LAPORAN TUGAS BESAR

IF2110/Algoritma dan Struktur Data

BNMO

Dipersiapkan oleh:

Kelompok B - K01

Muhammad Hanan	13521041
M. Farrel Danendra Rachim	13521048
Muhammad Bangkit Dwi Cahyono	13521055
Mutawally Nawwar	13521065
Louis Caesa Kesuma	13521069

Sekolah Teknik Elektro dan Informatika - Institut Teknologi Bandung

Jl. Ganesha 10, Bandung 40132

Daftar Isi

1. Ringkasan **4**

2. Penjelasan Tambahan Spesifikasi Tugas **4**
2. 1. Kulkas 4
2. 2. Waktu Pengolahan Makanan 5
2. 3. Rekomendasi Makanan 6
2. 4. WAIT X Y 6
2. 5. CATALOG 6
2. 6. COOKBOOK 6
2. 7. HELP 6
2. 8. START/EXIT 6

3. Struktur Data (ADT) **7**
3.1. Makanan 7
3.2. Time 8
3.3. Point 9
3.4. Simulator 10
3.5. List Statik 11
3.6. List Dinamis 13
3.7. Matriks 14
3.8. Mesin Karakter dan Mesin Kata 16
3.9. Queue dengan List Dinamik dan Infotype 19
3.10. Stack 21
3.11. Tree 23
3.12. Game 25
3.13. Map 26
3.14. Set 27

4. Program Utama **27**

5. Algoritma-Algoritma Menarik **30**
5.1. Algoritma Undo-Redo 30
5.2. Algoritma Kulkas 30
5.3. Algoritma Notifikasi 31

6. Data Test **31**

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.

6.1. Data Test START 31
6.2. Data Test MOVE 31
6.3. Data Test Pemesanan Makanan (Beserta DELIVERY dan INVENTORY) 32
6.4. Data Test Pengolahan Makanan (Beserta CATALOG dan COOKBOOK) 33
6.5. Data Test Makanan Kadaluarsa 34
6.6. Data Test Undo-Redo 35
6.7. Data Test HELP 36
6.8. Data Test WAIT 36
6.9. Data Test MASUKKULKAS/ KELUARKULKAS 37
6.10. Data Test REKOMENDASI 37
6.11. Data Test EXIT 38

7. Test Script 39

8. Pembagian Kerja dalam Kelompok 41

9. Lampiran 42
9.1. Deskripsi Tugas Besar 2 42
9.2. Notulen Rapat 42
9.3. Log Activity Anggota Kelompok 43
9.4. Form Asistensi Tugas Besar 45

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. Ringkasan

Permasalahan utama yang dihadapi dalam laporan ini adalah pembuatan sebuah program simulasi memasak bernama BNMO yang dibuat dalam bahasa C. Objektif dari program ini yakni memasak hidangan dari bahan makanan yang tersedia. Bahan makanan ini dapat diperoleh dengan memesan makanan melalui *command* BUY atau melalui inventory. Makanan juga mempunyai waktu kedaluwarsa – jika waktu kedaluwarsa telah terlampaui, makanan tidak bisa digunakan lagi (dibuang dari inventory). Adapun program dapat mengolah bahan makanan menjadi hidangan yang tersedia dengan berbagai macam operasi aksi, seperti MIX, CHOP, FRY, dan BOIL. Simulator berupa sebuah navigasi peta yang memiliki fitur-fitur untuk melengkapi objektif tersebut, seperti tempat telepon (T) untuk memesan makanan, tempat mixing (M) untuk melakukan operasi MIX, tempat menggoreng (F) untuk melakukan operasi FRY, dan seterusnya. Setiap *command* perpindahan dan pengolahan makanan dilakukan, satu menit pada simulator akan berjalan.

Kami menggunakan berbagai macam Abstract Data Type (ADT) untuk membangun program ini. Terdapat ADT sederhana seperti ADT Point, Waktu, Makanan, dan Simulator. Penyimpanan list makanan dan list resep memanfaatkan ADT List Statik. Gambaran peta pada simulator menggunakan ADT Matriks. Program ini menggunakan data yang diperoleh dari file konfigurasi berupa makanan, resep, dan peta, dengan memanfaatkan ADT Mesin Karakter dan Mesin Kata untuk melakukan pembacaan file tersebut. Inventory menggunakan konsep ADT Queue, sedangkan *command undo* dan *redo* menggunakan ADT Stack. Resep makanan didefinisikan pada file konfigurasi menggunakan ADT N-ary Tree. (Deskripsi tugas yang lengkap dicantumkan di Lampiran)

Dalam laporan ini akan dijelaskan penjelasan tambahan mengenai spesifikasi fitur yang belum rinci dari deskripsi tugas yang dilampirkan, penjelasan lebih lengkap mengenai ADT yang digunakan untuk membangun program simulasi ini, algoritma program utama, algoritma yang kami anggap menarik untuk dibahas di laporan ini, data test dari setiap fitur di program ini, dan skenario testing untuk semua fitur yang ada.

Dari hasil Tugas Besar ini, kami dapat menarik kesimpulan bahwa ADT dan struktur data secara umum sangatlah berguna untuk membangun algoritma dengan lebih efisien dan lebih mudah untuk dibaca. Tugas Besar ini juga lebih memberi pendalaman mengenai fungsi dan cara kerja khas dari tiap ADT. Misal, ADT N-ary Tree sangat cocok sebagai struktur data untuk sebuah resep makanan, karena sebuah hidangan memiliki bahan-bahan makanan yang dapat direpresentasikan melalui sebuah N-ary Tree.

2. Penjelasan Tambahan Spesifikasi Tugas

Pada tugas besar ini, kami mengerjakan beberapa fitur yang sekiranya kurang dijelaskan secara detail di pedoman tugas besar. Berikut adalah spesifikasi dari fitur tersebut.

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.

2. 1. Kulkas

Kulkas adalah salah satu fitur bonus yang kami implementasikan. Kulkas ini berfungsi menyimpan makanan yang dimiliki dari inventory sehingga waktu kedaluarsanya tidak berkurang, walaupun waktu terus berjalan saat simulator melakukan aktivitas lain. Fitur kulkas ini memanfaatkan ADT Matriks dengan isi elemen integer. Makanan dalam kulkas tidak bisa ditumpuk dan tidak boleh saling beririsan pada sel yang sama. Kami menetapkan ukuran kulkas sebesar 10 x 20 (20 kolom 10 baris). Kami juga menetapkan beberapa proses untuk mengelola kulkas:

- MASUKKULKAS X Y. Pengguna dapat mengetik command ini untuk menyimpan makanan yang dimiliki di inventory ke sebuah posisi di kulkas. X adalah indeks makanan yang ingin dimasukkan ke kulkas pada inventory pengguna, dan Y adalah sebuah nomor yang akan di-*assign* kepada makanan yang akan dimasukkan ke kulkas. Akhirnya, makanan akan direpresentasikan dalam sebuah bilangan (kami akan memanggilnya ID kulkas) dengan rumus: Makanan.id * 1000 + Y. Sehingga, kulkas dapat diimplementasikan dengan ADT Matriks yang menyimpan integer (dalam hal ini integer adalah ID kulkas dari proses meng-*assign* nomor Y ke makanan).

  Misalkan kita mengetik MASUKKULKAS 1 5. Pada saat itu, makanan pada indeks ke-1 di inventory adalah ayam mentah dengan ID makanan 12. Jadi, ID kulkas yang disimpan di dalam kulkas adalah 12 * 1000 + 5 = 12005.

  Kulkas tidak dapat menyimpan dua ID kulkas yang sama. Jika terdapat ID kulkas yang sama (berarti ID makanan sama dan Y sama) maka program akan meminta pengguna untuk memasukkan angka yang benar.

  Lalu, pengguna diminta untuk menginput 4 buah argumen berupa letak penyimpanan kulkas: Kiri atas, kanan bawah. Posisi harus divalidasi: Tidak boleh ada dua makanan yang menempati posisi kulkas yang sama, dan posisi harus di dalam range 10 x 20. Selain itu, size sebuah makanan harus dipertimbangkan juga: Jika tidak ada ruang yang cukup di dalam kulkas untuk memasukkan makanan dengan size tertentu (dengan memperhatikan bahwa tidak ada bagian dari kedua makanan yang bertabrakan), dikeluarkan pesan "Kulkas sudah penuh".
- KULKAS. Pengguna dapat melihat isi kulkas dengan mengetik command KULKAS. Kulkas akan terlihat seperti berikut:

  Bisa dilihat bahwa size makanan 11000 adalah 2 x 2.

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.

- KELUARKULKAS X. Untuk mengambil sebuah makanan dari kulkas ke inventory pengguna, pengguna dapat mengetik command KELUARKULKAS X dengan X adalah ID kulkas. Ruang yang dulunya ditempati makanan tersebut sekarang bisa ditempati makanan lain.

2. 2. Waktu Pengolahan Makanan

Dengan fitur ini, waktu pengolahan tiap-tiap makanan dapat berbeda-beda. Misalnya dibutuhkan waktu 10-15 menit untuk menggoreng ayam menjadi ayam goreng, tetapi hanya membutuhkan 3-5 menit untuk menggoreng telor. Dengan adanya fitur ini, waktu pengolahan makanan menjadi lebih realistis.

2. 3. Rekomendasi Makanan

Dengan fitur ini, pengguna bisa memperoleh rekomendasi makanan apa saja yang bisa dibuat berdasarkan makanan yang dimiliki di inventory. Fitur ini memanfaatkan konsep subset dan union pada himpunan (itulah mengapa kami menggunakan ADT Set) untuk mencocokkan bahan makanan apa saja yang dibutuhkan sesuai resep untuk membuat makanan baru dengan bahan makanan yang tersedia di inventory pemain.

2. 4. WAIT X Y

Simulator dapat menunggu waktu tertentu dengan command WAIT x y dengan x adalah jam dan y adalah menit untuk diam tidak melakukan apa-apa dan hanya menambah waktu. Contoh: WAIT 10 0 berarti pengguna diam selama 10 jam dan 0 menit. Nilai x dan y harus divalidasi agar berupa integer >= 0. Command WAIT juga harus memiliki tepat dua argumen (x dan y), jadi command WAIT tidak boleh ditulis seperti WAIT 10, WAIT 10 9 8, atau sekedar WAIT saja. Jika kedua hal tersebut sudah divalidasi, kami mengubah waktu terkini pada game sebanyak waktu yang diinput, dan semua aksi yang terjadi di luar tangan simulator (seperti makanan kedaluwarsa, makanan yang diantar sudah datang) akan di-apply.

2. 5. CATALOG

Command ini bertujuan untuk menampilkan semua bahan dan makanan yang tersedia pada aplikasi termasuk informasi detailnya, seperti nama makanan, durasi kedaluwarsa, aksi yang diperlukan (seperti fry, chop, buy dll), waktu pengantaran, dan ukuran makanan.

2. 6. COOKBOOK

Command ini bertujuan untuk menampilkan resep-resep yang tersedia pada sistem. Akan ditampilkan ke layar nama makanan yang akan dibuat, lalu di bawahnya ditulis prosesnya berupa aksi yang dilakukan terhadap bahan makanan serta bahan makanan yang diterapkan aksi tersebut.

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.

2. 7. HELP

Pengguna dapat mengetik command HELP untuk mencari tahu command-command apa saja yang dapat dioperasikan oleh pengguna, seperti MOVE X (X: NORTH/ SOUTH/ WEST/ EAST), BUY, FRY, CHOP, dan lain-lain. Daftar command ini akan ditampilkan pada layar.

2. 8. START/EXIT

Untuk memulai aplikasi, pengguna dapat mengetik command START. Secara otomatis, akan dilakukan konfigurasi file makanan, resep dan peta. Waktu dimulai dari 0:0:0, simulator, listMakanan dan listResep kosong dibuat, serta lokasi simulator dimulai dari kiri atas. Untuk keluar dari aplikasi, pengguna dapat mengetik command EXIT, yang akan menampilkan pesan terima kasih kepada pengguna dan akan menghentikan segala operasi yang pada saat itu sedang berjalan (seperti pengantaran dll), jadi state terkini aplikasi tersebut akan di-reset ke default jika pengguna ingin menjalankan program lagi (tidak dapat di-save).

3. Struktur Data (ADT)

3.1. Makanan

```c
/* Definisi Aksi */
typedef enum {
    BUY = 'T',
    FRY = 'F',
    CHOP = 'C',
    BOIL = 'B',
    MIX = 'M'
} ActionType;

/* Definisi Makanan */
typedef struct {
    int id;             /* ID makanan */
    char *name;         /* Nama makanan */
    TIME expiry;        /* Waktu kedaluwarsa */
    TIME delivery;      /* Lama pengiriman makanan */
    char location;      /* Lokasi aksi makanan */
    POINT size;         /* Size makanan (mxn) */
    TIME pengolahan;    /* Waktu pengolahan makanan*/
} Makanan;
```

id	name	expiry	location	delivery	size	pengolahan

Dari sketsa di atas, dapat dilihat bahwa ADT Makanan diimplementasikan dalam bentuk ADT sederhana, dan memiliki data berupa ID makanan (integer), nama makanan (string, misal

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.

