

LAPORAN TUGAS BESAR

IF2111 Algoritma dan Struktur Data

Aplikasi BNMO


Dipersiapkan oleh:

Kelompok 1

Wan Aufa Azis	18221001
Athira Dhyannis Tafkir	18221022
Dhafin Ghalib Luqman Hakim	18221023
Syasya Umaira	18221026
Muhammad Mumtaz	18221029

Sekolah Teknik Elektro dan Informatika - Institut Teknologi Bandung

Jl. Ganesha 10, Bandung 40132

	Sekolah Teknik Elektro dan Informatika ITB	Nomor Dokumen		Halaman
		<i>IF2111-TB1-03-01</i>		35
		<i>Revisi</i>	1	11 November 2022

Daftar Isi

1	Ringkasan	3
2	Penjelasan Tambahan Spesifikasi Tugas	5
2.1	Spesifikasi Fitur Bonus - Game Jari Bocil	5
3	Struktur Data (ADT)	9
3.1	ADT Array	9
3.2	ADT Mesin Kata	10
3.3	ADT Mesin Karakter	10
3.4	ADT Queue	10
3.5	ADT QueueDD	11
4	Program Utama	11
5	Algoritma-Algoritma Menarik	11
5.1	Prosedur Fiture	12
5.2	Prosedur ChooseMode	12
6	Data Test	13
6.1	Data Test MAIN MENU	13
6.2	Data Test START	13
6.3	Data Test LOAD GAME	14
6.4	Data Test CREATE GAME	15
6.5	Data Test LIST GAME	15
6.6	Data Test DELETE GAME	15
6.7	Data Test QUEUE GAME	16
6.8	Data Test PLAY GAME	17
6.8.1	Data Test GAME RNG	17
6.8.2	Data Test GAME DINER DASH	18
6.8.3	Data Test GAME JARI BOCIL	19
6.9	Data Test SKIP GAME	20
6.10	Data Test SAVE	21
6.11	Data Test HELP	21
6.12	Data Test QUIT	21
6.13	Data Test Command Lain	22
7	Test Script	22

8	Pembagian Kerja dalam Kelompok	26
9	Lampiran	27
9.1	Deskripsi Tugas Besar 1	27
	Command	27
a.	START	27
b.	LOAD <filename>	27
c.	SAVE <filename>	27
d.	CREATEGAME	27
e.	LISTGAME	27
f.	DELETGAME	27
g.	QUEUEGAME	28
h.	PLAYGAME	28
i.	SKIPGAME <n>	28
j.	QUIT	28
k.	HELP	28
l.	COMMAND LAIN	28
1.	RNG	28
2.	Diner Dash	29
3.	Game tambahan/ Buatan pemain	30
	Daftar ADT yang Digunakan	30
1.	ADT Array	30
2.	ADT Mesin Karakter dan Mesin Kata	30
3.	ADT Queue	30
	Bonus	30
9.2	Notulen Rapat	32
9.3	Log Activity Anggota Kelompok	35

1 Ringkasan

Permasalahan yang dihadapi oleh Indra dan Doni adalah munculnya *bug* di sistem BNMO yang mereka perbaiki dua bulan yang lalu dan sekarang kami harus membantu mereka untuk memprogram ulang robot video game console kesayangan mereka. BNMO atau dibaca Binomo merupakan suatu aplikasi untuk menjalankan permainan. Aplikasi ini diprogram menggunakan bahasa C yang berbasis *command line interface*. Untuk membuat aplikasi ini, kami memanfaatkan berbagai struktur data (ADT) seperti array, mesin kata, mesin karakter, dan queue. BNMO memiliki dan menjalankan beberapa fitur utama yaitu memainkan *game*, menambahkan *game* menghapus *game* serta mengurutkan *game* yang akan dimainkan. Saat BNMO dijalankan, BNMO akan menampilkan main menu yang berisikan *welcome page* dan menampilkan menu pilihan yaitu START dan LOAD. Nantinya main menu akan menerima *input command-command* yang ada yaitu START, LOAD, SAVE, CREATEGAME, LISTGAME, DELETGAME, QUEUEGAME, PLAYGAME, SKIPGAME, QUIT, HELP, dan COMMAND LAIN.

BNMO memiliki dua permainan utama yaitu RNG dan Diner Dash. Permainan yang pertama adalah RNG atau *Random Number Generator* adalah permainan menebak angka acak yang sudah ditentukan oleh program. Saat permainan dimulai, program sudah menentukan angka acak yang harus ditebak oleh pemain. Pemain harus menebak angka acak tersebut dengan menginput angka yang dikira benar kedalam program. Apabila angka yang ditebak benar maka permainan akan berakhir, namun bila angka yang ditebak salah maka program akan menampilkan “Lebih kecil” jika angka yang ditebak oleh pemain lebih besar dari yang ditentukan program dan “Lebih besar” jika angka yang ditebak oleh pemain lebih kecil dari yang ditentukan program. Nantinya program akan menunjukkan skor yang didapatkan oleh pemain berdasarkan seberapa cepat dia menebak angka acak tersebut dengan benar. Selanjutnya, permainan yang kedua adalah Diner Dash. Diner Dash adalah permainan mengantar makanan sesuai dengan urutan prioritasnya. Pada permainan ini ada tiga *command* valid yaitu COOK untuk memasak makanan, SERVE untuk menyajikan makanan, dan SKIP untuk tidak melakukan apa apa namun terhitung telah menyelesaikan satu putaran permainan. Di awal, permainan akan menampilkan daftar pesanan yang berisikan makanan, durasi memasak, ketahanan, harga dengan 3 buah antrian, daftar makanan yang dimasak yang berisikan makanan dan sisa durasi memasak, dan daftar makanan yang disajikan yang berisikan makanan dan sisa ketahanan masakan. Apabila pemain memasukkan command “COOK” untuk salah satu ID makanan maka pesanan tersebut akan masuk ke dalam daftar makanan yang dimasak. Ketika nanti durasi memasak mencapai 0 barulah pemain bisa memasukkan command “SERVE” untuk ID makanan tersebut, dan nantinya pesanan itu akan masuk ke dalam daftar makanan yang disajikan. Setelah itu barulah pesanan selesai. Permainan akan berakhir dan pemain dinyatakan menang apabila pemain dapat menyelesaikan 15 pesanan sedangkan pemain dinyatakan kalah apabila terdapat 7 pesanan pada daftar makanan yang dimasak. Nantinya di akhir permainan akan ada skor yang diambil dari total uang yang diterima pemain.

BNMO juga kami program untuk memiliki permainan tambahan yaitu Jari Bocil. Permainan yang terinspirasi dari permainan tradisional dengan menggunakan 10 jari yang dimodifikasi. Dalam permainan ini, pemain akan berhadapan dengan komputer untuk membuat

jumlah jari di kedua tangan lawan menjadi sama sama 5. Permainan akan berakhir jika jari di kedua tangan pemain atau komputer berjumlah sama sama 5.

Pengerjaan tugas besar ini membantu kami untuk memahami lebih lagi tentang bahasa C dan mengaplikasikan materi materi dasar yang diajarkan di kelas. Tugas besar ini bermanfaat bagi kami karena membuat kami belajar untuk menemukan ide dan alur dari membuat program hingga selesai.

