

# **LAPORAN TUGAS BESAR**

## **IF2110/Algoritma dan Struktur Data**

### **Willy Wangky's Playground**


Dipersiapkan oleh:

#### **Kelompok 4**

Dwianditya Hanif Raharjanto	13519046
Mohammad Afif Akromi	13519110
Fadel Ananda Doty	13519146
Muhammad Fawwaz Naabigh	13519206
Tanur Rizaldi Rahardjo	13519214

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-4-2</i>		<i>25</i>
		<i>Revisi</i>	<i>0</i>	<i>29 November 2020</i>

# Daftar Isi

<b>1 Ringkasan</b>	<b>3</b>
<b>2 Penjelasan Tambahan Spesifikasi Tugas</b>	<b>4</b>
2.1. Movement	4
2.2. Detail	4
2.3. Detail pada Office	5
2.4. Laporan pada Office	5
2.5. Legend pada Map	5
2.6. Pembacaan file Eksternal	5
<b>3 Struktur Data (ADT)</b>	<b>5</b>
3.1. ADT Point	5
3.2. ADT Jam	6
3.3. ADT List Implementasi Array	7
3.4. ADT Matriks	8
3.5. ADT Mesin Karakter + ADT Mesin Kata	8
3.6. ADT Queue (Priority Queue)	9
3.7. ADT Stack	10
3.8. ADT List Implementasi List Berkait	11
3.9. ADT Tree	12
3.10. ADT Graph (Variasi Multilist)	13
3.11. ADT Wahana	14
3.12. ADT Material	15
<b>4 Program Utama</b>	<b>15</b>
<b>5 Algoritma-Algoritma Menarik</b>	<b>15</b>
5.1. Algoritma Double Buffer	15
<b>6 Data Test</b>	<b>16</b>
6.1. Pindah Map	16
6.2. Buy Material	16
6.3. Build Wahana	16
6.4. Execute	16
6.5. Main	16
6.6. Detail	16

6.7. Detail pada Office	16
6.8. Laporan pada Office	17
6.9. Serve	17
6.10. Upgrade	17
6.11. Undo	17
6.12. Repair	17
<b>7 Test Script</b>	<b>17</b>
<b>8 Pembagian Kerja dalam Kelompok</b>	<b>20</b>
<b>9 Lampiran</b>	<b>21</b>
9.1. Deskripsi Tugas Besar 2	21
9.2. Notulen Rapat	22
9.3. Log Activity Anggota Kelompok	23
9.4. Form Asistensi	23

# 1 Ringkasan

Laporan ini dibuat dalam rangka pengerjaan tugas besar IF2110 Algoritma dan Struktur Data. Tugas besar ini bertujuan untuk membuat sebuah program dengan semua materi yang sudah dipelajari di mata kuliah IF2110 Algoritma dan Struktur Data. Program yang dibuat menggunakan bahasa pemrograman C. Program yang dibuat merupakan program wahana bermain yang dibuat untuk Willy Wangky. Program ini dapat mensimulasikan keadaan persiapan, yaitu keadaan ketika Willy Wangky dapat melakukan manajemen terhadap wahana. Program ini juga dapat mensimulasikan kegiatan wahana pada hari kerja. Program ini juga memungkinkan Willy Wangky sebagai *administrator* untuk melakukan kegiatan di sekitar wahana.

Laporan ini berisi spesifikasi program yang kami buat, struktur ADT yang kami definisikan pada program kami, algoritma program utama, algoritma yang menurut kami unik, data test, test script, pembagian kerja dalam kelompok, dan lampiran-lampiran dalam pengerjaan tugas besar ini.

Tugas besar ini secara umum merupakan implementasi dari berbagai macam *abstract data type* yang telah dipelajari di mata kuliah IF2110 Algoritma dan Struktur Data. Implementasi tersebut direpresentasikan pada berbagai macam mekanika permainan dalam Willy Wangky's World. Kesimpulan dari tugas besar ini adalah berbagai macam *abstract data type* berguna untuk menyelesaikan permasalahan tertentu yang ada di program ketika sedang membuat suatu program.

## 2 Penjelasan Tambahan Spesifikasi Tugas

Terdapat tambahan program yang diubah sehingga berbeda dengan spesifikasi tugas besar, beberapa diantaranya yaitu:

### 2.1 Movement

Pada spesifikasi tugas besar, pada saat main phase, player tidak dapat berjalan melewati icon building/wahana yang terdapat pada map. Akan tetapi, pada program yang kami buat, pada saat preparation phase player dapat berjalan melewati atau berada di atas icon wahana/building yang terdapat pada map.

### 2.2 Detail

Pada spesifikasi tugas besar, player hanya dapat melihat detail wahana pada saat berapa di main phase saja. Akan tetapi, pada program yang dibuat, player dapat melihat detail wahana pada saat preparation phase dan main phase. Syarat agar player dapat melihat detail wahana pada saat preparation phase adalah player harus berada tepat di atas icon wahana yang akan dilihat detailnya. Sementara syarat agar player dapat melihat detail wahana pada saat main phase adalah player harus berada di sebelah kanan, kiri, atas, atau bawah icon wahana yang akan dilihat detailnya, kemudian berjalan mengarah ke icon tersebut (contohnya apabila player berada di atas icon, player berjalan kebawah).

## 2.3 Detail pada Office

Pada spesifikasi tugas besar, apabila player berada di sebelah office dan mengetikkan perintah “detail” pada office, akan muncul daftar wahana yang dapat dipilih oleh player. Apabila player memilih wahana tersebut, akan muncul detail tentang wahana tersebut. Namun pada program yang kami buat, apabila player mengetikkan perintah “detail” pada office, akan muncul semua list wahana yang telah dibuat beserta detailnya yang berisi ID, nama, harga, durasi, lokasi, kapasitas, history upgrade, dll.

## 2.4 Laporan pada Office

Pada spesifikasi tugas besar, apabila player berada di sebelah office dan mengetikkan perintah “laporan” pada office, akan muncul daftar wahana yang dapat dipilih oleh player. Apabila player memilih wahana tersebut, akan muncul laporan tentang wahana tersebut. Namun pada program yang kami buat, apabila player mengetikkan perintah “laporan” pada office, akan muncul semua list wahana beserta laporannya yang berisikan ID, nama, dinaiki, total, dinaiki hari ini, dan total hari ini.

## 2.5 Legend pada Map

Pada program yang kami buat, terdapat perubahan icon pada map yang berbeda dengan yang terdapat pada spesifikasi tugas besar. Pada saat preparation phase, player direpresentasikan sebagai icon \*, sementara pada main phase, player direpresentasikan sebagai icon @. Antrean pada map direpresentasikan sebagai icon a. Office pada map direpresentasikan sebagai icon o. Dan wahana pada map direpresentasikan sebagai icon yang berbeda.

## 2.6 Pembacaan file Eksternal

Pembacaan file eksternal yang kami buat harus memberikan enter apabila memasukkan data baru pada file txt dikarenakan parsing oleh fileio.c yang kami buat.

