Gambar 6.7.2 Tampilan jika input yang dimasukkan pemain membuat snake bergerak ke tubuh sendiri. Turn tidak akan bertambah dan pemain diminta untuk memasukkan command lain.

```

TURN 3
Pilihan gerakan :
Atas (w) | Kiri (a) | Bawah (s) | Kanan (d)

Silahkan masukkan command anda: s
Anda terkena meteor.

Berikut merupakan peta permainan
+---+---+---+---+
|   |   | 1 |   |   |
+---+---+---+---+
|   |   | m |   |   |
+---+---+---+---+
|   |   | H |   |   |
+---+---+---+---+
|   |   | o |   |   |
+---+---+---+---+
| x |   |   |   |   |
+---+---+---+---+

TURN 4
Pilihan gerakan :
Atas (w) | Kiri (a) | Bawah (s) | Kanan (d)

Silahkan masukkan command anda: 

```

Gambar 6.7.3 Tampilan jika pemain terkena meteor

```

TURN 4
Pilihan gerakan :
Atas (w) | Kiri (a) | Bawah (s) | Kanan (d)

Silahkan masukkan command anda: d
Anda beruntung tidak terkena meteor!

Berikut merupakan peta permainan
+---+---+---+---+
|   |   |   |   |   |
+---+---+---+---+
|   |   |   |   |   |
+---+---+---+---+
|   |   | 1 | H |   |
+---+---+---+---+
|   |   | o |   |   |
+---+---+---+---+
| x | m |   |   |   |
+---+---+---+---+

TURN 5
Pilihan gerakan :
Atas (w) | Kiri (a) | Bawah (s) | Kanan (d)

Silahkan masukkan command anda: 

```

Gambar 6.7.4 Tampilan setelah pemain terkena meteor dan bergerak lagi. Badan snake yang berada di belakang bagian badan yang terkena meteor akan menyambung lagi dan nilainya akan terupdate

```

TURN 5
Pilihan gerakan :
Atas (w) | Kiri (a) | Bawah (s) | Kanan (d)

Silahkan masukkan command anda: s
Anda beruntung tidak terkena meteor!

Berikut merupakan peta permainan
+---+---+---+---+
|   |   |   |   |   |
+---+---+---+---+
|   |   |   |   |   |
+---+---+---+---+
|   |   | m | 1 |   |
+---+---+---+---+
|   |   | o | H |   |
+---+---+---+---+
| x |   |   |   |   |
+---+---+---+---+

TURN 6
Pilihan gerakan :
Atas (w) | Kiri (a) | Bawah (s) | Kanan (d)

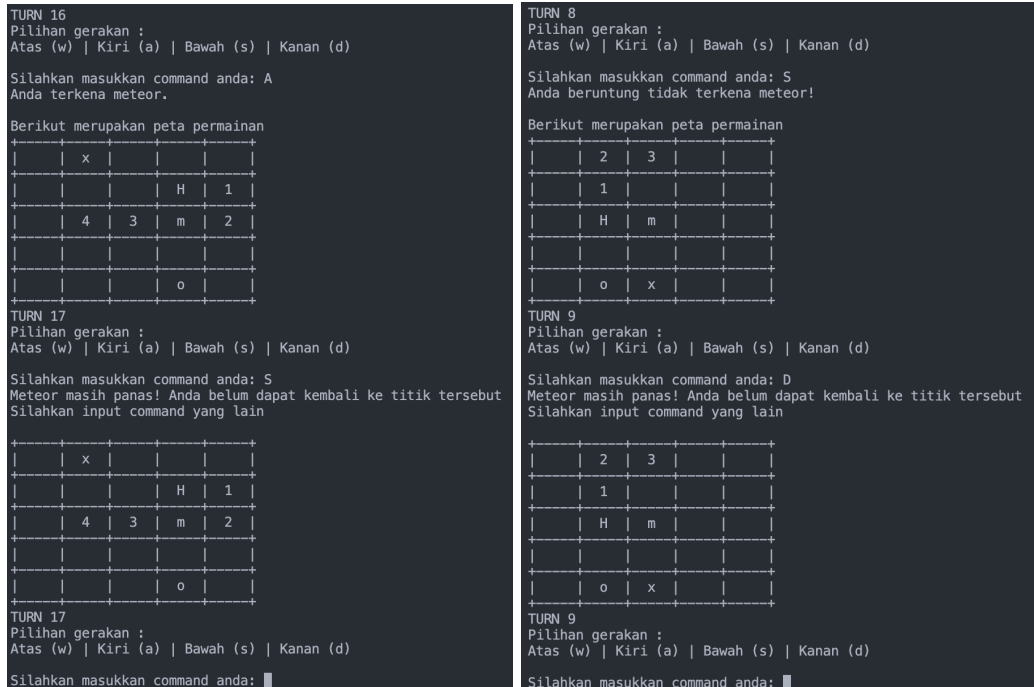
Silahkan masukkan command anda: a
Kepala terkena meteor. Game over!

+---+---+---+---+
|   | o |   |   |   |
+---+---+---+---+
|   |   |   |   |   |
+---+---+---+---+
|   |   |   | 2 |   |
+---+---+---+---+
|   |   | m | 1 |   |
+---+---+---+---+
| x |   |   |   |   |
+---+---+---+---+

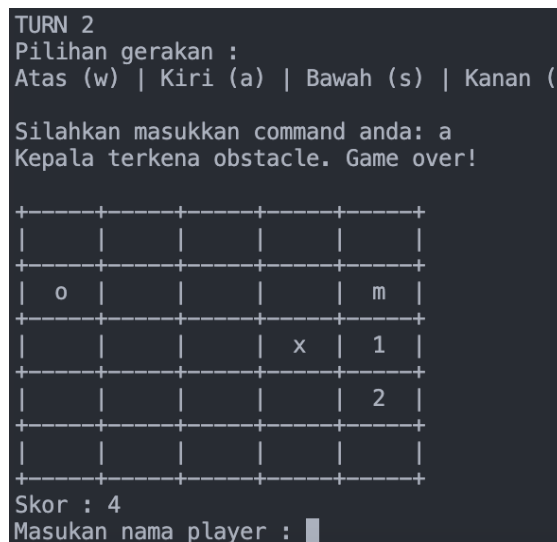
Skor : 4
Masukan nama player : 

