

**LAPORAN TUGAS BESAR**  
**IF2110 / Algoritma dan Struktur Data**

**BNMO COOKING SIMULATOR**


Dipersiapkan oleh :

Kelompok D - K03

13521001    Angger Ilham Amanullah  
13521005    Kelvin Rayhan Alkarim  
13521010    Muhamad Salman Hakim Alfarisi  
13521019    Ditra Rizqa Amadia  
13521023    Kenny Benaya Nathan  
13521031    Fahrian Afdholi

Sekolah Teknik Elektro dan Informatika - Institut Teknologi Bandung

Jl. Ganesha 10, Bandung 40132

	<b>Sekolah Teknik Elektro dan Informatika ITB</b>	<b>Nomor Dokumen</b>		<b>Halaman</b>
		<i>IF2110-TB-D-03</i>		<i>31</i>
		<i>Revisi</i>	<i>1</i>	<i>20-11-2022</i>



# Daftar Isi

<b>1 Ringkasan</b>	<b>4</b>
<b>2 Penjelasan Tambahan Spesifikasi Tugas</b>	<b>5</b>
2.1 Undo Redo	5
<b>3 Struktur Data (ADT)</b>	<b>5</b>
3.1 ADT Array of Character	5
3.2 ADT Boolean	5
3.3 ADT Character Machine	5
3.4 ADT Character Machine of Food	6
3.5 ADT Character Machine of Recipe	6
3.6 ADT Dynamic List	6
3.7 ADT Delivery	6
3.8 ADT Food	6
3.9 ADT Inventory	6
3.10 ADT Game State	6
3.11 ADT Static List	7
3.12 ADT Map	7
3.13 ADT Matrix	7
3.14 ADT Matrix of Character	7
3.15 ADT Point	7
3.16 ADT Queue	7
3.17 ADT Queue Linked	7
3.18 ADT Stack	8
3.19 ADT Stack of Game State	8
3.20 ADT String	8
3.21 ADT Time	8
3.22 ADT Tree	8
3.23 ADT Word Machine	8
3.24 ADT Word Machine of Food	8
3.25 ADT Word Machine of Recipe	8
<b>4 Program Utama</b>	<b>9</b>
<b>5 Algoritma-Algoritma Menarik</b>	<b>9</b>
5.1 Tree	9

<b>6 Data Test</b>	<b>12</b>
6.1 Data Map 1	12
6.2 Data Map 2	13
6.3 Data Move	13
6.4 Data Move (Undo)	15
6.5 Data Move (Redo)	16
6.6 Data Buy	16
6.7 Data Start	17
6.8 Data Inventory	18
6.9 Data Delivery	18
6.10 Data Wait X Y	18
6.11 Data Chop	19
6.12 Data Mix	19
6.13 Data Fry	20
6.14 Data Boil	20
6.15 Data Exit	21
6.16 Data Catalog	21
6.17 Data Cookbook	22
6.18 Data Notifikasi	22
<b>7 Test Script</b>	<b>23</b>
<b>8 Pembagian Kerja dalam Kelompok</b>	<b>25</b>
<b>9 Lampiran</b>	<b>25</b>
9.1 Deskripsi Tugas Besar	25
9.2 Notulen Rapat	26
9.2.1 Asistensi	26
9.3 Log Activity Anggota Kelompok	29
9.4 Progress Milestone	30

# 1 Ringkasan

Tugas Besar Algoritma dan Struktur Data (IF2110) 2022 adalah *project* membuat program simulasi memasak “BNMO Cooking Simulator” yang berbasis *command-line interface* (CLI) dengan menggunakan bahasa C. *User* dapat memainkan program ini dengan memasukkan *input command* sehingga dapat berpindah pada peta. Selain itu, *user* dapat melakukan pembelian makanan dan pengolahan makanan seperti *fry*, *chop*, *boil*, dan *mix*. Dalam program ini *user* juga dimudahkan dengan adanya fitur notifikasi untuk bahan makanan yang kedaluwarsa, fitur *delivery* untuk melihat bahan makanan yang sedang dikirim, fitur *inventory* untuk melihat bahan makanan yang disimpan dalam *inventory*, fitur *cookbook* untuk melihat resep, serta fitur *catalog* untuk melihat semua *list* bahan makanan yang ada pada program ini.

Pada laporan ini dibahas mengenai penjelasan tambahan spesifikasi tugas, struktur data, program utama, *data test*, *test script*, dan pembagian kerja kelompok.

## 2 Penjelasan Tambahan Spesifikasi Tugas

### 2.1 Undo Redo

Fitur Undo dan Redo dapat digunakan ketika *user* sudah melakukan aksi. Apabila belum terdapat aksi yang dilakukan, program akan meminta *command* lain. Hal ini dikarenakan, *command* ‘UNDO’ digunakan untuk membatalkan aksi, sedangkan *command* ‘REDO’ digunakan untuk membatalkan *undo*, apabila belum ada aksi maka kedua *command* tersebut tidak dapat dilakukan.

## 3 Struktur Data (ADT)

*Abstract Data Type* (ADT) yang digunakan pada program BNMO Cooking Simulator sebagai berikut.

### 3.1 ADT Array of Character

ADT *Array of Char* adalah *Array* dengan elemen-elemennya bertipe *char*. ADT ini digunakan sebagai pengganti *String*. *String* pada C tidak dapat diubah maupun di-*return*. Maka dari itu ADT *Array of Char* dijadikan alternatif ADT *String* yang lebih fleksibel. ADT ini didefinisikan dan diimplementasikan pada folder ‘*/src/adt/arrayofChar/*’.

### 3.2 ADT Boolean

ADT *Boolean* adalah ADT yang digunakan untuk mengirimkan nilai *true* atau *false* dalam pengecekan sebuah kondisi. ADT ini didefinisikan dan diimplementasikan pada folder ‘*/src/adt/boolean/*’.

### 3.3 ADT Character Machine

ADT *Character Machine* adalah ADT berisi metode-metode membaca suatu file konfigurasi. ADT ini didefinisikan dan diimplementasikan pada folder ‘*/src/adt/charmachine/*’.

### 3.4 ADT Character Machine of Food

ADT *Character Machine of Food* adalah ADT *Character Machine* yang digunakan untuk pembacaan file konfigurasi makanan. ADT ini didefinisikan dan diimplementasikan pada folder ‘/src/adt/charmachinefood/’.

### 3.5 ADT Character Machine of Recipe

ADT *Character Machine of Recipe* adalah ADT *Character Machine* yang digunakan dalam pembacaan file konfigurasi resep. ADT ini didefinisikan dan diimplementasikan pada folder ‘/src/adt/charmachineresep/’.

### 3.6 ADT Dynamic List

ADT *Dynamic List* adalah *array* dengan ukuran yang bersifat dinamis. Implementasi ADT ini meliputi konstruktor, selektor, interaksi *input/output*, dan operasi penambahan elemen. ADT ini didefinisikan dan diimplementasikan pada folder ‘/src/adt/listdin/’.