# 3 Struktur Data (ADT)

Dalam pengerjaan program, kami membuat beberapa ADT yang akan digunakan pada program utama, ADT tersebut yaitu ADT Point, ADT Jam, ADT List Implementasi Array, ADT Matriks, ADT Mesin Karakter, ADT Mesin Kata, ADT Queue (Priority Queue), ADT Stack, ADT List Implementasi List Berkait, ADT Tree, ADT Graph (Variasi Multilist). Kami memilih semua ADT di atas karena ADT tersebut cocok untuk menunjang struktur program utama. Misalnya saja ADT Point yang dapat merepresentasikan posisi player pada peta, ADT Jam yang kemudian akan dikombinasikan dengan ADT Queue untuk mensimulasikan antrian peserta, dan lain-lain.

## 3.1 ADT Point

- Sketsa struktur data  
ADT Point memiliki fungsi untuk mengolah perpindahan player di map. Pada ADT Point ini mengandung beberapa primitif diantaranya adalah:
  - Selektor
    1. Absis (P)
    2. Ordinat (P)

- `MakePoint(float X, float Y)`  
Membuat sebuah `POINT` dari komponen-komponennya
- `BacaPoint(POINT *P)`  
Membaca nilai absis dan ordinat dari keyboard dan membentuk `POINT P` berdasarkan dari nilai absis dan ordinat tersebut
- `TulisPOINT(POINT P)`  
Nilai `P` ditulis ke layar dengan format `(X,Y)`
- `Kuadran(POINT P)`  
Menghasilkan kuadran dari `P`: 1, 2, 3, atau `P`
- `PlusDelta(POINT P, float deltaX, float deltaY)`  
Mengirim salinan `P` yang absisnya adalah `Absis(P) + deltaX` dan ordinatnya adalah `Ordinat(P) + deltaY`
- `MirrorOf(POINT P, boolean SbX)`  
Menghasilkan salinan `P` yang dicerminkan terhadap salah satu sumbu
- `Geser(POINT *P, float deltaX, float deltaY)`  
`P` digeser, absisnya sebesar `deltaX` dan ordinatnya sebesar `deltaY`
- `IsOrigin(POINT P)`  
Menghasilkan `true` jika `P` adalah titik origin
- `EQ(POINT P1, P2)`  
Mengirimkan `true` jika `P1=P2`, jika absis dan ordinatnya sama
- Persoalan yang diselesaikan  
`ADT Point` digunakan untuk menentukan posisi player setiap waktu di map.
- Alasan pemilihan karena dalam permainan ini butuh sebuah koordinat untuk menentukan posisi player setiap saat
- Implementasi sebagai `point.h` dan `point.c` dan driver bernama `driver-point.c`

### 3.2 ADT Jam

- Sketsa struktur data  
Pada ADT jam ini mengandung beberapa primitif diantaranya adalah :
  - Selektor
    1. `Hour(J)`
    2. `Minute(J)`
  - `JAM MakeJAM(int H, int M)`  
Primitif ini berfungsi untuk membentuk suatu jam berkomponen `H` (jam) dan `M` (menit) yang valid.
  - `boolean IsJAMValid(int H, int M)`  
Primitif yang berfungsi untuk memvalidasi suatu masukan `H` (jam) dan `M` (menit) yang membentuk suatu jam. Jam yang valid adalah ketika  $23 \geq H \geq 0$  dan  $59 \geq M \geq 0$ .
  - `void BacaJAM (JAM *J)`  
Primitif ini berfungsi untuk mengecek suatu masukan `H` (jam) dan `M` (menit) apakah masukan tersebut valid membentuk jam atau tidak, jika valid proses akan berlanjut ke pembentukan jam dengan memanggil primitif `MakeJAM(int H, int M)` dan jika tidak valid akan otomatis mengulang masukan `H` dan `M` sampai valid.
  - `void TulisJAM (JAM J)`

Memiliki *initial state* berupa sembarang  $J$ . Kemudian  $J$  diproses hingga memiliki *final state* berupa menuliskan komponen nilai  $J$  ke layar dalam format  $HH$  hour (s)  $MM$  minute (s). Jika komponen  $MM$  pada  $J$  0 maka yang dituliskan dilayar hanya  $HH$  hour (s).

- long JAMToMenit (JAM  $J$ )  
Mengonversi jam menjadi menit.
- JAM MenitToJAM (long  $N$ )  
Mengonversi menit menjadi jam.
- JAM NextNMenit (JAM  $J$ , int  $N$ )  
Mengirim  $N$  menit setelah  $J$  dalam bentuk JAM.
- long Durasi (JAM  $JAw$  JAM  $JAkh$ )  
Mengirimkan nilai dari selisih antara  $JAkh$  dan  $JAw$  dalam menit. Jika  $JAw > JAkh$  maka  $JAkh$  adalah satu hari setelah  $JAw$ .
- Persoalan yang diselesaikan  
ADT jam memiliki fungsi untuk mengolah seluruh hal yang berkaitan dengan jam. Contohnya ketika kita ingin menghitung *time remaining*, pembentukan *current time* dan *opening time* pada *preparation phase*, jika pada *main phase* berarti pembentukan *current time* dan *closing time*.
- Alasan pemilihan sebab dalam permainan ini memiliki komponen waktu, jadi untuk mempermudah pengaksesan dan pembuatan komponen waktu primitif-primitif pada ADT jam ini dibuat.
- Implementasi sebagai jam.h dan jam.c serta *driver* bernama driver-jam.c

### 3.3 ADT List Implementasi Array

- Sketsa struktur data  
ADT ini digunakan untuk menyimpan daftar aksi yang dilakukan oleh pemain beserta durasinya. Selain itu, ADT ini digunakan untuk menyimpan daftar barang dan harganya.
  - Selektor
    1. length(list)
    2. max(list)
    3. listStr(list, idx)
    4. listInt(list, idx)
  - void makeList (int len, list \*ls)  
Digunakan untuk membuat sebuah list ls dengan melakukan alokasi dengan panjang len.
  - void deleteList (list \*ls)  
Melakukan penghapusan seluruh elemen list ls dengan melakukan dealokasi.
  - void append (List \*ls, ListTuple newElement)  
Berfungsi untuk menambahkan newElement menjadi elemen terakhir dari ls.
- Persoalan yang diselesaikan  
ADT ini berfungsi untuk menyimpan daftar aksi beserta durasi dari aksi dan menyimpan daftar barang beserta harga.
- Alasan pemilihan sebab permasalahan yang diatasi masing-masing menyimpan jumlah “info” yang sama untuk tiap elemen, maka ADT ini dirasa sangat cocok digunakan.
- Implementasi sebagai listarray.h dan listarray.c serta *driver* bernama driver-listarray.c

### 3.4 ADT Matriks

- Sketsa struktur data  
ADT ini memiliki beberapa primitif seperti di bawah ini,
  - Selektor
    1. rowLen(M)
    2. colLen(M)
    3. Elmt(M,i,j)
    4. occupiedAt(M,i,j)
    5. entityAt(M,i,j)
    6. buildingAt(M,i,j)
  - void makeMatrix(int row, int col, matrix \*mat)  
Membuat sebuah matriks mat sebesar row\*col.
  - void deleteMatrix (matrix \*mat)  
Digunakan untuk menghapus sebuah matriks mat.
- Persoalan yang diselesaikan  
ADT ini mengatasi persoalan mengenai map.
- Alasan pemilihan karena representasi map yang berupa sebuah bidang datar dengan 2 dimensi memiliki kesamaan dengan ADT matriks. Indeks matriks juga sangat berguna untuk mengakses koordinat pada map.
- Implementasi sebagai matrix.h dan matrix.c serta *driver* bernama driver-matrix.c