“ayam goreng”, tepung”), waktu kadaluarsa makanan (ADT Time), lokasi makanan di peta berupa koordinat kartesius (ADT Point), lama pengiriman makanan (ADT Time), ukuran makanan dengan representasi dua buah integer mxn (ADT Point), dan waktu pengolahan makanan (ADT Time). Selain itu, kami juga membuat enumerated type bernama ActionType untuk setiap command aksi yang direpresentasikan dengan sebuah karakter, yang nantinya akan berguna dalam navigasi di peta juga: BUY = 'T', FRY = 'F', CHOP = 'C', BOIL = 'B', MIX = 'M'.

ADT Makanan digunakan untuk merepresentasikan makanan berupa data yang terkandung di dalamnya yang akan digunakan dalam simulator ini. ADT ini menjadi komponen dasar untuk membangun fitur-fitur lain seperti Inventory, aksi pengolahan makanan, dan lain sebagainya. Kami memilih ADT ini karena membuat sebuah ADT untuk menyimpan berbagai data makanan akan membuat pembuatan program secara menyeluruh jauh lebih efisien daripada membuat variabel terpisah. Nama file: “makanan.h”, “makanan.c”

Adapun primitif dalam ADT Makanan adalah:

1. Selektor
   a. ID(M)
   b. NAME(M)
   c. EXP(M)
   d. DELIVERY(M)
   e. LOC(M)
   f. SIZE(M)
   g. PENGOLAHAN(M)
2. Konstruktor
   a. void createMakanan(Makanan *M, int id, char *name, TIME exp, char loc, TIME delivery, POINT size, TIME pengolahan)
3. Predikat
   a. boolean isMakananExpired(Makanan M)
   b. boolean isMakananDelivered(Makanan M)
   c. boolean isMakananBuyable(Makanan M)
   d. boolean isMakananFryable(Makanan M)
   e. boolean isMakananChopable(Makanan M)
   f. boolean isMakananBoilable(Makanan M)
   g. boolean isMakananMixable(Makanan M)

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.

3.2. Time

```
/* *** Definisi TYPE TIME <HH:MM:SS> *** */
typedef struct {
    int DD; /* integer [0..infinity] */
    int HH; /* integer [0..23] */
    int MM; /* integer [0..59] */
} TIME;
```

DD	HH	MM

Dari sketsa di atas, dapat dilihat bahwa ADT Time diimplementasikan dalam bentuk ADT sederhana, dan memiliki data berupa DD yaitu tanggal (integer), HH yaitu jam (integer 0-23), MM yaitu menit (integer 0-59). ADT ini digunakan untuk merepresentasikan waktu relatif dalam simulator, dengan tiap aksi default memajukan waktu sebesar semenit. ADT ini dapat menjadi pondasi untuk mengukur waktu delivery dan waktu expiry makanan. Kami memilih ADT ini karena pengukuran waktu akan menjadi lebih mudah. Nama file: “time.h”, “time.c”

Adapun primitif dalam ADT TIME adalah:

1. Selektor
   a. Day(T)
   b. Hour(T)
   c. Minute(T)
2. Validasi
   a. boolean IsTIMEValid (int D, int H, int M)
3. Konstruktor
   a. void CreateTime (TIME * T, int DD, int HH, int MM)
4. Baca/tulis
   a. void TulisTIME (TIME T)
   b. void TulisTIME2 (TIME T)
   c. void TulisTIME3 (TIME T)
   d. void TulisTIME4 (TIME T)
   e. void TulisTIME5 (TIME T)
5. Konversi
   a. long TIMEToMenit (TIME T)
   b. TIME MenitToTIME (long N)
6. Operasi relasional
   a. boolean TEQ (TIME T1, TIME T2)
   b. boolean TNEQ (TIME T1, TIME T2)

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.

c. boolean TLT (TIME T1, TIME T2)
d. boolean TGT (TIME T1, TIME T2)

7. Operasi aritmatika
   a. TIME NextMenit (TIME T);
   b. TIME NextNMenit (TIME T, int N);
   c. TIME PrevMenit (TIME T);
   d. TIME PrevNMenit (TIME T, int N);
   e. long Durasi (TIME TAw, TIME TAkh);

3.3. Point

```
typedef struct {
    int X; /* absis   */
    int Y; /* ordinat */
} POINT;
```

X	Y

Dari sketsa di atas dapat dilihat bahwa ADT Point diimplementasikan dalam bentuk ADT sederhana, dan memiliki data berupa X berupa absis (integer) dan Y berupa ordinat (integer). ADT ini digunakan merepresentasikan koordinat simulator di peta dengan koordinat kartesius. ADT ini dapat membantu mengetahui posisi terkini simulator. Kami memilih ADT ini karena merupakan ADT yang paling tepat sebagai navigasi simulator untuk peta aplikasi dalam bentuk 2 dimensi. Nama file: “point.h”, “point.c”

Adapun primitif dalam ADT POINT:

1. Selektor
   a. Absis(P)
   b. Ordinat(P)
2. Konstruktor
   a. void CreatePoint(POINT *P, int X, int Y)
3. Baca/tulis
   a. void ReadPoint(POINT *P)
   b. void WritePoint(POINT P)
   c. void WriteLokasi(POINT P)
4. Operasi relasional
   a. boolean SideBy(POINT P1, POINT P2)
5. Operasi move
   a. POINT MoveN(POINT P)

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.

b. POINT MoveE(POINT P)
c. POINT MoveS(POINT P)
d. POINT MoveW(POINT P)

3.4. Simulator

```
typedef struct {
    char* username;
    POINT lokasi;
    PrioQueue inventory;
} Simulator;
```

username	lokasi	inventory

Dari sketsa di atas dapat dilihat bahwa ADT Simulator diimplementasikan dalam bentuk ADT sederhana, dan memiliki data berupa username simulator, lokasi terkini simulator di peta (ADT Point), dan inventory yang dimiliki simulator (ADT PrioQueue). ADT ini digunakan untuk merepresentasikan data yang dimiliki simulator, dan berguna untuk melakukan berbagai aksi di aplikasi berdasarkan lokasi simulator dan inventory yang dimiliki. Kami memilih ADT ini karena username memiliki berbagai data yang sangat membantu untuk berjalannya aplikasi.
Nama file: “simulator.h”, “simulator.c”
Adapaun primitif dari ADT Simulator:

1. Selektor
   a. Username(x)
   b. Lokasi(x)
   c. Inventory(x)
2. Konstruktor
   a. void createSimulator(Simulator* s, char* Username, POINT lokasi, PrioQueue inventory)
3. Lain-lain
   a. void gantiUser(Simulator * s, char* username);
   b. void gantiLokasi(Simulator * s, POINT p);
   c. void gerakUser(Simulator * s, Map * peta, boolean * stuck, char x);
   d. void simulatorToState(Simulator s, PrioQueue deliverylist, TIME currentTime, ListDinMakanan currentNotif, MatrixKulkas currentKulkas, State* temp);

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.

e. void loadState(Simulator * s, PrioQueue * deliverylist, State st, char * currentUsername, ListDinMakanan * currentNotif, MatrixKulkas * currentKulkas, TIME * currentTime);
f. void copySimulator(Simulator * s1, Simulator * s2);
g. void setInventory(Simulator * s, PrioQueue inventory);
h. void MNorth(Map *MGame, boolean *stuck, Simulator *s);
i. void MEast(Map *MGame, boolean *stuck, Simulator *s);
j. void MSouth(Map *MGame, boolean *stuck, Simulator *s);
k. void MWest(Map *MGame, boolean *stuck, Simulator *s);
l. boolean cekSpotKosongKulkas(MatrixKulkas kulkas, int idxX, int idxY);
m. void penomorMakananKulkas(infotype * var, int id, MatrixKulkas kulkas, boolean * ada);
n. void hapusIdKulkas(infotype * var);
o. void ambilDariInventory (Simulator * s, int idx, infotype * var);
p. void pindahKeKulkas(infotype var, MatrixKulkas * kulkas, ListDinMakanan * currentNotif, int idX, int idY);
q. void ambilDariKulkas(Simulator * s, int idmakanan, MatrixKulkas * kulkas, ListDinMakanan * currentNotif);
r. boolean adjacent(Simulator s, Matrix peta, char x);
s. void displayListMakananAksi(ListStatikM listMakanan, ListStatikM *hasil, int *length, char aksi);
t. Tree getTreeFromMakanan(Makanan makanan, ListStatikT listResep);
u. Makanan getMakananFromId(int id, ListStatikM listMakanan);

3.5. List Statik

0	1	2	3	4	5	…	45	46	47	48	49

Di atas merupakan list statik yang memiliki 50 elemen. Karena tergolong sebagai list statik, ukuran array tidak dapat diubah. Kami menggunakan implementasi list statik secara implisit rata kiri. List kosong pada awalnya diinisialisasi memorinya dengan nilai MARK (kami menggunakan -999, serta nama MARK tergantung jenis data elemen list), serta elemen yang terisi harus kontigu, berarti tidak boleh ada mark di nilai yang terisi. Kami membuat list statik dengan tiga ADT yang berbeda sesuai tipe data elemen yang disimpannya. List ini merupakan array [0..49] of:

a. Makanan (ListStatikM)

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.

List statik ini menyimpan elemen-elemen dengan tipe makanan yang bisa digunakan di aplikasi. Elemen-elemen di list statik ini diperoleh dengan membaca file konfigurasi makanan. Nama file: “listtatikm.h”, “liststatikm.c”

```
/* Definisi elemen dan koleksi objek */
typedef int IdxType;
typedef struct {
   Makanan contents[CAPACITY_LISTSTATIKM];
} ListStatikM;
```

1. Selektor
   a. ELMT_LISTSTATIKM(l, i)
2. Konstruktor
   a. void createLSMakanan(ListStatikM *l);
3. Lain-lain
   a. int listLengthStatikM(ListStatikM l);
   b. void insertFood(ListStatikM *l, Makanan food);
   c. void printCatalog(ListStatikM l);

b. Tree (ListStatikT)

List statik ini menyimpan elemen dengan tipe data berupa tree. Hal ini dapat memudahkan penyimpanan berbagai resep yang dibaca dari file konfigurasi resep. elEff adalah elemen efektif yang dimiliki list. Nama file: “liststatikt.h”, “liststatik.c”

```
typedef struct {
   Tree list[MAX_RESEP];
   int elEff;
} ListStatikT;
```

Konstruktor: void CreateListTree(ListStatikT *lt);

c. POINT (ListStatikP)

List statik ini menyimpan elemen dengan tipe data POINT, yang dapat digunakan untuk menyimpan lokasi-lokasi pada file konfigurasi peta seperti S, F, B, dll, dengan bantuan ADT Matriks. Nama file: “liststatikp.h”, “liststatikp.c”

```
/* Definisi elemen dan koleksi objek */
typedef int IdxType;
typedef struct {
   POINT contents[CAPACITY_LISTSTATIKP];
} ListStatikP;
```

1. Selektor
   a. ELMT_LISTSTATIKP(l, i)
2. Konstruktor

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.

a. void createLSPoint(ListStatikP *l);

3. Lain-lain
   a. int listLengthStatikP(ListStatikP l);
   b. void insertPoint(ListStatikP *l, POINT p);

Kami menggunakan ADT List Statik karena ADT ini sangat cocok sebagai struktur data untuk menyimpan data yang sifatnya konstan/tidak berubah dan mudah dikelola - hal ini berlaku untuk daftar makanan, resep dan koordinat titik aksi yang diperoleh dari file konfigurasi. Perubahan list ini hanya bisa dilakukan pada awal proses aplikasi berjalan saat memasukkan command KONFIGURASI dengan mengubah file konfigurasi itu sendiri.

3.6. List Dinamis

Tidak seperti list statik, list dinamis dapat memungkinkan sebuah List yang jumlah elemennya tidak terbatas menggunakan array. Sesuai namanya, list ini dapat mengubah ukuran maksimumnya secara dinamis dengan cara mengalokasikan array baru yang lebih besar (jika sudah penuh dan ingin menambahkan elemen) atau array yang lebih kecil (jika "sparse", misal hanya 5 persen yang terisi).