### 3.7 ADT Delivery

ADT *Delivery* adalah ADT yang digunakan untuk fitur *Delivery* dengan memanfaatkan ADT *Food* dan *Queue Linked* untuk aksinya. Struktur Data *Linked List* digunakan pada *Priority Queue Delivery* agar mempermudah proses *dequeue* pada *list delivery*. ADT ini didefinisikan dan diimplementasikan pada folder ‘/src/adt/delivery/’.

### 3.8 ADT Food

ADT *Food* adalah ADT yang berisikan ID makanan, nama makanan, waktu *expired* makanan, waktu *delivery* makanan, dan aksi yang dilakukan. *Food* memanfaatkan ADT *String*, *Time*, dan *Word Machine* untuk struktur datanya. ADT ini didefinisikan dan diimplementasikan pada folder ‘/src/adt/food/’.

### 3.9 ADT Inventory

ADT *Inventory* adalah ADT yang digunakan untuk menggabungkan ADT *food*, dan *Queue linked*. Struktur Data *Linked List* digunakan pada *Priority Queue Delivery* agar mempermudah proses *dequeue* pada *list inventory*. ADT ini didefinisikan dan diimplementasikan pada folder ‘/src/adt/inventory/’.

### 3.10 ADT Game State

ADT *Game State* adalah ADT *field-field* berupa variabel untuk menyatakan suatu kondisi dari permainan. Field pada ADT ini yaitu posisi *simulator* pada peta, aksi yang dapat dilakukan *simulator*, dan waktu pada *simulator*. *Game State* memanfaatkan ADT *Matrix of Character* untuk membantu pemosisian *simulator* pada peta. ADT ini didefinisikan dan diimplementasikan pada folder ‘/src/adt/gameState/’.

### 3.11 ADT Static List

ADT Static List adalah *array* dengan ukuran yang bersifat statis. Implementasi ADT ini meliputi konstruktor, selektor, interaksi *input/output*, dan operasi penambahan elemen. ADT ini didefinisikan dan diimplementasikan pada folder ‘/adt/liststatik’.

### 3.12 ADT Map

ADT *Map* adalah ADT yang digunakan untuk membuat *map* dari *config*. *Map* memanfaatkan ADT *Word Machine* dan *Matrix of Character*. ADT ini didefinisikan dan diimplementasikan pada folder ‘/src/adt/map’.

### 3.13 ADT Matrix

ADT Matrix adalah ADT yang memiliki struktur matrix di dalamnya. ADT Matrix didefinisikan dan diimplementasikan pada folder ‘/src/adt/matrix’.

### 3.14 ADT Matrix of Character

ADT *Matrix of Character* adalah *Matrix* yang menyimpan elemen-elemennya dengan tipe karakter (*character*) sebagai elemennya. *Matrix of Character* ini memiliki field berupa ukuran maksimal matriks beserta dengan jumlah baris dan kolom efisiennya. ADT ini digunakan untuk menyimpan data peta dari konfigurasi. ADT ini digunakan untuk merepresentasikan *Map* pada permainan. ADT ini didefinisikan dan diimplementasikan pada folder ‘/src/adt/matrixChar’.

### 3.15 ADT Point

ADT *Point* adalah ADT yang terdiri atas tuple berupa nilai absis dan ordinat dalam tipe integer. Struktur data ini dipilih untuk merepresentasikan lokasi BNMO pada peta permainan. ADT ini didefinisikan dan diimplementasikan pada folder ‘/src/adt/point’.

### 3.16 ADT Queue

ADT *Queue* adalah *array* statis yang akses elemennya mengikuti aturan *first in, first out* (FIFO). *Queue* terdiri dari ukuran maksimal *array* dan *index head* serta *tail*. Struktur data ini untuk membantu proses pembuatan ADT *Queue Linked* untuk *delivery* dan *inventory*. Selain itu, *Queue* juga membantu dalam menunjukkan list makanan (baik yang bisa dibeli dan diolah) dan juga list notifikasi. ADT ini didefinisikan dan diimplementasikan pada folder ‘/src/adt/queue’.

### 3.17 ADT Queue Linked

ADT *Queue Linked* adalah sebuah ADT *Priority Queue* yang memanfaatkan struktur data *Linked List* agar memudahkan proses *dequeue* pada pengaplikasiannya nanti. ADT ini didefinisikan dan diimplementasikan pada folder ‘/src/adt/queuelinked’.

### 3.18 ADT Stack

ADT Stack adalah *array* statis yang akses elemennya mengikuti aturan *last in, first out* (LIFO). Implementasi ADT ini meliputi konstruktor, operasi pemeriksaan, operasi manipulasi elemen, dan operasi manipulasi kapasitas. ADT ini didefinisikan dan diimplementasikan pada folder ‘/src/adt/stack/’.

### 3.19 ADT Stack of Game State

ADT *Stack Of Game State* adalah ADT *Stack* yang akses elemennya mengikuti aturan *last in, first out* (LIFO) dengan elemen-elemennya bertipe ADT *GameState*. ADT ini digunakan untuk menyimpan riwayat *gameState* yang sudah ada secara berurutan. ADT ini mempermudah implementasi fitur *undo* dan *redo* pada permainan. ADT *Stack Of Game State* didefinisikan dan diimplementasikan pada folder ‘/src/adt/stackState/’.

### 3.20 ADT String

ADT String adalah ADT yang digunakan untuk membandingkan string (*compare string*) dari *input*. ADT ini didefinisikan dan diimplementasikan pada folder ‘/src/adt/string/’.

### 3.21 ADT Time

ADT *Time* adalah ADT untuk mengimplementasi waktu pada BNMO Cooking Simulator. ADT ini memiliki *field day* untuk hari, *hour* untuk jam, dan *minute* untuk menit. ADT *Time* didefinisikan dan diimplementasikan pada folder ‘/src/adt/time/’.

### 3.22 ADT Tree

ADT Tree adalah ADT yang digunakan untuk menyimpan resep. ADT Tree didefinisikan dan diimplementasikan pada folder ‘/src/adt/tree/’.

### 3.23 ADT Word Machine

ADT *Word Machine* adalah ADT untuk membaca isi dari suatu file dan menyimpan setiap katanya menjadi suatu variabel. ADT *Word Machine* digunakan untuk membantu membaca file konfigurasi. ADT ini didefinisikan dan diimplementasikan pada folder ‘/src/adt/wordmachine/’.

### 3.24 ADT Word Machine of Food

ADT *Word Machine of Food* adalah ADT *Word Machine* untuk membantu membaca isi file konfigurasi makanan. ADT ini didefinisikan dan diimplementasikan pada folder ‘/src/adt/wordmachinefood/’.

### 3.25 ADT Word Machine of Recipe

ADT *Word Machine of Food* adalah ADT *Word Machine* untuk membantu membaca isi file konfigurasi resep. ADT ini didefinisikan dan diimplementasikan pada folder ‘/src/adt/wordmachinerecipe/’.



