

**LAPORAN TUGAS BESAR
IF1210 DASAR PEMROGRAMAN**



Kelas: K01 / Kelompok: K

Anggota Kelompok:

Haidar Rafli Octavio Ramadhan	16523031
Galih Muhammad Syah Athaya	16523101
Kinan Athaya Barlian	16523171
Michael Dimas Sarono	19623131
Persada Ramiiza Abyudaya	19623241
Abdullah Al Ash Shidiq Kartabratna	16523241

**SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG
2024**

Pernyataan Kelompok

“Saya menyatakan bahwa saya mengerjakan tugas besar ini dengan sejujur-jujurnya, tanpa menggunakan cara yang tidak dibenarkan. Apabila di kemudian hari diketahui saya mengerjakan tugas besar ini dengan cara yang tidak jujur, saya bersedia mendapatkan konsekuensinya, yaitu mendapatkan nilai E pada mata kuliah IF1210 Dasar Pemrograman Semester 2 2023/2024.””

Yang mengeluarkan pernyataan,

Haidar Rafli Octavio Ramadhan (16523031)

Galih Muhammad Syah Athaya (16523101)

Kinan Athaya (16523171)

Michael Dimas Sarono (19623131)

Persada Ramiiza Abyudaya (19623241)

Abdullah Al Ash Shidiq Kartabrata (16523241)

Daftar Isi

Pernyataan Kelompok.....	2
Daftar Isi.....	3
Daftar Tabel.....	4
Daftar Gambar.....	5
Deskripsi Persoalan.....	6
Daftar Pembagian Kerja Kelompok.....	8
Checklist Hasil Penggerjaan Tugas Besar.....	9
Desain Command.....	11
Desain Kamus Data.....	25
Desain Dekomposisi Algoritmik dan Fungsional Program.....	41
Spesifikasi dalam Notasi Algoritmik.....	45
Lampiran Hasil Pengujian Program.....	92
Lampiran Form Asistensi.....	98

Daftar Tabel

Tabel Pembagian Kerja Kelompok.....	8
Tabel Checklist Hasil Penggerjaan Tugas Besar.....	9

Daftar Gambar

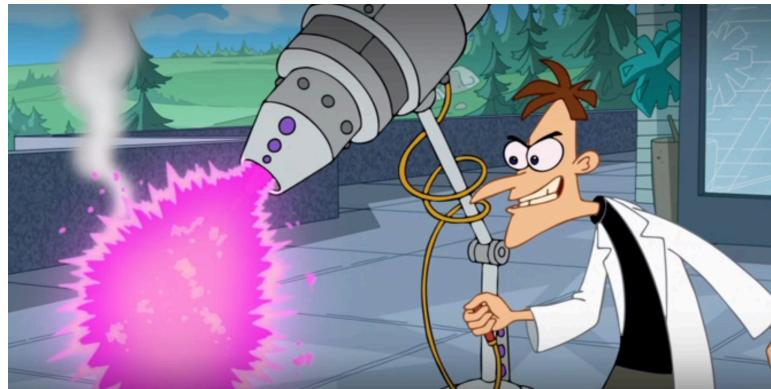
Gambar Pengujian F01 - Register.....	93
Gambar Pengujian F02 - Login.....	93
Gambar Pengujian F03 - Logout.....	93
Gambar Pengujian F04 - Menu & Help.....	93
Gambar Pengujian F07 - Inventory.....	94
Gambar Pengujian F08 - Battle.....	95
Gambar Pengujian F09 - Arena.....	95
Gambar Pengujian F10 - Shop & Currency.....	96
Gambar Pengujian F11 - Laboratory.....	97
Gambar Pengujian F12 - Shop Management.....	97
Gambar Pengujian F13 - Monster Management.....	98
Gambar Pengujian F14 - Load.....	98
Gambar Pengujian F15 - Save.....	98
Gambar Pengujian F16 - Exit.....	98
Gambar Form Asistensi 1.....	99
Gambar Form Asistensi 2.....	100

Deskripsi Persoalan



"Gambar Deskripsi Persoalan"

Di pinggiran kota Danville, Purry si Platypus, yang juga dikenal sebagai Agent P, sedang menghadapi tantangan yang besar. Dr. Asep Spakbor, seorang ilmuwan jahat, telah menciptakan monster-monster mengerikan yang mengancam keamanan kota Danville. Tugas Purry adalah untuk menghentikan ancaman ini, tetapi monster terbaru Dr. Asep Spakbor terlalu kuat bahkan bagi Purry.