Berikut kami meng-alokasikan sebuah list dinamik yang berukuran [0..999] dengan fungsi CreateListMakananDin(ListDinMakanan *l, int capacity) (hanya sebuah contoh):

0	1	2	3	4	5	…	995	996	997	998	999

Jika dalam suatu kasus list tersebut didealokasi, kapasitas list tersebut berubah kembali menjadi nol.

```
/* Definisi elemen dan koleksi objek */
typedef int IdxType;
typedef struct
{
    infotype *buffer;
    int nEff_Makanan;
    int capacity_Makanan;
} ListDinMakanan;
```

Dalam program ini kami menggunakan ADT ListDinMakanan yang menyimpan elemen berupa infotype dengan nama buffer (lihat 3.9), nEff_Makanan berupa integer yang menandakan banyaknya elemen yang disimpan di dalam list, dan capacity_Makanan berupa integer yang menandakan kapasitas terkini list yang bisa dimodifikasi. ADT ListDinMakanan dapat diimplementasikan ke dalam ADT state (lihat 3.10) pada aplikasi sebagai penyimpan makanan yang berubah (misal habis di-FRY/di-BOIL/kedaluarsa) pada waktu tertentu. Kami

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.

menggunakan prinsip list dinamis untuk kasus ini karena jumlah makanan yang dapat berubah selama aplikasi dijalankan tidak dapat ditentukan. Ada kemungkinan makanan yang disimpan di dalam list akan mencapai kapasitas batas jika pemain ingin mengolah banyak makanan. Keuntungan penggunaan list dinamis yaitu mudahnya mengubah kapasitas list dengan alokasi kalau list sudah penuh. Nama file: “listdinmakanan.h”, “listdinmakanan.c”

Adapun primitif ADT ListDinMakanan adalah:

1. Selektor
   a. NEFF_Makanan(l)
   b. BUFFER_Makanan(l)
   c. ELMT_LISTDINAMIS_Makanan(l, i)
   d. CAPACITY_LISTDINAMIS_Makanan(l)
   e. IdxType getFirstIdxS(ListDinMakanan l);
   f. IdxType getLastIdxS(ListDinMakanan l);
   g. int listMakananLengthDinamis(ListDinMakanan l);
2. Konstruktor
   a. void CreateListMakananDin(ListDinMakanan *l, int capacity);
   b. void dealocateListMakanan(ListDinMakanan *l);
3. Baca tulis
   a. void printListMakanan(ListDinMakanan l);
   b. void printListMakananUndo(ListDinMakanan l);
4. Lain-lain
   a. boolean isEmptyListMakanan(ListDinMakanan l);
   b. void insertLastMakanan(ListDinMakanan *l, infotype val);
   c. void deleteLastMakanan(ListDinMakanan *l, infotype *val);
   d. void expandListMakanan(ListDinMakanan *l, int num);
   e. void copyListDinMakanan(ListDinMakanan l1, ListDinMakanan *l2);

3.7. Matriks

	0	1	2	3	4	5	...	96	97	98	99
0											
1											
2											
3											
4											
5											
...											
96											
97											
98											
99											

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.

```
* * * * * * * * * * * *
*   S                   *
*           T     X     *
*   M             X     *
*                 X     *
*           X X X X     *
*   X                   *
*   X               C   *
*   X X X     F         *
*                       *
*               B       *
* * * * * * * * * * * *
```

Di atas merupakan matriks berukuran 100 x 100. Memori matriks yang dipakai selalu di ujung kiri atas. ADT Matriks ini digunakan sebagai representasi peta pada aplikasi. Peta diperoleh dari file konfigurasi, dan setelah dibaca, akan disimpan dalam ADT Matriks dengan bentuk-bentuk karakter tertentu. Contohnya bisa dilihat dari gambar di samping. Sehingga, bisa dibilang bahwa matriks ini merupakan matrix[0..49, 0..49] of char. Selain itu, ADT matriks digunakan sebagai fitur kulkas (bernama MatrixKulkas), dengan matriks menggunakan elemen infotype, berukuran 10 x 20. Kami memilih matriks karena ilustrasi pergerakan simulator pada peta paling mudah diimplementasikan melalui matriks, dengan perpindahan simulator melalui sebuah baris atau kolom, serta kulkas dengan bentuk matriks sangat berguna untuk mengelola item di dalamnya agar mudah dicari pengguna. Nama file: “matriks.h”, “matriksKulkas.h”, “matriks.c”, “matriksKulkas.c”

```c
typedef int IdxType; /* Index baris, kolom */
typedef char ElTypeChar;
typedef struct
{
   ElTypeChar mem[ROW_CAP][COL_CAP];
   int rowEff; /* banyaknya/ukuran baris yg terdefinisi */
   int colEff; /* banyaknya/ukuran kolom yg terdefinisi */
} Matrix;
```

```c
typedef int IdxType; /* Index baris, kolom */
typedef infotype ElType;
typedef struct
{
   ElType mem[ROW_CAP_K][COL_CAP_K];
   int rowEff; /* banyaknya/ukuran baris yg terdefinisi */
   int colEff; /* banyaknya/ukuran kolom yg terdefinisi */
} MatrixKulkas;
```

Primitif ADT Matrix:

1. Selektor
   a. ROW_EFF(M)
   b. COL_EFF(M)
   c. ELMT_MATRIX(M, i, j)
   d. IdxType getLastIdxRow(Matrix m);
   e. IdxType getLastIdxCol(Matrix m);

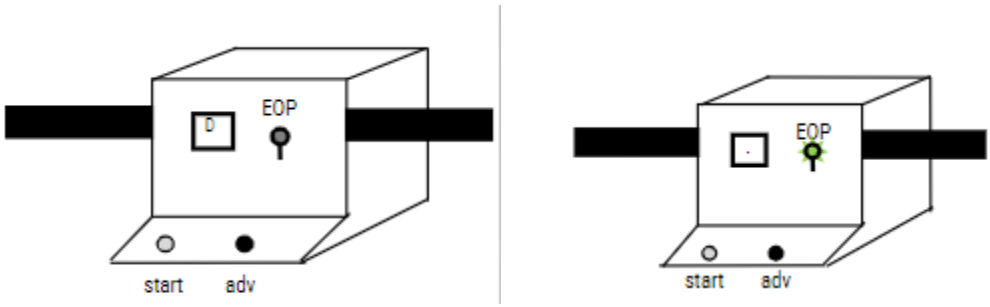
Template dokumen ini dan informasi yang dimilikinya adalah milik Sekolah Teknik Elektro dan Informatika ITB dan bersifat rahasia. Dilarang me-reproduksi dokumen ini tanpa diketahui oleh Sekolah Teknik Elektro dan Informatika ITB.

   f. boolean isIdxEffMatrix(Matrix m, IdxType i, IdxType j);
2. Konstruktor
   a. void createMatrix(int nRows, int nCols, Matrix *m);
3. Baca tulis
   a. void displayMatrix(Matrix m);
4. Lain-lain
   a. void copyMatrix(Matrix mIn, Matrix *mOut);
   b. int countElmtMatrix(Matrix m);
   c. boolean isSquare(Matrix m);

Primitif ADT MatrixKulkas:

1. Selektor
   a. ROW_EFFK(M)
   b. COL_EFFK(M)
   c. ELMTK(M, i, j)
   d. boolean isMatrixKulkasIdxValid(int i, int j);
   e. IdxType getLastIdxRowKulkas(MatrixKulkas m);
   f. IdxType getLastIdxColKulkas(MatrixKulkas m);
   g. boolean isIdxEffKulkas(MatrixKulkas m, IdxType i, IdxType j);
2. Konstruktor
   a. void createMatrixKulkas(int nRows, int nCols, MatrixKulkas *m);
3. Baca/tulis
   a. void displayMatrixKulkas(MatrixKulkas m);
4. Lain-lain
   a. void copyMatrixKulkas(MatrixKulkas mIn, MatrixKulkas *mOut);
   b. int countElmtKulkas(MatrixKulkas m);
   c. int countElmtDummy(MatrixKulkas m);
   d. boolean isFullKulkas(MatrixKulkas m);
   e. boolean isSquareKulkas(MatrixKulkas m);

3.8. Mesin Karakter dan Mesin Kata



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.

Di atas merupakan sketsa mesin karakter, dengan pita berupa deretan karakter dan diakhiri dengan MARK (dalam mesin ini MARK merupakan '\n' atau newline). Mesin ini berfungsi untuk membaca tiap karakter yang masuk ke dalam mesin (CC = Current Character). Jika CC = MARK, diraih End Of Process (EOP) dan pembacaan pita karakter selesai. Mesin karakter berguna sebagai pondasi utama untuk mesin kata.

Sedangkan untuk mesin kata, kami menggunakan model akuisisi versi I, yakni kata diakuisisi mulai dari karakter pertama sesudah akhir kata, dan sebuah boolean EndWord akan bernilai true jika kata terakhir telah diakuisisi dan diproses.

```
typedef struct {
    char TabWord[NMax1];
    int Length;
} Word;

typedef struct {
    Word TabWords[NMax2];
    int Length;
} ListWord;
```

ADT Word:

TabWord	Length

ADT ini digunakan sebagai representasi sebuah kata, dengan data TabWord[10] berupa char sebagai container penyimpan kata dengan indeks yang dipakai [0..99], serta Length berupa integer sebagai panjang kata.

ADT ListWord:

TabWords	Length

ADT ini digunakan sebagai penyimpanan kata-kata yang telah diakuisisi dalam suatu line dengan data TabWords[30] dengan elemennya berupa Word serta length berupa integer sebagai panjang.

ADT Word Machine ini digunakan sebagai parser command program (seperti FRY, BOIL, etc) dan pembacaan file-file konfigurasi ke dalam aplikasi. Kami memilih ADT Word Machine karena membaca input berupa string di bahasa C tidak terlalu praktis dibandingkan

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.

bahasa pemrograman yang lain, khususnya dalam pembacaan file eksternal. Sehingga, diperlukan sebuah ADT yang dapat membaca sekumpulan kata dan mengolahnya agar aplikasi dapat berjalan dengan lancar. Nama file: “charmachine.h”, “wordmachine.h”, “charmachine.c”, “wordmachine.c”

Adapun primitif ADT charmachine:

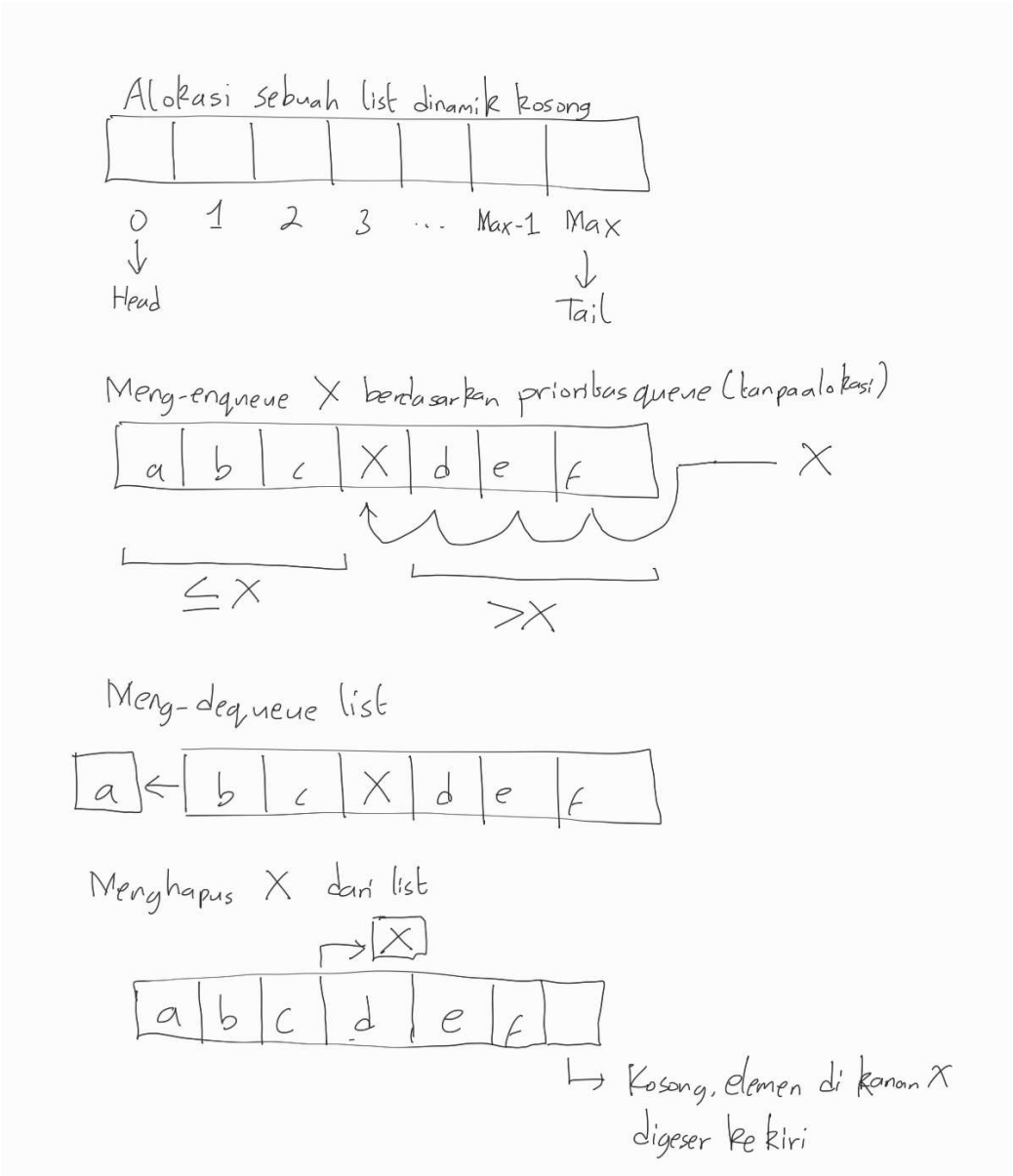
a. void START();
b. void ADV();
c. void STARTFILE(char *dir);
d. boolean isFileExist(char *dir);
e. void closeChar();

Dan ADT wordmachine:

a. void IgnoreBlanks();
b. void STARTWORD();
c. void STARTWORDFILE(char *dir);
d. void ADVWORD();
e. void CopyWord();
f. void ADVNewline();
g. void displayWord(Word w);
h. int wordToInt(Word currentWord);
i. void createListWord(ListWord *L);
j. ListWord readLine();
k. ListWord readLineFile();
l. boolean isWordEq(Word w1, Word w2);
m. void copyWord(Word w1, Word *w2);
n. Word concatWord(Word w1, Word w2);
o. char *ListWordToString(ListWord L);
p. char *wordToString(Word w);

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.

