Spesifikasi Tugas Besar

IF2010 Pemrograman Berorientasi Objek

Deadline Milestone 1: 21 November 2025 Pukul 20:10

Deadline Milestone 2: 13 Desember 2025 Pukul 12:00

Versi 2.2 (12/12/2025)

Changelog

---

1. Sabtu, 15 November 2025
    a. Rilis versi 1.0
2. Minggu, 16 November 2025
    a. Rilis versi 1.1
        i. Menambahkan *interface* wajib dan informasi terkait *trigger*, serta pengumuman asisten kelompok
3. Selasa, 18 November 2025
    a. Rilis versi 1.2
        i. Perbaikan *typo* pada map type C: Burger
4. Kamis, 20 November 2025
    a. Rilis versi 1.3
        i. Penambahan ketentuan **Plate Storage (P)**
5. Selasa, 25 November 2025
    a. Rilis versi 2.0
        i. Penambahan ketentuan terkait *concurrency* serta spesifikasi lainnya secara lebih lengkap

6. Selasa, 9 Desember 2025
    a. Rilis versi 2.1
        i. Penambahan informasi terkait bonus Random Level Generator
7. Jumat, 12 Desember 2025
    a. Rilis versi 2.2
        i. Pengunduran deadline Milestone 2
        ii. Penambahan informasi terkait demo dan teknis pengumpulan *source code*

Daftar Isi

Daftar Isi **3**
Pedoman **4**
Deskripsi Persoalan **6**
Spesifikasi Sistem **7**
Ketentuan Umum 7
Ketentuan Teknis 7
Spesifikasi Permainan **9**
Entitas Permainan 9
Maps 19
Chef Inventory 25
Actions 25
Stage Over 31
Flow Permainan **32**
Menu Game **34**
Bonus **35**
Kelompok **37**
QnA **37**
Asistensi **38**
Pengumpulan **39**
Milestone 1 - 21/11/2025 39
Milestone 2 - 13/12/2025 39
Demo **42**
Extras **43**

Pedoman

Berikut merupakan pedoman yang dapat Anda ikuti selama proses pengerjaan tugas besar

1. Lakukan **validasi** untuk setiap input yang akan dimasukkan.
2. Gunakan ***build automation tools*** seperti **Gradle** atau **Maven** untuk melakukan manajemen *build* dan *package*.
3. Apabila Anda ingin mengimplementasikan **Graphical User Interface (GUI)**, maka **rencanakan sejak awal**, tidak bisa dikerjakan mendadak di akhir. Pastikan program kalian cukup modular dari awal pengerjaan. Sebagai referensi, Anda bisa menerapkan MVC.
4. Dalam mengerjakan program, **lakukan asesmen untuk meninjau** bagian mana saja yang **perlu konkurensi** dan **tidak perlu konkurensi**. Tidak semua bagian perlu dikonkurensikan. Kalian cukup menerapkan konkurensi pada **aksi yang relevan**.
5. Penggunaan ***lock*** harus **dilakukan secara hati-hati**, jangan sampai menyebabkan *deadlock*. Apabila tidak menggunakan *synchronized* maupun *lock*, hati-hati pula terhadap *race condition*.
6. Buat program Anda **mudah diobservasi** sejak awal. Lakukan *logging* secara terstruktur untuk bagian-bagian kode yang dianggap penting. Alih-alih hanya menggunakan println(), coba eksplorasi pustaka *logger* seperti **SLF4J**. Anda **dapat memberi nama *thread*** untuk memudahkan proses *debug* melalui *log* yang dihasilkan.
7. Ketentuan penggunaan Generative AI dalam tugas besar ini adalah **bebas bertanggung jawab**. Jika terdapat penggunaan Generative AI yang tidak bertanggung jawab (mis. tidak mampu menjelaskan kode yang dibuat ketika demo ataupun asistensi) dalam pengerjaan tugas besar ini, **seluruh**

anggota kelompok akan **menerima sanksi BERAT sesuai dengan aturan akademik**.

8. Detail terkait atribut maupun *method* yang didefinisikan dalam spesifikasi ini bersifat **rekomendasi dan tidak saklek** (**dapat Anda modifikasi**) asalkan sesuai dengan tujuan permainan.
9. **KERJAKAN SECARA TERSTRUKTUR DAN KOORDINASIKAN PEKERJAAN ANDA**. Terapkan ***Git workflow*** (misalnya gunakan skema *branching*) untuk memastikan *commit* bermasalah dapat di-*revert* dengan mudah.
10. **Uji program kalian dengan saksama dan menyeluruh sebelum demo**. Hanya karena program kalian bisa di-*compile*, bukan berarti program kalian *foolproof*. Seluruh fitur juga harus diuji bersamaan (*integration test*), sebab fitur dapat saling mempengaruhi fitur lainnya. **GUNAKAN EMPATI DAN BERSIKAP INISIATIF, jangan hanya memedulikan pekerjaan Anda saja.**

Deskripsi Persoalan

Pada suatu hari yang tenang, **King Pop** terbangun dari tidur siangnya karena suara teriakan yang mengguncang seluruh istana. Kesal karena ketenangannya terganggu, ia segera memanggil para pengawal untuk mencari tahu apa yang terjadi di luar sana. Didapatkan informasi bahwa ribuan **Evil Nimons** menggila di tengah kota. Evil Nimons menjadi liar karena muak oleh kerasnya kehidupan di bawah kekuasaan King Pop yang korup, sehingga **kelaparan** melanda secara masif. Untuk mengantisipasi agar Evil Nimons tidak meluluhlantakkan pertahanan istana, King Pop mengundang duo *chef* legendaris, yaitu **Chef Kebin** dan **Chef Stewart**! Sebenarnya Kebin dan Stewart enggan bekerja sama dengan King Pop, namun tak ada lagi hal yang dapat mereka lakukan lagi selain **menerima tawaran Sang Raja**, lantaran terkena imbas usaha mereka bangkrut akibat daya beli para Nimons menurun. Kebin dan Stewart diminta oleh King Pop untuk **memasak dan menyajikan makanan secepat mungkin** kepada para Evil Nimons di *open kitchen* sekitar istana dengan harapan dapat menenangkan hati Evil Nimons. Mari bantu kedua *chef* jago ini dalam mengelola dapur agar dapat **menyelamatkan negara dari bencana kelaparan! HIDUP KING POP!!~~**

Spesifikasi Sistem

Ketentuan Umum

Anda diminta untuk membuat permainan berbasis ***Command Line Interface*** **(CLI)** bernama **Nimonscooked** menggunakan bahasa pemrograman **Java**. Tuliskan tahapan untuk melakukan kompilasi dan menjalankan program pada file README.md yang dikumpulkan bersama dengan *source code*. **Nimonscooked** dapat dimainkan oleh 1-4 orang pemain, namun untuk kemudahan dalam tugas besar ini akan ditentukan menjadi cukup **1 pemain** saja. Tujuan dari permainan ini dijelaskan pada bagian *Stage Over*. Untuk mewujudkan hal tersebut, ada beberapa aksi yang perlu dilakukan. Penjelasan tentang aksi dapat dibaca di sini. Gambaran besar *flow* permainan dapat dibaca di bagian ini. Setiap kelompok mendapatkan **satu tipe level map** yang harus dibuat. Pembagian tipe level map dapat dicek pada bagian ini. Untuk penjelasan tipe level map dapat dibaca di sini.