Dr. Asep Spakbor sedang menciptakan monster

Purry menyadari bahwa dia membutuhkan bantuan dari agen-agen lainnya untuk mengalahkan monster itu. Dia bergegas ke markas rahasia O.W.C.A. (Organisasi Warga Cool Abiez) di mana dia bertemu dengan agen-agen lainnya, yaitu kalian.



O.W.C.A

Purry mengatakan bahwa untuk mengalahkan Dr. Asep Spakbor, kalian harus bekerja sama untuk mengalahkan monster-monster kuat. Kalian harus merencanakan strategi yang matang agar dapat menyelesaikan misi ini dengan baik. Setelah mendengarkan perkataan Purry, kalian bersiap-siap untuk mengeksekusi rencana dengan teliti.

Purry pun meminta bantuan kalian untuk mencari dan melatih monster-monster sendiri untuk digunakan dalam pertempuran melawan Dr. Asep Spakbor. Kalian merasa tertantang oleh misi ini, namun kalian juga yakin bahwa dengan kerja tim dan tekad yang kuat, pasti bisa berhasil.

Kalian memutuskan untuk memulai misi pencarian monster di hutan terpencil yang diyakini menjadi tempat tinggal bagi banyak jenis monster. Petualangan kalian di hutan yang gelap dan misterius ini akan menguji keberanian dan ketangkasan kalian. Kalian harus siap menghadapi segala tantangan yang mungkin muncul di perjalanan ini demi keselamatan kota Danville.

Daftar Pembagian Kerja Kelompok

Tabel Pembagian Kerja Kelompok

Fitur	Implementasi	NIM Desainer	NIM Coder	NIM Tester
F00 - Random Number Generator	function random	16523101	16523101	16523101
F01 - Register	function register_account	16523241 19623241	16523241 19623241	19623241
F02 - Login	function wrong_password, exist_username	16523241	16523241	16523101 19623241
F03 - Logout	procedure logout	16523241	16523241	16523241
F04 - Menu & Help	function menu	16523241	16523241	16523241 19623241
F05 - Monster	function create_monster	16523101 19623131	16523101 19623131	16523101 19623131
F06 - Potion	procedure take_potion	16523101 19623131	16523101 19623131	16523101 19623131
F07 - Inventory	function inventory	16523101 19623131	16523101 19623131	16523101 19623131
F08 - Battle	function battle	16523101 19623131	16523101 19623131	16523101 19623131
F09 - Arena	function arena	16523101 19623131	16523101 19623131	16523101 19623131
F10 - Shop & Currency	function display_shop	16523171 16523101	16523171 16523101	16523101

F11 - Laboratory	function laboratory	16523171 19623241	16523171 19623241	19623241
F12 - Shop Management	function shop_manage ment	19623241	19623241	19623241
F13 - Monster Management	function monster_ma na gement	19623241	19623241	19623241
F14 - Load	function load	16523031	16523031	16523031
F15 - Save	procedure save	16523031	16523031	16523031
F16 - Exit	procedure exit	16523031	16523031	16523031

Checklist Hasil Pengerjaan Tugas Besar
Tabel Checklist Hasil Pengerjaan Tugas Besar

Fungsi	Design	Implementasi	Testing
F00 - Random Number Generator	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
F01 - Register	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
F02 - Login	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
F03 - Logout	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
F04 - Menu & Help	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
F05 - Monster	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
F06 - Potion	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
F07 - Inventory	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
F08 - Battle	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
F09 - Arena	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
F10 - Shop & Currency	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
F11 - Laboratory	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
F12 - Shop Management	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
F13 - Monster Management	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
F14 - Load	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
F15 - Save	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
F16 - Exit	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

Desain Command

1. F00 - Random Number Generator

Masukan:

- > limit bawah
- > limit atas

Keluaran:

angka integer random di antara kedua batasan

2. F01 - Register

Masukan:

- > Masukkan username
- > Masukkan password
- > Monster pilihan

Keluaran:

bila user memasukan masukan username dengan ketentuan karakter yang diperbolehkan hanya (alfabet huruf besar atau kecil, angka (0-9), underscore(_), stip (-), dan tidak ada username yang sama dengan sebelumnya, akan minta memilih monster pendamping pertama dan dicetak "**Selamat datang Agent username. Bersama monster pilihan, ayo kita lawan musuh musuh itu!**"

bila user memasukkan masukan username yang tidak sesuai dengan ketentuan karakter atau sudah ada dalam list username akan diminta mengisi ulang masukan