```

Gambar 6.7.5 Tampilan ketika pemain berhasil memakan sekaligus terkena meteor pada bagian head dari Snake. Permainan akan game over dan menampilkan skor dari pemain. Selain itu, program akan meminta nama player untuk dimasukkan ke scoreboard



Gambar 6.7.6 Tampilan jika pemain memasukkan command yang pada turn sebelumnya titik tersebut terkena meteor.



Gambar 6.7.7 Tampilan jika pemain mengunjungi obstacle. Permainan akan berakhir dan menampilkan skor dan program akan meminta nama player untuk dimasukkan ke scoreboard

```

TURN 4
Pilihan gerakan :
Atas (w) | Kiri (a) | Bawah (s) | Kanan (d)

Silahkan masukkan command anda: A
Anda beruntung tidak terkena meteor!

Berikut merupakan peta permainan
+-----+
| | | | | | |
+-----+
| H | 1 | 2 | 3 | | |
+-----+
| | | | | o | |
+-----+
| | | m | | | |
+-----+
| | x | | | | |
+-----+

TURN 5
Pilihan gerakan :
Atas (w) | Kiri (a) | Bawah (s) | Kanan (d)

Silahkan masukkan command anda: A
Anda beruntung tidak terkena meteor!

Berikut merupakan peta permainan
+-----+
| | | | m | | |
+-----+
| 1 | 2 | 3 | | H | |
+-----+
| | | | o | | |
+-----+
| | | | | | |
+-----+
| | x | | | | |
+-----+

TURN 6
Pilihan gerakan :
Atas (w) | Kiri (a) | Bawah (s) | Kanan (d)

Silahkan masukkan command anda:

```

Gambar 6.7.8 Tampilan jika pemain sudah berada di ujung petak dan memasukkan command yang membuat pergerakan tembus ke pada sisi seberang petak maka snake tetap dapat digerakkan ke seberang petak tersebut

```

Berikut merupakan peta permainan
+-----+
| | | | | | |
+-----+
| o | 6 | | | | |
+-----+
| 5 | H | 1 | | | |
+-----+
| | 4 | m | | x | |
+-----+
| | 3 | 2 | | | |
+-----+

TURN 43
Pilihan gerakan :
Atas (w) | Kiri (a) | Bawah (s) | Kanan (d)

Silahkan masukkan command anda: d
Snake tidak dapat bergerak ke manapun. Game over!