### 3.5 ADT Mesin Karakter + ADT Mesin Kata

- Sketsa struktur data ADT Mesin Karakter  
Pada ADT Mesin Karakter ini mengandung beberapa primitif diantaranya adalah :
  - State
    1. extern char CC
    2. extern boolean EOP
  - void START()  
Primitif ini berfungsi untuk memulai membaca pita karakter. Karakter pertama adalah yang pita posisinya pada jendela.
  - void ADV()  
Primitif ini berfungsi untuk memajukan pita karakter sebanyak satu karakter.
- Pada ADT Mesin Kata ini mengandung beberapa primitif diantaranya adalah :
  - State
    1. extern boolean EndKata
    2. extern Kata CKata
  - void wordInput()  
Primitif ini berfungsi sebagai awal mulai dari pengakuisisian kata yang ditampung dalam CKata. Awal mula CC sembarang dan diakhir proses ketika sudah berhasil mengakuisisi kata (ketika EndKata = true dan CC = Mark) CC karakter pertama sesudah karakter terakhir kata.
  - void IgnoreBlank()  
Primitif ini berfungsi untuk menghiraukan blank.
  - void ADVKATA()  
Primitif ini berfungsi untuk mengakuisisi kata berikutnya. Akuisisi kata dilakukan dengan prosedur SalinKata().



- `void SalinKata()`  
Primitif ini berfungsi untuk mengakuisisi kata, dan menyimpannya ke dalam `CKata`. Awal dari proses `CC` merupakan karakter pertama dari kata dan diakhir proses `CC = blank` atau `CC=Mark` dan `CC` adalah karakter sesudah karakter terakhir yang diakuisisi. Jika panjang kata melebihi `NMax`, maka sisa kata dipotong.
- Persoalan yang diselesaikan  
Disini ADT Mesin Karakter dan Mesin Kata berfungsi untuk menerima sebuah masukan seperti saat menerima masukan dari “Masukkan perintah”.
- Alasan pemilihan: Kami memilih ADT Mesin Karakter dan Mesin Kata ini karena ADT ini yang memproses sebuah masukan perintah.
- Implementasi sebagai `mesinkar.h`, `mesinkar.c`, `mesinkata.h` dan `mesinkata.c` serta *driver* bernama `driver-mesinkata.c`

### 3.6 ADT Queue (Priority Queue)

- Sketsa struktur data  
ADT Queue (Priority Queue) memiliki beberapa primitif berupa:
  - Selektor
    1. `Prio(e)`
    2. `QueueInfo(e)`
    3. `Head(Q)`
    4. `Tail(Q)`
    5. `QueueInfoHead(Q)`
    6. `QueueInfoTail(Q)`
    7. `QueueMaxEl(Q)`
    8. `QueueElmt(Q,i)`
  - `void PrintQueueChar(PrioQueueChar Q)`  
Mencetak isi queue ke layar
  - `void Dequeue(PrioQueueChar *Q, infotype *X)`  
F.S `x = nilai elemen HEAD` pada I.S, HEAD “maju” dengan mekanisme circular buffer. `Q` mungkin kosong
  - `void Enqueue(PrioQueueChar *Q, infotype X)`  
F.S `x` disisipkan pada posisi yang tepat sesuai dengan priority queue, terurut mengecil berdasarkan prio
  - `void QueueDealokasi(PrioQueueChar *Q)`  
F.S `Q` menjadi tidak terdefinisi lagi, `QueueMaxEl(Q)` diset 0
  - `void MakeEmptyQueue(PrioQueueChar *Q, int Max)`  
F.S Sebuah `Q` kosong terbentuk dan salah satu kondisi sbb  
Jika alokasi berhasil, Table memori dialokasi berukuran `QueueMaxEl` atau jika alokasi gagal, `Q` kosong dengan `QueueMaxEl`.
  - `int NBElmt(PrioQueueChar Q)`  
Mengirimkan banyaknya elemen queue. Mengirimkan 0 jika `Q` kosong
  - `boolean QueueIsFull(PrioQueueChar)`  
Mengirim true jika tabel penampung elemen `Q` sudah penuh yaitu mengandung elemen sebanyak `QueueMaxEl`
- Persoalan yang diselesaikan

ADT Queue (Priority Queue) digunakan untuk menyimpan skala prioritas dari pemain pada Main Phase. Misalnya Pengunjung yang sudah dilayani akan masuk lagi ke dalam queue pemain dengan prioritas lebih tinggi dan wahana yang sudah dipilih pada saat melayani akan dihapus dari daftar wahana yang ingin diikuti

Alasan pemilihan

Sebab dalam Main Phase player memiliki skala prioritas. Semakin tinggi skala prioritasnya berarti player tersebut sangat di nomor satu kan untuk bermain.

Persoalan yang diselesaikan

ADT Queue (Priority Queue) digunakan untuk menyimpan skala prioritas dari pemain pada Main Phase. Misalnya Pengunjung yang sudah dilayani akan masuk lagi ke dalam queue pemain dengan prioritas lebih tinggi dan wahana yang sudah dipilih pada saat melayani akan dihapus dari daftar wahana yang ingin diikuti

- Alasan pemilihan queue adalah paling cocok merepresentasikan antrian pada wahana pada program utama.
- Implementasi sebagai `prioqueuechar.h` dan `prioqueuechar.c` serta *driver* bernama `driver-prioqueuechar.c`.

### 3.7 ADT Stack

- Sketsa struktur data

ADT *Stack* memiliki beberapa primitif berupa,

- Selektor
  1. `Top(S)`
  2. `InfoTop(S)`
- `void CreateEmpty (Stack *S)`

Prosedur ini berguna untuk membuat sebuah *stack* *s* yang kosong dengan kapasitas `MaxEl (10)`.

- `boolean IsEmpty (Stack S)`

Mengirim nilai `true` jika *stack* *s* kosong. *Stack* kosong adalah *stack* yang mempunyai nilai `TOP Nil`.

- `boolean IsFull (Stack S)`

Mengirim `true` jika elemen *stack* *s* penuh.

- `void Push (Stack * S, infotype X)`

Berguna menambahkan elemen *x* sebagai elemen *stack* *s*. Kondisi awal *stack* *s* tidak penuh dan mungkin kosong dengan kondisi akhir elemen *x* menjadi `TOP` yang baru.

- `void Pop (Stack * S, infotype* X)`

Berfungsi untuk menghapus elemen *x* dari *stack* *s*. Kondisi awal *stack* *s* tidak kosong dan *x* merupakan elemen `TOP` lama.

- Persoalan yang diselesaikan

ADT *stack* digunakan untuk menyimpan aksi yang akan dilakukan saat wahana ditutup, seperti membangun wahana dan meng-*upgrade* wahana. Selain itu, ADT ini digunakan untuk melakukan *undo* aksi yang dilakukan saat wahana ditutup serta melakukan *stack* saat eksekusi aksi.

- Alasan pemilihan: Sebab dalam membangun wahana dan meng-*upgrade* wahana dapat dilakukan aksi *undo*, pemilihan ADT *Stack* dirasa sangat tepat. Primitif Pop pada ADT *Stack* dapat mengatasi hal tersebut.
- Implementasi sebagai *stack.h* dan *stack.c* serta *driver* bernama *driver-stack.c*.