3. F02 - Login

Masukan:

- > Masukkan username
- > Masukkan password

Keluaran:

bila user memasukkan username yang tidak terdaftar dalam list akan di cetak "**Username tidak terdaftar!**"

bila user memasukkan username yang terdaftar namun password yang salah akan dicetak "**Password salah!**"

bila username dan password benar dan role user sebagai agent akan dicetak
“Selamat datang, Agent username”

bila username dan password benar dan role user sebagai admin akan dicetak
“Selamat datang, yang mulia admin username”

4. F03 - Logout

Masukkan:
> panggilan command

Keluaran:
bila belum login akan dicetak
“Logout gagal!
Anda belum login”

bila telah login akan dicetak
“Logout Berhasil.”

5. F04 - Menu & Help

Masukkan:
> panggilan command

Keluaran:
bila sebelum login, akan dicetak
“===== HELP =====
Login belum dilakukan,silahkan login terlebih dahulu.
1. Login: Masuk ke dalam akun yang sudah terdaftar
2. Register: Membuat akun baru”

setelah login sebagai agen akan dicetak
“===== HELP =====
Halo Agent username. Kamu memanggil command HELP. Kamu memilih jalan yang benar, semoga kamu tidak sesat kemudian. Berikut adalah hal-hal yang dapat kamu lakukan sekarang:
1. Logout: Keluar dari akun yang sedang digunakan

- 2. Inventory:** Melihat owca-dex yang dimiliki oleh Agent
- 3. Shop:** Membeli potion atau monster
- 4. Battle:** Bertarung melawan monster!
- 5. Arena:** Latih dan tingkatkan kemampuan agen dan para monstermu!
- 6. Laboratory:** Upgrade monster yang kamu miliki
- 7. Save:** Menyimpan progress Anda
- 8. Exit:** Keluar dari game”

bila login sebagai admin akan dicetak

“===== HELP =====

Selamat datang, Admin. Berikut adalah hal- hal yang dapat kamu lakukan:

- 1. Logout:** Keluar dari akun yang sedang digunakan
- 2. Shop Management:** Melakukan manajemen pada SHOP sebagai tempat jual beli peralatan Agent
- 3. Monster Management:** Melakukan manajemen Monster
- 4. Save:** Menyimpan progress Anda
- 5. Exit:** Keluar dari game”

6. F05 - Monster

Masukkan:

> dictionary yang dibaca dari csv

Keluaran:

dictionary baru yang mengorganisir data yang dibaca dari csv dan menyesuaikan ATK Power, DEF Power, dan HP dengan level mereka

7. F06 - Potion

potion (str): Nama potion yang digunakan oleh monster, bisa berupa salah satu dari berikut:

- **'Strength Potion'**
- **'Resilience Potion'**
- **'Healing Potion'**

Modifikasi pada dictionary monster:

- **Jika potion adalah 'Strength Potion', nilai monster['ATK Power'] akan meningkat sebesar 5%**

- **Jika potion adalah 'Resilience Potion', nilai monster['DEF Power'] akan meningkat sebesar 5%**
- **Jika potion adalah 'Healing Potion', nilai monster['HP'] akan bertambah sebesar 25% dari monster['MaxHP'], namun tidak bisa melebihi monster['MaxHP']**

8. F07 - Inventory

Masukkan:

- > user (dict): Dictionary yang berisi informasi pengguna.
- > list_item (dict): Dictionary yang berisi daftar item yang dimiliki oleh pengguna.
- > list_mons (dict): Dictionary yang berisi daftar monster yang dimiliki oleh pengguna.
- > id (str): ID pengguna yang akan diambil informasi inventarisnya.

Keluaran:

suatu array yang memiliki informasi 'oc' pada index 0, monster yang dimiliki oleh user dengan 'ID' yang dimasukkan dan spesifikasinya pada index 1, jenis dan kuantitas potion pada index 2, dan 'ID' pada index 3

9. F08 - Battle

Masukan:

- > "1" untuk Menyerang lawan.
- > "2" untuk Menggunakan potion.
- > "3" untuk Melarikan diri.