```

Gambar 6.7.9 Tampilan jika snake tidak dapat bergerak kemana-mana lagi karena sudah terhalang oleh badan snake di titik titik sekitarnya. Permainan akan game over.

```

Berikut merupakan peta permainan
+-----+-----+-----+-----+
|   |   |   |   | H | o |
+-----+-----+-----+-----+
| 5 |   |   |   | 1 | 4 |
+-----+-----+-----+-----+
|   |   |   |   | 2 | 3 |
+-----+-----+-----+-----+
|   |   | m |   | x |   |
+-----+-----+-----+-----+
|   |   |   |   |   |   |
+-----+-----+-----+-----+
TURN 24
Pilihan gerakan :
Atas (w) | Kiri (a) | Bawah (s) | Kanan (d)

Silahkan masukkan command anda: d
Tail tidak dapat bertambah. Game over!

```

Gambar 6.7.10 Tampilan jika tail tidak dapat bertambah

6.8 Data Test LOAD

Load merupakan sebuah command yang digunakan untuk membaca file konfigurasi yang berisi nama permainan, nama pemain beserta skor tiap pemain, dan jumlah tiap pemain yang terdaftar dalam tiap permainan. Load harus diikuti oleh nama file yang ingin dibuka oleh pemain.

| | | |
|-------------------|--------------------|------------------|
| data > save2.txt | 16 Pokemon | 31 YI 21 |
| 1 8 | 17 Pokemon | 32 YA 20 |
| 2 RING | 18 HANGMAN | 33 6 |
| 3 Diner DASH | 19 TOWER OF HANOI | 34 Jake 58 |
| 4 HANGMAN | 20 SNAKE ON METEOR | 35 Finn 31 |
| 5 TOWER OF HANOI | 21 6 | 36 Marcelline 30 |
| 6 SNAKE ON METEOR | 22 BNMO 99 | 37 Mor 28 |
| 7 GAME ASAL | 23 Finn 80 | 38 Kil 12 |
| 8 TEST | 24 YAHU 30 | 39 Dip 10 |
| 9 Pokemon | 25 YIHA 15 | 40 0 |
| 10 10 | 26 YADU 10 | 41 1 |
| 11 TEST | 27 YUHU 5 | 42 Marshall 77 |
| 12 GAME ASAL | 28 4 | 43 0 |
| 13 GAME ASAL | 29 YO 44 | 44 0 |
| 14 TEST | 30 YU 30 | 45 0 |
| 15 TEST | | |
| 16 Pokemon | | |
| 17 Pokemon | | |
| 18 HANGMAN | | |

Gambar 6.8.1 Tampilan file save2.txt yang ingin dibaca

```

ENTER COMMAND: LOAD save2.txt
Save file berhasil dibaca. BNMO berhasil dijalankan.

```

Gambar 6.8.2 Tampilan jika terdapat file konfigurasi yang sesuai dengan command pengguna dan file berhasil dibaca

```

ENTER COMMAND: LOAD test.txt
File tidak exist

```

Gambar 6.8.3 Tampilan jika file konfigurasi tidak tersedia

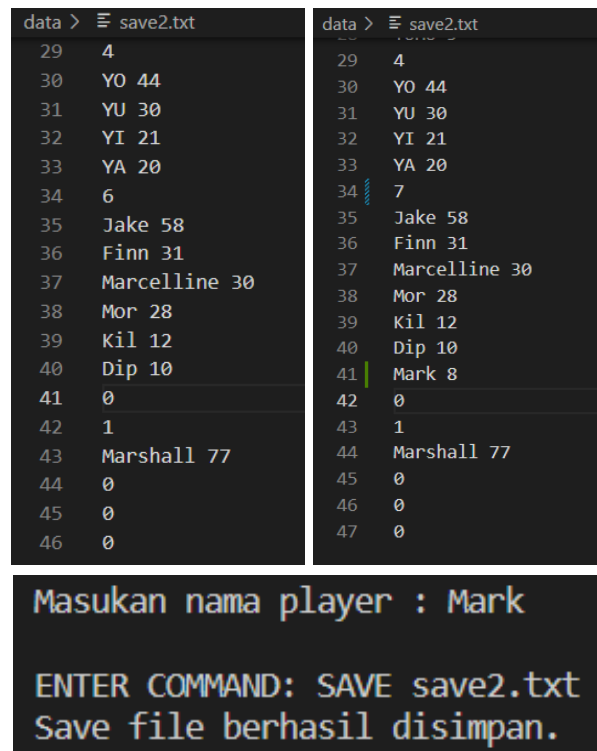
sehingga tidak berhasil dibaca

```
ENTER COMMAND: SCOREBOARD
Belum ada konfigurasi games yang dimuat
Muat konfigurasi games terlebih dahulu dengan command LOAD <filename>
```