2 Penjelasan Tambahan Spesifikasi Tugas

Pada pembuatan aplikasi BNMO terdapat satu game yang dibuat oleh kelompok kami bernama “Jari_Bocil”, game ini merupakan algoritma yang dapat disebut unik karena hanya dimiliki oleh kelompok kami. Secara singkat, *game* ini merupakan representasi bentuk digital dari permainan masa kecil kita sewaktu SD yang pada prakteknya menggunakan jari sebagai alat utama untuk bermain. Penjelasan lebih lanjut terkait spesifikasi *game* dan

2.1 Spesifikasi Fitur Bonus - Game Jari Bocil

Deskripsi Permainan Asli nya:

Permainan ini sejatinya merupakan permainan antara dua orang pemain. Permainan ini dimulai dengan kedua pemain mengeluarkan kedua tangannya dengan masing-masing tangan mengangkat satu buah jarinya. Selanjutnya diantara kedua pemain akan ada yang bermain terlebih dahulu dan ada yang bermain terakhir. Cara bermain nya sangatlah mudah, kamu hanya perlu menyerang tangan lawan yang ingin diserang menggunakan salah satu tanganmu untuk menyentuh lawan. Setelah itu, jumlah jari lawan di tangan yang kamu serang akan bertambah sesuai dengan jumlah jari pada tangan yang kamu gunakan untuk menyerang. Karena jumlah jari maksimal yang ada di masing-masing tangan hanya ada 5 buah, maka dari itu, setiap penjumlahan yang hasilnya sama dengan lima akan menjadi habis. Selain itu, setiap penjumlahan jari hasil penyerangan berjumlah lebih dari lima, akan dihitung mulai kembali dari satu. Jika dituliskan dalam bentuk fungsi, maka hasil penyerangan akan memiliki fungsi $f(x) = (x + n) \bmod 5$. Variabel x disini menandakan jumlah jari lawan yang diserang dan n menunjukkan jumlah jari pemain yang menyerang. Pemain bisa memilih untuk tidak menyerang lawan, namun pemain harus melakukan pergantian jumlah jari yang ada pada tangan kanan dan kiri nya dengan total jumlah jari yang sama. Sebagai contoh, jika pemain memiliki dua jari pada tangan kanan nya dan empat jari pada tangan kiri nya, pemain dapat menukar nya menjadi tiga jari pada masing masing tangan nya. Di akhir permainan, kondisi salah seorang pemain dikatakan menang ialah jika pemain tersebut dapat membuat kedua tangan lawan tidak memiliki jari lagi, dengan kata lain jumlah jari pada kedua tangan lawan sama dengan nol.

Implementasi dalam Program Game:

STEI- ITB	IF2111-TB1-03-01	Halaman 5 dari 35 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.		

Permainan ini cukup *simple* namun *tricky* dalam implementasinya karena harus menggunakan ADT yang disebutkan pada spesifikasi. Ketika mengimplementasikan permainan ini, kami tidak sepenuhnya memasukkan seluruh kerangka permainan. Kami memilih untuk menghilangkan opsi menukar jumlah jari kanan dan kiri pemain demi kemudahan pembuatan algoritma. Selain itu, kami juga mengganti pemain kedua menjadi komputer, sehingga dalam permainan ini hanya akan dilakukan pertarungan antara pemain (*user*) dengan komputer. Selain itu, konsep permainan kurang lebih akan sama. Pemain dan komputer memulai permainan dengan satu jari pada masing-masing tangan mereka. Selanjutnya, pemain akan menyerang terlebih dahulu (pemain akan selalu mendapat giliran pertama) baru kemudian komputer. Selanjutnya permainan akan berjalan sesuai aturan hingga diantara komputer ataupun pemain ada yang memenangkan permainan.

Program ini akan dimulai dengan memberikan output berupa kondisi jari di masing-masing tangan pemain dan komputer. Hal ini bertujuan agar pemain dapat menentukan strategi untuk menyerang. Lalu program akan menerima input dari pemain tentang tangan mana yang akan digunakan untuk menyerang, dan tangan mana yang akan diserang. Input akan berupa kata (“kanan/kiri”) dan akan mengulang input jika tidak valid (program akan mengulang sampai input user valid). Ada kasus khusus dimana pemain meminta untuk menyerang lawan menggunakan tangan yang sudah habis jari nya, pada kasus ini, akan muncul pesan bahwa jari pada tangan tersebut sudah habis dan penyerangan otomatis dilakukan menggunakan tangan yang satu lagi. Program menerima input dalam bentuk kata bukan tanpa alasan, namun untuk memenuhi spesifikasi program bonus yang diminta. Namun demi mempermudah pemrosesan, kami menambah sebuah prosedur bernama *PlayerInput()* untuk mengkonversi input yang bertipe bentukan kata menjadi sebuah integer 0 dan 1 (akan dijelaskan lebih lanjut nanti). Setelah menyerang, akan dilakukan perhitungan hasil penyerangan yang kemudian akan mengubah nilai jari di tangan kanan/kiri komputer.

Selanjutnya ialah giliran komputer untuk menyerang. Kami menggunakan function *srand()* dan *rand()* untuk *men-generate* sebuah *random number* dengan waktu saat ini sebagai *seed-nya*. Kedua fungsi ini kami import dari library “*stdlib.h*” dan “*time.h*”. Setelah menghasilkan sebuah *random number*, program akan melakukan operasi mod dengan 2 kepada *random number* tersebut. Tujuan nya ialah untuk menentukan apakah komputer akan menyerang menggunakan tangan mana dan tangan pemain mana yang akan diserang. Tentunya, kami juga melakukan penanganan kasus khusus, dimana tangan yang dipilih komputer sudah habis jari-nya sehingga program akan otomatis mengalihkan penyerangan pada tangan yang satunya. Setelah itu, komputer akan mengeksekusi penyerangan terhadap pemain dan akan dilakukan kalkulasi hasil penyerangan. Setelah pemain dan komputer melakukan penyerangan, akan kembali ditampilkan kondisi jari terkini dari pemain dan komputer selama belum ada kubu yang jari pada

kedua tangan nya habis. Jika ada, maka permainan akan selesai dan akan dimunculkan pesan bahwa pemain/komputer memenangkan permainan.

Prosedur Tambahan PlayerInput():

Prosedur ini merupakan *tool* untuk mengkonversi input dari user yang bertipe kata menjadi sebuah integer 0 dan 1. Proses nya mudah saja, cukup memanggil prosedur *STARTWORD()* dari ADT mesin_kata yang akan menerima input dari user dan menyimpan input dalam bentuk kata. Karena input yang diterima hanya kata “kanan” dan “kiri”, maka tidak perlu ada penanganan lagi dimana input yang dimasukkan lebih dari satu kata. Selanjutnya *currentword* hasil input akan dimasukkan ke sebuah prosedur *WordCompareString()* yang akan membandingkan kata yang di input dengan nilai string yang diharapkan. Jika kata yang di input merupakan kata “kanan”, maka akan mengembalikan nol sebagai nilai akhir. Namun jika kata yang di input merupakan kata “kiri” maka akan mengembalikan satu sebagai nilai akhir. Nilai nol dan satu ini melambangkan tangan mana yang akan digunakan sesuai definisi pada program.

Algoritma:

Berikut ialah algoritma prosedur yang digunakan untuk mengimplementasikan permainan ini dalam sebuah program.

```
void Jari_Bocil(){
    const int MaxFinger = 5;
    const int RIGHT = 0;
    const int LEFT = 1;
    int PlayerRF = 1, PlayerLF = 1, ComputerRF = 1, ComputerLF = 1;
    int PIHand, PIAttack, CIHand, CIAttack, i = 0;
    printf("Game Jari Bocil dimulai. Siapkan kedua tangan Anda untuk
    menyerang lawan!\n");
    while ((ComputerRF != 0 || ComputerLF != 0) && (PlayerRF != 0 ||
    PlayerLF != 0)) {
        if /*giliran player, i mod 2 = 0 */((i%2) == 0) {
            /*giliran player untuk menyerang, asumsi player selalu bermain
            duluan*/
            printf("Kondisi-mu sekarang:\n");
            printf("Jumlah jari tangan kanan: %i\n", PlayerRF);
            printf("Jumlah jari tangan kiri: %i\n", PlayerLF);
            printf("\n");
            printf("Kondisi-Komputer sekarang:\n");
            printf("Jumlah jari tangan kanan: %i\n", ComputerRF);
            printf("Jumlah jari tangan kiri: %i\n", ComputerLF);
            printf("\n");
            printf("Silahkan pilih tangan untuk menyerang (kanan/kiri):
            \n"); /*input tangan menyerang*/
            PIHand = PlayerInput();
```

```

        while (PIHand !=0 && PIHand !=1) { /*checker input tangan yang
menyerang valid*/
            printf("Input tidak valid! Input hanya bisa berupa
kanan/kiri\n");
            printf("Silahkan pilih tangan untuk menyerang (kanan/kiri):
\n");
            PIHand = PlayerInput();
        }
        /*Checker agar tidak menyerang dengan tangan yang sudah tidak
ada jari-nya*/
        if (PIHand == 0 && PlayerRF == 0) {
            printf("Tangan kanan kamu sudah habis jari-nya, penyerangan
akan dilakukan dengan tangan kiri!\n");
            PIHand = 1;
        }
        else if (PIHand == 1 && PlayerLF == 0) {
            printf("Tangan kiri kamu sudah habis jari-nya, penyerangan akan
dilakukan dengan tangan kanan!\n");
            PIHand = 0;
        }
        printf("Silahkan pilih tangan lawan untuk diserang
(kanan/kiri): \n"); /*input tangan diserang*/
        PIAttack = PlayerInput();
        while (PIAttack !=0 && PIAttack!=1) { /*checker input tangan
yang diserang valid*/
            printf("Input tidak valid! Input hanya bisa berupa
kanan/kiri\n");
            printf("Silahkan pilih tangan lawan untuk diserang
(kanan/kiri): \n");
            PIAttack = PlayerInput();
        }
        /*Perhitungan hasil setelah penyerangan*/
        if (PIHand == RIGHT && PIAttack == RIGHT) {ComputerRF =
(ComputerRF + PlayerRF) % MaxFinger;}
        else if (PIHand == RIGHT && PIAttack == LEFT) {ComputerLF =
(ComputerLF + PlayerRF) % MaxFinger;}
        else if (PIHand == LEFT && PIAttack == RIGHT) {ComputerRF =
(ComputerRF + PlayerLF) % MaxFinger;}
        else if (PIHand == LEFT && PIAttack == LEFT) {ComputerLF =
(ComputerLF + PlayerLF) % MaxFinger;}
        }
        else /*giliran komputer, i mod 2 = 1*/ {
            /*giliran computer untuk menyerang*/
            srand(time(NULL));
            int CIHand = rand()%2;
            int CIAttack = rand()%2;
            /*Checker agar tidak menyerang dengan tangan yang sudah tidak
ada jari-nya*/