### 3.8 ADT List Implementasi List Berkait

- Sketsa struktur data
  - Selektor
    1. Info(P)
    2. Next(P)
    3. First(L)
  - boolean LinIsEmpty (List L)  
Mengembalikan nilai true jika list L adalah list kosong dan mengembalikan nilai false bila list L bukan list kosong.
  - void LinCreateEmpty (List \*L)  
Digunakan untuk membuat sebuah list kosong.
  - address LinAlokasi (infotype X)  
Mengembalikan alamat hasil alokasi dari x jika alokasi berhasil dan mengembalikan nilai Nil jika alokasi gagal.
  - void DeLinAlokasi (address \*P)  
Melakukan dealokasi pada elemen alamat P.
  - address LinSearch (List L, infotype X)  
Mengembalikan alamat dari elemen dengan value x dari list L.
  - void InsVLast (List \*L, infotype X)  
Melakukan alokasi untuk elemen x, jika alokasi berhasil elemen x akan ditambahkan menjadi elemen terakhir yang baru dari list L. Jika alokasi gagal tidak akan terjadi perubahan.
  - void DelVLast (List \*L, infotype \*X)  
Elemen terakhir dari list L akan dihapus dan didealokasi serta disimpan dalam x.
  - void InsertAfter (address P, address Prec)  
Menambahkan elemen beralamat P setelah elemen beralamat Prec.
  - void InsertLast (List \*L, address P)  
Elemen beralamat P akan ditambahkan ke list L dan menjadi elemen terakhir yang baru.
  - void DelLast (List \*L, address \*P)  
Alamat P adalah alamat dari elemen terakhir dari list L. Alamat akan dihapus dari list L dan predesesor dari P akan menjadi elemen terakhir yang baru.
  - void PrintInfo (List L)  
Menampilkan seluruh nilai elemen dari list L.
  - int LinNBElmt (List L)  
Mengembalikan nilai dari banyak elemen dari list L.
- Persoalan yang diselesaikan  
ADT ini digunakan untuk menyimpan daftar *upgrade* yang telah dilakukan pada wahana.
- Alasan pemilihan: Dalam melakukan peng-*upgrade*-an, prosesnya dilakukan secara berurutan dan berkaitan. Oleh sebab itu, ADT List Implementasi List Berkait memiliki kesesuaian dengan sifat dari peng-*upgrade*-an.

- Implementasi sebagai listlinear.h dan listlinear.c serta *driver* bernama driver-linear.c

### 3.9 ADT Tree

- Sketsa struktur data

ADT *Tree* memanfaatkan ADT lain, yaitu ADT list rekursif. Adapun primitif dari ADT *Tree* yaitu,

- Selektor
  1. Akar(P)
  2. Left(P)
  3. Right(P)
- BinTree Tree (infotype Akar, BinTree L, BinTree R)  
Jika alokasi berhasil, akan menghasilkan pohon biner dari Akar, L, dan R. Jika gagal akan menghasilkan pohon kosong (Nil).
- void MakeTree (infotype Akar, BinTree L, BinTree R, BinTree \*P)  
Kondisi awal Akar, L, dan R terdefinisi. Akan membentuk sebuah pohon P dengan dari Akar, L, dan R jika alokasi berhasil. Jika gagal akan menghasilkan pohon kosong (P = Nil).
- addrNode AlokNode (infotype X)  
Jika alokasi berhasil akan mengembalikan Akar(P) = X, Left(P) = Nil, dan Right(P) = Nil. Jika gagal mengembalikan Nil.
- void DealokNode (addrNode P)  
Melakukan dealokasi addrNode P.
- boolean IsTreeEmpty (BinTree P)  
Mengembalikan nilai true jika P adalah pohon biner kosong.
- boolean IsTreeOneElmt (BinTree P)  
Mengembalikan nilai true jika P adalah pohon biner yang memiliki satu elemen.
- boolean IsUnerLeft (BinTree P)  
Mengembalikan nilai true jika P adalah pohon biner unerleft.
- boolean IsUnerRight (BinTree P)  
Mengembalikan nilai true jika P adalah pohon biner unerright.
- boolean IsBiner (BinTree P)  
Mengembalikan nilai true jika P adalah pohon biner yang tidak kosong di unerright dan unerleft.
- void PrintTree (BinTree P, int h)  
Menampilkan pohon di layar dengan h sebagai indentasi.
- boolean SearchTree (BinTree P, infotype X)  
Mengembalikan nilai true jika pohon P memiliki elemen x.
- int NbElmt (BinTree P)  
Mengirimkan jumlah elemen (node) pohon P.
- int NbDaun (BinTree P)  
Mengirimkan jumlah daun (node) pohon P.
- int Level (BinTree p, infotype X)  
Mengirimkan level dari elemen node x pada pohon P.
- int Tinggi (BinTree P)  
Mengirimkan tinggi pohon P.
- void AddDaun (BinTree \*P, infotype X, infotype Y, boolean Kiri)

Kondisi awal pohon tidak kosong dan  $x$  adalah daun dari pohon  $P$ . Kondisi akhir menambahkan  $y$  sebagai anak kiri  $x$  jika  $Kiri = true$ , atau sebagai anak kanan  $x$  jika  $Kiri = false$ .

- `void DelDaun (BinTree *P, infotype X)`

Menghapus semua daun bernilai  $x$ .

- `List MakeListDaun (BinTree P)`

Menghasilkan list dengan elemennya adalah semua daun pohon  $P$  jika semua alokasi list berhasil. Daun terkiri akan menjadi list paling pertama.

- `List MakeListPreorder (BinTree P)`

Menghasilkan list dengan elemennya adalah semua daun pohon  $P$  dengan urutan preorder jika semua alokasi list berhasil.

- `List MakeListLevel (BinTree P, int N)`

Menegembalikan sebuah list dengan elemennya merupakan semua elemen pohon  $P$  pada level  $N$ .

- Persoalan yang diselesaikan

ADT ini digunakan sebagai pohon untuk upgrade wahana.

- Alasan pemilihan: Struktur ADT pohon sesuai dengan kondisi upgrade yang diharapkan.

### 3.10 ADT Graph (Variasi Multilist)

- Sketsa struktur data

- Selektor

1. `Vertex(P)`

1. `Next(P)`

2. `NumVertices(G)`

3. `AdjList(G)`

- `addressGraphNode createNode(int X);`

Membuat node kosong dengan nilai vertexnya  $x$  dan next NULL

- `addrGraph createAGraph(int vertices);`

Membuat graph kosong dengan banyak simpul sebanyak vertices yang kemudian dibuat adjacency list

- `void addEdge(addrGraph graph, int src, int dst);`

Menambahkan sisi ke graph dari source ke destination

- `void printGraph(addrGraph graph);`

Menampilkan graph secara keseluruhan berupa banyak simpulnya dan simpul lain yang terhubung dengan simpul tersebut

- `boolean isGraphConnected(addrGraph graph, int src, int dest);`

Mengecek apakah dua buah simpul terhubung, akan mereturn true jika terhubung dan false jika tidak terhubung

- Persoalan yang diselesaikan

Persoalan yang diselesaikan dari ADT ini adalah persoalan mengecek apakah dua buah map saling terhubung. Apabila dua map terhubung, maka player dapat melakukan perpindahan ke map sebelahnya.

- Alasan pemilihan: Alasan pemilihan ADT ini sebagai solusi dari permasalahan karena graph paling tepat merepresentasikan dua buah objek saling terhubung, dalam hal ini map.