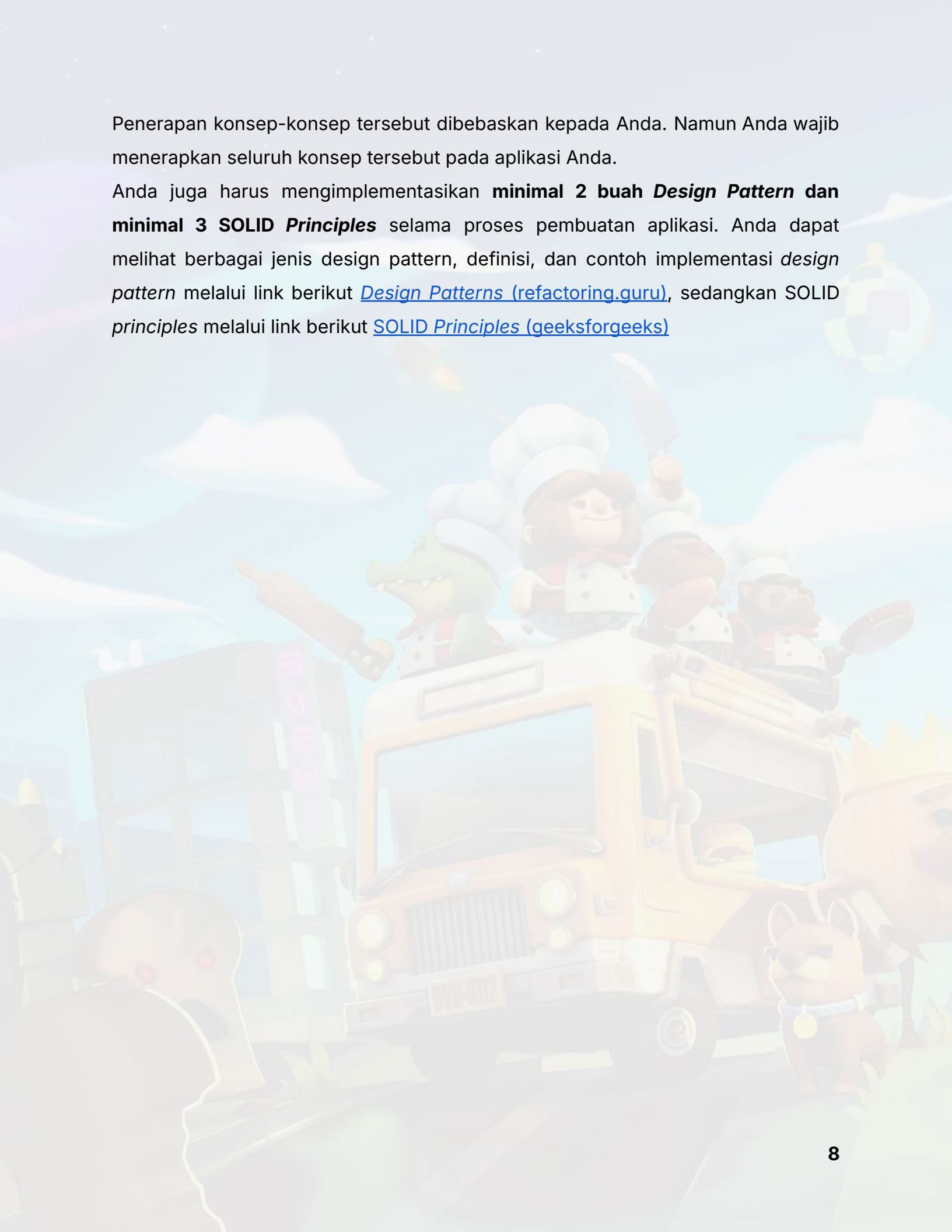
Ketentuan Teknis

Berikut adalah hal-hal yang wajib diimplementasikan di aplikasi yang Anda buat. Perhatikan konsep-konsep OOP serta desain dari aplikasi kalian!

1. *Inheritance*
2. *Abstract Class* / *Interface*
3. *Polymorphism*
4. *Generics*
5. *Exceptions*
6. *Collections*
7. *Concurrency* (Contoh implementasi *concurrency*)

Penerapan konsep-konsep tersebut dibebaskan kepada Anda. Namun Anda wajib menerapkan seluruh konsep tersebut pada aplikasi Anda.

Anda juga harus mengimplementasikan **minimal 2 buah *Design Pattern* dan minimal 3 SOLID *Principles*** selama proses pembuatan aplikasi. Anda dapat melihat berbagai jenis design pattern, definisi, dan contoh implementasi *design pattern* melalui link berikut *Design Patterns* (refactoring.guru), sedangkan SOLID *principles* melalui link berikut SOLID *Principles* (geeksforgeeks)



Spesifikasi Permainan

Entitas Permainan

1. ***Chef Player***

   Chef Player adalah entitas yang dikendalikan oleh pemain. Dalam permainan ini *chef* berjumlah **dua**. Setiap *chef* dapat bergerak, mengambil, menaruh, serta menggunakan stasiun di dapur. *Chef* memiliki posisi, arah hadap, dan satu slot inventori tangan yang dapat memegang **satu item**.

   ### 1.1. Atribut

   Chef Player **setidaknya memiliki atribut sebagai berikut**.

Nama Atribut	Tipe	Keterangan
id	String	Identitas unik untuk tiap *chef*
name	String	Nama pemain atau karakter *chef*
position	Position	Posisi koordinat chef pada *grid* peta
direction	Direction	Arah hadap *chef* (UP, DOWN, LEFT, RIGHT)
inventory	Item (nullable)	*Item* yang sedang dipegang (maksimal satu)
currentAction	Enum	Menandakan aksi yang sedang dilakukan oleh Chef tersebut.

   ### 1.2. Perilaku dan Aturan

   #### 1.2.1. Pergerakan

   a. *Chef* dapat bergerak ke atas, bawah, kiri, dan kanan menggunakan input *keyboard* (mis. **`WASD`**)

   b. *Chef* tidak dapat bergerak ke petak berisi dinding, *station*, atau *chef* lain

c. Perubahan posisi dan arah hadap terjadi secara atomik tiap langkah

1.2.2. Inventori

a. *Chef* hanya dapat memegang **satu item** dalam satu waktu
b. *Item* yang dapat dipegang: *Ingredient*, *Dish*, *Kitchen Utensils*, atau kombinasi *Kitchen Utensils* dan *Dish*
c. ~~Jika *Chef* sedang memegang ***Plate* bersih**, dan item di depan adalah *ingredient* dalam kondisi akhirnya atau *dish*, maka *item* akan otomatis **digabungkan ke *Plate*** (*plating*) dan *plate* tersebut diletakkan pada tempat *ingredient* atau *dish* tadi.~~
d. *Plate* yang kotor tidak dapat digunakan untuk *plating* sebelum dicuci di Washing Station

1.2.3. Interaksi

a. *Chef* dapat berinteraksi dengan *station* di arah hadapnya menggunakan tombol aksi.
b. Jenis interaksi tergantung pada tipe station di depan:
    i. *Ingredients Storage*: mengambil bahan mentah (RAW)
    ii. *Cutting Station*: memotong bahan menjadi CHOPPED
    iii. *Cooking Station*: memasak bahan menjadi COOKED atau BURNED
    iv. *Plate Storage*: mengambil *plate* bersih yang disediakan di awal game atau *plate* yang telah digunakan (kotor)
    v. *Assembly Station*: menyusun *dish* di atas *plate*
    vi. *Serving Counter*: menyajikan *dish* ke pelanggan
    vii. *Washing Station*: meletakkan dan mencuci *plate* kotor menjadi bersih, serta mengambil *plate* bersih
    viii. *Trash Station*: membuang ~~item~~ *ingredient* atau *dish* dari tangan
    ix. Jika di depan tidak ada *station*, interaksi menjadi ***Pick Up / Put Down*** terhadap *item* yang ada di lantai

1.2.4. Ganti *Chef*

i. Pemain mengendalikan dua *Chef* dalam satu level, namun hanya satu *Chef* yang aktif dan dapat digerakkan pada satu waktu.

ii. *Chef* yang tidak aktif akan tetap diam di posisinya dan mempertahankan *item* yang sedang dibawa, sehingga inventori setiap *chef* juga terpisah (item yang dibawa oleh *Chef* A tidak otomatis pindah ke *Chef* B).

iii. Misal saat *Chef* A sedang melakukan aksi, lalu berpindah ke *Chef* B, *Chef* A tetap melanjutkan pekerjaannya. Setelah *Chef* A menyelesaikan pekerjaannya, *Chef* A hanya diam di tempat.

1.2.5. *Busy State*

i. Aksi-aksi berdurasi dijalankan pada *thread* terpisah menggunakan mekanisme *concurrency*.

2. *Item*

Item merupakan objek-objek fisik yang dapat diambil, dipindahkan, dan digunakan oleh *chef* di dapur. Semua benda yang bisa dipegang atau diproses termasuk kategori *item*, di antaranya ***ingredient***, ***dish***, dan ***kitchen utensils***. *Item* merupakan ***superclass* abstrak**.