3.9. Queue dengan List Dinamik dan Infotype



Di atas merupakan sketsa cara kerja prioqueue dengan menggunakan list dinamik. Tidak seperti list statik, list dinamik dapat diubah-ubah ukurannya dengan cara alokasi. Selain itu, elemen dalam prioqueue juga terurut membesar berdasarkan waktu expiry-nya. Berikut merupakan definisi dari prioqueue:

ADT PrioQueue:

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.

```c
typedef int address;        /* indeks tabel */
/* Contoh deklarasi variabel bertype PrioQueue : */
/* Versi I : tabel dinamik, Head dan Tail eksplisit, ukuran disimpan */
typedef struct {
    infotype * T;           /* tabel penyimpan elemen */
    address HEAD;           /* alamat penghapusan */
    address TAIL;           /* alamat penambahan */
    int MaxElPrioqueue;     /* Max elemen queue */
} PrioQueue;
```

Data dalam PrioQueue adalah T, HEAD, TAIL, dan MaxElPrioqueue. T merupakan tabel penyimpan elemen berupa list berisi infotype (akan dijelaskan di bawah), HEAD adalah alamat paling depan dalam queue berupa integer, TAIL adalah alamat paling belakang dalam queue berupa integer, dan MaxElPrioqueue adalah elemen maksimum dalam queue berupa integer.

ADT infotype:

```c
/* Definisi elemen dan address */
typedef struct {
    TIME time;              /* berisi waktu dalam menit */
    Makanan info;           /* makanan */
} infotype;
```

ADT infotype yang menjadi elemen dalam PrioQueue terdiri atas time dan info. time merupakan waktu yang sudah diubah ke dalam menit dengan tipe data long, dan info adalah makanan yang disimpan dalam PrioQueue berupa tipe data Makanan.

PrioQueue berfungsi sebagai inventory pada aplikasi. Inventory berfungsi untuk mengecek makanan yang dimiliki dan mengurutkannya berdasarkan waktu kedaluarsanya. Untuk menambah makanan pada inventory, diperlukan pembacaan queue sehingga makanan tersebut disimpan pada urutan yang benar pada queue sesuai waktu kedaluarsanya. Jika waktu kedaluarsa makanan telah mencapai nol, makanan yang ada di urutan paling depan akan terbuang dari inventory secara otomatis. PrioQueue juga diperlakukan sebagai list dinamik karena tidak menutup kemungkinan kalau makanan dari inventory digunakan dan diolah sehingga makanan tersebut tersingkir dari PrioQueue sebelum menjadi HEAD pada queue tersebut.

Kami menggunakan PrioQueue dengan pendekatan list dinamik karena makanan yang disimpan dalam inventory jumlahnya sering berubah-ubah seiring waktu, baik karena makanan kedaluarsa, penambahan makanan baru, maupun penggunaan makanan dari inventory untuk diolah. Dengan memanfaatkan prinsip list dinamik, ukuran inventory bisa disesuaikan berdasarkan kapasitas inventory dan makanan yang tersedia di inventory. Dengan memanfaatkan

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.

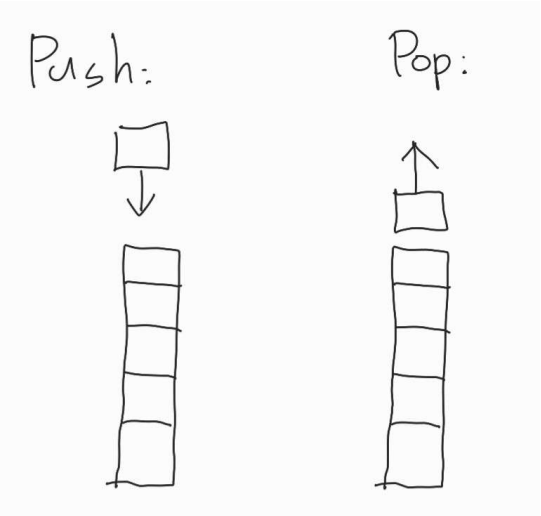
PrioQueue, makanan bisa diurutkan sesuai waktu kedaluarsanya sehingga lebih mudah diatur. Nama file: “Prioqueue.h”, “Prioqueue.c”

Adapun primitif ADT PrioQueue adalah:

1. Selektor
   a. Head(Q)
   b. Tail(Q)
   c. InfoHead(Q)
   d. InfoTail(Q)
   e. MaxElPrioqueue(Q)
   f. Elmt(Q,i)
2. Konstruktor/Destruktor
   a. void CreateEmptyPrioqueue (PrioQueue * Q, int Max);
   b. void DeAlokasi(PrioQueue * Q);
3. Prototipe
   a. boolean IsEmptyPrioqueue (PrioQueue Q);
   b. boolean IsFullPrioqueue (PrioQueue Q);
   c. int NBElmtPrioqueue (PrioQueue Q);
4. Add/delete
   a. void Enqueue1 (PrioQueue * Q, infotype X);
   b. void Enqueue (PrioQueue * Q, infotype X);
   c. void Dequeue (PrioQueue * Q, infotype * X);
   d. void Delete (PrioQueue * Q, int id, infotype * X) ;
   e. void DeleteAt (PrioQueue * Q, int idx, infotype * X);
   f. void CopyQueue (PrioQueue *Q1, PrioQueue * Q2);
5. Operasi tambahan
   a. void PrintPrioQueue (PrioQueue Q);
   b. void PrintPrioQueueDelivery (PrioQueue Q);
6. Pengurusan inventory dan delivery list
   a. void PasstimeQueue(PrioQueue * Q, int x, ListDinMakanan * notif);
   b. void PassTimeDelivery(PrioQueue * deliList, PrioQueue * destination, int x, ListDinMakanan * notif);
7. Pengakses
   a. int IndexOfPrioqueue(PrioQueue Q, infotype x);
   b. infotype GetElmtPrioqueue(PrioQueue Q, int idx);
   c.

3.10. Stack

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.



ADT Stack digunakan untuk menyimpan state pada simulator pada satu waktu tertentu di dalam struktur data mirip array, dengan pengubahan elemen hanya pada elemen di puncak.

State berisi data mengenai waktu dengan tipe data TIME, koordinat simulator terkini dengan tipe data POINT, inventory yang dimiliki simulator pada saat itu serta makanan yang sedang dalam proses delivery dengan tipe PrioQueue, kulkas yang dimiliki simulator dengan tipe MatrixKulkas, dan listDinMakanan yang mengandung elemen dengan tipe data makanan yang sudah berubah, baik itu hasil olahan setelah di-FRY dan lain sebagainya atau kedaluarsa.

Stack dibuat dengan versi I, yaitu dengan menyimpan tabel dan alamat TOP secara eksplisit. Stack memiliki data berupa list berisi state dengan kapasitas maksimal 1000, dan elemen TOP/puncak berupa integer.

Stack ini berfungsi untuk mengimplementasikan command UNDO/REDO pada aplikasi. Akan terdapat dua stack yang digunakan: stack main dan stack secondary. Tiap kali ada perubahan state, state yang baru akan dimasukkan ke stack main. Jika dilakukan command UNDO, elemen TOP pada stack main akan di-pop, dan elemen tersebut akan di-push ke stack secondary. Jika dilakukan command REDO, elemen TOP pada stack secondary akan di-pop, dan elemen tersebut akan di-push ke stack main. Kami memanfaatkan ADT Stack karena ADT ini sangat berguna untuk pengelolaan state dari awal sampai akhir dengan hanya memfokuskan state terkini dan state sebelumnya. Nama file: “stack.h”, “stack.c”.

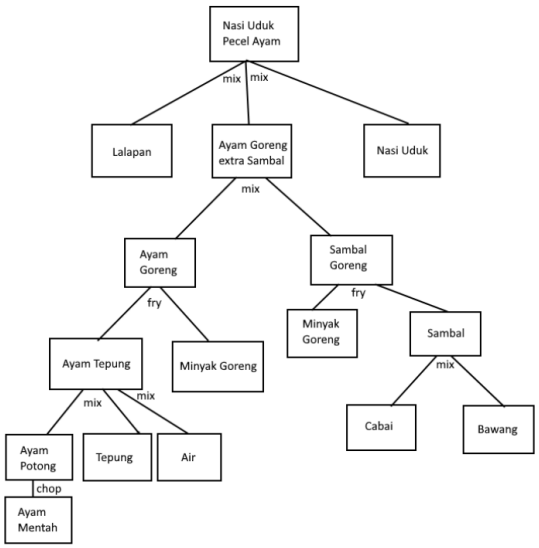
```
typedef struct {
  TIME waktu; // Waktu terkini
  POINT koordinat; // Posisi simulator
  PrioQueue inventory; // Prioqueue berupa makanan dalam inventory yang disertai waktu expired
  PrioQueue delivery; // Prioqueue berupa waktu delivery
  MatrixKulkas kulkas; // Isi kulkas terkini
  ListDinMakanan listMakanan; // List makanan yang berubah (misal abis di-FRY/di-BOIL/kedaluarsa)
} State; // State yang disimpan di dalam stack
typedef int address;   /* indeks tabel */

/* Contoh deklarasi variabel bertype stack dengan ciri TOP : */
/* Versi I : dengan menyimpan tabel dan alamat top secara eksplisit*/
typedef struct {
  State T[MaxStack]; /* tabel penyimpan elemen */
  address TOP;  /* alamat TOP: elemen puncak */
} Stack;
/* Definisi stack S kosong : S.TOP = Nil */
/* Elemen yang dipakai menyimpan nilai Stack T[0]..T[MaxStack-1] */
/* Jika S adalah Stack maka akses elemen : */
    /* S.T[(S.TOP)] untuk mengakses elemen TOP */
    /* S.TOP adalah alamat elemen TOP */
```


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.