Gambar 6.8.4 Tampilan jika pemain memasukkan command selain LOAD tanpa melakukan LOAD terlebih dahulu

6.9 Data Test SAVE

Save merupakan sebuah command untuk menyimpan *state game* pemain, nama pemain beserta skornya, dan hasil scoreboard tiap permainan. Command ini harus diikuti dengan nama file yang akan disimpan pada disk. Command Save dikatakan bekerja dengan baik dan benar jika *state game* pemain, nama pemain beserta skornya, dan hasil scoreboard tiap permainan berhasil disimpan dalam file yang telah disebutkan namanya. File yang disebutkan dapat berupa file lama yang akan di-"update" atau file baru yang ingin dibentuk.



Gambar 6.9.1 Tampilan perbandingan file sebelum dan sesudah ada pemain yang melakukan SAVE setelah bermain Hangman

7 Test Script

Seperti yang telah disebutkan sebelumnya, diperlukan *testing* untuk memastikan bahwa fitur-fitur dan command-command yang ada dalam program dapat berjalan dengan baik dan

benar. Oleh karena itu, dalam bagian Test Script ini akan dijelaskan beberapa skenario test yang dimungkinkan untuk semua fitur dan command.

Berikut skenario test dijelaskan lebih lanjut dalam bentuk tabel.

| No. | Fitur yang Dites | Tujuan Testing | Langkah-Langkah Testing | Input Data Test | Hasil yang Diharapkan | Hasil yang Keluar |
|-----|------------------|---|--|--------------------------------|---|---|
| 1 | Scoreboard | Memeriksa apakah Scoreboard dapat menampilkan output yang sesuai dengan jumlah skor permainan tiap pemain | Masukkan command SCOREBOARD | 6.1 Data Test SCOREBOARD | Menampilkan scoreboard setiap game dari yang skornya tertinggi hingga ke rendah | Sesuai Gambar 6.1.1, 6.1.2, 6.1.3, dan 6.1.4 |
| 2 | Reset Scoreboard | Memeriksa apakah Scoreboard dapat di-reset dan menampilkan output yang sesuai | Masukkan command RESET SCOREBOARD | 6.2 Data Test RESET SCOREBOARD | Menghapus scoreboard untuk game tertentu/seluruh game (tergantung input pengguna) | Sesuai Gambar 6.2.1, 6.2.2, dan 6.2.3 |
| 3 | History | Memeriksa apakah program dapat menampilkan apa saja yang telah dimainkan dengan output yang sesuai | Masukkan command HISTORY <n> dengan n adalah jumlah history game yang ingin ditampilkan | 6.3 Data Test HISTORY <n> | Melihat game yang telah dimainkan (dari yang paling terakhir dimainkan) | Sesuai gambar 6.3.1 dan 6.3.2 |
| 4 | Reset History | Memeriksa apakah program dapat menghapus history permainan dengan output yang sesuai | Masukkan command RESET HISTORY | 6.4 Data Test RESET HISTORY | Menghapus history permainan | Sesuai gambar 6.4.1 dan 6.4.2 |
| 5 | Hangman | Memeriksa apakah permainan Hangman dapat | Melakukan queue game dan memasukkan nomor 3 pada game yang ingin dimasukkan pada antrian. Lalu, Hangman akan | 6.5 Data Test Hangman | Menampilkan huruf yang ditebak jika terdapat huruf tersebut di | Sesuai Gambar 6.5.1, 6.5.2, 6.5.3, 6.5.4, 6.5.5, 6.5.6, 6.5.7, 6.5.8, |