```



```

        if (CIHand == 0 && ComputerRF == 0) {CIHand = 1;}
        else if (CIHand == 1 && ComputerLF == 0) {CIHand = 0;}
        if (CIHand == 0) {printf("Tangan yang dipilih komputer untuk
menyerang: KANAN\n");}
        else if (CIHand == 1) {printf("Tangan yang dipilih komputer
untuk menyerang: KIRI\n");}
        if (CIAttack == 0) {printf("Tangan yang dipilih komputer untuk
diserang: KANAN\n");}
        else if (CIAttack == 1) {printf("Tangan yang dipilih komputer
untuk diserang: KIRI\n");}
        printf("\n"); /*Ngasih jarak aja biar enak liatnya*/
        /*Penghitungan hasil setelah penyerangan*/
        if (CIHand == RIGHT && CIAttack == RIGHT) {PlayerRF = (PlayerRF
+ ComputerRF) % MaxFinger;}
        else if (CIHand == RIGHT && CIAttack == LEFT) {PlayerLF =
(PlayerLF + ComputerRF) % MaxFinger;}
        else if (CIHand == LEFT && CIAttack == RIGHT) {PlayerRF =
(PlayerRF + ComputerLF) % MaxFinger;}
        else if (CIHand == LEFT && CIAttack == LEFT) {PlayerLF =
(PlayerLF + ComputerLF) % MaxFinger;}
    }
    i++;
}
if (ComputerRF == 0 && ComputerLF == 0) { /*kondisi player menang*/
    printf("Selamat Anda telah memenangkan permainan, jangan
sombong yaaa!\n"); /*pesan player menang, komputer kalah*/
    /*print score*/
}
else { /*kondisi komputer menang*/
    printf("Yaaah masa kalah sama komputer, Ayok coba lagi!\n");
/*pesan komputer menang, player kalah*/
    /*print score*/
}
}
}

```

3 Struktur Data (ADT)

Pada pembuatan aplikasi BNMO ini, kami menggunakan beberapa struktur data untuk menyelesaikan persoalan-persoalan pada Tugas Besar yaitu ADT Array, ADT Mesin Kata, ADT Mesin Karakter, dan ADT Queue.

3.1 ADT Array

Pada ADT Array, kami hanya menggunakan dua *file* yang terdiri dari satu *file header* dan satu *file implementasi* dari *file header* berupa file c. Dalam implementasi *array* ini kami menggunakan konsep *array* dinamis yang telah diajarkan di kelas maupun saat praktikum. Adapun fungsi yang kami tambahkan kedalam *file array* yang kami buat namun tidak ada pada *file array* dinamis saat praktikum yaitu fungsi InsertLast, InsertFirst, DeleteLast, DeleteFirst,

Printarray, Reversearray, dan Copyarray. Dalam ADT Array strukturnya terdiri dari sebuah *pointer* A yang akan menyimpan nilai bertipe *word*, IdxType bertipe *integer*, Capacity bertipe *integer*, dan Neff menyimpan nilai efektif elemen bertipe *integer*. ADT Array membantu menyimpan *game* yang dimiliki oleh *user*. Alasan digunakannya ADT Array ini karena dapat digunakan sebagai tempat menyimpan *game* yang dimiliki oleh *user*. ADT Array diimplementasikan sebagai ADT Array dengan nama *file header* “Array.h”

3.2 ADT Mesin Kata

Pada ADT Mesin Kata terdapat TabWord yang merupakan *container* penyimpan kata, dengan indeks [0..NMax-1] bertipe *character* dan *Length* bertipe *integer*. Terdapat juga nilai *extern* EndWord bertipe *boolean* dan nilai *extern* currentWord bertipe *word*. ADT Mesin Kata memiliki 8 fungsi primitif, yaitu IgnoreBlanks() untuk mengabaikan satu atau beberapa *container* kosong, STARTWORD() untuk memulai membaca kata di suatu *container* dengan meminta *input* dari *user*, STARTWORDFILE() untuk memulai membaca *file* yang telah dimasukkan oleh *user* di suatu *container*, ADVWORD() untuk melanjutkan ke kata selanjutnya bila *container* kosong, CopyWord() untuk menyalin kata dan disimpan dalam current word, WordToString() untuk menerima kata dalam bentuk *word* lalu mengubahnya ke dalam bentuk *string*, WordCompareString() untuk membandingkan kata dengan *string* dan akan menghasilkan *true* jika sama, dan PrintKata() untuk menuliskan tipe bentukan kata ke layar. ADT Mesin Kata membantu untuk membaca karakter atau kata pada program. Alasan digunakannya ADT Mesin Kata ini karena dapat mempermudah program dalam membaca sebuah karakter atau kata. ADT Mesin Kata diimplementasikan sebagai ADT Mesin Kata dengan nama *file header* “Mesin_Kata.h”

3.3 ADT Mesin Karakter

Dalam ADT Mesin Karakter strukturnya terdiri dari sebuah nilai *extern* currentChar (Current Character) yang bertipe *character* dan nilai *extern* EOP (End of Pita) yang bertipe *boolean*. Pada ADT Mesin Karakter memiliki 5 fungsi primitif, yaitu START() untuk mulai membaca karakter di suatu pita, STARTFILE() untuk membaca karakter didalam pita yang berupa *file*, ADV() untuk memajukan pita satu karakter, GetCC() untuk mengirimkan *current character*, dan IsEOP() untuk mengirimkan *true* apabila *current character* adalah *mark*. ADT Mesin Karakter membantu untuk membaca atau memasukkan karakter ke dalam program. Alasan digunakannya ADT Mesin Karakter ini karena dapat membantu program dalam membaca karakter, menggeser atau memajukan karakter, memasukkan karakter, dan ADT Mesin Karakter dibutuhkan untuk mesin kata. ADT Mesin Karakter diimplementasikan sebagai ADT Mesin Karakter dengan nama *file header* “Mesin_Karakter.h”

3.4 ADT Queue

Dalam ADT Queue strukturnya terdiri dari *buffer* yang merupakan *container* penyimpan kata dengan indeks [CAPACITY] bertipe *word*, idxHead bertipe *integer*, dan idxTail bertipe *integer*. Pada ADT Queue memiliki 7 fungsi primitif, yaitu CreateQueue() untuk alokasi dan membuat sebuah *queue* kosong, isEmpty() akan mengirim *true* jika *queue* kosong, isFull()

akan mengirim *true* jika tabel penampungan elemen *queue* sudah penuh, *length()* untuk mengirimkan banyaknya elemen *queue*, *enqueue()* untuk menambahkan *value* pada *queue*, *dequeue()* untuk menghapus *value* pada *queue*, dan *displayQueue()* untuk menuliskan isi *queue* dengan traversal, *queue* ditulis di antara kurung siku antara dua elemen dipisahkan dengan separator “koma”, tanpa tambahan karakter di depan, di tengah, atau di belakang, termasuk spasi dan *enter*. Alasan digunakannya ADT Queue ini karena dapat membantu program dalam membuat antrian yang dapat menambahkan, mengurangi, menampilkan *queue*, dan menjadi parameter untuk user memainkan game. ADT Queue diimplementasikan sebagai ADT Queue dengan file *header* “Queue.h”