- Implementasi sebagai graph.h dan graph.c serta *driver* bernama driver-graph.c

### 3.11 ADT Wahana

- Sketsa struktur data

- Selektor

1. ID (W)
2. Harga (W)
3. WahanaDurasi (W)
4. Kapasitas (W)
5. Frekuensi (W)
6. Penghasilan (W)
7. PenghasilanHari (W)
8. FrekuensiHari (W)
9. StatusWahana (W)
10. Nama (W)
11. Deskripsi (W)
12. Gambar (W)

ADT Wahana memiliki beberapa primitif berupa:

- Wahana\* createWahanaByID(Wahana\* W, int ID)

Menghasilkan Wahana yang selektornya sudah di assign semua dengan input parameter berupa ID.

- void destroyWahana(Wahana\* W)

Menghapus wahana dengan dealokasi pointer Wahana\*

- void setStatusWahana(Wahana\* W, int newStatus)

Mengeset status baru wahana apakah wahana tersebut broken atau baik-baik saja

- boolean searchWahanaByID(Wahana\* W, int ID)

Mencari Wahana berdasarkan ID nya

- int getHargaWahanaByID(Wahana\* W, int ID)

Mengambil harga masing-masing wahana berdasarkan ID wahana tersebut

- int getWahanaDurasiByID(Wahana\* W, int ID)

Mengambil durasi wahana berdasarkan ID nya

- int getKapasitasByID(Wahana\* W, int ID)

Mengambil kapasitas masing-masing wahana berdasarkan IDnya

- int getFrekuensiByID(Wahana\* W, int ID)

Mengambil data pengunjung wahana tiap hari berdasarkan IDnya

- int getStatusWahanaByID(Wahana\* W, int ID)

Mengambil data status wahana apakah broken atau masih dalam kondisi baik-baik saja

- int getIndexByID(Wahana\* W, int ID)

Mengambil data index wahana berdasarkan wahana IDnya

- int getCharByID(Wahana\* W, int ID)

Mengambil data char wahana berdasarkan IDnya

- Persoalan yang diselesaikan

Sebab pada game ini setiap Wahana bisa di upgrade atau bahkan bisa rusak ketika di dimainkan. Oleh karena itu, butuh sebuah ADT yang bisa mengcover masalah tersebut. ADT Wahana sangat cocok untuk menjawab permasalahan tersebut.

- Alasan pemilihan: Alasan pemilihan ADT ini sebagai solusi untuk memecahkan masalah adalah karena ADT ini memiliki fungsi-fungsi yang dapat menunjang program utama.
- Implementasi sebagai wahana.h, dan driver-wahana.c

### 3.12 ADT Material

- Sketsa struktur data
  - Selektor
    1. MaterialID (M)
    2. MaterialHarga (M)
    3. MaterialNama (M)
    4. MaterialCount (M)
  - boolean searchMaterialByID(Material\*m, int ID)  
Mengembalikan nilai true jika material dengan id ID berada dalam m.
  - int getHargaMaterialByID(Material\*m, int ID)  
Mengembalikan nilai harga dari material m ber-id ID.
  - int getCountMaterialByID(Material\*m, int ID)  
Digunakan untuk menghitung banyaknya material ber-id ID dalam m.
  - void setCountMaterialByID(Material\*m, int ID, int val)  
Prosedur ini akan mengatur banyak material m ber-id ID sebanyak val.
- Persoalan yang diselesaikan  
ADT ini digunakan untuk mengatasi permasalahan yang berkaitan dengan material dalam pembangunan wahana.
- Alasan pemilihan: ADT ini dipilih dan dibuat agar sesuai mungkin dengan apa yang dibutuhkan dalam segala hal yang berkaitan dengan material dalam pembangunan wahana.
- Implementasi sebagai material.h, fileio.h, fileio.c, dan driver-fileio.c

## 4 Program Utama

Program utama tugas besar ini berada dalam file main.c. Pada file ini, hal pertama yang dilakukan adalah pemanggilan fungsi `system` untuk melakukan clear screen untuk membersihkan sisa-sisa text pada terminal. Lalu program akan masuk ke fungsi `startGame` yang akan menginisialisasi program utama. kemudian program akan melakukan konfigurasi seperti memanggil loading bar, melakukan penggambaran terhadap map, dan melakukan penggambaran terhadap info/status yang berada di sebelah kanan map. Kemudian program akan memasuki game loop yang terdiri dari fungsi `prepDay` untuk preparation phase dan `playDay` untuk main phase. Program akan memasuki fungsi `endGame` apabila pemain memutuskan untuk keluar dari permainan.

## 5 Algoritma-Algoritma Menarik

### 5.1 Algoritma Double Buffer

Algoritma ini digunakan untuk menggambar peta pada terminal. Algoritma ini menggunakan basis dasar perbandingan antara dua matriks yang berbeda. Apabila terdapat suatu perubahan pada salah satu matriks, perubahan akan dibandingkan dengan matriks satunya lagi. Kemudian perubahan ini akan dimasukkan ke dalam matriks yang lama. Algoritma ini digunakan pada saat melakukan penggambaran pada peta. Algoritma ini menarik karena dapat mengurangi flickering yang terjadi apabila tidak.

## **6 Data Test**

### **6.1 Pindah Map**

Fitur yang dites adalah perpindahan pemain pada peta, hasil yang seharusnya diberikan adalah pemain dapat melakukan pergerakan bebas tanpa terhalangi icon legend di map pada saat preparation phase dan pemain akan tidak dapat melewati icon legend di map saat berada di main phase.

### **6.2 Buy Material**

Fitur yang dites adalah pembelian material pada preparation phase. Hasil yang seharusnya diberikan adalah pemain dapat melakukan pembelian material pada preparation phase selama uang mencukupi.

### **6.3 Build Wahana**

Fitur yang dites adalah pembangunan wahana pada preparation phase. Hasil yang seharusnya didapat adalah pemain dapat melakukan pembangunan wahana apabila uang dan material mencukupi pada saat preparation phase.

### **6.4 Execute**

Fitur yang dites adalah melakukan perintah execute saat berada pada preparation phase. Hasil yang diharapkan adalah semua aksi pada preparation phase tersimpan kemudian player akan masuk ke main phase.

### **6.5 Main**

Fitur yang dites adalah melakukan perintah main saat berada pada preparation phase. Hasil yang diharapkan adalah semua aksi pada preparation phase tidak tersimpan kemudian player akan masuk ke main phase.

### **6.6 Detail**

Fitur yang dites adalah detail wahana apabila player berdiri di atas wahana pada preparation phase atau berdiri di sebelah wahana dan berjalan menuju wahana pada main phase. Hasil yang diharapkan adalah dapat memunculkan detail wahana.

### **6.7 Detail pada Office**

Fitur yang dites adalah detail wahana yang dilakukan pada saat player memasuki office. Hasil yang diharapkan adalah memunculkan daftar semua wahana yang sudah dibangun dan menampilkan detailnya.

### **6.8 Laporan pada Office**

Fitur yang dites adalah laporan wahana yang dilakukan pada saat player memasuki office. Hasil yang diharapkan adalah memunculkan daftar semua wahana yang sudah dibangun dan menampilkan laporannya.