## 4 Program Utama

Program dimulai dengan memuat kondisi awal permainan. Kondisi awal peta dimuat dengan membaca *file* konfigurasi peta yang dibuat oleh *user*, membaca *file* konfigurasi makanan, dan membaca *file* konfigurasi resep. Kondisi permainan diawali dengan posisi *simulator* sesuai pada konfigurasi peta, waktu yang menunjukkan pukul 5 pagi, dan aksi yang dapat dilakukan *simulator*. Kondisi awal ini dimasukkan ke dalam *stack* kondisi permainan untuk menyimpan riwayat kondisi permainan selama permainan berlangsung. Setelah seluruh kondisi dimuat, program menampilkan status kondisi permainan yakni posisi *simulator* saat ini, jam pada permainan, notifikasi, dan peta.

Program meminta masukan perintah dari *user*. Apabila perintah yang dimasukkan *user* terdaftar pada program, program utama akan memanggil program lain sesuai perintah yang *user* masukkan. Apabila perintah tidak terdaftar, program utama tidak akan mengubah kondisi permainan, dan tetap meminta *user* memasukkan perintah. Program lain yang dipanggil oleh program utama akan memperbarui kondisi permainan, dan program utama akan menampilkan permainan sesuai kondisi yang sudah diperbarui.

Adapun perintah yang diterima program, sebagai berikut :

- 'START' : memulai program
- 'EXIT' : keluar dari program
- 'MOVE NORTH' / 'MOVE SOUTH' / 'MOVE EAST' / 'MOVE WEST' : memindahkan simulator pada peta
- 'BUY' : pembelian makanan
- 'CHOP' : pemotongan makanan yang ada pada *inventory*
- 'MIX' : *mix* makanan yang ada pada *inventory*
- 'FRY' : penggorengan makanan yang ada pada *inventory*
- 'BOIL' : makanan yang ada pada *inventory*
- 'UNDO' : undo aksi
- 'REDO' : redo aksi
- 'WAIT X Y' : mempercepat waktu x jam y menit
- 'CATALOG' : menampilkan seluruh makanan yang ada pada program
- 'COOKBOOK' : menampilkan resep.

## 5 Algoritma-Algoritma Menarik

### 5.1 Tree

```

Address newNode(ElType val)
{
    Address p;
    p = (Address) malloc (sizeof(Node));
    if(p != NULL){
        NEXT(p) = NULL;
        INFO(p) = val;
    }
    return p;
}

```

```

boolean isEmpty(List l){
    return FIRST(l) == NULL;
}
void insertFirst(List *l, ElType val){
    Address p = newNode(val);
    if (p!= NULL){
        if (!isEmpty(*l)) NEXT(p) = FIRST(*l);
        else NEXT(p) = NULL;
        FIRST(*l) = p;
    }
}
void createTree(List *l){
    FIRST(*l) = NULL;
}
void tree(int arr[][100],List l[][100],int maxElmt){
    for(int x=0;x<maxElmt;x++){
        Address p1 = newNode(arr[x][0]);
        for(int i=2;i<arr[x][1]+2;i++){
            List l1;
            createTree(&l1);
            insertFirst(&l1,arr[x][i]);
            Address pNext = FIRST(l1);
            NEXT(pNext) = p1;
            l[x][i-2] = l1;
        }
    }
}

void makeTree(List treeResep[100][100]){
    STARTWORD();
    int length;

```

```

length = wordToInteger();
int arr[length+5][100];
ADVWORD();
int mode = 0, x = 0;
while(retval != EOF){
    if(mode == 0){
        mode = 1;
        int temp = wordToInteger();
        arr[x/3][0] = temp;
        ADVWORD();
    }
    else if(mode == 1){
        mode = 2;
        int temp = wordToInteger();
        arr[x/3][1] = temp;
        ADVWORD();
    }
    else if(mode == 2){
        mode = 0;
        for(int i=2; i<2+arr[x/3][1]; i++){
            int temp = wordToInteger();
            arr[x/3][i] = temp;
            ADVWORD();
        }
    }
    x++;
}
tree(arr, treeResep, length);
}

```

Algoritma *tree* ini kita gunakan untuk menyimpan resep yang ada pada *config*, *tree* ini juga berguna untuk mencari apakah bahan bahan yang dibutuhkan untuk membuat sebuah makanan ada di *inventory*. Algoritma ini bekerja dengan memasukkan *child* dengan *insertFirst* pada list dan membuat *new node* untuk *parent* dan menggabungkannya dengan child sehingga terbentuklah tree dengan konsep *down to top*.

## 6 Data Test

### 6.1 Data Map 1

Peta dengan panjang 10 dan lebar 10. Posisi *simulator* pada titik (0,0), posisi telepon pada (4,1), posisi *cooking area* pada (8,6), posisi *frying area* pada (6,7), posisi *mixing area* pada (1,2), dan posisi *boiling area* pada (6,9).

```
config > ≡ map.txt
1  10 10
2  S#####
3  ####T##X##
4  #M#####X##
5  #####X##
6  ####XXXX##
7  #X#####
8  #X#####C#
9  #XXX##F###
10 #####
11 #####B###
```

Gambar 6.1.1

Hasil yang diharapkan yaitu program dapat menyimpan dan menampilkan peta sesuai konfigurasi. Posisi *simulator* sesuai dengan konfigurasi dan *simulator* dapat melakukan aksi apabila berada di sekitar area aksi yang diminta. *Simulator* dapat bergerak pada area kosong. Perintah *undo* dan *redo* dapat mengembalikan posisi *simulator* pada posisi seharusnya.

```
BNMO COOKING SIMULATOR

Posisi Anda: (0,0)
Waktu: 00:05:00
Notifikasi: -

* * * * *
* S           *
*   T       X   *
*   M         X   *
*         X     *
*       X X X X   *
*   X           *
*   X           C *
*   X X X       F *
*               *
*         B       *
* * * * *
Masukkan perintah: □
```

Gambar 6.1.2

## 6.2 Data Map 2

Peta dengan panjang 5 dan lebar 8. Posisi *simulator* pada titik (1,1), posisi telepon pada (2,1), posisi *cooking area* pada (3,4), posisi *frying area* pada (5,2), posisi *mixing area* pada (2,3), dan posisi *boiling area* pada (0,7).

```
config > ≡ map.txt
1 5 8
2 #####B
3 #ST##X##
4 #####F##
5 ###M##X##
6 ###C####
```

Gambar 6.2.1

Hasil yang diharapkan yaitu program dapat menyimpan dan menampilkan peta sesuai konfigurasi. Posisi *simulator* sesuai dengan konfigurasi dan *simulator* dapat melakukan aksi apabila berada di sekitar area aksi yang diminta. *Simulator* dapat bergerak pada area kosong. Perintah *undo* dan *redo* dapat mengembalikan posisi *simulator* pada posisi seharusnya.

```
BNMO COOKING SIMULATOR

Posisi Anda: (1,1)
Waktu: 00:05:00
Notifikasi: -

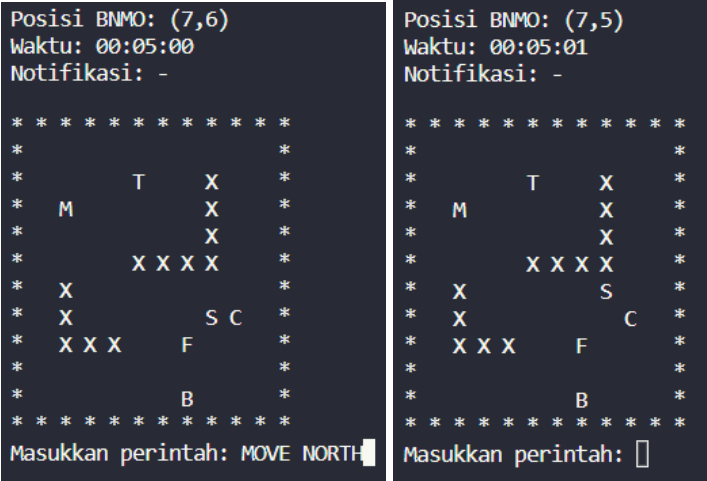
* * * * *
*           B *
*  S T     X  *
*           F  *
*     M     X  *
*           C  *
* * * * *
Masukkan perintah: □
```