3.5 ADT QueueDD

Dalam ADT QueueDD strukturnya terdiri dari *buffer* yang merupakan *container* penyimpan kata dengan indeks [CAPACITY] bertipe *food* yang menyimpan id, durasi, ketahanan, harga bertipe *integer*. Terdapat juga *idxHead* bertipe *integer*, dan *idxTail* bertipe *integer*. Pada ADT QueueDD sama dengan ADT Queue memiliki 7 primitif, memiliki penjelasan fungsi yang sama juga dengan ADT Queue, yaitu *MakeQueue()*, *IsEmpty()*, *IsFull()*, *Len()*, *Enqueue()*, *DequeueAt()*, dan *DisplayQueue()*. Terdapat satu perbedaan dengan ADT Queue adalah *DequeueAt()*, *DequeueAt()* digunakan untuk menghapus *value* pada *queue* dengan *idx* sesuai dengan inputan *user*. Alasan digunakannya ADT QueueDD ini karena dapat membantu program dalam membuat antrian yang dapat menambahkan, mengurangi pada inputan indeks sesuai *user*, dan menampilkan *queue* pada *games* Dinner Dash. ADT QueueDD diimplementasikan sebagai ADT QueueDD dengan file *header* “QueueDD.h”.

4 Program Utama

Program utama dengan file “main.c” meng-include semua file ADT yang ada. Program utama dari aplikasi BNMO dimulai dengan menampilkan *main menu* yang berisikan *welcome page* dan menampilkan pilihan *menu* untuk *start* dan *load game*. Dengan Prosedur CHOOSEMODE user bisa memilih untuk melakukan start atau load game. Setelah itu, *main menu* akan menerima *input* dari *user* untuk lanjut memainkan permainan baru atau melanjutkan permainan yang sudah pernah dimainkan sebelumnya. Apabila user memilih start maka program akan melanjutkan dengan mengakses file config, namun jika user memilih load maka program akan melanjutkan dengan mengakses file game user sebelumnya. Selanjutnya program akan menampilkan fitur fitur yang ada yaitu CREATEGAME, LISTGAME, DELETGAME, QUEUEGAME, PLAYGAME, SKIPGAME, SAVE, HELP, dan QUIT lalu program akan terus berjalan sampai user memilih fitur QUIT.

5 Algoritma-Algoritma Menarik

Di dalam program yang kami buat, terdapat 2 buah algoritma yang bisa dikategorikan sebagai unik. Dua buah algoritma yang unik ini adalah prosedur FITURE dan prosedur

CHOOSEMODE. Dua buah algoritma ini kami kategorikan sebagai kategori algoritma menarik karena kami tidak menggabungkan kedua fitur ini di “main.c” tapi kami pisah di “console.c”

5.1 Prosedur Fiture

```
void FUTURE() {
    printf("-----\n");
    printf("|  Ketik HELP untuk melihat penjelasan fitur  |\n");
    printf("-----\n");
    printf("\nFitur pada BNMO yang bisa anda pilih :\n");
    printf(">> CREATE GAME\n");
    printf(">> LIST GAME\n");
    printf(">> DELETE GAME\n");
    printf(">> QUEUE GAME\n");
    printf(">> PLAY GAME\n");
    printf(">> SKIP GAME\n");
    printf(">> SAVE\n");
    printf(">> HELP\n");
    printf(">> QUIT\n");
}
```

Algoritma prosedur Fiture berguna menampilkan seluruh fitur BNMO yang dapat dipilih oleh *user* untuk memilih fitur pada BNMO yang ingin dijalankan. Algoritma ini menjadi sebuah algoritma menarik karena berada dalam file “console.c” bukan digabung di dalam file “main.c”.

5.2 Prosedur ChooseMode

```
void CHOOSEMODE(int *mode, char *file){
    printf("Silahkan memilih mode START/LOAD: "); STARTWORD2();
    PrintKata(currentWord); printf("\n"); // HAPUS
    if(WordCompareString(currentWord, "START") &&
currentWord.Length==5) {
        *mode=1;
    } else if (WordCompareString(currentWord, "LOAD")){
        ADVWORD2(); PrintKata(currentWord); printf("\n"); //HAPUS
        WordToString(currentWord, file);
        *mode=2;
    } else {
        printf("Ketikkan mode yang benar START/LOAD\n\n");
    }
}
```

Algoritma prosedur CHOOSEMODE berguna untuk memilih mode awal apakah player memilih *START* atau *LOAD*. Algoritma ini menjadi sebuah algoritma menarik karena berada dalam *file* “console.c” bukan digabung di dalam *file* “main.c”

6 Data Test

6.1 Data Test MAIN MENU

Pada tes ini dilakukan pengujian untuk memastikan bahwa program dapat berjalan dan menampilkan tampilan awal dari aplikasi BNMO. Pada tampilan awal aplikasi BNMO ini menampilkan *main menu*. Adapun cara melakukan kompilasi program ini dengan mengetikkan “gcc main.c ADT/QueueDD/QueueDD.c ADT/Array/Array.c ADT/Queue/Queue.c ADT/MesinKata/mesin_karakter.c ADT/MesinKata/mesin_kata.c Procedure/Fungsi_Kecil.c Procedure/Game.c console.c -o BNMO” pada terminal.



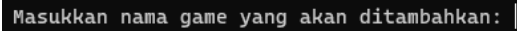
Gambar 1. Tampilan Judul dan Main Menu

6.2 Data Test START

Pada tes ini akan menjalankan *input start* serta memastikan fungsi tersebut dapat berjalan. Adapun pada gambar 2 merupakan tampilan saat akan memasukkan *input* dan gambar 3 merupakan hasil tampilan setelah memasukkan *input*.

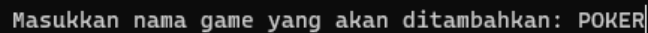
6.4 Data Test CREATE GAME

Pada tes ini akan menjalankan fitur *create game* yang akan dimasukkan pada tampilan gambar 3. Setelah pengguna masuk ke fitur *create game*, akan muncul tampilan sesuai gambar 5 dan diharapkan pengguna memasukkan nama *game* yang akan dibuat. Setelah berhasil *create game* akan muncul tampilan sesuai gambar 7.



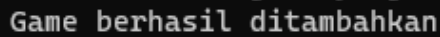
```
Masukkan nama game yang akan ditambahkan: |
```

Gambar 5. Tampilan Setelah Memasukkan *Create Game* Pada Fitur BNMO



```
Masukkan nama game yang akan ditambahkan: POKER|
```

Gambar 6. Tampilan Saat Memasukkan Nama *Game* Yang Akan Ditambahkan

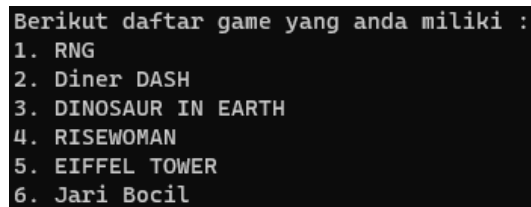


```
Game berhasil ditambahkan
```

Gambar 7. Tampilan Setelah Memasukkan Nama *Game*

6.5 Data Test LIST GAME

Pada tes ini akan menjalankan fitur *list game*. Pengguna akan memasukkan *list game* pada tampilan di gambar 3. Setelah melakukan hal tersebut akan muncul tampilan sesuai gambar 8.



```
Berikut daftar game yang anda miliki :  
1. RNG  
2. Diner DASH  
3. DINOSAUR IN EARTH  
4. RISEWOMAN  
5. EIFFEL TOWER  
6. Jari Bocil
```

Gambar 8. Tampilan Setelah Memasukkan *List Game* Pada Fitur BNMO

6.6 Data Test DELETE GAME

Pada tes ini akan menjalankan fitur *delete game*. Pengguna akan memasukkan *delete game* pada tampilan gambar 3. Setelah memasukkan *delete game* akan muncul tampilan sesuai gambar 9. Untuk menghapus game yang diinginkan, pengguna akan memasukkan nomor *game* yang tersedia. Jika *game* tersebut merupakan *game default* akan muncul tampilan sesuai gambar 10. Namun, apabila *game* tersebut bukan *game default* maka akan muncul sesuai gambar 11.

```

Berikut daftar game yang anda miliki :
1. RNG
2. Diner DASH
3. DINOSAUR IN EARTH
4. RISEWOMAN
5. EIFFEL TOWER
6. Jari Bocil

Masukkan nomor game yang ingin dihapus :

```

Gambar 9. Tampilan Setelah Memasukkan Delete Game Pada Fitur BNMO

```

Game default tidak bisa dihapus!