| | | | | | | |
|---|-----------------|--|---|-------------------------------|--|--|
| | | dimainkan dan mengeluarkan output yang sesuai | dimainkan jika pemain memasukkan command PLAY GAME dan Hangman merupakan urutan teratas pada queue game. Kemudian, pemain akan masuk ke menu hangman dan dapat memilih apakah akan langsung bermain atau menambahkan kata ke dalam list tebakan. Jika pemain memilih untuk bermain, program akan meminta input tebakan dari pemain. | | dalam kata yang harus ditebak. Mengurangi kesempatan menebak jika tidak terdapat huruf tersebut dalam kata yang harus ditebak. Menampilkan skor jika kesempatan menebak sudah habis. | 6.5.9, 6.5.10, 6.5.11, dan 6.5.12 |
| 6 | Tower of Hanoi | Memeriksa apakah permainan Tower of Hanoi dapat dimainkan dan mengeluarkan output yang sesuai | Melakukan queue game dan memasukkan nomor 4 pada game yang ingin dimasukkan pada antrian. Lalu, Tower of Hanoi akan dimainkan jika pemain memasukkan command PLAY GAME dan Tower of Hanoi merupakan urutan teratas pada queue game. Kemudian, pemain memasukkan input tiang A/B/C sebagai tiang tujuan/tiang asal. | 6.6 Data Test Tower of Hanoi | Memindahkan piringan jika input yang dimasukkan pemain valid. Menampilkan "INPUT INVALID" jika input yang dimasukkan invalid. Menampilkan skor jika pemain berhasil memindahkan seluruh piringan ke tiang C. | Sesuai Gambar 6.6.1, 6.6.2, 6.6.3, dan 6.6.4 |
| 7 | Snake on Meteor | Memeriksa apakah permainan Snake on Meteor dapat dimainkan dan mengeluarkan output yang sesuai | Melakukan queue game dan memasukkan nomor 5 pada game yang ingin dimasukkan pada antrian. Lalu, Snake on Meteor akan dimainkan jika pemain memasukkan command PLAY GAME dan Snake on Meteor merupakan urutan teratas pada queue game. Kemudian, pemain memasukkan arah gerak snake (s/d/a/w) hingga gameOver. | 6.7 Data Test Snake on Meteor | Menambahkan badan snake jika snake berhasil memakan makanan. Menghapus badan snake jika bagian badan terkena meteor. Menampilkan skor jika kepala snake terkena meteor/obstacle atau snake tidak dapat | Sesuai Gambar 6.7.1, 6.7.2, 6.7.3, 6.7.4, 6.7.5, 6.7.6, 6.7.7, 6.7.8, 6.7.9 dan 6.7.10 |

| | | | | | | |
|---|------|---|--|--------------------|---|--|
| | | | | | bergerak karena sekelilingnya sudah dipenuhi badan snake. | |
| 8 | Load | Memeriksa apakah file berhasil dibaca dan BNMO berhasil dijalankan | Melakukan compile dan memasukkan command LOAD yang disertai nama file. | 6.8 Data Test Load | Jika terdapat save file yang tersedia, save file yang dipilih berhasil dibaca dan permainan berhasil dijalankan. Apabila file tidak tersedia, menampilkan pesan bahwa File tidak exist. | Sesuai Gambar 6.8.1, 6.8.2, 6.8.3, dan 6.8.4 |
| 9 | Save | Memeriksa apakah state game pemain, nama pemain beserta skornya, dan hasil scoreboard tiap permainan berhasil disimpan dalam file | Melakukan compile dan masukkan command SAVE beserta nama file. | 6.9 Data Test Save | Menyimpan state game pemain, nama pemain beserta skornya, dan hasil scoreboard tiap permainan ke dalam suatu file. | Sesuai Gambar 6.9.1 |

8 Pembagian Kerja dalam Kelompok

Berikut tabel daftar pembagian kerja dalam kelompok.

| Nama | NIM | Pembagian Kerja |
|------------------------|----------|---|
| Ghaylan Muhammad Fatih | 18221042 | TOWER OF HANOI |
| Gracia Theophilia | 18221078 | HANGMAN, membuat laporan, mengontak asisten |
| Fikri Naufal Hamdi | 18221096 | SNAKE ON METEOR, mengontak asisten |

| | | |
|---------------------|----------|--|
| Reinhart Wisely Lim | 18221154 | SCOREBOARD, RESET SCOREBOARD, HISTORY, RESET HISTORY |
| Esther Regina | 18221086 | SNAKE ON METEOR, membuat laporan |

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. Pada tugas sebelumnya, kalian telah berhasil membuat Indra dan Doni bahagia dengan mengimplementasikan fitur-fitur dasar. Kini, Indra dan Doni ingin melakukan pengembangan lebih lanjut dengan menambahkan fitur serta permainan pada BNMO.