Gambar 6.2.2

## 6.3 Data Move

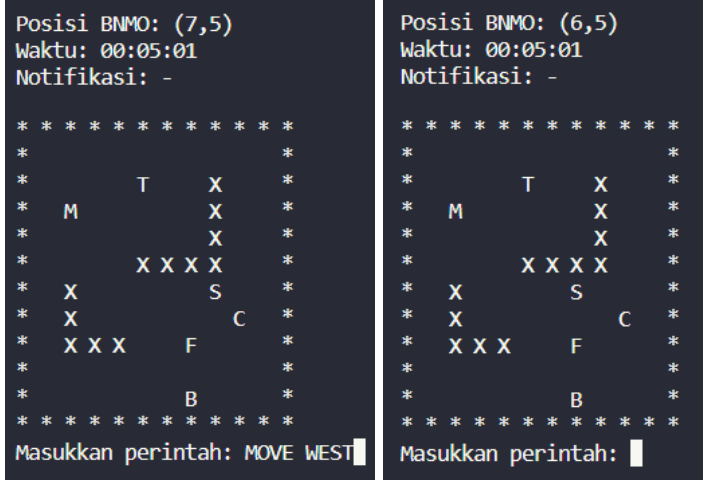
Simulator (S) dapat bergerak 4 arah mata angin yaitu utara, timur, selatan, dan barat. Simulator tidak dapat mengisi posisi yang sudah terisi elemen lain seperti T, M, X, S, C, B,

dan \*. Apabila posisi awal S (7,6) lalu memasukkan command ‘MOVE NORTH’ maka S akan berada pada posisi (7,5).



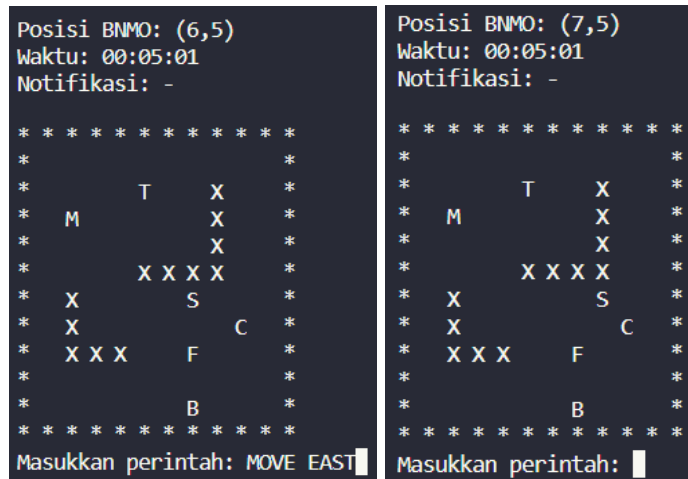
Gambar 6.3.1

Apabila posisi awal S (7,5) lalu memasukkan command ‘MOVE EAST’ maka S akan berada pada posisi (6,5).



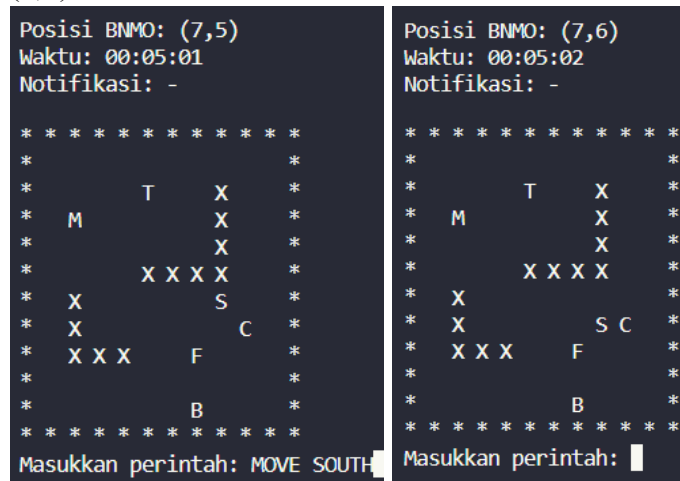
Gambar 6.3.2

Apabila posisi awal S (6,5) lalu memasukkan command ‘MOVE NORTH’ maka S akan berada pada posisi (7,5).



Gambar 6.3.3

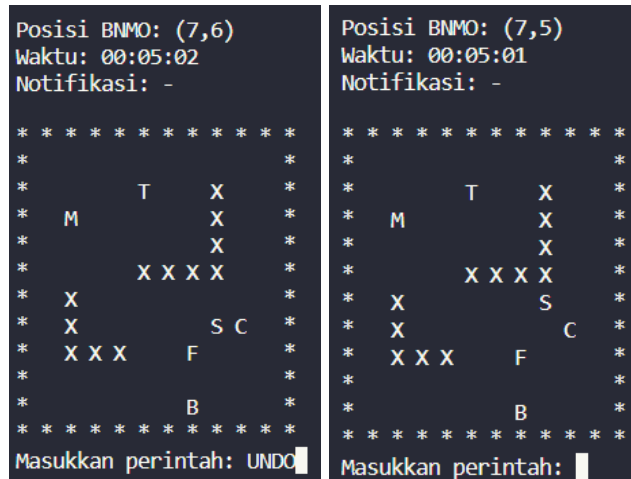
Apabila posisi awal S (7,5) lalu memasukkan command ‘MOVE SOUTH’ maka S akan berada pada posisi (7,6).



Gambar 6.3.4

## 6.4 Data Move (Undo)

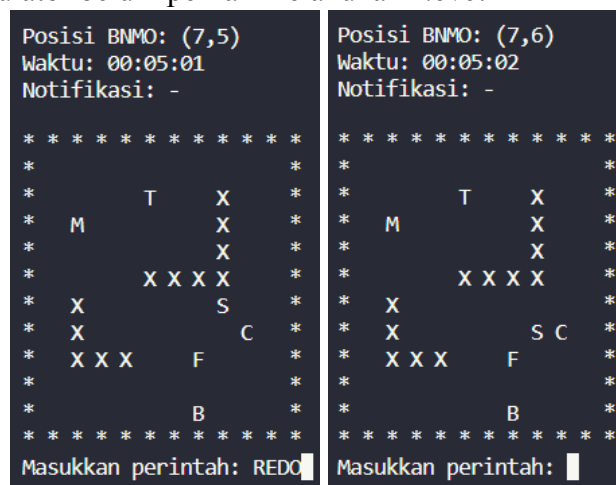
User dapat melakukan *undo action move* pada peta dengan memasukkan input ‘UNDO’. Seperti yang terlihat pada gambar, ketika menerima masukan ‘UNDO’ simulator akan kembali ke posisi yang sebelumnya (7,5). *Undo* tidak dapat dilakukan jika sebelumnya simulator belum pernah melakukan *move*.