```

Gambar 10. Tampilan Jika Game Default Dihapus

```

Game berhasil dihapus!

```

Gambar 11. Tampilan Jika Game Yang Dihapus Bukan Game Default

6.7 Data Test QUEUE GAME

Pada tes ini akan menjalankan fitur *queue game*. Pengguna akan memasukkan *queue game* pada tampilan gambar 3. Setelah memasukkan *queue game* akan muncul tampilan sesuai gambar 12 dan jika masih belum ada *queue game* maka pada tampilan antrian akan kosong. Pada tampilan tersebut pengguna diharapkan memasukkan nomor *game* yang akan dimainkan. Setelah berhasil memasukkan nomor, akan muncul tampilan sesuai gambar 13. Jika pengguna mengetikkan kembali *queue game* maka akan muncul tampilan sesuai gambar 14.

```

Berikut daftar game yang anda miliki :
1. RNG
2. Diner DASH
3. DINOSAUR IN EARTH
4. RISEWOMAN
5. EIFFEL TOWER
6. Jari Bocil

Berikut adalah daftar antrian game-mu:
Nomor Game yang mau ditambahkan ke antrian:

```

Gambar 12. Tampilan Setelah Memasukkan Queue Game Pada Fitur BNMO

```

Nomor Game yang mau ditambahkan ke antrian: 1
Game RNG dimasukkan kedalam antrian.

```

Gambar 13. Tampilan Memasukkan Nomor Game Ke Antrian


```

Berikut daftar game yang anda miliki :
1. RNG
2. Diner DASH
3. DINOSAUR IN EARTH
4. RISEWOMAN
5. EIFFEL TOWER
6. Jari Bocil

Berikut adalah daftar antrian game-mu:
1. RNG

Nomor Game yang mau ditambahkan ke antrian: |

```

Gambar 14. Tampilan Antrian Game

6.8 Data Test *PLAY GAME*

Pada menjalankan fitur play game, pengguna harus memasukkan game terlebih dahulu dengan fitur *queue game*. Jika belum terdapat game di antrian maka akan muncul tampilan seperti gambar 15. Akan tetapi jika terdapat antrian *game* pada fitur *play game* maka akan muncul tampilan sesuai gambar 16.

```

Berikut adalah daftar antrian game-mu:

Tidak ada game di dalam antrian untuk dimainkan, silahkan masukkan game kedalam antrian terlebih dahulu!

```

Gambar 15. Tampilan Saat Tidak Terdapat *Game* Untuk Dijalankan

```

-----
| Ketik HELP untuk melihat penjelasan fitur |
-----

Fitur pada BNMO yang bisa anda pilih :
>> CREATE GAME
>> LIST GAME
>> DELETE GAME
>> QUEUE GAME
>> PLAY GAME
>> SKIP GAME
>> SAVE
>> HELP
>> QUIT
Masukkan perintah: PLAY GAME

Berikut adalah daftar antrian game-mu:
1. RNG
2. Diner DASH
3. Jari Bocil

Loading...

```

Gambar 16. Tampilan Jika Terdapat Antrian *Game*

6.8.1 Data Test *GAME RNG*

Pada tes ini akan dijalankan game RNG. Pada game ini pengguna akan menebak angka dari 1-100 dan akan memasukkan angka tersebut sesuai tampilan gambar 17. Apabila *game* tersebut selesai, akan muncul tampilan gambar 18.

```

Game RNG dimulai. Uji keberuntungan anda dengan menebak X. X bilangan 0 s.d 100
Tebakan: |

```

Gambar 17. Tampilan Awal Permainan RNG

```

Game RNG dimulai. Uji keberuntungan anda dengan menebak X. X bilangan 0 s.d 100
Tebakan: 50
X Lebih Besar
Tebakan: 60
X Lebih Besar
Tebakan: 70
X Lebih Kecil
Tebakan: 65
X Lebih Besar
Tebakan: 68
X Lebih Besar
Tebakan: 69
Selamat, anda benar menebak X yaitu 69

```

Gambar 18. Tampilan Saat Angka Berhasil Ditebak

6.8.2 **Data Test GAME DINER DASH**

Pada tes ini akan menjalankan game Diner Dash. Tampilan awal game ini akan seperti gambar 19. Untuk memasak makanan pada *game* ini dapat menggunakan *command cook* sesuai gambar 20. Untuk menghadirkan makanan tersebut dapat menggunakan *command serve* dan akan muncul tampilan seperti gambar 21. *Game* akan selesai jika antrian makanan sudah melebihi dari 7 makanan yang tampilannya sesuai gambar 22.

```

Selamat Datang di Diner Dash!

Saldo : 0

Daftar Pesanan
Makanan | Durasi memasak | Ketahanan | Harga
-----|-----|-----|-----
M0      | 2              | 3         | 16334
M1      | 1              | 5         | 25724
M2      | 4              | 4         | 36962

Daftar Makanan yang sedang dimasak
Makanan | Sisa durasi memasak
-----|-----
|
Daftar Makanan yang dapat disajikan
Makanan | Sisa ketahanan makanan
-----|-----
|

MASUKKAN COMMAND: |

```

Gambar 19. Tampilan Awal *Game* Diner Dash

```

MASUKKAN COMMAND: COOK M0
Berhasil memasak M0

Saldo : 0

Daftar Pesanan
Makanan | Durasi memasak | Ketahanan | Harga
-----|-----|-----|-----
M0      | 2              | 3         | 16334
M1      | 1              | 5         | 25724
M2      | 4              | 4         | 36962
M3      | 5              | 1         | 38145

Daftar Makanan yang sedang dimasak
Makanan | Sisa durasi memasak
-----|-----
M0      | 2

Daftar Makanan yang dapat disajikan
Makanan | Sisa ketahanan makanan
-----|-----

```

Gambar 20. Tampilan Saat Memasak

```

MASUKKAN COMMAND: SERVE M0
Berhasil mengantarkan M0

Saldo : 16334

Daftar Pesanan
Makanan | Durasi memasak | Ketahanan | Harga
-----|-----|-----|-----
M1      | 1              | 5         | 25724
M2      | 4              | 4         | 36962
M3      | 5              | 1         | 38145
M4      | 2              | 3         | 19961
M5      | 2              | 1         | 21942
M6      | 3              | 2         | 42391

Daftar Makanan yang sedang dimasak
Makanan | Sisa durasi memasak
-----|-----
M2      | 3

Daftar Makanan yang dapat disajikan
Makanan | Sisa ketahanan makanan
-----|-----
M1      | 3

```

Gambar 21. Tampilan Saat Menghidangkan Makanan

Game over, antrian sudah melebihi 7

Gambar 22. Tampilan Saat *Game* Selesai

6.8.3 **Data Test GAME JARI BOCIL**

Pada tes ini akan menjalankan *game* Jari Bocil. Tampilan awal *game* ini akan menampilkan kondisi awal jari komputer dan pemain seperti gambar 23. Selanjutnya program akan meminta input tangan yang akan digunakan untuk menyerang dan tangan yang diserang seperti pada gambar 24. Program juga akan menampilkan pilihan tangan yang dipilih komputer untuk menyerang dan diserang seperti pada gambar 25. Untuk

penanganan kasus khusus dimana pemain memilih tangan yang jari nya sudah habis, maka akan otomatis dialihkan ke tangan yang satunya dan muncul pesan seperti pada gambar 26. Terakhir jika pemain menang ataupun kalah dalam permainan, akan muncul pesan yang akan mengakhiri permainan seperti pada gambar 27.

```
Game Jari Bocil dimulai. Siapkan kedua tangan Anda untuk menyerang lawan!
Kondisi-mu sekarang:
Jumlah jari tangan kanan: 1
Jumlah jari tangan kiri: 1

Kondisi-Komputer sekarang:
Jumlah jari tangan kanan: 1
Jumlah jari tangan kiri: 1
```