Berikut primitif ADT Stack:

1. Selektor
   a. Top(S)
   b. InfoTop(S)
   c. InfoWaktu(X)
   d. InfoKoordinat(X)
   e. InfoInventory(X)
   f. InfoKulkas(X)
   g. InfoNotif(X)
   h. InfoDelivery(X)
2. Konstruktor
   a. void CreateEmptyStack(Stack *S);
3. Predikat
   a. boolean IsStackEmpty(Stack S);
   b. boolean IsStackFull(Stack S);
4. Operasi utama
   a. void Push(Stack * S, State X);
   b. void Pop(Stack * S, State* X);
   c. void EmptyStack(Stack * S);
   d. void Undo(Stack * SMain, Stack * SSecondary, State * currentState);
   e. void Redo(Stack * SMain, Stack *SSecondary, State *currentState);
   f. void ReallocateStack(Stack * S1);

3.11. Tree



ADT Tree digunakan untuk menyimpan resep, dengan contoh nasi uduk pecel ayam seperti gambar di samping, dengan nasi uduk pecel ayam sendiri sebagai root dari tree dan bahan-bahannya sebagai childrennya. Tiap penghubung antara parent dan child merupakan aksi-aksi pengolahan makanan seperti FRY dan MIX. Untuk root dari tree ada kemungkinan tidak hanya satu, dan resep perlu disimpan di list statik. Tree kami menggunakan pointer of pointer of Node pada struct Node sehingga kami dapat membuat array of node pointer. Pointer tersebut kami namai struct-nya Address. Sebuah TreeNode (sebuah

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.

elemen pada Tree) memiliki data bernama info yang berisikan ID Makanan, bukan elemen dengan ADT Makanan, sehingga tipe datanya yaitu integer; array of node pointer [0..9] dengan tipe elemen Address; dan childEff yang merupakan banyak children yang dimiliki sebuah node (misal ayam goreng memiliki 2 childEff, ayam tepung memiliki 3 childEff) dengan tipe integer. Kami menggunakan N-ary tree karena sebuah parent dapat memiliki jumlah child yang berbeda-beda (misal untuk gambar di samping, ayam goreng memiliki dua child sedangkan ayam tepung memiliki tiga child). Dengan menggunakan N-ary tree ini, kami dapat memetakan sebuah resep dengan efisien dan teratur agar bahan-bahan makanan tidak tercampur. Nama file: “tree.c”, “tree.h”

```c
typedef int ElTypeInt;
typedef struct treeNode* Address;
typedef struct treeNode {
    ElTypeInt info;
    Address children[MAX_CHIlD];
    int childEff;
} TreeNode;


/* Definisi Pohon */
/* pohon kosong p = NULL */

typedef Address Tree;
```

Berikut primitif ADT Tree:

1. Selektor
   a. ROOT(p)
   b. CHILDREN(p)
   c. CHILD_EFF(p)
2. Konstruktor
   a. Tree NewTree (ElTypeInt root, Tree children[]);
   b. void CreateTree (ElTypeInt root, Tree children[], Tree *p);
   c. Address newTreeNode(ElTypeInt val);
3. Lain-lain
   a. void addChild(Tree *p, Tree x);
   b. void deallocTreeNode (Address p);
   c. boolean isTreeEmpty (Tree p);
   d. boolean isOneElmt (Tree p);
   e. void printPreorder(Tree p);

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.

3.12. Game

```
typedef enum {
    MAKANAN = 1,
    RESEP = 2,
    PETA = 3
} ConfigType;

typedef struct {
    TIME currentTime;
    ListStatikM listMakanan;
    ListStatikT listResep;
    Map map;
    boolean endGame;
} Game;
```

ADT Game adalah ADT yang kami buat sendiri pada umumnya berguna sebagai penyimpan data aplikasi ini setelah dilakukan konfigurasi. Berikut sketsa ADT-nya:

listMakanan	listResep	map	endGame

ADT Game terdiri atas komponen-komponen berikut: listMakanan dengan tipe data listStatikM yang berisi semua makanan yang bisa digunakan di aplikasi, listResep dengan tipe data listStatikT yang berisi semua resep yang bisa digunakan di aplikasi, map dengan tipe data Map yang memiliki matriks karakter di peta, letak S (simulator) berupa POINT, dan daftar-daftar titik aksi seperti B, F, dll dalam ListStatikP, serta endGame dengan tipe Boolean yang menentukan apakah game masih berjalan atau tidak. ADT Game pada umumnya dioperasikan di program utama (main.c). List statik makanan, resep, dan koordinat titik aksi semuanya disimpan di dalam ADT Game ini. Penyimpanan data tersebut dapat memastikan aplikasi berjalan sesuai konfigurasi. Tidak hanya itu, terdapat enumerated type (tipe integer dengan konstanta yang di-assign ke value tersebut) bernama ConfigType untuk membedakan file konfigurasi apa yang akan dibaca sehingga metode pembacaan yang digunakan juga disesuaikan: MAKANAN = 1, RESEP = 2, PETA = 3. Kami memilih untuk membuat ADT baru untuk menyelesaikan masalah ini karena organisasi dan implementasi dalam mengolah makanan berdasarkan resep pada titik-titik tertentu akan berjalan lebih mudah jika semua hal esensial yang membuat game berfungsi dapat disimpan di dalam suatu struktur data. Nama file: “game.c”, “game.h”

Berikut primitif ADT Game:

a. void startGame(Game *game, Simulator *s);
b. boolean isEndGame(Game g);

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.

3.13. Map

```
typedef struct
{
    Matrix Peta;
    //POINT S_Map;
    ListStatikP T_Map;
    ListStatikP M_Map;
    ListStatikP F_Map;
    ListStatikP C_Map;
    ListStatikP B_Map;
    ListStatikP X_Map;
} Map;
```

Peta	S_Map	T_Map	M_Map	F_Map	C_Map	B_Map	X_Map

ADT Map mengandung data Peta dengan tipe data Matriks yang berfungsi menyimpan representasi peta dalam bentuk karakter, S_Map dalam bentuk POINT untuk menyimpan posisi terkini simulator, dan sisanya berupa ListStatikP (list statik yang menyimpan elemen POINT) untuk menyimpan berbagai koordinat titik aksi: T_Map untuk tempat menelepon, M_Map untuk tempat mixing, F_Map untuk tempat menggoreng, C_Map untuk tempat memotong, B_Map untuk tempat merebus, dan X_Map untuk tembok yang tidak bisa dilalui dan tidak bisa diapa-apakan. ADT ini digunakan untuk menyatakan koordinat-koordinat dari titik-titik penting pada peta dan menghubungkannya pada matriks peta yang telah dikonfigurasi. T_Map, M_Map, F_Map, C_Map, dan B_Map semuanya bersifat konstan dan tidak dapat diubah karena diperoleh dari file konfigurasi, namun koordinat simulator (S) dapat berubah-ubah sesuai command navigasi dari pengguna (MOVE). Kami memilih untuk membuat ADT baru untuk menyelesaikan persoalan ini karena membungkus matriks dengan daftar koordinat titik aksinya akan memudahkan untuk menentukan apakah simulator adjacent dengan titik aksi dan memastikan bahwa simulator tidak dapat menabrak atau melewati titik-titik aksi tersebut. Nama file: “map.c”, “map.h”

Berikut primitif ADT Map:

1. Selektor
    a. PETA(M)
    b. TMap(O)
    c. MMap(O)
    d. FMap(O)
    e. CMap(O)
    f. BMap(O)
    g. XMap(O)

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.

2. Konstruktor
   a. void CreateMap(Map *MGame, int X, int Y);
3. Display
   a. void DisplayMap(Map MGame, POINT sim);

3.14. Set

```
/* Definisi elemen dan koleksi objek */
typedef int IdxType;
typedef struct {
   int contents[CAPACITY_SET]; /* memori tempat penyimpan elemen (container) */
} Set;
```

ADT Set sekedar terlihat seperti ADT List Statik, namun dalam ADT ini kami membuat beberapa fungsi dan prosedur yang cocok untuk menerapkan konsep-konsep set seperti subset. ADT Set ini menyimpan integer berupa 0 dan 1, bergantung apa suatu bahan makanan tersedia/dibutuhkan di dalam inventory atau resep atau tidak. ADT Set ini berguna untuk mengimplementasikan fitur bonus rekomendasi makanan. Kami menggunakan ADT Set karena dengan konsep subset antara himpunan resep dan himpunan inventory, rekomendasi makanan yang diraih akan jauh lebih mudah dan akurat bagi pemain yang ingin membuat makanan berdasarkan inventory yang tersedia. Nama file: “set.c”, “set.h”.

Berikut primitif ADT Set:

1. Selektor
   a. ELMT_SET(s, i)
2. Konstruktor
   a. void createSet(Set *s);
3. Lain-lain
   a. int LengthSet(Set s);
   b. boolean isAnggota(int a, Set s);
   c. boolean isSubset(Set s, Set t);
   d. void makeSetFromListMakanan(Set *s, ListStatikM daftarMakanan);
   e. void makeSetFromInventory(Set *s, ListStatikM daftarMakanan, PrioQueue inventory);

4. Program Utama

Nama file: “main.c”

Saat program utama (“main.c” di folder core) dijalankan, splash screen ditampilkan dengan startMenu, dan sebuah listWord dibuat. Pengguna dapat mengetik command START atau EXIT pada tahap ini. Tiap input dari pengguna divalidasi - jika input salah, pengguna diminta untuk melakukan input lagi. Input command START yang valid akan disimpan di dalam listWord dan akan dilakukan prosedur startGame yang melakukan konfigurasi file makanan, resep dan

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.

peta secara otomatis. Waktu dimulai dari 0:0:0, simulator, listMakanan dan listResep kosong dibuat, serta lokasi simulator dimulai dari kiri atas. Grafik peta dan waktu terkini ditampilkan ke layar. game.Endgame yang menandakan apakah proses aplikasi berjalan atau tidak dinyatakan dengan false.

Pada tahap ini, pengguna dapat mengetik command. Input akan dimsukkan ke listWord, dan dari sebuah daftar command yang valid, akan dicek apakah input sesuai salah satu command dalam daftar tersebut. Fungsi MenuInput akan mengembalikan sebuah integer yang berkorespondensi dengan command yang diinput:

- 0: EXIT. Program akan berhenti, dengan game.Endgame di-assign dengan nilai true. Dilakukan validasi untuk memastikan bahwa command tidak memiliki argumen.
- 1: BUY. Pengguna dapat membeli makanan. Jika pengguna adjacent dengan titik lokasi 'T', akan ditampilkan daftar makanan yang dapat dibeli, dan pengguna dapat memilih makanan yang akan dibeli sesuai nomor makanan. Makanan yang dibeli akan dimasukkan ke delivery list, dan akan diantar sesuai waktu delivery-nya. Dilakukan validasi untuk memastikan bahwa command tidak memiliki argumen.
- 2: FRY. Pengguna hanya bisa menggoreng jika pengguna adjacent dengan titik lokasi 'F', dan jika masih terdapat minyak goreng di dalam inventory. Saat pengguna mengetik command FRY, akan ditampilkan daftar makanan yang bisa di-fry. Jika pengguna mengetik angka 0, command FRY dibatalkan. Jika pengguna ingin melanjutkan memasak, makanan yang telah digoreng akan dimasukkan ke inventory. Dilakukan validasi untuk memastikan bahwa command tidak memiliki argumen.
- 3: CHOP. Pengguna hanya bisa memotong jika pengguna adjacent dengan titik lokasi 'C'. Saat pengguna mengetik command CHOP, akan ditampilkan daftar makanan yang bisa di-chop. Jika pengguna mengetik angka 0, command CHOP dibatalkan. Jika pengguna ingin melanjutkan memasak, makanan yang telah dipotong akan dimasukkan ke inventory. Dilakukan validasi untuk memastikan bahwa command tidak memiliki argumen.
- 4: BOIL. Pengguna hanya bisa merebus jika pengguna adjacent dengan titik lokasi 'B'. Saat pengguna mengetik command BOIL, akan ditampilkan daftar makanan yang bisa di-boil. Jika pengguna mengetik angka 0, command BOIL dibatalkan. Jika pengguna ingin melanjutkan memasak, makanan yang telah direbus akan dimasukkan ke inventory. Dilakukan validasi untuk memastikan bahwa command tidak memiliki argumen.
- 5: MIX. Pengguna hanya bisa mencampur jika pengguna adjacent dengan titik lokasi 'B'. Saat pengguna mengetik command MIX, akan ditampilkan daftar makanan yang bisa di-mix. Jika pengguna mengetik angka 0, command MIX dibatalkan. Jika pengguna ingin melanjutkan memasak, makanan baru hasil mix akan dimasukkan ke inventory. Dilakukan validasi untuk memastikan bahwa command tidak memiliki argumen.

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.

- 6: HELP. Akan ditampilkan daftar command yang dapat dimanfaatkan pengguna. Dilakukan validasi untuk memastikan bahwa command tidak memiliki argumen.
- 7: INVENTORY. Akan ditampilkan daftar makanan yang dimiliki di inventory, berupa nama makanan dan sisa waktu kedaluarsanya, dengan PrintPrioQueue.
- 8: DELIVERY. Akan ditampilkan daftar makanan yang sedang diantar, berupa nama makanan dan sisa waktu pengantarannya, dengan PrintPrioQueueDelivery. Dilakukan validasi untuk memastikan bahwa command tidak memiliki argumen.
- 9: MOVE X. Command ini berguna sebagai navigasi simulator pada peta. Dilakukan validasi untuk memastikan bahwa command memiliki tepat satu argumen berupa: NORTH, WEST, SOUTH, atau EAST. Jika input valid, gerakan juga akan dianalisis: apakah gerakan simulator menabrak suatu titik aksi atau dinding/tembok dengan boolean stuck dalam prosedur gerakUser. Aksi simulator akan dikonversi ke state dan state tersebut akan di-push ke stack undo. Jika boolean stuck True, pergerakan gagal. Sebaliknya, lokasi baru simulator di-rewrite.
- 10: CATALOG. Akan ditampilkan makanan yang dapat digunakan dan diolah beserta informasinya. Dilakukan validasi untuk memastikan bahwa command tidak memiliki argumen.
- 11: COOKBOOK. Akan ditampilkan berbagai resep yang dapat digunakan. Masing-masing makanan dalam resep diperoleh dari tree resep dengan mengambil ID-nya, untuk kemudian dicetak nama makanan tersebut serta aksi yang dilakukan terhadap masing-masing makanan tersebut berdasarkan data “location” dalam ADT Makanan. Dilakukan validasi untuk memastikan bahwa command tidak memiliki argumen.
- 12: WAIT X Y. Pengguna dapat menunggu waktu tertentu dengan command WAIT x y dengan x adalah jam dan y adalah menit untuk diam tidak melakukan apa-apa dan hanya menambah waktu. Dilakukan validasi untuk memastikan bahwa command memiliki tepat dua argumen integer >= 0. Data simulator terkini diubah menjadi state dengan prosedur simulatorToState, dan state tersebut di-push ke stack undo (agar bisa meng-undo). game.currentTime diubah waktunya sesuai dengan input jam dan menit. Lokasi dan waktu terbaru ditampilkan di layar.
- 13: UNDO. Command UNDO bertujuan untuk membatalkan command yang dilakukan oleh simulator dan mengembalikan state aplikasi ke sebelum command itu dieksekusi. State terkini di-pop dari stack undo dan di-push ke stack redo, dan state tersebut diubah menjadi ADT simulator dan menyimpannya di simulator dengan loadState. Jika stack undo kosong, pesan “Tidak bisa UNDO” ditampilkan.
- 14: REDO. Command REDO bertujuan untuk membatalkan command UNDO. State terkini di-pop dari stack redo dan di-push ke stack undo, dan state tersebut diubah

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.

menjadi ADT simulator dan menyimpannya di simulator dengan loadState. Jika stack redo kosong, pesan "Tidak bisa REDO" ditampilkan.