2.1 *Ingredient*

Ingredient adalah bahan baku yang digunakan untuk membuat menu makanan. Contoh ingredient: tomat, daging, beras, nori, dll. Setiap *ingredient* **harus berada dalam salah satu *state* berikut**.

State	Keterangan
RAW	Mentah
CHOPPED	Sudah dipotong
COOKING	Sedang dimasak
COOKED	Sudah dimasak
BURNED	Hangus

Setiap ingredient disarankan mengimplementasikan **interface** berikut:

```
public interface Preparable {

    boolean canBeChopped();
    boolean canBeCooked();
    boolean canBePlacedOnPlate();

    void chop();
    void cook();
}
```

2.2 *Dish*

Dish adalah ~~hidangan akhir~~ suatu hidangan yang merupakan hasil kombinasi beberapa *ingredient* yang sudah diproses ~~sesuai resep~~ (disatukan di suatu piring). Contohnya adalah di map sushi, Nori + Udang yang disatukan di piring merupakan hidangan yang VALID (dapat disajikan), meskipun tidak ada dalam resep. ~~Setiap dish **setidaknya memiliki atribut sebagai berikut.**~~

~~**Nama Atribut**~~	~~**Tipe**~~	~~**Keterangan**~~
~~name~~	~~String~~	~~Nama hidangan~~
~~components~~	~~List<Preparable>~~	~~List ingredient dalam dish~~

2.3 *Kitchen Utensils*

Kitchen Utensils adalah wadah makanan untuk penyajian atau untuk masak, seperti *Plate*, *Boiling Pot*, *Frying Pan*, dan Oven. *Boiling Pot* hanya dapat digunakan untuk memasak beras dan pasta. *Frying pan* hanya dapat digunakan untuk *ingredient* tertentu dengan status CHOPPED. Oven tidak portabel (diam di *cooking station*) dan hanya dapat digunakan untuk memasak *pizza*. Setiap peralatan **setidaknya memiliki atribut sebagai berikut**.

Nama Atribut	Tipe	Keterangan
contents	Set<Preparable>	Daftar *ingredient* yang saat ini berada di dalam *kitchen utensils*

Setiap alat masak (*boiling pot*/*frying pan*/oven) disarankan implementasikan **interface** berikut:

```
public interface CookingDevice {

    boolean isPortable();
    int capacity();
    boolean canAccept(Preparable ingredient);

    void addIngredient(Preparable ingredient);
    void startCooking();
}
```

3. ***Station***

Station merupakan area kerja tempat *chef* melakukan aksi pemrosesan terhadap *item*. Setiap *station* memiliki fungsi tertentu dan hanya dapat digunakan untuk aksi spesifik. Pemain hanya dapat berinteraksi dengan *station* apabila berdiri di petak yang berdekatan (*adjacent cell*). Jenis-jenis *station* yang ada di permainan Nimonscooked adalah sebagai berikut:

Nama	Fungsi
***Cutting Station* (C)**	Memotong bahan mentah menjadi bentuk siap olah, serta dapat juga digunakan untuk menaruh bahan atau merakit hidangan seperti *Assembly Station*. Proses memotong membuat *chef* dalam keadaan *busy state*. Jika *chef* meninggalkan *cutting station* saat progres memotong sedang berlangsung, progres akan berhenti namun tidak mengulang dari awal, sehingga saat *chef* kembali memotong, akan melanjutkan sisa progres
***Cooking Station* (R)**	Memasak bahan mentah atau *chopped* menjadi matang **dengan bantuan *kitchen utensils*** (*boiling pot*/*frying pan* untuk level yang menggunakan *stove*, dan oven untuk level tipe D). Proses masak dijalankan secara otomatis saat *ingredients* dimasukkan ke *kitchen utensils* di atas *cooking station*. Selama proses ini berlangsung, *chef* tidak harus terus berada di *adjacent cell* sampai proses selesai. Namun harus segera mengangkat *kitchen utensils* berisi *ingredients* sebelum waktu untuk BURNED
***Assembly Station* (A)**	Menyediakan permukaan kosong untuk menaruh bahan dan merakit hidangan

***Washing Station* (W)**	Mencuci *plate* kotor agar dapat digunakan kembali. *Plate* kotor dapat ditumpuk lebih dari satu *plate*. Namun, tetap dicuci satu per satu. *Washing station* dibagi menjadi dua bagian, satu untuk mencuci, satu untuk menaruh piring yang telah dicuci. Proses mencuci membuat *chef* dalam keadaan *busy state*. Jika *chef* meninggalkan cucian saat progres sedang berlangsung, progres akan berhenti namun tidak mengulang dari awal, sehingga saat *chef* kembali mencuci, akan melanjutkan sisa progres
***Serving Counter* (S)**	Menyajikan *dish* yang sudah selesai **dengan *plate*** kepada pelanggan
***Ingredient Storage* (I)**	Menyediakan stok bahan mentah dari satu jenis *ingredient*. Jumlah stok bahan mentah tak terbatas. *Station* ini juga dapat digunakan untuk menaruh bahan atau merakit hidangan seperti *Assembly Station*
***Plate Storage* (P)**	Menampung piring bersih dan piring yang telah digunakan. Notes: - Pada awal permainan semua piring bersih akan berada pada tumpukan *stack*.

	- Piring kotor yang berasal dari *serving* akan langsung dikirim ke bagian paling atas *stack*. - Piring ~~bersih~~ hanya bisa diambil 1 per 1 ~~- Piring kotor dapat langsung diambil semuanya (jika ada 2 piring kotor pada tumpukan piring paling atas, maka saat player mengambil piring akan terambil 2 piring kotor)~~ - Piring bersih dapat diambil apabila tidak ada piring kotor yang berada di atas tumpukan - Tidak dapat melakukan *drop* item apapun pada station ini
***Trash Station* (T)**	Membuang bahan atau isi hidangan yang tidak dibutuhkan (hidangan salah, gosong, dsb).

4. ***Order***

Order merupakan representasi pesanan pelanggan yang harus diselesaikan. *Order* muncul secara berurutan selama permainan berlangsung. Jika ada dua order yang sama lalu salah satu diselesaikan, selesaikan yang paling awal masuk. Jenis dish yang datang harus dibuat **random** namun tetap sesuai tipe levelnya. Setiap satu *order* berhasil diselesaikan, akan muncul satu *order* baru. *Order* terus bermunculan hingga permainan berakhir. Banyaknya *order* maksimal yang ditampilkan boleh Anda batasi misalnya 4 atau 5 *order*. Tiap *order* memiliki beberapa atribut penting seperti yang dijelaskan pada tabel berikut.

Atribut	Tipe Data	Deskripsi	Contoh Nilai
Posisi *Order*	Integer	Nomor urut pesanan yang aktif di permainan	1
Recipe	Recipe	Resep atau menu yang diminta oleh pelanggan	Objek Recipe Kappa Maki
Reward	Integer (skor / poin)	Nilai skor yang diperoleh pemain jika *order* berhasil disajikan dengan benar.	+120
Penalty	Integer (skor / poin)	Nilai penalti jika order gagal atau waktu limit *order* habis.	−50
Time Limit	Integer (Seconds)	Batas waktu maksimal untuk menyelesaikan 1 pesanan	60 (1 Menit)

5. ***Recipe***

Recipe adalah spesifikasi atau aturan yang mendefinisikan sebuah hidangan yang harus dibuat oleh pemain. Setiap *recipe* terdiri dari daftar *ingredient* beserta kondisi pemrosesan yang harus dipenuhi, misalnya bahan harus dalam keadaan sudah dipotong, sudah dimasak, atau mentah, sesuai kebutuhan resep. Berikut contoh resep.

Nama resep	Ingredients
Kappa Maki	Nori (Mentah), Nasi (*Cooked*), Timun (*Chopped*)

Maps

Setiap kelompok akan mendapatkan **satu tipe map** sesuai pembagian kelompok. Semua *map* dirancang dengan dimensi yang sama (14 x 10) namun memiliki *layout* dan penempatan *station* yang berbeda untuk menciptakan pengalaman *gameplay* yang unik.