## 6.9 Serve

Fitur yang dites adalah bagaimana player dapat melayani queue saat berada di antrean. Hasil yang diharapkan adalah queue berkurang dan orang yang sedang mengantre memasuki wahana.

## 6.10 Upgrade

Fitur yang dites adalah upgrade yang dilakukan terhadap wahana yang sudah dibangun pada saat preparation phase. Hasil yang diharap adalah wahana berhasil terupgrade apabila uang, materi mencukupi.

## 6.11 Undo

Fitur yang dites adalah undo yang dapat dilakukan pada preparation phase. Hasil yang diharap adalah player dapat membatalkan aksi yang baru saja ia lakukan di preparation phase.

## 6.12 Repair

Fitur yang dites adalah perbaikan wahana. Hasil yang diharap adalah player dapat memperbaiki wahana yang rusak pada saat preparation phase atau main phase dan uang player mencukupi.

# 7 Test Script

**Tabel 7.1** Tabel *Test Script*.

No	Fitur yang Dites	Tujuan Testing	Langkah-Langkah Testing	Input Data Test	Hasil yang Diharapkan	Hasil yang Keluar
1	Pindah map	Mengetahui apakah pemain dapat melakukan perpindahan antara map	1. Memulai game baru 2. Melakukan pergerakan ke gerbang yang berada pada map	Pergerakan w a s d	Pemain dapat memasuki gerbang dan berpindah map	Pemain dapat melakukan perpindahan ke map lain
2	Buy material	Mengetahui apakah pemain dapat melakukan pembelian material apabila uang mencukupi	1. Memulai game baru 2. Memasuki preparation phase 3. Memasukkan perintah “buy” pada console 4. Memilih material yang akan dibeli 5. Memasukkan jumlah material yang akan dibeli	Mengetikkan perintah buy, memilih ID material dari tabel daftar material yang muncul, mengetikkan jumlah material yang akan dibeli	Pemain dapat melakukan pembelian material dengan jumlah tertentu dengan syarat uang yang dimiliki mencukupi	Pemain dapat melakukan pembelian material apabila sudah memenuhi syarat-syarat yang diperlukan.
3	Build wahana	Mengetahui apakah pemain dapat melakukan pembangunan wahana apabila uang dan material mencukupi	1. Memulai game baru 2. Memasuki preparation phase 3. Memilih spot kosong pada peta 4. Memasukkan perintah “build” pada console 5. Memilih ID wahana yang akan dibangun	Mengetikkan perintah build, memilih ID dari tabel daftar wahana yang tersedia	Pemain dapat melakukan pembangunan pada map. Syarat agar pemain dapat melakukan pembangunan adalah pembangunan harus dilakukan pada spot kosong pada peta, kemudian uang dan	Pemain dapat melakukan pembangunan apabila sudah memenuhi syarat-syarat yang diperlukan.

					material yang dimiliki harus mencukupi.	
4	Execute	Mengetahui apakah pemain dapat menjalankan aksi-aksi yang sudah dilakukan pada saat preparation day dan masuk ke main day	<ol style="list-style-type: none"> <li>1. Memulai game baru</li> <li>2. Melakukan aksi seperti membangun wahana, membeli material</li> <li>3. Memasukkan perintah "execute" pada console</li> <li>4. Memilih "y" atau "n"</li> </ol>	Mengetikkan <code>execute</code> setelah melakukan berbagai macam aksi pada saat preparation day lalu mengetik <code>y</code> atau <code>n</code>	Aksi yang sudah dilakukan akan tersimpan lalu pemain akan masuk ke main phase	Aksi yang sudah dilakukan tersimpan kemudian pemain akan masuk ke main phase
5	Main	Mengetahui apakah pemain dapat pindah ke main phase dari preparation phase	<ol style="list-style-type: none"> <li>1. Memulai game baru</li> <li>2. Memasukkan perintah "main" pada console</li> <li>3. Mengetik "y" atau "n"</li> </ol>	Mengetikkan <code>main</code> pada saat preparation phase, lalu mengetik <code>y</code> atau <code>n</code>	Pemain dapat berpindah ke main phase pada saat berada di preparation phase, apabila pemain mengetikkan perintah <code>y</code> setelah main, maka fase game akan masuk ke mode main phase tanpa menjalankan aksi-aksi yang sudah dilakukan pada saat preparation phase. Apabila pemain mengetikkan <code>n</code> , tidak akan masuk ke main phase dan akan tetap berada di preparation day.	Pemain dapat masuk ke main phase tanpa mengeksekusi aksi yang telah dilakukan pada preparation phase apabila memilih <code>y</code> dan pemain akan tetap berada pada preparation phase apabila memilih <code>n</code>
6	Detail	Mengetahui apakah pemain dapat melihat status wahana apabila sedang berdiri disamping wahana lalu berjalan ke arah wahana pada saat main phase atau pada saat berdiri tepat di atas icon wahana pada saat preparation phase	<ol style="list-style-type: none"> <li>1. Memulai game baru</li> <li>2. Berada pada preparation phase/main phase</li> <li>3. Mengarahkan player ke sebelah bangunan yang akan dicek</li> <li>4. Mengetikkan <code>detail</code> pada console</li> </ol>	Mengetik <code>detail</code> pada saat berada di sebelah wahana	Pemain dapat melihat detail wahana dengan syarat pemain berada pada sebelah wahana tersebut dan kemudian berjalan ke arah wahana tersebut apabila melakukannya pada saat main phase dan pemain berada tepat pada icon wahana apabila melakukannya pada preparation phase.	Pemain dapat melihat detail wahana apabila memenuhi syarat.
7	Detail pada office	Mengetahui apakah pemain dapat melihat detail wahana apabila sedang berada di dalam office	<ol style="list-style-type: none"> <li>1. Memulai game baru</li> <li>2. Membangun wahana pada saat preparation phase</li> <li>3. Melakukan perintah <code>execute</code></li> <li>4. Mengarahkan player ke sebelah office</li> <li>5. Pada saat berada di sebelah office, mengetikkan <code>detail</code> pada console</li> </ol>	Mengetikkan <code>detail</code> pada console pada saat berada di office	Pemain dapat melihat detail semua wahana yang sudah dibangun pada saat preparation phase yang meliputi nama, ID, lokasi, harga, durasi, kapasitas, deskripsi, dan history upgrade..	Pemain dapat melihat detail semua wahana apabila sudah berada di samping office.