Gambar 6.4.1

## 6.5 Data Move (Redo)

User dapat melakukan *action redo move* pada peta dengan memasukkan input 'REDO'. Seperti yang terlihat pada gambar, ketika menerima masukkan 'REDO' simulator akan kembali ke posisi yang setelahnya (kebalikan dari *undo*) (7,6). *Redo* tidak dapat dilakukan jika sebelumnya simulator belum pernah melakukan *move*.



Gambar 6.5.1

## 6.6 Data Buy

User dapat melakukan pembelian bahan makanan dengan memasukkan command 'BUY' ketika simulator (S) berada di dekat T (Telefon). Setelah itu, user dapat melihat bahan makanan yang tersedia pada toko, waktu kedaluwarsa bahan makanan, serta waktu pengiriman. Setelah user memilih bahan makanan yang dibeli, program akan menampilkan output makanan berhasil dipesan, dan barang akan masuk ke inventory sesuai waktu pengiriman. Setelah itu, program akan meminta user untuk menekan tombol 'enter' untuk kembali ke *map*





## 6.8 Data Inventory

Ketika *user* memasukkan *command* 'INVENTORY', program akan menampilkan bahan makanan yang ada pada inventory beserta waktu kedaluwarsa. Setelah itu, program akan meminta user untuk menekan tombol 'enter' untuk kembali ke *map*.

```
=====
|                               INVENTORY                               |
|=====|
| No. Nama ----- Waktu Kedaluwarsa |
| 1. Ayam Goreng ----- 00:00:08 |
| 2. Ayam Tepung ----- 00:00:23 |
| 3. Tepung ----- 00:00:58 |
| 4. Minyak Goreng ----- 00:01:23 |
| 5. Ayam Potong ----- 00:01:58 |
| 6. Ayam Mentah ----- 00:23:53 |
|=====|
| Tekan enter menutup inventory |
=====
```

Gambar 6.8.1

## 6.9 Data Delivery

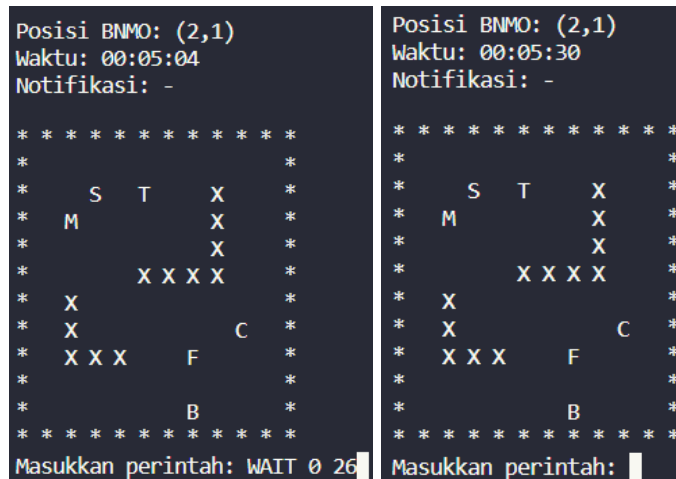
Ketika *user* memasukkan *command* 'DELIVERY', program akan menampilkan bahan makanan yang sedang dalam pengiriman beserta sisa waktu pengiriman. Setelah itu, program akan meminta user untuk menekan tombol 'enter' untuk kembali ke *map*.

```
=====
|                               DELIVERY                               |
|=====|
| No. Nama ----- Waktu Delivery |
| 1. Ayam Mentah ----- 00:00:15 |
|=====|
| Tekan enter menutup inventory |
=====
```

Gambar 6.9.1

## 6.10 Data Wait X Y

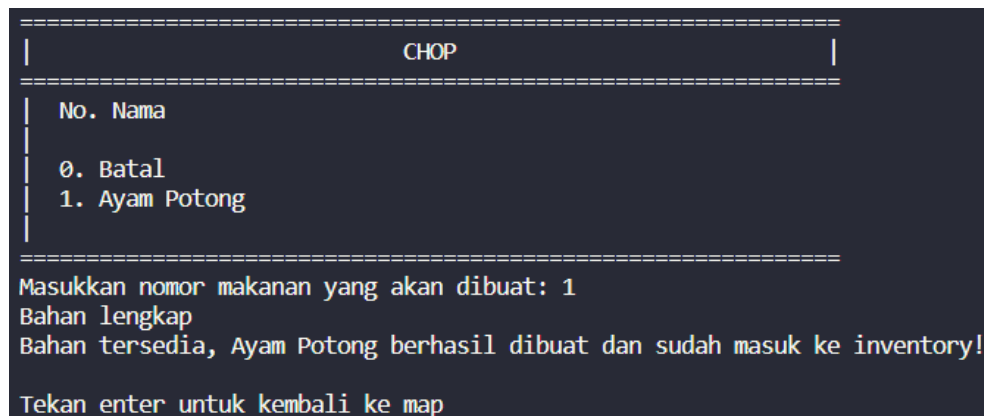
Ketika *user* memasukkan *command* 'WAIT X Y' dengan X adalah jam dan Y adalah menit, *user* dapat mempercepat waktu ( *time skip* ) selama X dan Y.



Gambar 6.10.1

## 6.11 Data Chop

Fitur *chop* (pemotongan) dapat digunakan ketika simulator berada di sekitar C (*chop*). *User* dapat memasukkan *command* 'CHOP', lalu program akan menampilkan bahan makanan yang dapat dipotong. Ketika bahan makanan yang dibutuhkan untuk pemotongan tersedia, bahan makanan hasil pemotongan akan masuk ke dalam *inventory*. Tetapi, jika bahan makanan untuk dipotong tidak tersedia, program akan menolak melakukan *chop*. Setelah itu, program akan meminta *user* untuk menekan tombol 'enter' untuk kembali ke *map*.