Gambar 23. Tampilan Saat *Game* Dimulai

```
Silahkan pilih tangan untuk menyerang (kanan/kiri):
Kanan
Input tidak valid! Input hanya bisa berupa kanan/kiri
Silahkan pilih tangan untuk menyerang (kanan/kiri):
kanan
Silahkan pilih tangan lawan untuk diserang (kanan/kiri):
kiri
```

Gambar 24. Tampilan Saat *Game* Menerima *Input* dari Pemain

```
Tangan yang dipilih komputer untuk menyerang: KIRI
Tangan yang dipilih komputer untuk diserang: KANAN
```

Gambar 25. Tampilan Pilihan *Input* Komputer

```
Tangan kanan kamu sudah habis jari-nya, penyerangan akan dilakukan dengan tangan kiri!
```

Gambar 26. Tampilan Penanganan Kasus Khusus

```
Selamat Anda telah memenangkan permainan, jangan sombong yaaa!
```

```
Yaaah masa kalah sama komputer, Ayok coba lagi!
```

Gambar 27. Tampilan Saat *Game* Selesai

6.9 Data Test SKIP GAME

Pada tes ini akan menjalankan fitur *skip game*. Pengguna akan memasukkan *skip game* nomor game yang dilewati pada tampilan gambar 3. Jika belum terdapat antrian dan menginginkan skip game maka akan muncul tampilan seperti gambar 28. Jika terdapat antrian game dan menginginkan skip game akan muncul tampilan seperti gambar 29. Untuk *game* yang dilewati, program akan langsung menjalankan game setelahnya.

```
Masukkan perintah: SKIPGAME 1

Berikut adalah daftar antrian game-mu:

Tidak ada permainan lagi dalam daftar game-mu
```

Gambar 28. Tampilan jika belum terdapat antrian

```
Masukkan perintah: SKIPGAME 1

Berikut adalah daftar antrian game-mu:
1. Jari Bocil
2. Diner DASH
3. Jari Bocil
Berikut adalah daftar antrian game-mu:
1. Diner DASH
2. Jari Bocil

Loading...

Selamat Datang di Diner Dash!
```

Gambar 29. Tampilan saat sudah berhasil *skip game*

6.10 Data Test SAVE

Pada tes ini akan menjalankan fitur *save*. Pengguna akan memasukkan *save* pada tampilan gambar 3 beserta nama *file* dengan format *file* txt seperti gambar 30.

```
Masukkan perintah: SAVE PLAYER1.TXT
```

Gambar 30. Tampilan Fitur *Save*

6.11 Data Test HELP

Pada tes ini akan menjalankan fitur *help*. Pengguna akan memasukkan *help* pada tampilan gambar 3. Setelah memasukkan *help* akan muncul tampilan sesuai gambar 31.

```
FITUR-FITUR BNMO:
1. START      -> merupakan menu awal untuk memulai BNMO dengan pilihan game default.
2. LOAD       -> merupakan menu awal untuk membaca dan membuka file save yang berisikan history permainan dari player.
3. SAVE       -> merupakan menu untuk menyimpan data setelah adanya perubahan dari player.
4. CREATE GAME -> merupakan menu untuk menambahkan game baru pada daftar game.
5. LIST GAME  -> merupakan menu untuk menampilkan daftar game yang tersedia untuk player.
6. DELETE GAME -> merupakan menu untuk menghapus sebuah game dari daftar game, dengan syarat:
   game yang dihapus adalah game tambahan dan game yang terdapat pada queue saat itu tidak bisa dihapus.
7. QUEUE GAME -> merupakan menu untuk mendaftarkan game kedalam list queue game yang akan dimainkan oleh player.
8. PLAY GAME  -> merupakan menu untuk memainkan game.
9. SKIP GAME  -> merupakan menu untuk melewati game yang tidak ingin dimainkan, sebanyak yang diinginkan.
10. QUIT      -> merupakan menu untuk keluar dari BNMO.
```

Gambar 31. Tampilan Fitur *Help*

6.12 Data Test QUIT

Pada tes ini akan menjalankan fitur *quit*. Pengguna akan memasukkan *quit* pada tampilan gambar 3. Setelah memasukkan *quit* akan muncul tampilan sesuai gambar 33. Pada tampilan

tersebut pengguna dapat memilih untuk menyimpan *game* atau tidak dan setelah itu akan langsung keluar dari aplikasi

```
Masukkan perintah: QUIT
Apakah anda ingin save? (Y/N)
```

Gambar 32. Tampilan Fitur *Quit*

6.13 Data Test Command Lain

Pada tes ini apabila pengguna salah memasukkan perintah akan muncul *command* lain seperti gambar 33.

```
Masukkan perintah: ABCD
Command tidak dikenali, silahkan masukkan command yang valid.
```

Gambar 33. Tampilan *Command* Lain

7 Test Script

No.	Fitur yang Dites	Tujuan Testing	Langkah-Langkah Testing	Input Data Test	Hasil yang Diharapkan	Hasil yang Keluar
1.	Fitur MAIN MENU	Memeriksa apakah program bisa menampilkan tampilan awal dari aplikasi BNMO	Pada saat menjalankan program, akan keluar <i>main menu</i>	Data Test 6.1	Memunculkan <i>main menu</i> yang terdapat 2 pilihan yaitu <i>START</i> atau <i>LOAD GAME</i>	Sesuai yang diharapkan
2	Fitur START	Memeriksa apakah fitur <i>START GAME</i> bisa berjalan dengan baik dan permainan bisa dimulai dan dijalankan	Memasukkan <i>command START</i> pada <i>main menu</i>	Data Test 6.2	Memunculkan fitur pada BNMO yang dapat dipilih, kemudian akan diminta <i>input</i> fitur	Sesuai yang diharapkan
3.	Fitur LOAD	Memeriksa apakah permainan bisa di <i>load</i> atau fitur	Memasukkan <i>command LOAD</i> pada <i>main menu</i> untuk <i>file</i> yang sudah di <i>save</i>	Data Test 6.3	berhasil mengakses <i>file</i> yang telah disimpan oleh	Sesuai yang diharapkan

		<i>LOAD</i> bisa berhasil dijalankan			<i>user</i> sebelumnya	
4.	Fitur SAVE	Memeriksa apakah fitur <i>SAVE</i> bisa dijalankan	Memasukkan <i>command SAVE</i> untuk menyimpan <i>file game</i>	Data Test 6.10	<i>Game</i> berhasil disimpan dan dapat di <i>load</i>	Sesuai yang diharapkan
5.	Fitur CREATE GAME	Memeriksa apakah fitur <i>CREATE GAME</i> bisa dijalankan	Memasukkan <i>command CREATE GAME</i> , kemudian memasukkan inputan nama <i>game</i>	Data Test 6.4	<i>Game</i> baru berhasil ditambahkan pada <i>list game</i>	Sesuai yang diharapkan
6.	Fitur LIST GAME	Memeriksa apakah fitur LIST GAME bisa dijalankan dan menampilkan <i>list game</i> yang ada	Masukkan <i>command LIST GAME</i>	Data Test 6.5	memunculkan <i>list game</i> yang dimiliki	Sesuai yang diharapkan
7.	Fitur DELETE GAME	Memeriksa apakah fitur <i>DELETE GAME</i> bisa dijalankan dan menghapus <i>game</i> yang diinginkan	Memasukkan <i>input command DELETE GAME</i> , kemudian memasukkan inputan urutan <i>game</i> yang ingin dihilangkan	Data Test 6.6	Menghapus <i>game</i> yang telah dipilih sesuai ketentuan	Sesuai yang diharapkan
8.	Fitur QUEUE GAME	Memeriksa apakah fitur <i>QUEUE GAME</i> bisa dijalankan dan menambahkan antrean <i>game</i>	Memasukkan <i>input command QUEUE GAME</i> , kemudian memasukkan inputan urutan <i>game</i> yang ingin dimainkan.	Data Test 6.7	Berhasil memasukkan <i>game</i> yang dipilih ke dalam list	Sesuai yang diharapkan
9.	Fitur PLAY GAME	Memeriksa apakah fitur <i>PLAY GAME</i> bisa dijalankan	Memasukkan <i>input command PLAY GAME</i> untuk memainkan <i>game</i> yang terdapat pada antrian pertama	Data Test 6.8	<i>Game</i> yang terdapat pada <i>queue</i> pertama berhasil dimainkan	Sesuai yang diharapkan