8	Laporan pada office	Mengetahui apakah pemain dapat melihat laporan wahana apabila sedang berada di dalam office	<ol style="list-style-type: none"> <li>1. Memulai game baru</li> <li>2. Membangun wahana pada saat preparation phase</li> <li>3. Melakukan perintah execute</li> <li>4. Mengarahkan player ke sebelah office</li> <li>5. Pada saat berada di sebelah office, mengetikkan laporan pada console</li> </ol>	Mengetikkan laporan pada console saat berada di office	Pemain dapat melihat laporan semua wahana yang sudah dibangun pada saat preparation phase yang meliputi nama, ID, dinaiki, total hasil, dinaiki hari ini, dan total hasil hari ini.	Pemain dapat melihat laporan semua wahana apabila sudah berada di samping office.
9	Serve	Mengetahui apakah pemain dapat melakukan serve saat berada pada main phase dan berada di sebelah antrean kemudian mengarah ke antrian	<ol style="list-style-type: none"> <li>1. Memulai game</li> <li>2. Membangun wahana pada preparataion phase</li> <li>3. Melakukan perintah execute</li> <li>4. Mengarahkan player ke sebelah antrean</li> <li>5. Mengarahkan w a s d ke arah antrian</li> <li>6. Menetik perintah serve pada console</li> </ol>	Mengarahkan w a s d ke arah antrian kemudian mengetikkan perintah serve	Pemain dapat melayani orang yang sedang mengantri pada wahana, setelah di serve maka orang tersebut akan memasuki wahana dan keluar dari antrean.	Pemain dapat melakukan serve pada orang yang sedang mengantre yang mengakibatkan orang tersebut keluar dari antrean dan masuk wahana.
10	Upgrade	Mengetahui apakah pemain dapat melakukan upgrade saat berada di preparation phase dan sedang berdiri diatas icon wahana yang akan di upgrade	<ol style="list-style-type: none"> <li>1. Memulai game</li> <li>2. Membangun wahana</li> <li>3. Mengetikkan perintah upgrade ke console</li> <li>4. Memilih dari list upgrade yang ada</li> </ol>	Mengetikkan perintah upgrade ke console	Pemain dapat melakukan upgrade terhadap wahana yang sudah dibangun sebelumnya pada saat preparation phase apabila material dan uang mencukupi.	Pemain dapat melakukan upgrade apabila syarat-syarat disamping terpenuhi.
11	Undo	Mengetahui apakah pemain dapat melakukan aksi undo apabila sedang berada di preparation phase dan sudah melakukan aksi sebelumnya.	<ol style="list-style-type: none"> <li>1. Memulai game</li> <li>2. Melakukan aksi pada preparation phase</li> <li>3. Mengetikkan perintah undo ke console</li> </ol>	Mengetikkan perintah undo ke console	Pemain dapat membatalkan aksi yang telah dilakukan sebelumnya pada saat berada pada preparation phase.	Pemain dapat membatalkan aksi yang telah dilakukan sebelumnya saat berada di preparation phase.
12	Repair	Mengetahui apakah pemain dapat melakukan aksi repair pada wahana yang rusak pada saat preparation phase maupun main phase	<ol style="list-style-type: none"> <li>1. Memulai game</li> <li>2. Berada pada preparation phase/main phase</li> <li>3. Apabila pada preparation phase, harus berada di atas wahana</li> <li>4. Apabila pada main phase, harus berada di sebelah wahana dan berjalan mengarah ke building</li> <li>5. Mengetikkan perintah repair ke console</li> <li>6. Apabila wahana rusak, ketik y atau n</li> </ol>	Mengetikkan perintah repair ke console	Pemain dapat melakukan reparasi terhadap wahana yang rusak dengan syarat berada di atas icon wahana apabila berada di preparation phase atau berada di sebelah wahana kemudian berjalan mengarah ke arah wahana apabila berada di main phase. Apabila wahana tidak rusak akan	Pemain dapat melakukan perbaikan terhadap wahana apabila memenuhi syarat yang ditentukan.

			7. Apabila wahana tidak rusak, akan muncul pesan bahwa wahana tidak rusak		dimunculkan pesan bahwa wahana tidak rusak, sementara apabila wahana rusak akan dimunculkan pesan bahwa wahana telah diperbaiki apabila uang mencukupi dan memilih untuk memperbaiki wahana.	
13	Prepare	Mengetahui apakah pemain dapat kembali ke preparation phase saat sedang berada pada main phase	1. Berada pada main phase 2. Mengetik prepare pada console	Mengetik prepare pada console pada saat berada di main phase	Pemain dapat berpindah ke preparation phase apabila mereka sedang berada di main phase, kemudian queue akan dikosongkan dan akan kembali ke preparation phase.	Pemain dapat berpindah ke preparation phase, queue pada main phase akan dikosongkan.

## 8 Pembagian Kerja dalam Kelompok

**Tabel 8.1** Tabel Pembagian Tugas.

NIM	Pembagian Tugas
13519046	ADT Jam, ADT Mesin Kata, Laporan
13519110	ADT Point, ADT Queue, Laporan
13519146	ADT Linked List, ADT Graph, Fileio, Laporan
13519206	ADT Stack, ADT Tree, Fileio, Laporan
13519214	ADT List, ADT Matriks, Engine, Laporan, Integrasi

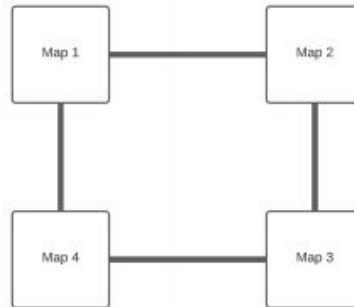
## 9 Lampiran

### 9.1 Deskripsi Tugas Besar 2

Willy Wangky telah menerima program yang kamu dan teman-temanmu telah buat untuk mengurus tiket taman bermainnya. Namun, taman bermain yang dibuat oleh Oompa Loompa dianggap kurang memuaskan oleh Willy Wangky. Maka dari itu, ia menginginkan kamu untuk membuat program dalam bahasa C untuk mensimulasikan taman bermain sehingga ia dapat mendesain taman bermainnya terlebih dahulu sebelum dibangun oleh Oompa Loompa.

Wahana bermain Willy Wangky memiliki 4 petak lahan yang dihubungkan oleh sebuah lorong. Salah satu dari 4 petak lahan yang dimiliki merupakan lahan khusus. Pada lahan tersebut, terdapat minimal tempat penyimpanan bahan-bahan bangunan, sebuah antrian dan sebuah office di mana Willy Wangky dapat melihat hal-hal berikut:

1. Melihat detail dari wahana.
2. Melihat laporan penjualan wahana.



Contoh peta taman bermain

Sebelum hari dimulai, Willy Wangky ingin mempersiapkan segala sesuatu untuk wahana bermainnya. Kegiatan yang akan dilakukan saat persiapan adalah sebagai berikut:

1. Membeli bahan-bahan bangunan apabila uang mencukupi.
2. Membangun wahana baru di taman bermain untuk menambah pemasukan apabila bahan bangunan mencukupi.
3. Melakukan upgrade pada wahana yang sudah ada di taman bermain apabila bahan bangunan mencukupi.

Karena Willy Wangky juga sibuk mengurus pabrik coklatnya (selain mengurus wahana bermainnya), program simulasi harus mampu menyimpan kegiatan yang akan dilakukan saat persiapan. Kegiatan terakhir harus dimasukkan sebelum kegiatan sebelumnya dikarenakan Willy Wangky ingin melakukan Undo apabila aksi terakhir yang dia lakukan kurang memuaskan.

Selain mensimulasikan persiapan, program juga harus dapat mensimulasikan kegiatan pada saat hari kerja (saat taman bermain dibuka). Pada saat taman bermain dibuka:

1. Terdapat pengunjung yang datang setiap waktu tertentu dan mengantri untuk masuk ke dalam taman bermain dengan daftar jenis wahana yang ingin diikuti.
2. Setelah mengikuti sebuah wahana, pengunjung akan mengantri lagi dalam antrian dengan prioritas lebih tinggi. Dengan kata lain, pengunjung yang sudah dilayani harus dilayani terlebih dahulu.