Gambar 6.11.1

## 6.12 Data Mix

Fitur *Mix* dapat digunakan ketika simulator berada di sekitar M (*mix*). *User* dapat memasukkan *command* 'MIX', lalu program akan menampilkan bahan makanan yang dapat di-*mix*. Ketika bahan makanan yang dibutuhkan untuk di-*mix* tersedia, bahan makanan hasil *mix* akan masuk ke dalam *inventory*. Tetapi, jika bahan makanan untuk di-*mix* tidak tersedia, program akan menolak melakukan *mix*. Setelah itu, program akan meminta *user* untuk menekan tombol 'enter' untuk kembali ke *map*.

```
=====
|                                     MIX                                     |
=====
| No. Nama                                                                    |
| 0. Batal                                                                    |
| 1. Ayam Tepung                                                                |
| 2. Kopi Hitam                                                                |
|                                                                              |
=====
Masukkan nomor barang yang ingin dibuat: 1
Bahan tersedia, Ayam Tepung berhasil dibuat dan sudah masuk ke inventory!
Tekan enter untuk kembali ke map
```

Gambar 6.12.1

### 6.13 Data Fry

Fitur *Fry* ( penggorengan ) dapat digunakan ketika simulator berada di sekitar F (*fry*). *User* dapat memasukkan *command* ‘FRY’, lalu program akan menampilkan bahan makanan yang dapat digoreng. Ketika bahan makanan yang dibutuhkan untuk digoreng tersedia, bahan makanan hasil *fry* akan masuk ke dalam *inventory*. Tetapi, jika bahan makanan untuk digoreng tidak tersedia, program akan menolak melakukan *fry*. Setelah itu, program akan meminta *user* untuk menekan tombol ‘enter’ untuk kembali ke *map*.

```
=====
|                                     FRY                                     |
=====
| No. Nama                                                                    |
| 0. Batal                                                                    |
| 1. Ayam Goreng                                                                |
|                                                                              |
=====
Masukkan nomor barang yang ingin dibuat: 1
Bahan tersedia, Ayam Goreng berhasil dibuat dan sudah masuk ke inventory!
Tekan enter untuk kembali ke map
```

Gambar 6.13.1

### 6.14 Data Boil

Fitur *Boil* (rebus) dapat digunakan ketika simulator berada di sekitar B (*boil*). *User* dapat memasukkan *command* ‘BOIL’, lalu program akan menampilkan bahan makanan yang dapat direbus. Ketika bahan makanan yang dibutuhkan untuk direbus tersedia, bahan makanan hasil *boil* akan masuk ke dalam *inventory*. Tetapi, jika bahan makanan untuk direbus tidak tersedia, program akan menolak melakukan *boil*. Setelah itu, program akan meminta *user* untuk menekan tombol ‘enter’ untuk kembali ke *map*.

```
=====
|                                     BOIL                                     |
=====
| No. Nama                                                                    |
|                                                                              |
| 0. Batal                                                                    |
| 1. Air Panas                                                                |
|                                                                              |
=====
Masukkan nomor barang yang ingin dibuat: 1
Bahan tersedia, Air Panas berhasil dibuat dan sudah masuk ke inventory!
Tekan enter untuk kembali ke map
```

Gambar 6.14.1

## 6.15 Data Exit

Ketika *user* memasukkan command ‘EXIT’, program akan berhenti dan keluar. *Command exit* dapat diterima program ketika di awal program dan di-map.

```
Masukkan perintah: EXIT
PS C:\Users\acer\Documents\GitHub\Tubes_Alstrukdat>
```

Gambar 6.15.1

## 6.16 Data Catalog

Ketika *user* memasukkan command ‘CATALOG’, program akan menampilkan list seluruh data mencakup semua bahan makanan yang tersedia dalam aplikasi. Setelah itu, program akan meminta *user* untuk menekan tombol ‘enter’ untuk kembali ke *map*. List bahan makanan tersebut berisi nama bahan makanan/minuman, waktu kedaluwarsa, aksi yang diperlukan, hingga waktu pengirimannya.

```
=====
|                                     CATALOG                                    |
=====
| No. Nama ----- Waktu Kedaluwarsa ----- Aksi Yang Diperlukan ----- Waktu Pengiriman |
|                                                                              |
| 1. Ayam Mentah ----- 01:00:00 ----- Buy ----- 15 menit |
| 2. Ayam Potong ----- 00:02:05 ----- Chop ----- 5 menit |
| 3. Tepung ----- 00:01:05 ----- Buy ----- 30 menit |
| 4. Minyak Goreng ----- 00:01:30 ----- Buy ----- 1 jam |
| 5. Ayam Tepung ----- 00:00:30 ----- Mix ----- |
| 6. Ayam Goreng ----- 00:00:15 ----- Fry ----- 5 menit |
| 7. Air ----- 05:00:05 ----- Buy ----- 30 menit |
| 8. Air Panas ----- 00:05:00 ----- Boil ----- |
| 9. Kapal Api ----- 25:00:00 ----- Buy ----- 5 hari 10 menit |
| 10. Kopi Hitam ----- 00:01:45 ----- Mix ----- 5 menit |
|                                                                              |
=====
Tekan enter menutup inventory
█
```

Gambar 6.16.1

## 6.17 Data Cookbook

Ketika *user* memasukkan command ‘COOKBOOK’, program akan menampilkan list bahan makanan yang bisa diolah sendiri, baik itu dengan cara *mix*, *chop*, *fry*, *maupun* *boil*. Selain itu, program akan mengirim pula aksi pengolahan apa yang bisa dilakukan hingga seluruh bahan makanan yang dibutuhkan untuk mengolah makanan tersebut.

```
=====
|                                     COOKBOOK                               |
=====
(Aksi yang diperlukan - bahan...)

1. Ayam Potong
Chop - Ayam Mentah
2. Air Panas
Boil - Air
3. Ayam Tepung
Mix - Ayam Potong - Tepung
4. Ayam Goreng
Fry - Ayam Tepung - Minyak Goreng
5. Kopi Hitam
Mix - Air Panas - Kapal Api
```

Gambar 6.17.1

## 6.18 Data Notifikasi

Ketika pesanan sampai dan masuk ke dalam *inventory* maka program akan menampilkan bahan makanan yang sampai pada notifikasi. Selain itu, ketika makanan dalam *inventory* kedaluwarsa, maka program juga akan menampilkan bahan makanan yang kedaluwarsa pada notifikasi.

```
Posisi Kelvin: (3,1)
Waktu: 00:05:16
Notifikasi :
1. Pesanan Ayam Mentah telah tiba
2. Ayam Goreng sudah kadaluwarsa
* * * * *
*           *
*   S T   X   *
*   M       X   *
*           X   *
*       X X X X   *
*   X           *
*   X           C   *
*   X X X   F       *
*           *
*           B       *
* * * * *
Masukkan perintah: █
```