Legenda

C = *Cutting Station* (Stasiun pemotongan)

R = *Cooking Station* (Stove/Oven)

A = *Assembly Station* (Stasiun perakitan)

S = *Serving Counter* (Konter penyajian)

W = *Washing Station* (*Sink* untuk cuci piring)

I = *Ingredient Storage* (Penyimpanan bahan/*Crate*)

P = *Plate Storage* (Penyimpanan piring)

T = *Trash Station* (Tempat sampah)

X = *Wall*/*Obstacle* (Dinding/Rintangan)

. = *Walkable Space* (Ruang untuk berjalan)

V = *Spawn Chef Point* (Tempat awal *chef* di awal *game*)

MAP TYPE A: SUSHI MAP

Resep:

1. Kappa Maki: Nori (*Raw*) + Nasi (*Cooked*) + Timun (*Chopped*)
2. Sakana Maki: Nori (*Raw*) + Nasi (*Cooked*) + Ikan (*Raw*)
3. Ebi Maki: Nori (*Raw*) + Nasi (*Cooked*) + Udang (*Cooked*)
4. Fish Cucumber Roll: Nori (*Raw*) + Nasi (*Cooked*) + Ikan (*Raw*) + Timun (*Chopped*)

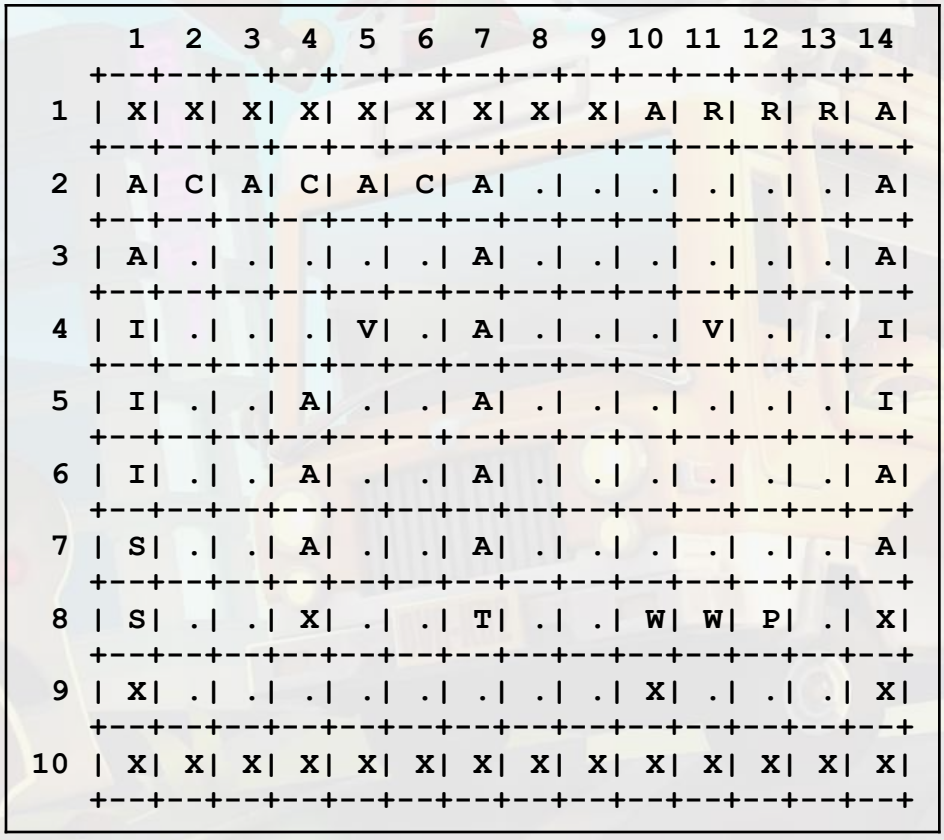
Bahan yang Tersedia:

- Beras (perlu dimasak → Nasi)
- Nori (pakai mentah)
- Timun (perlu dipotong)
- Ikan (perlu dipotong)
- Udang (perlu dipotong, lalu dimasak)

Kitchen Utensils yang Tersedia:

- 2 *Boiling Pot* di atas *Stove*
- 1 *Frying Pan* di atas *Stove*
- 4 *Plate*

Layout



MAP TYPE B: PASTA MAP

Resep:

1. Pasta Marinara: Pasta (*Cooked*) + Tomat (*Cooked*)
2. Pasta Bolognese: Pasta (*Cooked*) + Daging (*Cooked*)
3. Pasta Frutti di Mare: Pasta (*Cooked*) + Udang (*Cooked*) + Ikan (*Cooked*)

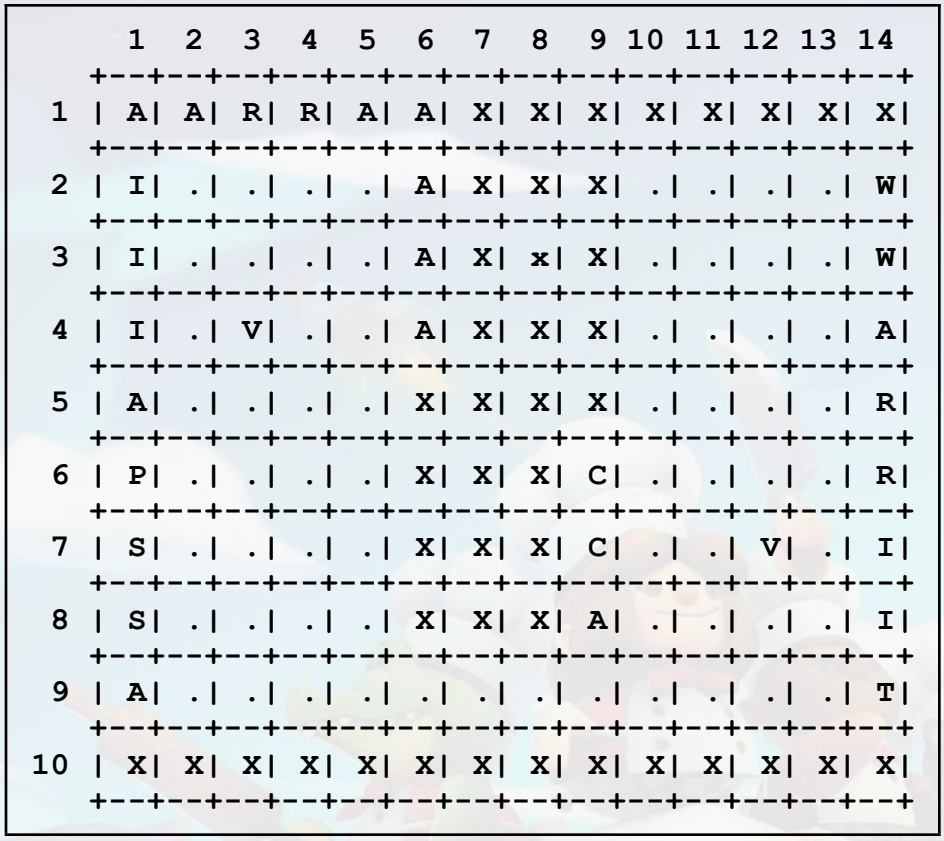
gBahan yang Tersedia:

- Pasta (perlu dimasak)
- Tomat (perlu dipotong, lalu dimasak)
- Udang (perlu dipotong, lalu dimasak)
- Ikan (perlu dipotong, lalu dimasak)
- Daging (perlu dipotong, lalu dimasak)

Kitchen Utensils yang Tersedia:

- 2 *Boiling Pot* di atas *Stove*
- 2 *Frying Pan* di atas *Stove*
- 4 *Plate*

Layout



```
     1  2  3  4  5  6  7  8  9 10 11 12 13 14
   +--+--+--+--+--+--+--+--+--+--+--+--+--+--+
 1 | A| A| R| R| A| A| X| X| X| X| X| X| X| X|
   +--+--+--+--+--+--+--+--+--+--+--+--+--+--+
 2 | I| .| .| .| .| A| X| X| X| .| .| .| .| W|
   +--+--+--+--+--+--+--+--+--+--+--+--+--+--+
 3 | I| .| .| .| .| A| X| x| X| .| .| .| .| W|
   +--+--+--+--+--+--+--+--+--+--+--+--+--+--+
 4 | I| .| V| .| .| A| X| X| X| .| .| .| .| A|
   +--+--+--+--+--+--+--+--+--+--+--+--+--+--+
 5 | A| .| .| .| .| X| X| X| X| .| .| .| .| R|
   +--+--+--+--+--+--+--+--+--+--+--+--+--+--+
 6 | P| .| .| .| .| X| X| X| C| .| .| .| .| R|
   +--+--+--+--+--+--+--+--+--+--+--+--+--+--+
 7 | S| .| .| .| .| X| X| X| C| .| .| V| .| I|
   +--+--+--+--+--+--+--+--+--+--+--+--+--+--+
 8 | S| .| .| .| .| X| X| X| A| .| .| .| .| I|
   +--+--+--+--+--+--+--+--+--+--+--+--+--+--+
 9 | A| .| .| .| .| .| .| .| .| .| .| .| .| T|
   +--+--+--+--+--+--+--+--+--+--+--+--+--+--+
10 | X| X| X| X| X| X| X| X| X| X| X| X| X| X|
   +--+--+--+--+--+--+--+--+--+--+--+--+--+--+
```

MAP TYPE C: BURGER MAP

Resep:

1. Classic Burger: Roti + Daging (*Cooked*)
2. Cheeseburger: Roti + Daging (*Cooked*) + Keju (*Chopped*)
3. BLT Burger: Roti + *Lettuce*(*Chopped*) + Tomat(*Chopped*) + Daging(*Cooked*)
4. Deluxe Burger: Roti + *Lettuce* (*Chopped*) + Daging (*Cooked*) + Keju (*Chopped*)

Bahan yang Tersedia:

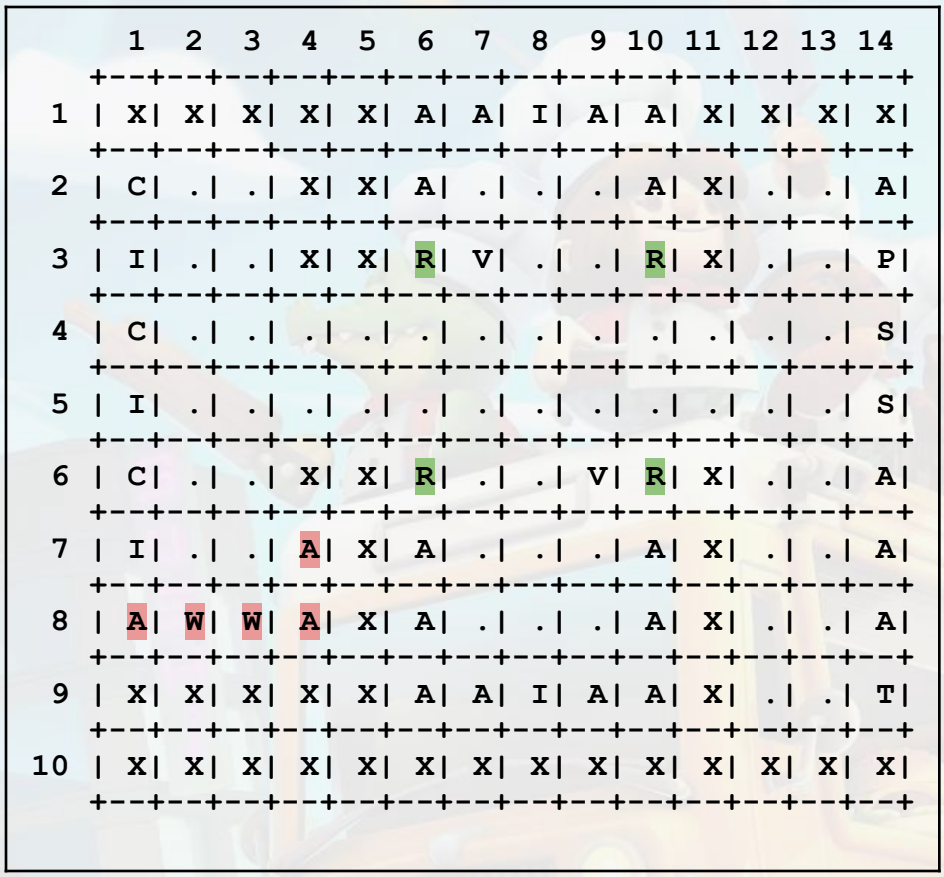
- Roti (pakai mentah)
- Daging (perlu dipotong, lalu dimasak)
- Keju (perlu dipotong)

- *Lettuce* (perlu dipotong)
- Tomat (perlu dipotong)

Kitchen Utensils yang Tersedia:

- 4 *Frying Pan* di atas *Stove*
- 4 *Plate*

Layout



MAP TYPE D: PIZZA MAP

Resep (Semua resep berikut perlu dimasak di oven setelah di-assemble):

1. Pizza Margherita: Adonan (*Chopped*) + Tomat (*Chopped*) + Keju (*Chopped*)
2. Pizza Sosis: Adonan (*Chopped*) + Tomat (*Chopped*) + Keju (*Chopped*) + Sosis (*Chopped*)

3. Pizza Ayam: Adonan (*Chopped*) + Tomat (*Chopped*) + Keju (*Chopped*) + Ayam (*Chopped*) [lalu semua dimasak di dalam **Oven**]

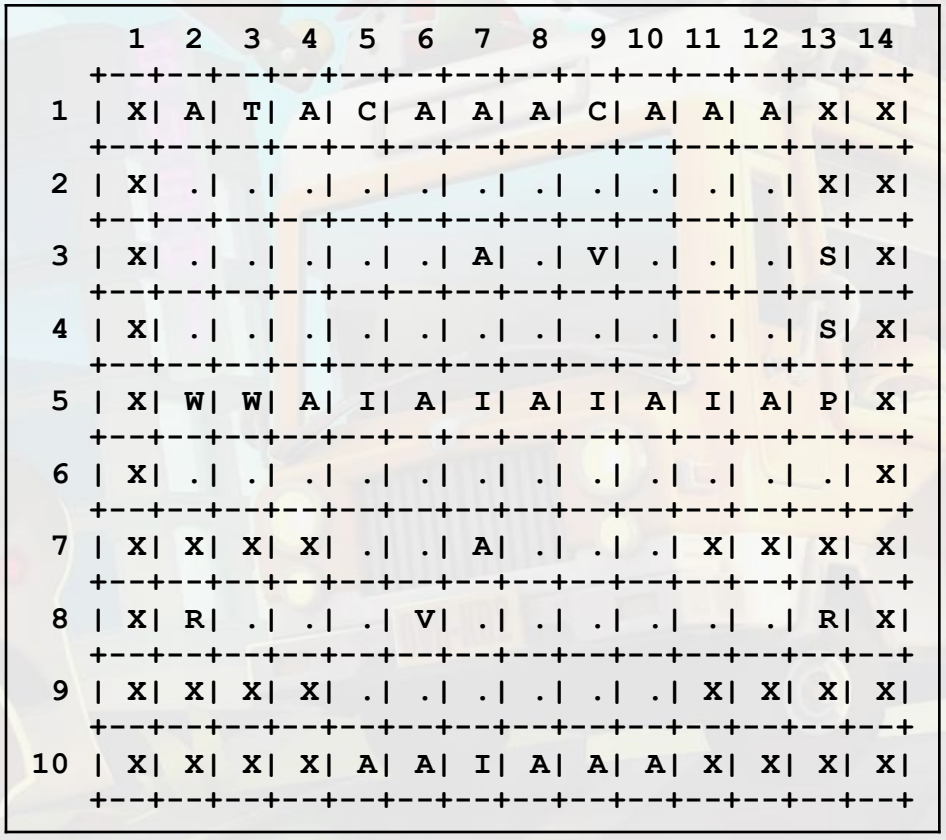
Bahan yang Tersedia:

- Adonan (perlu dipotong)
- Tomat (perlu dipotong)
- Keju (perlu dipotong)
- Sosis (perlu dipotong)
- Ayam (perlu dipotong)

Kitchen Utensils yang Tersedia:

- 2 Oven
- 3 *Plate*

Layout



Chef Inventory

Setiap *chef* hanya dapat memiliki 1 slot inventori. Slot tersebut dapat digunakan untuk mengangkat semua jenis *item*. Apabila slot telah terisi maka *chef* tidak dapat mengambil *item* lain. Pengecualian berlaku apabila yang dipegang oleh *chef* saat ini adalah *kitchen utensils* dan *item* yang hendak diambil adalah *ingredients* (*edible*) atau *dish*. Apabila *item* tersebut berada di *kitchen utensils*, maka *item* yang dipegang oleh *chef* akan berubah menjadi *kitchen utensils* dengan *ingredients* (edible) atau *dish* yang diambil tadi. Apabila yang dipegang oleh *chef* saat ini *kitchen utensils* berisi *ingredients* (*edible*) yang akan dipindahkan ke *kitchen utensils* atau *trash* di depan, maka isi *ingredients* akan berpindah ke tujuan dan inventori *chef* menjadi *kitchen utensils* kosong. ~~Selain itu, piring kotor dapat ditumpuk dan dibawa seluruhnya untuk diletakkan di *washing station*.~~

Actions

Aksi yang dapat dilakukan oleh *player* adalah sebagai berikut:

1. **Move**

Aspek	Deskripsi
Trigger	W (atas), A (kiri), S (bawah), D (kanan)
Input	Arah gerak
Proses	*Chef* berpindah posisi ke arah yang dituju jika *tile* kosong dan tidak ada halangan
Output	Posisi dan arah hadap *chef* berubah
Kegagalan	Jika petak tujuan berisi dinding, *station*, atau *chef* lain → gerak batal

2. **Pick Up / Drop**

Aspek	Deskripsi
Trigger	Jika di depan chef ada item atau tile kosong, lalu pemain menekan tombol *Pickup/Drop* (Default: **C**, namun boleh diganti sesuai desain kontrol tiap kelompok)
Input	*Item* di depan *chef* atau item di inventori *chef*
Proses	- Jika tangan kosong → ambil *item* di depan. - Jika tangan memegang *plate* bersih dan *item edible* di depan → gabungkan (*plating*). Terdapat 2 kondisi plating yang dapat terjadi: 1. Jika item di *kitchen utensils*, *plating* akan dilakukan di *inventory chef*. 2. Jika item di luar *kitchen utensils*, *plating* akan dilakukan di tempat *item* tersebut berada. - Jika tangan memegang *item* dan *tile* depan kosong → letakkan item - Jika mengambil bahan dari *Ingredient Storage*, inventori *chef* akan otomatis terisi oleh bahan *ingredient* terkait - Jika di *Ingredient Storage* terdapat item di atasnya, yang diambil terlebih dahulu adalah *item* di atasnya sebelum dapat mengambil *ingredient* yang ada di dalam *storage*
Output	*Item* berpindah antara tangan dan *tile* sesuai konteks
Kegagalan	*Put down* tidak dapat dilakukan jika piring bersih dimasukkan ke *washing station* atau *cooking station* tanpa *boiling pot/frying pan*. Begitupula piring kotor yang telah dimasukkan ke *washing station* tidak dapat diambil lagi kecuali piring telah dicuci

3. **Use station**

a. *Cutting Action*

Aspek	Deskripsi
Trigger	Pemain menekan tombol *Interact* (Default: **V**, namun boleh diganti sesuai desain kontrol tiap kelompok)
Station	*Cutting Station* [C]
Input	*Ingredient* dengan *state* RAW.

Proses	Chef akan memotong bahan RAW selama durasi yang telah ditentukan hingga bahan berubah menjadi CHOPPED. Chef akan masuk ke dalam *busy state*
Output	*Ingredient* berubah *state* menjadi CHOPPED.
Catatan	Jika aksi dihentikan ketika bahan masih dipotong, progress bar akan berhenti namun tidak hilang.
Kegagalan	Jika *item* bukan *ingredient* mentah atau *station* penuh.
Durasi (Second)	3 Detik

b. *Cooking Action*

Aspek	Deskripsi
Trigger	Pemain memasukkan suatu bahan yang dapat diterima ke alat masak di *cooking station*
Station	*Cooking Station* [R]
Input	*Ingredient* dengan *state* RAW atau CHOPPED
Proses	Bahan yang telah dimasukkan ke dalam alat masak, akan diproses tanpa perlu campur tangan Chef. Bahan yang telah dimasak selama lebih dari **24 detik** akan otomatis **BURNED** dan harus dibuang.
Output	*Ingredient* matang (COOKED)
Kegagalan	Jika tidak ada *boiling pot*/*frying pan* yang diletakkan pada cooking station, tidak akan bisa memasak.
Durasi (Second)	12 Detik untuk menjadi **COOKED**

c. *Assembly* (*Plating*) *Action*

Aspek	Deskripsi
Trigger	Pemain menekan tombol untuk *Pick Up*/*Drop*
Station	*Assembly Station* [A], *Cutting Station* [C], *Ingredient*

	Storage [I], dan *Cooking Station* [R]
Input	*Plate* bersih dan *Ingredient* dengan *state*-nya yang **paling akhir**
Proses	- Kombinasikan bahan menjadi *dish* - *Dish* disimpan di atas *plate*, sesuai kondisi: > Piring bersih di tangan dan *ingredient* berada di [A], [C], atau [I] tanpa *kitchen utensils* (*ingredient* biasanya dalam *state* RAW atau CHOPPED), *plating* dilakukan dan diletakkan di [A], [C], atau [I] > Piring bersih di tangan dan *ingredient* berada di dalam *kitchen utensils* (*ingredient* biasanya dalam *state* COOKED), *plating* dilakukan dan diletakkan di inventori *chef* > *Ingredient* di dalam *kitchen utensils* di tangan dan piring bersih di [A], [C], atau [I], *ingredients* akan pindah ke piring bersih, lalu *kitchen utensils* menjadi kosong di inventori *chef*
Output	*Dish*. *Plate* menjadi kotor setelah *dish* di-*serve*
Kegagalan	Saat *plate* kotor tidak dapat di-*plating*

d. *Washing Action*

Aspek	Deskripsi
Trigger	Pemain menekan tombol *Interact* (Default: **V**, namun boleh diganti sesuai desain kontrol tiap kelompok)
Station	*Washing Station* (*Sink*) [W]
Input	(Beberapa) *Plate* kotor
Proses	*Chef* mencuci *plate* kotor yang sudah di-*store* di dalam *Washing Station* satu per satu. *Plate* yang sudah bersih akan muncul di bagian lain dari tempat mencuci.
Output	*Plate* menjadi bersih
Kegagalan	Jika input bukan *plate* kotor, tidak dapat dicuci.
Durasi (Second)	3 detik untuk **SETIAP** plate

e. *Serving Action*

Aspek	Deskripsi
Trigger	Pemain menekan tombol untuk *Drop*
Station	*Serving Counter* [S]
Input	*Dish* lengkap di atas *plate* bersih
Proses	- Validasi kecocokan dish dengan seluruh *order* aktif - Jika ada yang cocok, berikan skor - Jika tidak ada yang cocok, berikan penalti - Kembalikan piring ke bagian paling atas dari *Plate Storage* dengan kondisi kotor **10 detik** setelah dish disajikan
Output	*Order* terhapus, skor diperbarui, *plate* menjadi kotor di *plate storage*. Jika tidak cocok, *order* tidak dihapus
Kegagalan	Jika tidak ada *plate* yang diserahkan, tidak dapat menyajikan hidangan. Jika chef menyajikan hidangan yang tidak ada di order, maka hidangan tersebut akan dimakan **Kak Jendra (Hilang). Tetapi, piring akan kembali ke *Plate Storage* dengan kondisi kotor.**

f. *Trash Action*

Aspek	Deskripsi
Trigger	Pemain menekan tombol untuk *Drop*
Station	*Trash Station* [T]
Input	*Item* apapun, baik *ingredient* maupun *kitchen utensils* berisi masakan
Proses	*Item* dibuang. Jika *item* adalah *kitchen utensils*, hanya isinya saja yang dihapus
Output	*Item* terhapus dari inventori atau *tile*
Kegagalan	Tidak dapat membuang *kitchen utensils*