Program simulasi juga harus memungkinkan Willy Wangky untuk:

1. Berjalan mengelilingi wahana bermain yang telah dibangun (wahana yang dibangun sendiri).
2. Melayani pengunjung yang berada dalam antrian.
3. Dikarenakan wahana memiliki kemungkinan untuk rusak, maka Willy Wangky harus mendatangi wahana yang bersangkutan untuk memperbaiki wahana tersebut.
4. Masuk ke dalam office dan melihat detail atau laporan wahana.

Berdasarkan deskripsi game di atas, buatlah sebuah program yang dapat mensimulasikan game tersebut menggunakan bahasa C. Interaksi pengguna dan komputer dilakukan melalui command line interface (CLI). Pengguna memasukkan perintah-perintah yang akan dijelaskan pada bagian game mechanics dan command.

## 9.2 Notulen Rapat

### Rapat 1 November 2020

Mereview spesifikasi tugas besar, melakukan pembagian tugas untuk membuat ADT sesuai dengan spesifikasi

### Rapat 5 November 2020

Asistensi pertama dan pengumpulan ADT yang telah dibuat

### Rapat 16 November 2020

Membuat laporan dan menginvite semua anggota kelompok ke dalam laporan

### Rapat 18 November 2020

Pembahasan mekanisme game, membuat preparation game, menggambar di engine.c

### Rapat 19 November 2020

Asistensi kedua dan melanjutkan isi program

### Rapat 20 November 2020

Menyesuaikan ADT dan preparation phase

### Rapat 26 November 2020

Melanjutkan isi program (debugging dan testing preparation phase)

### Rapat 27 November 2020

Melanjutkan isi program (mulai pengerjaan main phase, mengecek isi laporan)

### Rapat 28 November 2020

Melanjutkan isi program (debugging dan testing main phase)

### Rapat 29 November 2020

Finalisasi program dan laporan

## 9.3 Log Activity Anggota Kelompok

Dilakukan oleh semua anggota kelompok secara berbarengan/paralel

**Tabel 9.1** Tabel *Log Activity* Semua Anggota Kelompok.

Tanggal	Kegiatan
1 November 2020	Pembagian Tugas ADT
5 November 2020	Asistensi Pertama, pengumpulan ADT yang sudah dibuat
16 November 2020	Membuat laporan (inisialisasi awal dan membuat gdocs berdasarkan template)
18 November 2020	Pembahasan mekanisme game, melanjutkan isi laporan
19 November 2020	Asistensi Kedua
20 November 2020	Melanjutkan isi program, menyelesaikan sebagian besar preparation phase dan menyesuaikan ADT yang dibuat agar sesuai dengan preparation phase
26 November 2020	Debugging dan testing preparation phase

27 November 2020	Mulai pengerjaan main phase, mengecek isi laporan
28 November 2020	Debugging dan testing main phase
29 November 2020	Finalisasi program dan laporan

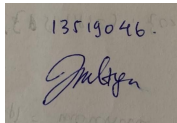
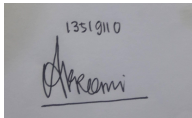
## 9.4 Form Asistensi

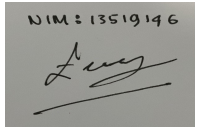



### Form Asistensi Tugas Besar IF2110/Algoritma dan Struktur Data Sem. 1 2019/2020

No. Kelompok/Kelas : 04/K-2  
 Nama Kelompok : mikum  
 Anggota Kelompok (Nama/NIM) :  
 1. Dwianditya Hanif/13519046  
 2. Mohammad Afif Akromi/13519110  
 3. Fadel Ananda Dotty/13519146  
 4. Muhammad Fawwaz Naabigh/13519206  
 5. Tanur Rizaldi Rahardjo/13519214  
 6.

Asisten Pembimbing : Aliffiqri Agwar/13517107

#### Asistensi I

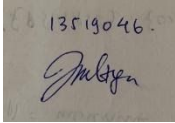
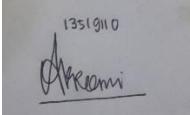
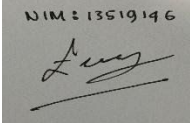


<b>Tanggal : 5 November 2020</b>	<b>Catatan Asistensi:</b> <b>1. Q: Penjelasan lebih detail mengenai modul dan driver ?</b> <b>A : Modul adalah file .h dan .c sedangkan driver adalah file main nya. 1 modul itu berisi adt dan adt baru untuk menunjang 1 fitur. Jadi modul itu per fitur. Hal ini untuk pengecekan saja kalau program tidak berjalan akan dicek sebenarnya yang eror itu di main nya atau pada adt nya.</b> <b>2. Q : untuk movement nya kalau realtime bagaimana ?</b> <b>A : Mengikuti spesifikasi saja, jadi movement nya tetap input perintah tidak realtime.</b> <b>3. Q : Untuk kesabaran itu bagaimana ya ?</b> <b>A : selama tidak di serve tetap di antrean, urutannya bergantung prioritas yang ada. Dan akan hilang ketika timer habis. Timer di definisikan sendiri di adt jam.</b> <b>4. Q : apakah struktur data yang (ADT) di spesifikasi harus adad ?</b> <b>A: iya karena itu menjadi patokan penilaian.</b>
<b>Tempat : Via Google Meet</b>	
<b>Kehadiran Anggota Kelompok:</b>	
No NIM Tanda tangan 1  2  3	

 <p>4 13519206</p>  <p>5 13519214</p>  <p>6</p>	<p>5. Q : apakah idle tidak menambah waktu ? A : iya, memakan waktu hanya ketika memasukkan perintah saja</p> <p>6. Saat preparation hanya membangun saja. Player tidak ada pergerakan sama sekali, jadi ada semacam kursor untuk membangun. Pesan : ikuti saja adt yang ada, karena patokan nilai dosen itu per adt nya.</p>
	 Tanda Tangan Asisten:

## Asistensi II

Tanggal : 19 November 2020	Catatan Asistensi:
Tempat : Via Google Meet	<p>Q : Apakah player simbolnya harus P di map ? A : tidak, asalakan ada di legenda</p> <p>Q : Apakah ada ketentuan state.txt ? A : Dibebaskan</p> <p>Q : Apakah primitive ADT harus dipakai semua ? A : Bergantung kebutuhan saja untuk pemakaian primitive, dan ADT boleh di modif.</p> <p>Q : Apakah representasi ADT Graph itu Linked List ? A : iya, karena di kelas nanti diajarkannya representasi Linked list</p> <p>Q : Maksud lebih detail mengenai pohon upgrade itu bagaimana ? A : Maksudnya itu adalah cabang dari pohon itu adalah pilihan untuk upgradenya mau yang seperti apa.</p> <p>Untuk map sesuai dengan yang ada di spesifikasi (gambar awal) yaitu ada 4 map dan setiap map ada 2 koneksi (pintu).</p>



<p><b>Kehadiran Anggota Kelompok:</b></p> <p>No</p> <p>NIM</p> <p>Tanda tangan</p> <p>1</p>  <p>2</p>  <p>3</p>  <p>4</p> <p>13519206</p>  <p>5</p> <p>13519214</p>  <p>6</p>	
	<p><b>Tanda Tangan Asisten:</b></p> <p><a href="https://docs.google.com/document/d/1U8zAhRR9zaNZ6FqKPxECCZHSqEHfB2uIMvmWfPJgFD8/edit">https://docs.google.com/document/d/1U8zAhRR9zaNZ6FqKPxECCZHSqEHfB2uIMvmWfPJgFD8/edit</a></p>