Gambar 6.18.1

## 7 Test Script

No .	Fitur yang Dites	Tujuan Testing	Langkah-Langkah Testing	Input Data Test	Hasil yang Diharapkan	Hasil yang Keluar
1	Pembacaan konfigurasi peta dan pergerakan <i>simulator</i>	Memeriksa apakah program dapat menampilkan peta sesuai konfigurasi <i>user</i>	Mengubah konfigurasi peta dengan dimensi 10x10	Data Map 1	Program menyimpan dan menampilkan peta sesuai <i>input</i> dan <i>simulator</i> dapat bergerak	Program berhasil menampilkan peta sesuai <i>input</i> dan <i>simulator</i> dapat bergerak.
2	Pembacaan konfigurasi peta dan pergerakan <i>simulator</i>	Memeriksa apakah program dapat menampilkan peta sesuai konfigurasi <i>user</i>	Mengubah konfigurasi peta dengan dimensi 5x8	Data Map 2	Program menyimpan dan menampilkan peta sesuai <i>input</i> dan <i>simulator</i> dapat bergerak	Program berhasil menampilkan peta sesuai <i>input</i> dan <i>simulator</i> dapat bergerak.
3	Move	Memeriksa apakah program dapat memindahkan simulator sesuai arah yang diinginkan	Memasukkan <i>command</i> 'MOVE NORTH', 'MOVE SOUTH', 'MOVE WEST', 'MOVE EAST'	Data Move	Program dapat memindahkan simulator sesuai <i>command</i>	Program berhasil memindahkan simulator sesuai <i>command</i>
4	Undo move	Memeriksa apakah program dapat melakukan <i>undo move</i> .	Memasukkan <i>command</i> 'UNDO'	Data Move (Undo)	Program dapat melakukan <i>undo move</i> pada <i>map</i>	Program berhasil melakukan <i>undo move</i> pada <i>map</i>
5	Redo move	Memeriksa apakah program dapat melakukan <i>redo move</i> .	Memasukkan <i>command</i> 'REDO'	Data Move (Redo)	Program dapat melakukan <i>redo move</i> pada <i>map</i> .	Program berhasil melakukan <i>redo move</i> pada <i>map</i> .
6	Buy	Memeriksa apakah program dapat melakukan pembelian bahan makanan pada toko	Memasukkan <i>command</i> 'BUY'	Data Buy	Program dapat melakukan pembelian dan menyimpan bahan makanan pembelian pada <i>inventory</i> .	Program berhasil melakukan pembelian dan menyimpan bahan makanan pembelian pada <i>inventory</i> .
7	Start	Memeriksa apakah Inisiasi program dapat berjalan	Melakukan <i>run file</i> 'main.exe'	Data START	Program dapat berjalan dan menampilkan <i>splash screen</i> .	Program berhasil berjalan dan menampilkan <i>splash screen</i> .

No .	Fitur yang Dites	Tujuan Testing	Langkah-Langkah Testing	Input Data Test	Hasil yang Diharapkan	Hasil yang Keluar
8	Inventory	Memeriksa apakah program dapat menampilkan <i>inventory</i>	Memasukkan <i>command</i> 'INVENTORY'	Data Inventory	Program dapat menampilkan bahan makanan yang terdapat pada <i>inventory</i>	Program berhasil menampilkan bahan makanan yang terdapat pada <i>inventory</i>
9	Delivery	Memeriksa apakah program dapat melakukan <i>delivery</i>	Memasukkan <i>command</i> 'DELIVERY'	Data Delivery	Program dapat melakukan pengiriman bahan makanan yang sudah dipesan.	Program berhasil melakukan pengiriman bahan makanan yang sudah dipesan.
10	Wait X Y	Memeriksa apakah program dapat mempercepat waktu	Memasukkan <i>command</i> 'WAIT X Y', (X jam, Y menit)	Data Wait X Y	Program dapat mempercepat waktu	Program berhasil mempercepat waktu
11	Chop	Memeriksa apakah program dapat melakukan <i>chop</i>	Memasukkan <i>command</i> 'CHOP' ketika simulator berada disekitar 'C' pada <i>map</i> .	Data Chop	Program dapat melakukan <i>chop</i> pada bahan makanan yang tersedia	Program berhasil melakukan <i>chop</i> pada bahan makanan yang tersedia
12	Mix	Memeriksa apakah program dapat melakukan <i>mix</i>	Memasukkan <i>command</i> 'MIX' ketika simulator berada disekitar 'M' pada <i>map</i> .	Data Mix	Program dapat melakukan <i>mix</i> pada bahan makanan yang tersedia	Program berhasil melakukan <i>mix</i> pada bahan makanan yang tersedia
13	Fry	Memeriksa apakah program dapat melakukan <i>fry</i>	Memasukkan <i>command</i> 'FRY' ketika simulator berada disekitar 'F' pada <i>map</i> .	Data Fry	Program dapat melakukan <i>fry</i> pada bahan makanan yang tersedia	Program berhasil melakukan <i>fry</i> pada bahan makanan yang tersedia
14	Boil	Memeriksa apakah program dapat melakukan <i>boil</i>	Memasukkan <i>command</i> 'BOIL' ketika simulator berada disekitar 'B' pada <i>map</i> .	Data Boil	Program dapat melakukan <i>boil</i> pada bahan makanan yang tersedia	Program berhasil melakukan <i>boil</i> pada bahan makanan yang tersedia
15	Exit	Memeriksa apakah program dapat di- <i>exit</i>	Memasukkan <i>command</i> 'EXIT'	Data Exit	Program dapat di- <i>exit</i>	Program berhasil di- <i>exit</i>
16	Catalog	Memeriksa apakah program dapat	Memasukkan <i>command</i> 'CATALOG'	Data Catalog	Program dapat menampilkan list seluruh	Program berhasil menampilkan



No .	Fitur yang Dites	Tujuan Testing	Langkah-Langkah Testing	Input Data Test	Hasil yang Diharapkan	Hasil yang Keluar
		menampilkan list seluruh bahan makanan yang tersedia pada program			bahan makanan yang tersedia pada program	list seluruh bahan makanan yang tersedia pada program
17	Cookbook	Memeriksa apakah program dapat menampilkan resep makanan yang dapat diolah	Memasukkan <i>command</i> 'COOKBOOK'	Data Cookbook	Program dapat menampilkan resep makanan yang dapat diolah	Program berhasil menampilkan resep makanan yang dapat diolah
18	Notifikasi pesanan Sampai	Memeriksa apakah program dapat menampilkan notifikasi ketika pesanan sampai	Mempercepat waktu / melakukan gerakan sehingga waktu berjalan	Data Notifikasi	Program dapat menampilkan notifikasi ketika pesanan sampai	Program berhasil menampilkan notifikasi ketika pesanan sampai
19	Notifikasi makanan kedaluwarsa	Memeriksa apakah program dapat menampilkan notifikasi ketika bahan makanan kedaluwarsa	Mempercepat waktu / melakukan gerakan sehingga waktu berjalan	Data Notifikasi	Program dapat menampilkan notifikasi ketika bahan makanan kedaluwarsa	Program berhasil menampilkan notifikasi ketika bahan makanan kedaluwarsa

## 8 Pembagian Kerja dalam Kelompok

No.	NIM	Tugas
1	13521001	Fitur makanan, fitur resep, fitur pengolahan makanan, laporan, dan testing.
2	13521005	Fitur inventory, fitur pemesanan makanan, fitur delivery, laporan, dan testing.
3	13521010	Inisiasi, laporan, README, dan testing.
4	13521019	Fitur peta, <i>simulator</i> , program utama, laporan, README, dan testing.
5	13521023	Fitur inventory, fitur pemesanan makanan, fitur delivery, laporan, dan testing.
6	13521031	Fitur makanan, fitur resep, fitur pengolahan makanan, laporan, dan testing.

## 9 Lampiran

### 9.1 Deskripsi Tugas Besar

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*”.

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 membuat program simulasi untuk ditiru BNMO. Oleh karena itu, Doni meminta bantuan kalian untuk membuat program simulasi tersebut.