		dan bisa memainkan <i>game</i>				
10.	Fitur SKIPGAME	Memeriksa apakah fitur <i>SKIP GAME</i> bisa dijalankan dapat melewati antrean sejumlah yang diinginkan oleh <i>user</i>	Memasukkan <i>input command SKIP GAME</i> untuk melewati <i>game</i> sebanyak yang diinginkan	Data Test 6.9	<i>Queue</i> permainan berhasil dilewati sebanyak yang diinginkan, lalu dilanjutkan dengan langsung memainkan <i>game</i> selanjutnya yang terdapat di <i>queue</i>	Sesuai yang diharapkan
11 .	Fitur QUIT	Memeriksa apakah fitur <i>QUIT</i> bisa dijalankan dan bisa keluar dari program	Memasukkan input <i>command QUIT</i> untuk keluar dari program	Data Test 6.12	memunculkan pilihan untuk <i>save file</i> , lalu keluar dari program	Sesuai yang diharapkan
12.	Fitur HELP	Memeriksa apakah fitur <i>HELP</i> bisa dijalankan dan menampilkan penjelasan tentang fungsi program	Memasukkan <i>input command HELP</i> untuk mendapatkan penjelasan mengenai fungsi program	Data Test 6.11	memunculkan penjelasan mengenai fungsi program	Sesuai yang diharapkan
13.	Fitur COMMAND LAIN	Memeriksa apakah fitur <i>COMMAND LAIN</i> bisa dijalankan dan bisa membedakan inputan <i>command</i> yang sesuai dengan tidak	Memasukkan <i>input command</i> dan memeriksa apakah <i>command</i> valid	Data Test 6.13	Memunculkan inputan <i>command</i> baru, bila <i>command</i> yang dimasukkan tidak valid	Sesuai yang diharapkan

14.	Fitur Game RNG	Memeriksa apakah <i>game</i> RNG bisa berjalan dengan baik	Memasukkan <i>game</i> RNG ke dalam <i>QUEUE GAME</i> , lalu input <i>PLAY GAME</i> untuk memainkan <i>game</i>	Data Test 6.8.1	<i>Game</i> berhasil dimainkan sesuai dengan ketentuan <i>game</i>	Sesuai yang diharapkan
15.	Fitur Game DINER DASH	Memeriksa apakah <i>game</i> <i>DINER DASH</i> bisa berjalan dengan baik	Memasukkan <i>game</i> dinner dash kedalam <i>QUEUE GAME</i> , lalu input <i>PLAY GAME</i> untuk memainkan <i>game</i>	Data Test 6.8.2	<i>Game</i> berhasil dimainkan sesuai dengan ketentuan <i>game</i>	Sesuai yang diharapkan
16.	Fitur Game Jari Bocil	Memeriksa apakah <i>game</i> JARI BOCIL bisa berjalan dengan baik	Memasukkan <i>game</i> jari bocil kedalam <i>QUEUE GAME</i> , lalu input <i>PLAY GAME</i> untuk memainkan <i>game</i>	Data Test 6.8.3	<i>Game</i> berhasil dimainkan sesuai dengan ketentuan <i>game</i>	Sesuai yang diharapkan

8 Pembagian Kerja dalam Kelompok

No.	Fitur/ADT	NIM Coder	NIM Tester
1	ADT Array	18221001	18221001, 18221022, 18221023, 18221026, 18221029
2	ADT Mesin Karakter	18221001	18221001, 18221022, 18221023, 18221026, 18221029
3	ADT Mesin Kata	18221001	18221001, 18221022, 18221023, 18221026, 18221029
4	ADT Queue	18221001	18221001, 18221022, 18221023, 18221026, 18221029
5	ADT QueueDD	18221001	18221001, 18221022, 18221023, 18221026, 18221029
6	Game	18221001, 18221023	18221001, 18221022, 18221023, 18221026, 18221029
7	Console Game	18221001, 18221022, 18221023, 18221026, 18221029	18221001, 18221022, 18221023, 18221026, 18221029
8.	Main Program	18221001, 18221022, 18221023, 18221026, 18221029	18221001, 18221022, 18221023, 18221026, 18221029

9 Lampiran

9.1 Deskripsi Tugas Besar 1

Command

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

a. START

START merupakan salah satu command yang dimasukkan pertama kali oleh pemain ke BNMO. Setelah menekan Enter, dibaca file konfigurasi default yang berisi list game yang dapat dimainkan.

b. LOAD <filename>

LOAD merupakan salah satu command yang dimasukkan pertama kali oleh pemain ke BNMO. Memiliki satu argumen yaitu filename yang merepresentasikan suatu *save file* yang ingin dibuka. Setelah menekan Enter, akan dibaca save file <filename> yang berisi list game yang dapat dimainkan, histori dan scoreboard game, lebih detailnya bisa dilihat pada Konfigurasi Sistem.

c. SAVE <filename>

SAVE merupakan command yang digunakan untuk menyimpan state game pemain saat ini ke dalam suatu file. Command SAVE memiliki satu argumen yang merepresentasikan nama file yang akan disimpan pada disk.

d. CREATEGAME

CREATEGAME merupakan command yang digunakan untuk menambahkan game baru pada daftar game. Spesifikasi game yang dibuat dapat dilihat pada section [Spesifikasi Game](#)

e. LISTGAME

LISTGAME merupakan command yang digunakan untuk menampilkan daftar game yang disediakan oleh sistem.

f. DELETGAME

DELETGAME merupakan command yang digunakan untuk menghapus sebuah game dari daftar game. Adapun aturan penghapusan game adalah:

- Game yang dapat dihapus hanya game yang dibuat secara custom oleh pengguna.
- 5 game pertama pada file konfigurasi tidak dapat dihapus.
- Game yang saat itu terdapat di dalam queue game tidak dapat dihapus.

g. QUEUEGAME

QUEUEGAME merupakan command yang digunakan untuk mendaftarkan permainan kedalam list. List dalam queue akan hilang ketika pemain menjalankan command **QUIT**.

h. PLAYGAME

PLAY GAME merupakan command yang digunakan untuk memainkan sebuah permainan. Game yang dimainkan adalah game dengan urutan pertama di antrian game. Ketika salah satu permainan dimulai, sistem akan menjalankan game sesuai pada section [Spesifikasi Game](#). Permainan selain yang dispesifikasikan pada [Spesifikasi Game](#) akan menampilkan pesan bahwa game tidak dapat dimainkan.

i. SKIPGAME <n>

SKIPGAME merupakan command yang digunakan untuk melewati permainan sebanyak n.

j. QUIT

Keluar dari program.

k. HELP

Bantuan command-command yang disebutkan di atas. Tampilan dan kata-kata dibebaskan.

l. COMMAND LAIN

Command-command lain selain yang disebutkan diatas tidak valid.

Keluar dari program.

Spesifikasi Game

1. RNG

BNMO tidak selalu menikmati *game* yang sudah pasti *outcome*-nya. Karena itu, ia suka dengan *game* yang melibatkan RNG (*Random number generator*). Berikut adalah spesifikasi *game* ini:

- Setiap permainan dimulai dengan program sudah menentukan sebuah angka acak X.
- Di setiap giliran, pemain diberi kesempatan menebak angka X. *Game* akan memberi tahu apakah tebakan pemain dibandingkan terhadap X lebih besar atau lebih kecil.
- Permainan selesai jika pemain menebak angka X dengan benar.
- Skor untuk *game* ini tergantung dengan seberapa cepat pemain menebak X. Formula skor dibebaskan.
- Batasan X dan maksimal giliran dibebaskan.

Catatan:

- Gunakan fungsi pembangkit angka acak bawaan bahasa c.
- Secara *default*, pembangkit angka acak bawaan bahasa c menghasilkan *pseudo random number* yang artinya angka yang dihasilkan memiliki pola dan tidak benar-benar acak. Cari tahu bagaimana membangkitkan angka acak dengan benar-benar acak.