4. Switch Chef

Aspek	Deskripsi
Trigger	Pemain menekan tombol *Switch Chef* (Default: **B**, namun boleh diganti sesuai desain kontrol tiap kelompok)
Input	Misalnya saat ini sedang mengendalikan *Chef* A
Proses	- *Chef* otomatis berpindah ke *chef* lainnya. - Apabila *chef* pertama sedang dalam *busy state* (memotong atau mencuci), lalu *switch* ke *chef* berikutnya, maka *chef* pertama tetap melanjutkan pekerjaannya hingga selesai. Setelah selesai, *chef* pertama tetap diam di tempat sampai *chef* kembali di-*switch*
Output	Pemain dapat mengendalikan *Chef* B, sedangkan *Chef* A menjadi tidak aktif

Stage Over

Stage dapat berakhir apabila salah satu kondisi berikut terpenuhi:

1. Time's Up!!! (Waktu Habis)

Trigger: Timer Countdown mencapai **0.**

Action:

- Waktu mulai permainan boleh Anda tentukan sendiri, misalnya dimulai dari **3 menit**
- Tentukan batas minimum untuk *pass* level Anda
- Game berhenti menerima order baru
- Hitung total *score* dan tampilkan
- Evaluasi hasil (*Pass*/*Fail*)

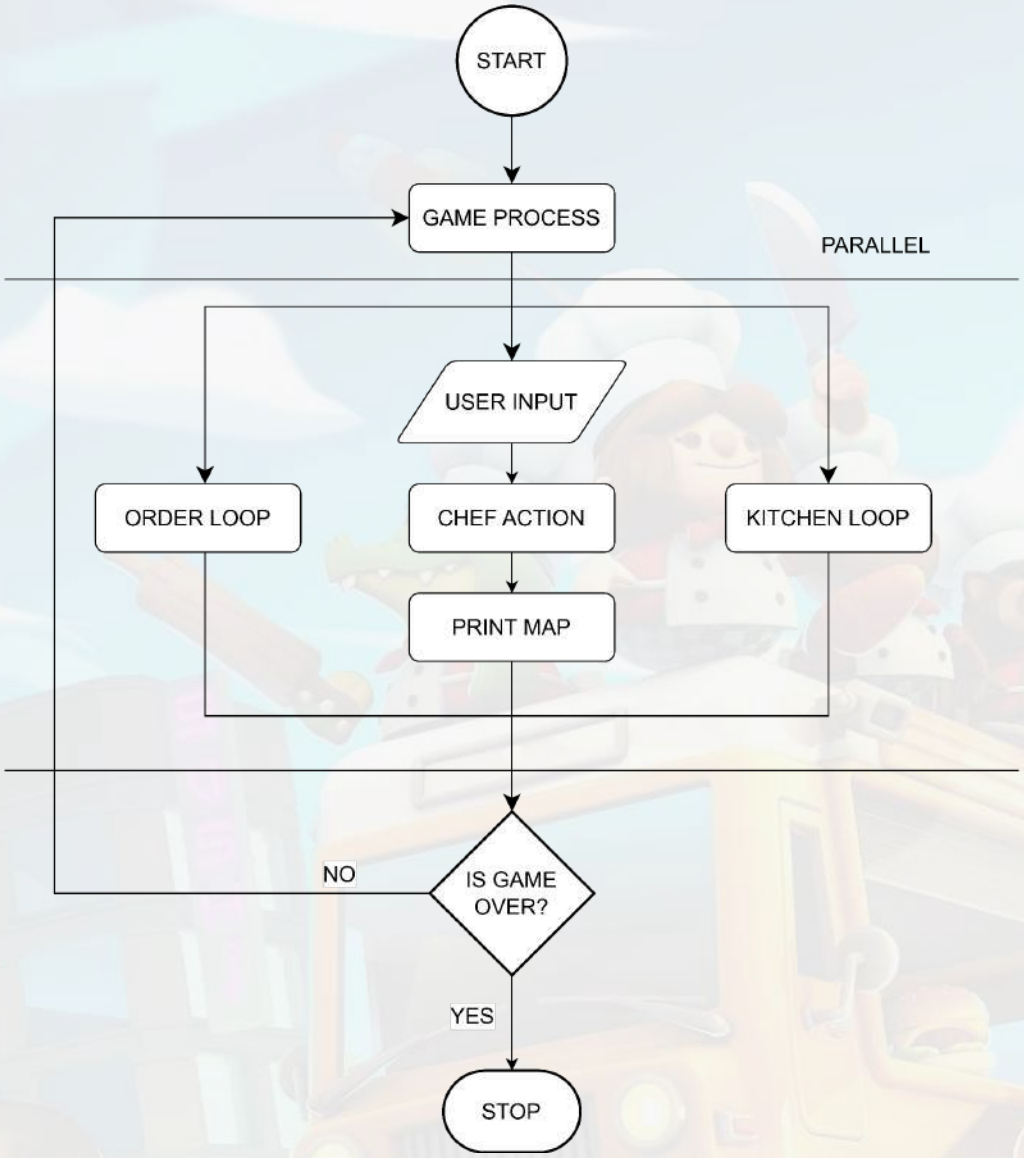
2. *Too Many Failed Orders*

Trigger: Player gagal menyelesaikan X orders berturut-turut (misal: 5 orders)

Action:

- Stage langsung berakhir dengan status FAIL
- Player harus *retry stage*

Flow Permainan



1. ***Order Loop***

   Proses *new order* masuk ke antrian yang harus dikerjakan oleh *chef*. Setiap ada satu *order* berhasil terselesaikan atau durasi *order* telah *expired* akan muncul satu *new order* kembali.

2. ***Kitchen Loop***

   Kitchen loop meliputi proses saat suatu *order* telah dilayani, piring yang telah digunakan untuk *serving* akan kembali ke *plate storage* dalam **10 detik** kemudian dengan kondisi piring sudah kotor. *Chef* harus mencuci piring tersebut sebelum dapat kembali digunakan. *Kitchen loop* juga termasuk proses masak yang otomatis berjalan akibat *trigger* dari *chef*.

Menu Game

Menu *Game* merupakan tampilan utama yang mengatur alur permainan dari sisi pengguna. Menu ini menjadi antarmuka awal sebelum pemain masuk ke dalam *stage* permainan.

Menu dibagi menjadi beberapa bagian sebagai berikut:

1. ***Main Menu***

   Tampil saat program pertama dijalankan.

   a. Pilihan:

      1. *Start Game* → Memulai permainan.
      2. *How to Play* → Menampilkan panduan kontrol dan mekanisme game.
      3. *Exit* → Keluar dari permainan.

2. ***Stage Select Menu***

   Menampilkan *stage* yang dapat dimainkan.

   a. Pemain yang telah menyelesaikan *stage*, status stage jadi SUCCESS.

   b. Pemain dapat melihat *preview* peta dan target skor sebelum memulai.

3. ***Result Screen* (*Post-Stage Menu*)**

   Muncul setelah stage berakhir.

   a. Menampilkan total skor, jumlah *order* sukses/gagal, dan status kelulusan (*Pass*/*Fail*).

   b. Jika stage berhasil diselesaikan, akan muncul animasi *Stage Cleared* dan tombol “Back to Menu”.

Bonus

---

1. **GUI**

   Kalian disarankan untuk mengimplementasi GUI pada tugas besar ini. GUI membuat game kalian memiliki UI dan visual yang membuat game menjadi lebih hidup dan mudah berinteraksinya. Dalam melakukan implementasi GUI, kalian dibebaskan untuk menggunakan *package* apapun, contoh *package* GUI yang dapat digunakan yaitu Java Swing (sudah bawaan dari Java sehingga kalian tidak perlu meng-*install* sendiri), atau JavaFX (*package third-party* sehingga kalian perlu meng-*install*-nya). Contoh implementasi program menggunakan GUI dapat dilihat referensi kodenya pada tautan ini. Perlu diingat bahwa kalian diberi kebebasan dalam implementasi dan tidak harus mengikuti alternatif pada referensi. Untuk mempermudah proses pengembangan dan kolaborasi, **sangat disarankan** menggunakan **MVC (*Model*, *View*, *Controller*)**.
2. **Dash**

   Kombinasi tombol tertentu dapat mengaktifkan *dash*, sehingga *chef* akan bergerak maju lebih dari 1 kotak (bisa 2 atau 3 atau 4, silakan eksplor konfigurasi yang paling *balance*). *Dash* harus memiliki **cooldown**.
3. **Lempar**

   Chef dapat melempar *ingredients* yang belum dimasak. Lempar berarti sama saja seperti menaruh *item* pada jarak lebih dari 1 kotak (bisa 2 atau 3 atau 4, silakan eksplor konfigurasi yang paling *balance*). Lemparan harus divalidasi terlebih dahulu, apabila lemparan melewati tembok, maka item akan jatuh tepat di sebelum tembok tersebut. Lemparan juga harus dapat ditangkap oleh *chef* lainnya.