Buatlah sebuah permainan berbasis CLI (command-line interface). Sistem ini dibuat dalam bahasa C dengan menggunakan struktur data yang sudah kalian pelajari di mata kuliah ini. Kalian boleh menggunakan (atau memodifikasi) struktur data yang sudah kalian buat untuk praktikum pada tugas besar ini. Library yang boleh digunakan hanya `stdio.h`, `stdlib.h`, `time.h` dan `math.h`.

9.2 Notulen Rapat

Kami melakukan rapat kelompok sebanyak sekali yang bertujuan untuk pembagian tugas masing-masing anggota kelompok.

Berikut notulen rapat tersebut.

Rapat 1: Jumat, 18 November 2022

Topik Pembahasan: Pembagian Tugas Masing-Masing Anggota

Platform Dilaksanakannya Rapat: Zoom

Hasil Rapat:

Hasil rapat berupa pembagian tugas (membuat command maupun fitur) untuk Tugas Besar 2 sebagai berikut.

- Reinhart Wisely Lim: SCOREBOARD, RESET SCOREBOARD, HISTORY <n>, dan RESET HISTORY
- Gracia Theophilia: HANGMAN
- Ghaylan Muhammad Fatih: TOWER OF HANOI
- Esther Regina dan Fikri Naufal Hamdi: SNAKE ON METEOR

9.3 Log Activity Anggota Kelompok

Berikut tabel *log activity* anggota kelompok.

| | | |
|---|---|----------------------------|
| STEI- ITB | 2 | Halaman 35 dari 40 halaman |
| Template dokumen ini dan informasi yang dimilikinya adalah milik Sekolah Teknik Elektro dan Informatika ITB dan bersifat rahasia. Dilarang me-reproduksi dokumen ini tanpa diketahui oleh Sekolah Teknik Elektro dan Informatika ITB. | | |

| NIM | Deskripsi | Tanggal |
|--|--|------------------|
| 18221042 18221078 18221096 18221154 18221086 | Melakukan pembagian tugas melalui <i>platform</i> Zoom. | 18 November 2022 |
| 18221154 | Membuat ADT Map dan SCOREBOARD | 19 November 2022 |
| 18221154 | Membuat HISTORY <n> dan RESETHISTORY | 20 November 2022 |
| 18221154 | Membuat ADT Stack | 22 November 2022 |
| 18221154 | Membuat ADT Array of Map | 24 November 2022 |
| 18221096 | Membuat ADT listdp, ADT point, dan permainan Snake on Meteor | 24 November 2022 |
| 18221042 | Membuat permainan Tower of Hanoi | 24 November 2022 |
| 18221078 | Membuat permainan Hangman dan membuat laporan bagian ringkasan, deskripsi tugas besar 2, dan notulen rapat | 25 November 2022 |
| 18221042 18221078 18221096 18221154 18221086 | Asistensi 1 | 25 November 2022 |
| 18221096 | Membuat ADT StackInt | 26 November 2022 |
| 18221078 | Update Hangman dan membuat laporan bagian ADT Stack, ADT StackInt, ADT Map, dan ADT Array of Map | 28 November 2022 |
| 18221086 | Update Snack on Meteor, RNG, dan ADT listdp | 29 November 2022 |
| 18221078 | Membuat laporan bagian ADT Point | 30 November 2022 |
| 18221086 | Membuat laporan bagian program utama bagian Snake on Meteor dan ADT Linked List | 30 November 2022 |

| | | |
|--|---|-----------------|
| 18221042 18221078 18221096 18221154 18221086 | Asistensi 2 | 1 Desember 2022 |
| 18221078 | Update Hangman dan Help serta membuat laporan bagian penjelasan tambahan spesifikasi Hangman dan Scoreboard, algoritma menarik random string, data test, test script. | 1 Desember 2022 |
| 18221086 | Membuat laporan bagian algoritma menarik linked list bertipe pointer, data test, dan test script | 1 Desember 2022 |
| 18221154 | Update ADT, Skip Game, dan Play Game | 1 Desember 2022 |
| 18221096 | Update Snake on Meteor | 2 Desember 2022 |
| 18221042 18221078 18221096 18221154 18221086 | Finalisasi | 2 Desember 2022 |