- 15: MASUKKULKAS X Y. Command ini bertujuan untuk menyimpan sebuah makanan dari inventory ke kulkas. (penjelasan lebih lanjut di bab 2.1)
- 16: KELUARKULKAS X. Command ini bertujuan untuk mengambil sebuah makanan dari kulkas (penjelasan lebih lanjut di bab 2.1)
- 17: KULKAS. Command ini bertujuan untuk menampilkan isi kulkas.
- 18: REKOMENDASI. Command ini menunjukkan daftar rekomendasi makanan untuk pengguna berdasarkan makanan yang tersedia di inventory.

5. Algoritma-Algoritma Menarik

Ada beberapa algoritma menarik yang kelompok kami temukan dalam pengerjaan tugas besar kali ini. Berikut beberapa algoritma menarik.

5.1. Algoritma Undo-Redo

Algoritma undo-redo yang kami gunakan pada program ini idenya adalah menyimpan data-data dari simulator, list yang berisi notifikasi, *current time* dari game, *delivery list*, dan isi kulkas kedalam *datatype* state. State tersebut kemudian disimpan kedalam stack undo setiap ada command baru dari user, namun jika input command dari user melebihi 23 command maka program akan menghapus state paling tua dari stack tersebut agar memori yang digunakan program tidak besar.

Saat user melakukan undo, maka program akan menyimpan *current state* kedalam stack redo dan mengambil state teratas dari stack undo dan menyimpannya kedalam data-data current. Untuk redo cara kerjanya sama seperti undo tapi terbalik. Jika user melakukan undo dan kemudian menjalankan sebuah command baru, maka stack redo akan otomatis dihapus dan user tidak bisa melakukan redo lagi.

Salah satu error yang kami dapatkan adalah saat kami mengolah data-data yang kami miliki, menyimpannya ke stack undo, dan melakukan undo. Errornya berupa data yang kami ambil dari stack undo tersebut tidak di-load secara sempurna, dan terkadang ada beberapa data yang tertimpa dengan data yang lama. Kami mengatasinya dengan cara menyalin seluruh data yang kami punya kedalam temp dan state yang kami masukkan adalah state yang berisi data temp tersebut, dan kemudian mengolah data current berdasarkan command masukan user.

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.

5.2. Algoritma Kulkas

Algoritma kulkas yang kami pakai memungkinkan size makanan selain 1x1 dimasukkan kedalam kulkas. Algoritma yang kami gunakan adalah dengan mengambil makanan dari *inventory* dan kemudian menomori makanan tersebut, jika terdapat makanan yang sama dengan nomor yang sama maka program akan menampilkan pesan *error* sehingga tidak mungkin ada makanan yang sama dengan nomor yang sama.

Penomoran makanan tersebut dengan cara mengalikan ID makanan yang akan dimasukkan dengan 1000 dan kemudian ditambahkan dengan nomor input dari user.

Penyimpanan makanan tersebut dengan cara memasukkan index untuk point kiri atas dari makanan tersebut, dan index untuk point kanan bawah dari makanan tersebut. Algoritma seperti ini memungkinkan user untuk memasukkan makanan secara vertikal atau horizontal. Tentunya program akan mengecek apakah ada makanan pada tempat tersebut dan apakah index masukan pengguna sesuai dengan size makanannya.

Pengambilan makanan tersebut dengan cara memasukkan ID kulkas makanan tersebut, dan kemudian program akan menghapus makanan tersebut dari kulkas dan mengembalikan ID-nya seperti awal.

5.3. Algoritma Notifikasi

Algoritma yang kami gunakan adalah algoritma yang menggunakan data dari makanan yang termodifikasi. Jika ada sesuatu hal yang berkaitan dengan makanan yang berubah, seperti contohnya kadaluarsa maka program akan memasukkan data makanan tersebut kedalam listnotifikasi. Kemudian program akan menampilkan seluruh notifikasi yang terjadi saat user menggunakan command tersebut. Notifikasi ditampilkan dengan cara melihat data lokasi pada makanan tersebut.

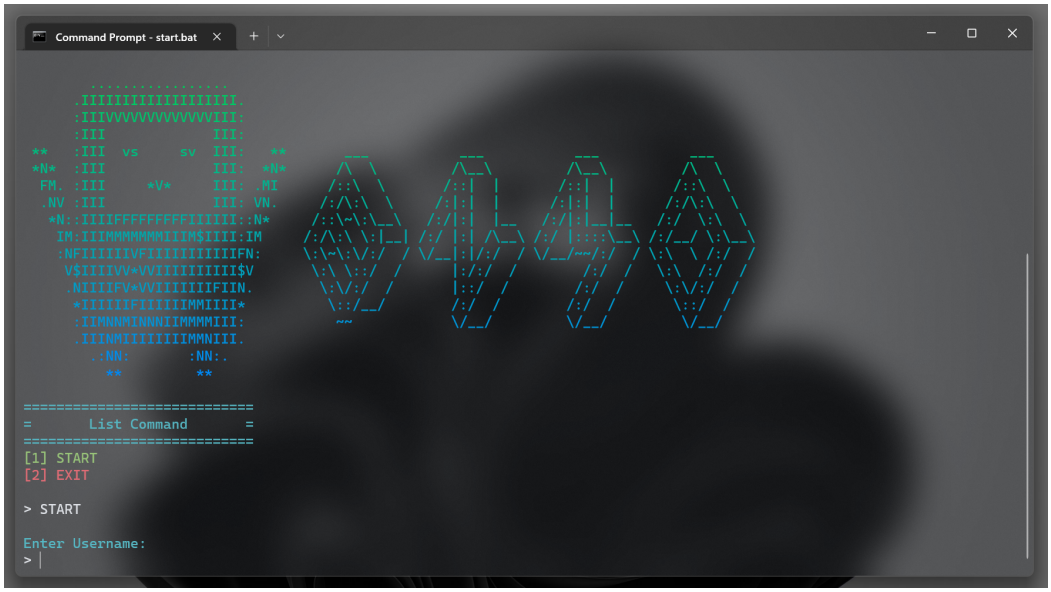
Penampilan notifikasi terdapat 2 macam, yaitu penampilan notifikasi biasa dan penampilan notifikasi saat undo. Penampilan notifikasi saat undo tentunya juga melihat data lokasi pada makanan tersebut.

6. Data Test

6.1. Data Test START

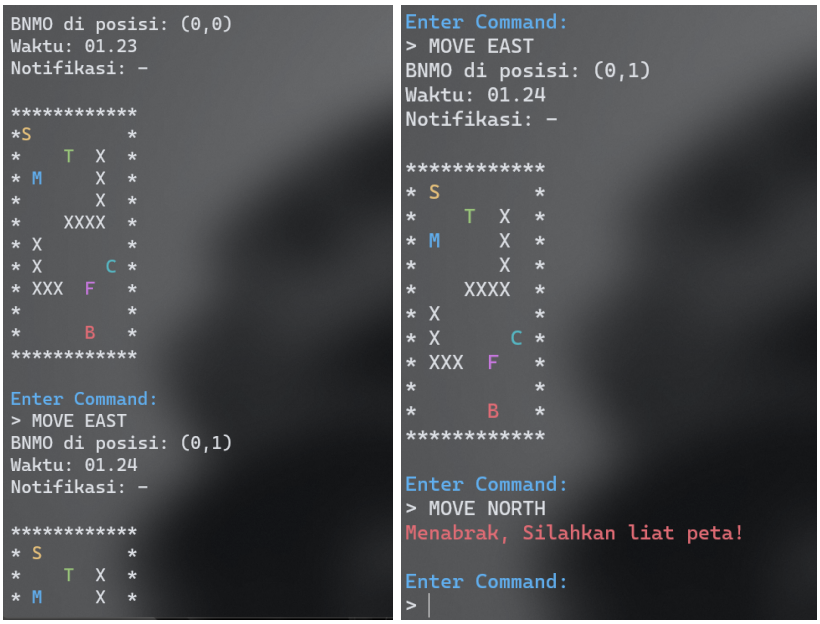
Saat memulai permainan, pemain dapat memilih untuk memulai permainan atau keluar dari game. Berikut tampilan jika pemain memanggil command START.

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.



6.2. Data Test MOVE

Pemain dapat berpindah pada peta dengan command MOVE. Ada 4 pilihan atau argumen untuk command MOVE, yakni MOVE NORTH, MOVE SOUTH, MOVE WEST, atau MOVE EAST. Berikut contoh tampilannya, dilengkapi dengan jika menabrak (input tidak valid).



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.

6.3. Data Test Pemesanan Makanan (Beserta DELIVERY dan INVENTORY)

Pemain dapat melakukan aksi BUY atau membeli makanan jika *adjacent* dengan lokasi 'T' (Telephone) pada peta. Pemain akan memilih makanan apa saja yang bisa dibeli dan makanan akan diantar berdasarkan waktu antar tiap makanan, makanan akan ditampilkan pada delivery list. Berikut contoh dari pemesanan makanan hingga masuk ke inventory.

```
BNMO di posisi: (0,4)
Waktu: 01.27
Notifikasi: -

************
*    S     *
*    T  X  *
* M     X  *
*       X  *
*    XXXX  *
* X        *
* X      C *
* XXX  F   *
*          *
*      B   *
************

Enter Command:
> BUY

=====================
=        BUY        =
=====================
List Bahan Makanan Yang Bisa Dibeli:
  [1] Ayam Mentah (15 menit)
  [2] Tepung (30 menit)
  [3] Minyak Goreng (1 jam)

Kirim 0 untuk exit.
```

```
Enter Command:
> BUY

=====================
=        BUY        =
=====================
List Bahan Makanan Yang Bisa Dibeli:
  [1] Ayam Mentah (15 menit)
  [2] Tepung (30 menit)
  [3] Minyak Goreng (1 jam)

Kirim 0 untuk exit.

Enter Command:
> 1

Berhasil memesan Ayam Mentah. Ayam Mentah akan diantar dalam 15 menit.

Enter Command:
> DELIVERY

      Food Is On The Way
=============================================
No  Nama                Arrive In
=============================================
 1  Ayam Mentah         15 menit

Enter Command:
>
```

```
Enter Command:
> WAIT 0 15
BNMO di posisi: (0,4)
Waktu: 01.44
Notifikasi:
- Ayam Mentah telah masuk ke inventory

************
*    S     *
*    T  X  *
* M     X  *
*       X  *
*    XXXX  *
* X        *
* X      C *
* XXX  F   *
*          *
*      B   *
************

Enter Command:
> INVENTORY

          INVENTORY
=============================================
No  Nama                Expired In
=============================================
 1  Ayam Mentah         1 hari
```


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.