2. Diner Dash

Indra dan Doni juga suka permainan yang menegangkan. Oleh karena itu, ia ingin ada sebuah game Diner Dash dalam BNMO. Secara singkat, Diner Dash merupakan permainan mengantar makanan namun terurut berdasarkan prioritasnya. Berikut adalah spesifikasi game ini:

- Terdapat 3 command yang dapat dilakukan pada game, yaitu **COOK** dan **SERVE**
 - **COOK** merupakan command yang bertujuan untuk memasak makanan
 - **SERVE** merupakan command yang bertujuan untuk menyajikan makanan kepada pelanggan.
 - **SKIP** merupakan command yang bertujuan untuk menyelesaikan 1 putaran tanpa melakukan apa apa (seluruh durasi memasak dan ketahanan berkurang 1, pelanggan bertambah 1).
- Command **selain COOK dan SERVE** dianggap tidak valid dan **tidak terhitung sebagai satu putaran**.
- Command **COOK dan SERVE yang tidak valid juga tidak terhitung sebagai satu putaran**
- Permainan akan dimulai dengan 3 pelanggan. Setiap pelanggan hanya dapat memesan satu makanan. Untuk setiap makanan, terdapat informasi tentang ID makanan yang dihasilkan secara *increment* (M01, M02, M03, dst), durasi memasak, harga makanan, serta ketahanan makanan. Semua informasi tersebut akan didapatkan secara random dengan menggunakan **random number generator**. Durasi dan ketahanan makanan akan berkisar diantara 1-5. Sedangkan, harga makanan akan berkisar diantara 10000 - 50000.
- Kapasitas dari pemain adalah memasak 5 makanan dalam waktu yang sama. Pelanggan yang dilayani adalah pelanggan yang duluan memasuki antrian.
- Permainan selesai apabila antrian melebihi 7 pelanggan atau jumlah pelanggan yang sudah dilayani mencapai 15 pelanggan.
- Pada setiap putaran, akan terdapat 1 pelanggan baru.
- Pada setiap putaran, seluruh durasi dari makanan yang sedang dimasak akan berkurang 1. Ketika durasi makanan mencapai 0, maka makanan sudah dapat di SERVE.

- Pada setiap putaran, seluruh ketahanan dari makanan yang sudah dapat disajikan akan berkurang 1. Ketika ketahanan makanan mencapai 0, maka makanan sudah akan hangus dan harus ulang dimasak.
- Ketika makanan sudah di SERVE, maka makanan dapat diantar kepada pelanggan dan pelanggan dapat meninggalkan antrian. Setelah pelanggan meninggalkan antrian, maka pemain akan menerima uang
- SERVE hanya dapat digunakan untuk pesanan yang berada di paling depan.
- Skor akhir dari pemain adalah total uang yang diterima oleh pemain.

3. Game tambahan/ Buatan pemain

Game buatan pemain yang dibuat menggunakan command CREATE GAME akan langsung selesai dan masuk ke tahap game over dengan skor akhir berupa integer random.

Daftar ADT yang Digunakan

Anda diwajibkan menggunakan ADT di bawah ini. Selain itu, Anda dapat pula menggunakan ADT lain, namun cantumkan analisis alasan kenapa menggunakan ADT tersebut pada laporan.

1. ADT Array

ADT ini digunakan untuk merepresentasikan daftar game yang terdapat dalam sistem.

2. ADT Mesin Karakter dan Mesin Kata

ADT ini digunakan untuk melakukan parsing command program dan pembacaan file konfigurasi ke dalam aplikasi.

3. ADT Queue

ADT ini digunakan untuk menentukan urutan bermain game serta salah satu game, yaitu Diner Dash.

Bonus

Pada tugas besar ini, terdapat beberapa fitur yang berupa bonus. Fitur-fitur ini tidak wajib untuk dikerjakan dan bobotnya lebih kecil dibanding fitur utama.

1. Tambahan game custom

Mahasiswa dibebaskan menambahkan jenis permainan. Permainan harus dibuat dari scratch dalam artian tidak boleh menggunakan library selain dari library yang ditentukan. Adapun aturan pembuatan game custom adalah:

STEI- ITB	IF2111-TB1-03-01	Halaman 30 dari 35 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.		

1. Menggunakan salah satu ADT yang telah dipelajari di kelas (ADT Point, ADT Array, ADT Mesin Karakter, dan sebagainya).
2. Memiliki output berupa score yang bertipe integer.



9.2 Notulen Rapat

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

No. Kelompok/Kelas : 1 / K03
Nama Kelompok : Kelompok Satu
Anggota Kelompok (Nama/NIM) :
1. Wan Aufa Aziz / 18221001
2. Athira Dhyannis / 18221022
3. Dhafin Ghalib Lukman Hakim / 18221023
4. Syasya Umaira / 18221026
5. Muhammad Mumtaz / 18221029



Asisten Pembimbing : Graciella Valeska Liander





Asistensi I


Tanggal : 4 November 2022	Catatan Asistensi: Sepertinya ada kesalahan di pointernya, nanti dibantu cek biasanya masalah di pointer karena kita banyak pake pointer Arraynya coba di ganti jadi word aja, karena bisa jadi bugnya di array word to string gunakan array of word gapapa kalo array of word nyimpen tuple struktur filenya masih berentakan misalnya bikin adt terus di dalamnya masukin array terus di dalam arraynya masukin bawah2nya exe jangan lupa di hapus drivernya taruh aja gapapa console pake makearray gabisa di atas harus di startword di startword gaperlu output list gamenya akan kebaca langsung mastiin fitur, playgame cuma boleh dari queue game, skip game dari queue game dan otomatis langsung di play untuk delete, pas mau save di clear dulu terus baru di masukin lagi game-gamenya algoritma menarik, kl mau load atau save kita pindah mode satu atau dua dulu itu bisa di masukin ke algoritma menarik data test bisa liat ke spek, commandnya apa ekspektasi dari program kita apa (ga harus ikut spek) test script commandnya apa, realitanya yang keluar apa dari program kita
Tempat : Zoom	
Kehadiran Anggota Kelompok:	
No NIM Tanda tangan 1 18221001  2 18221029  3 18221026	

 4 18221022 	
	Tanda Tangan Asisten: 

Asistensi II

Tanggal : 11 November 2022	Catatan Asistensi: Recap progress : codingan di github udh di push yang terbaru buat adtnya udah lengkap semua, di bagian prosedur juga udah ada implementasi game termasuk bonus buat data juga udah ada, main dan consolenya juga udah recap kendala: untuk codingannya lebih ke ragu soalnya di kita masih ada yang memakai string.h dipake di mesin kata, boleh ga ya kak? sebenarnya gaboleh import library manggil string.h kalo ga manggil string.h boleh. berarti implementasi sendiri perintah load dan save di spek berbarengan dengan nama file, kalo kita input satu satu dulu boleh ga ya? digabung aja kalo bisa sesuain sama spek soalnya berhubungan sama pengecekan juga kalo ga sama kan bisa eror bingung di isi folder bin, isinya itu apa aja ya kak?
Tempat : Zoom	
Kehadiran Anggota Kelompok: No NIM Tanda tangan 1 18221001   2 18221022	

 <p>3 18221023</p>  <p>4 18221026</p>  <p>5 18221029</p> 	<p>make file tujuannya buat simplify cara executenya, kalo ada make filenya perintah gcc gitunya udah kepake, sebenarnya optional, ntar ditulis ajadi readme nya sample save data itu contoh data yang udah di save harusnya gimana</p> <p>ada error code di proses pemanggilan queue, aku emang ada dua queue diner dash sama queue? kalo di main tadi include apa? queue aja kalo pas di compile kan semua udh jadi 1 program yg sama, bisa dibilang ga kepisah sesuai folder, jadi dia sempet bingung</p> <p>di queue diner dash sebenarnya queue dan ga queue konsepnya aku pake indeks yang mana yang di dequeue jadi makenya dequeue at gitu.</p> <p>demo prefer online apa offline? kalo offline ketemu langsung sama asistennya, kemungkinan besar online sih.</p> <p>demo kapan? harusnya di range minggu depan, tunggu info selanjutnya aja</p>
	Tanda Tangan Asisten:

	
--	---

9.3 Log Activity Anggota Kelompok

Timeline Kelompok

No	Waktu	Keterangan
1	Minggu, 30 Oktober 2022	<ul style="list-style-type: none"> Meet kelompok untuk pembagian tugas
2	Jumat, 4 November 2022	<ul style="list-style-type: none"> Asistensi 1 Meet kelompok membahas hasil asistensi
3	Rabu, 9 November 2022	<ul style="list-style-type: none"> Penyatuan program utama Penyelesaian bonus (game tambahan)
5	Jumat, 11 November 2022	<ul style="list-style-type: none"> Asistensi 2 Penyelesaian laporan Melakukan <i>finishing</i> pada code program