4. **Random Level Generator**

   Buat *generator map* dengan tetap memperhatikan *dishes* yang akan dibuat dan *station* yang diperlukan. Jangan sampai ada *station* yang tidak dapat digunakan. Saran algoritma yang dapat digunakan adalah sebagai berikut MST (*Minimum Spanning Tree*) *Based Path Connectivity*, *Cellular Automata* or *Random Walk + Validation*, atau *Room Graph + Connectivity Validation*. **BONUS INI HARUS MENGIMPLEMENTASIKAN ALGORITMA DI ATAS. BUKAN HANYA MEMBUAT BANYAK TEMPLATE LALU DI-*RANDOM***

5. **Design Pattern**

   Terapkan 5 *design pattern* pada OOP dan beri penjelasan.

6. **Kreativitas**

   Kalian bebas menambahkan fitur apapun di luar spesifikasi. Semakin kompleks implementasi dari fitur terkait, semakin besar apresiasi yang akan diberikan oleh asisten.

Kelompok

Kelompok tugas besar akan diacak dengan masing-masing kelompok beranggotakan 4 - 5 orang. Pembagian kelompok dan tipe level *map* dapat dilihat pada sheets ini 🛠️ IF2010 Pembagian Kelompok & Asisten

QnA

Jika ada spesifikasi yang belum jelas, dapat ditanyakan melalui form QnA berikut ini

🛠️ QnA Tugas Besar IF2010 Pemrograman Berorientasi Objek 2025/2026

Asistensi

Asistensi dilakukan dengan ketentuan sebagai berikut.

1. Setiap kelompok akan mendapatkan satu orang Asisten Pembimbing.
2. Asistensi wajib dilakukan sebanyak **minimal dua kali**, satu kali sebelum *milestone* 1 dan yang kedua sebelum *milestone* 2. Berikut adalah ketentuan asistensi wajib.
    a. Asistensi 1 paling lambat tanggal **21 November 2025** pukul **16.00**.
    b. Asistensi 2 paling lambat tanggal **9 Desember 2025** pukul **16.00**.
3. Waktu dan tempat asistensi ditentukan berdasarkan perjanjian asisten dan kelompok.
4. Asistensi yang dilaksanakan secara daring harus dilakukan secara **on-cam**.
5. Perjanjian waktu asistensi harus dilakukan paling lambat **H-3** waktu asistensi.
6. Asistensi **bukan sesi untuk membahas ulang spesifikasi**, tetapi untuk klarifikasi atau bertanya bagian yang tidak dimengerti. Asisten **tidak akan membahas kembali spesifikasi saat asistensi**.
7. Paling lambat **1 jam** setelah asistensi selesai, masing-masing anggota kelompok yang hadir pada asistensi wajib mengisi form asistensi secara daring melalui Google Form.
    a. Form catatan asistensi: di sini
    b. Sertakan juga bukti kehadiran asistensi berupa screenshot atau foto dengan asisten.
    c. Tidak diperkenankan untuk meng-copy paste catatan kemajuan milik orang lain.

Pengumpulan

Milestone 1 - 21/11/2025

Pada Milestone 1, Anda perlu mengumpulkan link repository GitHub dan 2 dokumen, yaitu pembagian tugas setiap anggota kelompok dan struktur dari objek-objek yang terdapat pada game (disarankan menggunakan *class diagram*). **Harap mengundang asisten** yang bersangkutan **pada *repository* GitHub**. Akun GitHub asisten akan dituliskan pada sheets pembagian kelompok. Tidak ada aturan penamaan khusus *repository* GitHub. Format dari kedua dokumen adalah PDF dan wajib dikumpulkan pada form ini.

Milestone 2 - 13/12/2025

Pada Milestone 2, anda perlu mengumpulkan Source Code, Buklet, dan Log Activity. Pengumpulan akan dilakukan dengan membuat **release** pada Repository Github kelompok masing-masing. Release harus sudah mencakup Source Code, Buklet dan Log Activity. Waktu Release tidak boleh melewati deadline pengumpulan. Berikut adalah spesifikasi source code dan buklet. Di saat pengumpulan, pastikan repository Github tugas besar sudah di set menjadi **publik**. Pengumpulan dilakukan pada link form ini.
Berikut adalah spesifikasi source code, buklet, dan log activity.

Source Code

Teknik yang digunakan dalam pembuatan aplikasi permainan Nimonscooked dibebaskan, namun terdapat beberapa konsep OOP yang wajib diimplementasikan di aplikasi yang Anda buat, yaitu:

1. Inheritance
2. Abstract Class / Interface
3. Polymorphism
4. Generics
5. Exceptions
6. Collections
7. Concurrency

Program harus diimplementasikan menggunakan bahasa Java. Tuliskan tahapan untuk melakukan kompilasi dan menjalankan program pada file README.md. Kumpulkan file ZIP *source code* beserta README.md pada **olympia**.

Buklet

Pada Tugas Besar kali ini, format laporan diubah menjadi bentuk buklet. Buklet yang dibuat memiliki ketentuan sebagai berikut.

1. Berukuran A5
2. Tidak lebih dari 15 halaman
3. Format PDF
4. Berisi setidaknya:
    a. Halaman cover, yang berisi nama dan nomor kelompok, serta nama dan NIM anggota kelompok.
    b. User manual dan deskripsi jalannya permainan (gameplay).
    c. Nama, peran, dan pembagian tugas anggota kelompok.
    d. Struktur final dari objek-objek yang terdapat pada game (disarankan menggunakan class diagram)
    e. Design pattern yang digunakan
    f. Cerita tentang proses pengembangan
        i. Pada bagian ini kalian menceritakan proses pengembangan Tugas Besar ini dari awal sampai akhir. Kalian boleh mencantumkan screenshot rapat, hasil ideasi, corat-coret

Miro/Jamboard, atau dokumentasi apapun yang mendukung jalan cerita kalian.

ii. Cantumkan juga rancangan awal dan apa saja perubahan yang dilakukan (apabila implementasi beda dari rancangan) dan kenapa perlu dilakukan perubahan tersebut.

5. Dibuat sekreatif dan semenarik mungkin. Desain, tipografi, pewarnaan, dan tata letak dibebaskan kepada kalian.
6. Bahasa yang digunakan tidak harus baku, tetapi tetap serius dan terkesan friendly.

Log Activity

Log Activity harus memuat progress masing - masing anggota dan disatukan menjadi satu dokumen. Tabel di dalam dokumen **minimal** harus memiliki format sebagai berikut.

Kebin / 18224000

Waktu	**Kegiatan**	**Hasil**
30 November 2025	Kegiatan 1	Fitur A lancar tanpa bug
3 Desember 2025	Kegiatan 2	Class B masih ada bug X

Demo

1. Mahasiswa harus mengisi slot asisten penguji dan jadwal yang disepakati sesuai asisten pada link ini.
2. Demo akan dilakukan baik secara luring ataupun daring dengan menggunakan Google Meet, sesuai kesepakatan dengan asisten penguji.
3. Demo yang dilaksanakan secara daring harus dilakukan secara **on-cam**.
4. Mahasiswa menghubungi asisten demo untuk melakukan perjanjian jadwal demo.
5. Mahasiswa mempersiapkan program dan menjalankan eksekusi program sesuai dengan instruksi dari asisten penguji.
6. Penilaian demo dilakukan dengan membandingkan hasil eksekusi program dengan hasil yang diharapkan.
7. Demo dapat dilakukan pada waktu yang akan ditentukan kemudian dengan pembagian asisten yang berbeda.

Extras

Bang ini mah **bukan** Nimonscooked tapi **Wallahi I'm Cooked** bang!

~Jendra

~Maulvi

Ini semua salahnya Kak Jendra yaaaaaa. Aku ga ikut ikut (kaya kucing maulvi)

~Nuel

Buruan Jendra laperr

~Roihan

Jangan lupa dimatiin kompornya awas kebakaran.

~Iki