6.4. Data Test Pengolahan Makanan (Beserta CATALOG dan COOKBOOK)

Pemain dapat melakukan aksi MIX, CHOP, FRY, BOIL (pengolahan makanan) jika memiliki bahan yang diperlukan di inventory. Pemain dapat melihat makanan apa saja yang bisa dibuat dan dibeli dari command CATALOG dan COOKBOOK. Setelah itu pemain dapat membuat suatu makanan berdasarkan resep makanan yang tersedia. Berikut contoh pengolahan makanan (termasuk yang gagal).

```
Enter Command:
> CATALOG
                    LIST MAKANAN
==================================================================================================
No  Nama                Expired in              Action                  Delivery time           Size
==================================================================================================
 1  Ayam Mentah         1  hari                 BUY                     15 menit               2 x 2
 2  Ayam Potong         2  jam                  CHOP (2  menit)         0                      2 x 2
 3  Tepung              1  jam                  BUY                     30 menit               1 x 1
 4  Minyak Goreng       1  jam   30 menit       BUY                     1  jam                  1 x 2
 5  Ayam Tepung         30 menit                MIX  (5  menit)         0                      2 x 2
 6  Ayam Goreng         15 menit                FRY  (10 menit)         0                      2 x 2

Enter Command:
> COOKBOOK

        COOKBOOK
==========================================
No  Resep
==========================================
 1  Ayam Tepung
    MIX - Ayam Potong - Tepung
 2  Ayam Potong
    CHOP - Ayam Mentah
 3  Ayam Goreng
    FRY - Ayam Tepung - Minyak Goreng
```

```
Enter Command:
> FRY

=======================
=         FRY         =
=======================
List Bahan Makanan yang Bisa Dibuat:
  [1] Ayam Goreng

Kirim 0 untuk exit.

Enter Command:
> 1

Gagal membuat Ayam Goreng karena kamu tidak memiliki bahan berikut:
  [1] Ayam Tepung
  [2] Minyak Goreng
```


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.

```
Enter Command:
> CHOP

======================
=        CHOP        =
======================
List Bahan Makanan yang Bisa Dibuat:
  [1] Ayam Potong

Kirim 0 untuk exit.

Enter Command:
> 1
Ayam Potong selesai dibuat dan sudah masuk inventory!

Enter Command:
> INVENTORY

INVENTORY
==============================================
No  Nama                 Expired In
==============================================
 1  Ayam Potong          2 jam
```

```
INVENTORY
==============================================
No  Nama                 Expired In
==============================================
 1  Ayam Tepung          22 menit
 2  Minyak Goreng        1 jam 7 menit

Enter Command:
> FRY

====================
=        FRY       =
====================
List Bahan Makanan yang Bisa Dibuat:
  [1] Ayam Goreng

Kirim 0 untuk exit.

Enter Command:
> 1
Ayam Goreng selesai dibuat dan sudah masuk inventory!

Enter Command:
> INVENTORY

INVENTORY
==============================================
No  Nama                 Expired In
==============================================
 1  Ayam Goreng          15 menit
```

```
INVENTORY
==============================================
No  Nama                 Expired In
==============================================
 1  Mie                  20 menit

Enter Command:
> BOIL

====================
=        BOIL      =
====================
List Bahan Makanan yang Bisa Dibuat:
  [1] Bakmie Kuah

Kirim 0 untuk exit.

Enter Command:
> 1
Bakmie Kuah selesai dibuat dan sudah masuk inventory!

Enter Command:
> INVENTORY

INVENTORY
==============================================
No  Nama                 Expired In
==============================================
 1  Bakmie Kuah          30 menit
```

```
INVENTORY
==============================================
No  Nama                 Expired In
==============================================
 1  Tepung               21 menit
 2  Ayam Potong          1 jam 51 menit

Enter Command:
> MIX

======================
=        MIX         =
======================
List Bahan Makanan yang Bisa Dibuat:
  [1] Ayam Tepung

Kirim 0 untuk exit.

Enter Command:
> 1
Ayam Tepung selesai dibuat dan sudah masuk inventory!

Enter Command:
> INVENTORY

INVENTORY
==============================================
No  Nama                 Expired In
==============================================
 1  Ayam Tepung          30 menit
```

6.5. Data Test Makanan Kadaluarsa

Makanan yang ada di inventory dapat kadaluwarsa jika sudah melebihi waktu kadaluwarsanya. Berikut contoh saat makanan kadaluwarsa.

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.

```
> INVENTORY

          INVENTORY
==========================================
No  Nama                Expired In
==========================================
 1  Ayam Goreng         15 menit

Enter Command:
> WAIT 0 15
BNMO di posisi: (6,6)
Waktu: 03.23
Notifikasi:
- Ayam Goreng telah kadaluarsa :(

************
*          *
*    T  X  *
* M     X  *
*       X  *
*    XXXX  *
* X        *
* X    S C *
* XXX  F   *
*          *
*      B   *
************

Enter Command:
> INVENTORY

Inventorynya kosong.. :(
```

6.6. Data Test Undo-Redo

Dalam permainan ini, pemain dapat meng-*undo* apa yang telah pemain perbuat dan juga me-*redo*, state akhir permainan di save dalam stack yang nantinya dapat dimanfaatkan oleh pemain jika semisal terjadi aksi yang tidak diinginkan. Undo-Redo berlaku pada semua keadaan pada game, seperti waktu, posisi, inventory, delivery list, dll. Semuanya akan muncul pada notifikasi setelah melakukan Undo-Redo. Berikut contoh Undo–Redo.

```
> INVENTORY

Inventorynya kosong.. :(

Enter Command:
> UNDO
BNMO di posisi: (6,6)
Waktu: 03.08
Notifikasi:
- Ayam Goreng dikembalikan ke inventory

************
*          *
*    T  X  *
* M     X  *
*       X  *
*    XXXX  *
* X        *
* X    S C *
* XXX  F   *
*          *
*      B   *
************

Enter Command:
> INVENTORY

          INVENTORY
==========================================
No  Nama                Expired In
==========================================
 1  Ayam Goreng         15 menit
```

```
> INVENTORY

          INVENTORY
==========================================
No  Nama                Expired In
==========================================
 1  Ayam Goreng         15 menit

Enter Command:
> REDO
BNMO di posisi: (6,6)
Waktu: 03.23
Notifikasi:
- Ayam Goreng telah kadaluarsa :(

************
*          *
*    T  X  *
* M     X  *
*       X  *
*    XXXX  *
* X        *
* X    S C *
* XXX  F   *
*          *
*      B   *
************

Enter Command:
> INVENTORY

Inventorynya kosong.. :(
```


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.

6.7. Data Test HELP

Pemain dapat melihat command-command apa saja yang sekiranya dapat dijalankan di game. HELP ini juga berfungsi agar pemain tidak salah mengetikkan command. Berikut contoh penggunaan command HELP.

```
Enter Command:
> HELP

      LIST COMMAND
=====================================
No  Command
=====================================
 1  MOVE X (X: NORTH/ SOUTH/ WEST/ EAST)
 2  BUY
 3  FRY
 4  CHOP
 5  BOIL
 6  MIX
 7  WAIT X Y (X: jam, Y: menit)
 8  INVENTORY
 9  DELIVERY
10  CATALOG
11  COOKBOOK
12  UNDO
13  REDO
14  MASUKKULKAS X Y (X: index makanan di Inventory, Y: nomor yang ingin dipakai)
15  KELUARKULKAS X (X: ID makanan di kulkas)
16  KULKAS
17  REKOMENDASI
18  HELP
19  EXIT
```

6.8. Data Test WAIT

Dalam permainan ini, pemain dapat menggunakan command WAIT untuk mempercepat waktu, ini berfungsi supaya mempercepat waktu saat delivery atau hanya sekedar ingin mempercepat waktu. Berikut contohnya.

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.

```
Enter Command:
> MOVE EAST
BNMO di posisi: (0,5)
Waktu: 01.45
Notifikasi: -

************
*     S    *
*    T  X  *
* M     X  *
*       X  *
*    XXXX  *
* X        *
* X      C *
* XXX  F   *
*          *
*      B   *
************

Enter Command:
> WAIT 0 15
BNMO di posisi: (0,5)
Waktu: 02.00
Notifikasi: -
```

```
Enter Command:
> DELIVERY

      Food Is On The Way
==============================================
No   Nama                    Arrive In
==============================================
 1   Tepung                  30 menit

Enter Command:
> WAIT 0 45
BNMO di posisi: (0,4)
Waktu: 02.49
Notifikasi:
- Tepung telah masuk ke inventory
```

6.9. Data Test MASUKKULKAS/ KELUARKULKAS

Dikarenakan makanan pada inventory memiliki waktu kadaluwarsa, pemain di sini dapat menyimpan makanannya di kulkas agar makanannya tidak kadaluwarsa. Saat di kulkas makanan tidak berkurang waktu kadaluwarsanya sehingga pemain dapat memanfaatkannya jika ingin makanannya awet. Berikut contohnya.

```
Enter Command:
> MASUKKULKAS 1 6
    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0
    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0
    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0
    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0
    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0
    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0
    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0
    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0
    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0
    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0

Masukkan index pojok kiri atas, dan index pojok kanan bawah
Urutannya: kiri atas kanan bawah
> 0 0 0 0

Enter Command:
> KULKAS
10006|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0
    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0
    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0
    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0
    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0
    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0
    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0
    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0
    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0
    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0
```


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.

```
10006|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0
    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0
    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0
    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0
    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0
    0|    0|    0|    0|    0|11008|11008|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0
    0|    0|    0|    0|    0|11008|11008|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0
    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0
    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0
    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0

Enter Command:
> KELUARKULKAS 11008
10006|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0
    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0
    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0
    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0
    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0
    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0
    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0
    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0
    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0|    0

Enter Command:
> INVENTORY

          INVENTORY
=============================================
No  Nama                Expired In
=============================================
 1  Ayam Mentah         23 jam 42 menit
```

6.10. Data Test REKOMENDASI

Dalam permainan ini, jika pemain memiliki bahan makanan di Inventory tetapi tidak tahu ingin membuat apa dengan bahannya, maka pemain dapat memasukkan command REKOMENDASI agar aplikasi merekomendasikan makanan yang bisa dibuat dari bahan yang ada di Inventory pemain. Berikut contohnya.

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.

```
Enter Command:
> INVENTORY

          INVENTORY
=================================================
No   Nama                  Expired In
=================================================
 1   Ayam Mentah           23 jam 42 menit

Enter Command:
> REKOMENDASI

REKOMENDASI
==============================
No   Nama
==============================
 1   Ayam Potong
```

```
Enter Command:
> INVENTORY

Inventorynya kosong.. :(

Enter Command:
> REKOMENDASI

Inventory kosong, tidak ada rekomendasi makanan.
```

6.11. Data Test EXIT

Jika pemain ingin memberhentikan game, pemain dapat memasukkan command EXIT, program akan memberhentikan game.

```
Enter Command:
> EXIT

Semangat yak ><, anakIF
Terima kasih telah menggunakan BNMO
```

7. Test Script

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.

No	Fitur yang Dites	Tujuan Testing	Langkah-Langkah Testing	Input Data Test	Hasil yang Diharapkan	Hasil yang Keluar
1	START	Untuk memulai permainan	Memberi command "START"	Data test 1	Permainan dimulai dengan meminta username pemain.	Sesuai yang diharapkan.
2	MOVE	Untuk berpindah-pindah pada peta	Memberi command "MOVE" diikuti dengan arah yang valid, yaitu "EAST", "WEST", "SOUTH", "NORTH".	Data test 2	Simulator berpindah tempat jika tidak menabrak, dan memberikan pesan jika menabrak	Sesuai yang diharapkan.
3	BUY	Membeli makanan yang tersedia dari config	Memberi command "BUY" dengan syarat harus *adjacent* dengan lokasi "T".	Data test 3	Menampilkan list makanan yang dapat dibeli dilanjutkan dengan memilih makanan yang dibeli	Sesuai yang diharapkan.
4	DELIVERY	Untuk menampilkan makanan yang sedang dipesan.	Memberi command "DELIVERY".	Data test 3	Menampilkan list makanan yang masih dalam waktu pemesanan.	Sesuai yang diharapkan.
5	INVENTORY	Untuk menampilkan makanan yang ada pada inventory simulator	Memberi command "INVENTORY"	Data test 3	Menampilkan list makanan dan bahan yang ada pada inventory simulator	Sesuai yang diharapkan.
6	MIX	Untuk mencampurkan 2 bahan makanan yang bisa di-mix dan terdapat pada inventory	Memberi command "MIX" dengan syarat harus *adjacent* dengan lokasi "M".	Data test 4	Menampilkan list makanan yang dapat dibuat dengan cara mix. Dilanjutkan dengan memilih makanan yang ingin dibuat, menampilkan pesan gagal mix jika bahan makanannya tidak lengkap dan menampilkan bahan yang kurang tersebut.	Sesuai yang diharapkan.
7	CHOP	Untuk memotong bahan makanan yang bisa dipotong dan	Memberi command "CHOP" dengan syarat harus *adjacent* dengan lokasi "C"	Data test 4	Menampilkan list makanan yang dapat dibuat dengan cara	Sesuai yang diharapkan.

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.