## 9.2 Notulen Rapat

### 9.2.1 Asistensi


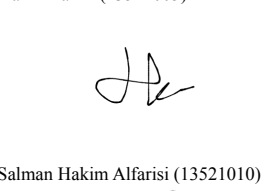
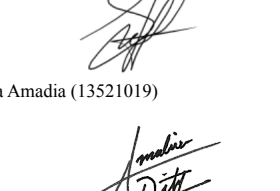
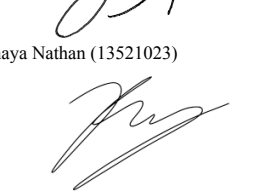
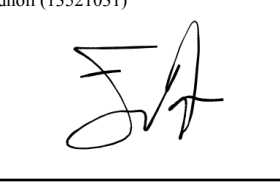


**Form Asistensi Tugas Besar  
IF2110/Algoritma dan Struktur Data  
Sem. 1 2021/2022**

No. Kelompok/Kelas : D / K03  
Nama Kelompok : D  
Anggota Kelompok (Nama/NIM) : 1. Angger Ilham Amanullah / 13521001  
2. Kelvin Rayhan Alkarim / 13521005  
3. Muhamad Salman Hakim Alfarisi / 13521010  
4. Ditra Rizqa Amadia / 13521019  
5. Kenny Benaya Nathan / 13521023  
6. Fahrian Afdholi / 13521031








Asisten Pembimbing : Widya Anugrah Putra

---

## Asistensi I

<b>Tanggal : 26 Oktober 2022</b>	<b>Catatan Asistensi:</b>  QnA  Q: input via console atau txt? (Fahrian)  A: Dua2nya, dari console untuk command, sedangkan dari file untuk konfigurasi seperti menu, bahan, dll.  Q: Untuk poin konfigurasi makanan, lokasi aksi makanan itu gimana ya? (Kenny)  A: Lokasi aksi makanan tergantung pada peta, kalo buy di B, mix di M, dll.  Q: Peta minimal 10x10, kalau misal di bawah minimal, tampilan errornya gimana?  A: Dikasih minimal aja, kalo error ditanya lagi aja sampe bener  Q: Kalo dibuat shortcut untuk gerak2nya kek MOVE SOUTH jadi MS, MOVE NORTH jadi MN boleh ga?  A: Ikutin di spek aja  Q: Kalo posisi di 7,5 udah pasti di Fry, apa harus tetep masukin command Fry?  A: Tetep pake command, karena bisa lewat doang. Belum pasti fry
<b>Tempat : Video Conference ( Google Meet )</b>	
<b>Kehadiran Anggota Kelompok:</b> <div style="text-align: center;">No NIM Tanda tangan</div> <ol style="list-style-type: none"> <li>Angger Ilham Amanullah (13521001) </li> <li>Kelvin Rayhan Alkarim (13521005) </li> <li>Muhamad Salman Hakim Alfarisi (13521010) </li> <li>Ditra Rizqa Amadia (13521019) </li> <li>Kenny Benaya Nathan (13521023) </li> <li>Fahrian Afdholi (13521031) </li> </ol>	
	<b>Tanda Tangan Asisten:</b>  

## Asistensi II

Tanggal : 3 November 2022	Catatan Asistensi:
Tempat : Video Conference ( Google Meet )	
<p><b>Kehadiran Anggota Kelompok:</b></p> <p>No NIM Tanda tangan</p> <ol style="list-style-type: none"> <li>Angger Ilham Amanullah (13521001) </li> <li>Kelvin Rayhan Alkarim (13521005) </li> <li>Muhamad Salman Hakim Alfarisi (13521010) </li> <li>Ditra Rizqa Amadia (13521019) </li> <li>Kenny Benaya Nathan (13521023) </li> <li>Fahrian Afdholi (13521031) </li> </ol>	
	<p><b>Tanda Tangan Asisten:</b></p> 

### 9.3 Log Activity Anggota Kelompok

Berikut adalah *log activity* anggota kelompok :

1. Angger Ilham Amanullah (13521001) :
  - 7 November 2022 : *update ADT*
  - 8 November 2022 : *fix resep, fix time, fix food, add foodlist*
  - 15 November 2022 : *add buy, update tree, add boil, add chop, delete buy, update simulator, fix adt, add fry, add mix, update food*
  - 17 November 2022 : *add fry, add mix, update food, update notification*
  - 18 November 2022 : *update boil, update fry, update mix, update chop*
  - 19 November 2022 : *update cookbook*
2. Kelvin Rayhan Alkarim (13521005) :
  - 27 Oktober 2022 : *add ADT*
  - 28 Oktober 2022 : *update food*
  - 8 November 2022 : *update ADT, update prioqueue time*
  - 9 November 2022 : *update food*
  - 15 November 2022 : *update inventory*
  - 16 November 2022 : *update inventory*
  - 17 November 2022 : *update food, merge branch, add delivery, debug*
  - 18 November 2022 : *update boil, update food config, update recipe config*
3. Muhamad Salman Hakim Alfarisi (13521010) :
  - 26 Oktober 2022 : *add inisiasi*
  - 31 Oktober 2022 : *add welcome.txt*
  - 2 November 2022 : *update wordmachine*
  - 19 November 2022 : *update readme*
  - 20 November 2022 : *membuat laporan, update readme, update mix.c*
4. Ditra Rizqa Amadia (13521019) :
  - 4 November 2022 : *update folder structure, update include path, add command parser*
  - 5 November 2022 : *add map*
  - 6 November 2022 : *update map, update inisiasi, add simulatore movement, add main, add undo and redo*
  - 7 November 2022 : *add wait and time state, add available action state, update readme*
  - 14 November 2022 : *debug map configuration app*
  - 15 November 2022 : *add username input*
  - 17 November 2022 : *merge branch, update time, fix conflict push, add clear.c, update inventory, integrate delivery app*
  - 18 November 2022 : *integrate time to delivery app, fix conflict, fix display chop*
5. Kenny Benaya Nathan (13521023) :
  - 27 Oktober 2022 : *add ADT time, merge branch*
  - 11 November 2022 : *update prioqueue*

15 November 2022 : *update prioqueue timedel*  
 18 November 2022 : *update delivery*  
 19 November 2022 : *update message, add catalog, add cookbook*

6. Fahrian Afdholi (13521031) :  
 5 November 2022 : *add tree*  
 6 November 2022 : *fix tree*  
 9 November 2022 : *fix tree*  
 10 November 2022 : *fix tree, merge branch*  
 18 November 2022 : *fix tree*

## 9.4 Progress Milestone

Tanggal		03-11-2022
No	Fitur	Progress (0-100%)
1.	Command Parser	100%
2.	Inisiasi	
	a. Splash Screen	100%
	b. Command START	100%
	c. Command EXIT	100%
3.	Simulator	
	a. ADT Simulator	100%
4.	Makanan	
	a. Membaca makanan dari file	100%
	b. ADT Makanan	100%
	c. Command CATALOG	100%
6.	Peta	
	a. Membaca peta dari file	100%
	b. Command MOVE NORTH/EAST/SOUTH/WEST	100%

7.	Mekanisme Waktu	
	a. ADT Waktu	100%
	b. Waktu bertambah seiring command yg valid	100%
8.	Laporan (50%)	100%