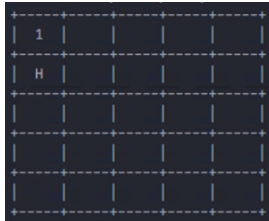





9.4 Lain-Lain


9.4.1 Form Asistensi 1

**Form Asistensi Tugas Besar 2
IF2110/Algoritma dan Struktur Data
Sem. 1 2022/2023**

No. Kelompok/Kelas : 12/K02
 Nama Kelompok : Kelompok 12
 Anggota Kelompok (Nama/NIM) :
 1. Ghaylan Muhammad Fatih/18221042
 2. Gracia Theophilia/18221078
 3. Fikri Naufal Hamdi/18221096
 4. Reinhart Wisely Lim/18221154
 5. Esther Regina/18221086
 Asisten Pembimbing : Jason Stanley Yoman

Asistensi I

| | |
|--|--|
| Tanggal : 25 November 2022 | <p>Catatan Asistensi:</p> <ol style="list-style-type: none"> 1. Bagaimana untuk penambahan ekor di titik (0,1) dalam game Snake on Meteor seperti kasus berikut?  <p>Urutan prioritas untuk generate tail bisa dilihat di spesifikasi, antara ke kiri atau ke kanan tapi lebih prioritas secara horizontal. Untuk kasus ini, prioritas ke kanan terlebih dahulu.</p> <ol style="list-style-type: none"> 2. Seandainya badan terkena obstacle (game Snake on Meteor) bagaimana kak? Kalau badan tidak mungkin terkena obstacle karena badan selalu ikut arah kepalanya. 3. Ingin memastikan, untuk penambahan ekor berarti hanya special case di titik (0,0) saja kah kak (game Snake on Meteor)? Jadi di titik (0,0) membesar secara skalar? Iya, betul. |
| Tempat : Zoom | |
| <p>Kehadiran Anggota Kelompok:</p> <p>No NIM Tanda tangan</p> <p>1 18221042</p>  <p>2 18221078</p>  <p>3 18221096</p>  <p>4 18221154</p>  <p>5 18221086</p>  | |

| | |
|--|--|
| | Tanda Tangan Asisten:  Jason Stanley Yoman |
|--|--|

9.4.2 Form Asistensi 2



Form Asistensi Tugas Besar 2 IF2110/Algoritma dan Struktur Data Sem. 1 2022/2023




No. Kelompok/Kelas : 12/K02
 Nama Kelompok : Kelompok 12
 Anggota Kelompok (Nama/NIM) :

- 1. Ghaylan Muhammad Fatih/18221042
- 2. Gracia Theophilia/18221078
- 3. Fikri Naufal Hamdi/18221096
- 4. Reinhart Wisely Lim/18221154
- 5. Esther Regina/18221086

Asisten Pembimbing : Jason Stanley Yoman

Asistensi 2

| | |
|---|---|
| Tanggal : 1 Desember 2022 | Catatan Asistensi: <ol style="list-style-type: none"> 1. Di spek tertulis, lokasi makanan tidak bisa muncul di bagian ekor sama badan snake, apakah makanan juga tidak bisa muncul di bagian di head juga? Include headnya juga. 2. Meteor baru muncul di turn 2 kan? Iya, betul. 3. Apakah perlu ngeprint lagi kondisi terkena meteor yang baru sebelum bergerak? Gak perlu, printnya abis dia bergerak aja. |
| Tempat : Zoom | |
| Kehadiran Anggota Kelompok: No NIM Tanda tangan 1 18221042  2 18221078  | |

| | |
|---|--|
| <p>3 18221096</p>  <p>4 18221154</p>  <p>5 18221086</p>  | |
| | <p>Tanda Tangan Asisten:</p>  <p>Jason Stanley Yoman</p> |