No	Fitur yang Dites	Tujuan Testing	Langkah-Langkah Testing	Input Data Test	Hasil yang Diharapkan	Hasil yang Keluar
		terdapat pada inventory			dipotong. Dilanjutkan dengan memilih makanan yang dapat dibuat.	
8	FRY	Untuk menggoreng makanan yang bisa digoreng dan terdapat pada inventory	Memberi command "FRY" dengan syarat harus *adjacent* dengan lokasi "F"	Data test 4	Menampilkan list makanan yang dapat dibuat dengan cara digoreng. Dilanjutkan dengan memilih makanan yang dapat dibuat.	Sesuai yang diharapkan.
9	BOIL	Untuk merebus makanan yang dapat direbus dan terdapat pada inventory	Memberi command "BOIL" dengan syarat harus *adjacent* dengan lokasi "B"	Data test 4	Menampilkan list makanan yang dapat dibuat dengan cara direbus. Dilanjutkan dengan memilih makanan yang dapat dibuat	Sesuai yang diharapkan.
10	CATALOG	Untuk menampilkan informasi secara detail dari makanan-makanan yang ada di config	Memberi command "CATALOG"	Data test 4	Menampilkan list makanan yang ada di config dengan informasi detail seperti nama, waktu expired, action yang dapat dilakukan, waktu delivery, dan ukuran makanannya.	Sesuai yang diharapkan.
11	COOKBOOK	Untuk menampilkan resep makanan yang ada pada config	Memberi command "COOKBOOK"	Data test 4	Menampilkan resep makanan yang ada pada config beserta dengan aksi dan bahan yang dibutuhkan.	Sesuai yang diharapkan.
12	UNDO	Untuk me-*undo* yang dilakukan simulator	Memberi command "UNDO"	Data test 6	Kembali ke state sebelumnya.	Sesuai yang diharapkan.
13	REDO	Untuk me-redo yang dilakukan simulator	Memberi command "REDO"	Data test 6	Mengulang ke state yang telah di-undo sebelumnya.	Sesuai yang diharapkan.

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.

No	Fitur yang Dites	Tujuan Testing	Langkah-Langkah Testing	Input Data Test	Hasil yang Diharapkan	Hasil yang Keluar
14	HELP	Untuk menampilkan command yang valid pada game ini.	Memberi command “HELP”	Data test 7	Menampilkan 19 command yang dapat ditulis.	Sesuai yang diharapkan.
15	WAIT	Untuk mempercepat waktu.	Memberi command “WAIT X Y”, dengan X adalah dalam jam dan Y dalam menit.	Data test 8	Current time atau waktu menjadi sesuai dengan penjumlahan durasi wait dengan waktu sebelumnya..	Sesuai yang diharapkan.
16	MASUKKULKAS	Untuk memasukkan makanan yang ada pada inventory simulator ke dalam kulkas	Memberi command “MASUKKULKAS X Y”, dengan X adalah indeks atau urutan makanan di inventory dan Y adalah nomor unik makanan di kulkas	Data test 9	Memilih lokasi makanan yang ingin dimasukkan ke dalam kulkas dengan indeks yang sesuai pada kulkas, jika sudah maka makanan yang dimasukkan tadi akan hilang dari inventory dan akan masuk kulkas	Sesuai yang diharapkan.
17	KELUARKULKAS	Untuk mengeluarkan makanan yang ada di kulkas untuk dimasukkan kembali ke inventory	Memberi command “KELUARKULKAS X”, dengan X adalag id unik makanan di kulkas.	Data test 9	Makanan yang dipilih tadi akan hilang dari kulkas dan akan dimasukkan kembali ke inventory	Sesuai yang diharapkan.
18	REKOMENDASI	Untuk menampilkan rekomendasi makanan yang dapat dibuat dari bahan dan makanan yang ada pada inventory simulator.	Memberi command “REKOMENDASI”.	Data test 10	Menampilkan makanan yang dapat dibuat dari bahan dan makanan yang ada pada inventory simulator.	Sesuai yang diharapkan.
19	EXIT	Untuk keluar dan mengakhiri permainan.	memberi command “EXIT”.	Data test 11	Permainan akan berakhir.	Sesuai yang diharapkan.

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.

8. Pembagian Kerja dalam Kelompok

No	NIM	Nama	Pembagian kerja
1	13521041	Muhammad Hanan	Debugging, ADT list statik, ADT list dinamis, ADT map, ADT point, utils, laporan.
2	13521048	M. Farrel Danendra Rachim	Laporan (sebagian besar), Stack & UNDO/REDO, ADT List Dinamik, ADT State, ADT Simulator, merapikan kode
3	13521055	Muhammad Bangkit Dwi Cahyono	ADT Makanan, ADT Mesin Karakter & Kata, struktur kode, alur aplikasi, ADT color, game, utils, debugging, file konfigurasi, membuat command, merapikan kode, membuat size makanan dan waktu pengolahan makanan, interface, laporan.
4	13521065	Mutawally Nawwar	ADT Tree, ADT Time, laporan, ADT Set, rekomendasi makanan.
5	13521069	Louis Caesa Kesuma	ADT Simulator, kulkas, prioqueue, infotype, mekanisme penyimpanan dan pengambilan makanan kulkas, mekanisme wait untuk inventory dan delivery list, mekanisme notifikasi, mekanisme notifikasi saat undo/redo, debugging, laporan.

9. Lampiran

9.1. Deskripsi Tugas Besar 2

BNMO (dibaca: Binomo) adalah sebuah robot game milik Indra dan Doni. Akhir-akhir ini, Indra baru saja menjalin hubungan spesial dengan perempuan bernama Siska Kol. Dan dalam dekat waktu, Indra akan mengajak Siska Kol ke rumah untuk makan malam bersama Doni dan BNMO. Oleh karena itu, Indra meminta bantuan BNMO dan Doni untuk membantu mempersiapkan makan malam spesial tersebut. Saat itu juga, BNMO langsung tertarik untuk mengerjakan bagian masak karena ia sangat sering melihat video memasak di aplikasi toktok dan sangat terngiang-ngiang dengan “mari kita cobaaa”.

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.

Namun, ada masalah. BNMO tidak tahu cara memasak dan Doni tidak bisa membantu persiapan karena ada hal lain. BNMO tidak bisa belajar dari video youcub karena BNMO adalah sebuah komputer sehingga hal yang paling mudah untuk dilakukan adalah membuatkan program simulasi untuk ditiru BNMO. Oleh karena itu, Doni meminta bantuan kalian untuk membuatkan program simulasi tersebut.

9.2. Notulen Rapat

Rapat 1: Minggu, 23 Oktober 2022

- Kita bagi-bagi tugas dulu aja, mulai dari ADT yang sederhana dulu dari 1-6. ADT Tree dikerjainnya nanti aja soalnya kita belum diajarin. Nanti tanya kakak asistennya aja.
- Masih bingung undo/redo-nya, itu kita perlu masukin commandnya ke dalam stack terus kalau undo itu buat inversnya, atau kita taruh state terkini dalam stacknya? Nanti tanya kakaknya aja
- Size makanan itu seberapa ya? Apa sizenya bisa melebihi 1x1? Nanti tanya kakaknya aja
- Udah disetup repo githubnya, jangan lupa download CMake-nya buat nge-compile.
- Asistensinya kapan? Kira-kira Selasa aja ya, nanti japri aja kakaknya.

Rapat 2: Sabtu, 29 Oktober 2022

- Commit-nya ke branch dulu supaya bisa pull-request.
- Kita bagi-bagi tugas lagi karena ADT-nya udah selesai:

  Next Task:

  - yang ngerjain tree → bikin untuk read configuration + create tree
  - yang ngerjain map → bikin untuk read config + create map (inisiasi map + print)
  - simulator → create + load
  - stack → bantu simulator untuk stack (inisiasi waktu)
  - main/ alur program/ game
  - adt makanan → inisiasi config file
  - tambahan, yang bikin TIME → bikin setTime biar bisa akses currentTime
- Untuk sementara di driver masing2 aja karena mainnya on progres. Kalau mau ngepush komen aja agar ga berpengaruh ke main.
- main.c belum dibuat apa-apa.
- Udah pada pake CMake? Itu lumayan bagus kalau mau nge-debug sebelum ngepush.
- Tiap kali udah selesai nge-commit-nya, isi log activity.
- Move north move south ga dijadiin prosedur aja?

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.

9.3. Log Activity Anggota Kelompok

	Bangkit	Farrel	Louis	Nawwar	Hanan
23/10/2022	Initial commit Repository GitHub				
24/10/2022	Structure code + splash screen	Selesai laporan bab 1			
25/10/2022					
26/10/2022					
27/10/2022		Sudah commit stack.c			
28/10/2022		Selesai commit stack.c			Add ADT Point, ADT list statik dan dinamis.
29/10/2022		Selesai laporan bab 3, update bab 9	Selesai Simulator (spesifikasi wajib)	add adt matrix, time	Update list statik
30/10/2022		Update undo & redo	Selesai Prioqueue		
31/10/2022		Buat ADT state		add read config resep,matrix kulkas	Update list statik, add read config map, add ADT map
01/11/2022	ADT machine, ADT makanan FIX	Tambah listdinmakanan.h beserta fungsinya		Update read config resep	Update ADT point, ADT map, update read config map

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.

02/11/2022					Fix ADT map
03/11/2022					
04/11/2022					
05/11/2022					
06/11/2022	Debugging MOVE, WAIT, add size makanan (bonus)				
07/11/2022				Adding cookbook	
08/11/2022	Debugging UNDO REDO		Selesai kulkas	Add feature fry, boil, mix, chop	Add feature Buy.
09/11/2022					
10/11/2022					
11/11/2022	Adding interface	Tambah ADT set, Debugging		Adding bonus rekomendasi makanan, Fixing bug at read config resep	Debugging
12/11/2022	Fixing bugs on DELIVERY and PASSTIME dengan kelompok	Debugging	Selesai notif		Debugging
13/11/2022					
14/11/2022				Adding undo redo di	

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.

				mengolahMakanan	
15/11/2022	Adding username (fixed bugs)				
16/11/2022	Debugging				
17/11/2022	Rapih-rapih and finishing, selesain laporan		Fix kasus jika ada makanan yang delivery timenya 0, menambah beberapa algoritma menarik, fix notifikasi, dan debugging		Debugging
18/11/2022	Rapih-rapih + debugging lagi	Rapihin laporan			Debugging dan selesain dan rapihin laporan
19/11/2022					
20/11/2022					

9.4. Form Asistensi Tugas Besar

Form Asistensi Tugas Besar
IF2110/Algoritma dan Struktur Data
Sem. 1 2021/2022

No. Kelompok/Kelas : B/1
Nama Kelompok : B
Anggota Kelompok (Nama/NIM) :
1. Muhammad Hanan/13521041
2. M. Farrel Danendra Rachim/13521048

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.

3. M. Bangkit Dwi Cahyono/13521055
4. Mutawally Nawwar/13521065
5. Louis Caesa Kesuma/13521069

Asisten Pembimbing : Kadek Surya Mahardika

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.

Asistensi I

Tanggal : 25 Oktober 2022	**Catatan Asistensi:**
Tempat : Zoom	● Untuk undo, perlu menyimpan state kondisi waktu, inventory, lokasi, dan lain-lain? Apakah undo sampai awal sekali? Tanya saja di QnA. ● Anggap saja size makanan dalam kulkas itu 1x1. Jadi maksimal kapasitas 200 makanan. ● Tree belum diajarkan di kelas pada saat asistensi ini. Slidenya juga belum dikasih. Cari saja referensinya di Ding Dong. ● Misal ada ayam goreng bisa dilakukan aksi di-fry. bisa di-mix, di-chop. Bisa, tapi kalau aksinya hanya satu saja boleh saja. ● Jika program ditutup apa perlu di-save? Boleh saja tambahkan fungsionalitasnya, tapi tidak wajib. ● Konfigurasi file makanan, resep, dll apakah beda filenya atau disatukan? Mestinya dipisah-pisah. ● Untuk menyimpan state yang dibaca yang diubah secara runtime. ● Kalau ada dua makanan yang waktu kedaluwarsa sama, keduanya ditulis di inventory ● Minyak goreng termasuk makanan ● Milestone yang diluar tabel tidak perlu dicantumkan. ● Apa progress milestone dinilai? Belum dibahas oleh asisten – mungkin masuk mungkin tidak.

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.

Kehadiran Anggota Kelompok: No NIM Tanda tangan 1 13521041 2 13521048 3 13521055 4 13521065 5 13521069	

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.

	Tanda Tangan Asisten: Kadek Surya Mahardika 13519165

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.

Asistensi II

Tanggal : 3 November 2022	**Catatan Asistensi:**
Tempat : Zoom	● Demonstrasi program sementara ● Kalau command invalid tidak ditunjukkan peta. ● Laporan juga sudah lumayan. ● Kalau sketsa ADTnya gt gapapa kan kak? Gapapa.

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.

Kehadiran Anggota Kelompok: No NIM Tanda tangan	
1 13521041 	
2 13521048 	
3 13521055 	
4 13521065 	
5 13521069 	

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.

	Tanda Tangan Asisten: Kadek Surya Mahardika 13519165

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.