Keluaran:

Bila perintah yang dipilih adalah "1":

- Monster Anda akan menyerang lawan.**
- Program akan menampilkan serangan yang dilakukan dan jumlah damage yang diberikan.**
- Program akan menampilkan status lawan setelah diserang.**

Bila perintah yang dipilih adalah "2":

- Program akan menampilkan daftar potion yang tersedia.**
- Anda dapat memilih potion untuk digunakan.**
- Jika potion berhasil digunakan, efeknya akan diterapkan pada monster Anda.**

4. **Jumlah potion yang tersedia akan berkurang.**
5. **Jika tidak ada potion yang tersedia atau sudah digunakan sebelumnya, program akan memberikan pesan yang sesuai.**

Bila perintah yang dipilih adalah "3":

1. **Anda akan mencoba melarikan diri dari pertarungan.**
2. **Ada kemungkinan berhasil atau gagal.**
 - **Jika berhasil, program akan memberikan pesan bahwa Anda berhasil melarikan diri.**
 - **Jika gagal, program akan memberikan pesan bahwa Anda gagal melarikan diri dan pertarungan akan berlanjut.**

Setelah setiap giliran, status monster akan ditampilkan, termasuk nama, ATK Power, DEF Power, HP, dan Level.

Setelah pertarungan selesai:

- **Jika monster Anda masih hidup dan berhasil mengalahkan lawan, program akan memberikan pesan bahwa Anda menang, menampilkan detail lawan yang dikalahkan, dan memberikan jumlah koin yang Anda dapatkan.**
- **Jika monster Anda kalah, program akan memberikan pesan bahwa Anda kalah, menampilkan detail monster Anda yang kalah, dan mengurangi jumlah koin yang Anda miliki.**

Program akan mengembalikan inventory yang telah diperbarui setelah pertarungan.

10. F09 - Arena

Masukan:

- > "1" untuk Menyerang lawan.
- > "2" untuk Menggunakan potion.
- > "3" untuk Melarikan diri.

Keluaran:

Bila perintah yang dipilih adalah "1":

1. **Monster Anda akan menyerang lawan.**
2. **Program akan menampilkan serangan yang dilakukan dan jumlah damage yang diberikan.**
3. **Program akan menampilkan status lawan setelah diserang.**

Bila perintah yang dipilih adalah "2":

1. **Program akan menampilkan daftar potion yang tersedia.**
2. **Anda dapat memilih potion untuk digunakan.**
3. **Jika potion berhasil digunakan, efeknya akan diterapkan pada monster Anda.**
4. **Jumlah potion yang tersedia akan berkurang.**
5. **Jika tidak ada potion yang tersedia atau sudah digunakan sebelumnya, program akan memberikan pesan yang sesuai.**

Bila perintah yang dipilih adalah "3":

1. **Anda akan mencoba melarikan diri dari pertarungan.**
2. **Ada kemungkinan berhasil atau gagal.**
3. **Jika berhasil, program akan memberikan pesan bahwa Anda berhasil melarikan diri.**
4. **Jika gagal, program akan memberikan pesan bahwa Anda gagal melarikan diri dan pertarungan akan berlanjut.**
5. **Setelah setiap giliran, status monster akan ditampilkan, termasuk nama, ATK Power, DEF Power, HP, dan Level.**

Setelah setiap stage selesai:

- **Jika monster Anda masih hidup dan berhasil mengalahkan lawan, program akan memberikan pesan bahwa Anda menang, menampilkan detail lawan yang dikalahkan, dan meningkatkan stage.**
- **Jika monster Anda kalah, program akan memberikan pesan bahwa Anda kalah, menampilkan detail monster Anda yang kalah, dan mengakhiri pertarungan.**

Setelah arena selesai:

1. **Program akan memberikan jumlah hadiah yang Anda dapatkan berdasarkan jumlah stage yang telah diselesaikan.**
2. **Program akan menampilkan statistik total damage yang diberikan dan diterima selama arena.**
3. **Program akan mengembalikan inventory yang telah diperbarui setelah arena.**

11. F10 - Shop & Currency

Masukan:

- > lihat
- > beli
- > jenis

> kuantitas

Keluaran:

bila dipilih opsi untuk melihat:

1. bila dipilih monster:

akan di display table monster lengkap dengan ATK Power, DEF Power, dan HP monsternya serta harga untuk membelinya

2. bila dipilih potion:

akan di display table potion dengan nama dan harga untuk membelinya

bila dipilih opsi untuk membeli:

1. bila dipilih monster:

akan diminta index dari monster yang ingin dibeli dan jumlah monster yang ingin dibeli. bila OC mencukupi, barang akan dimasukkan ke inventory

2. bila dipilih potion:

akan minta jenis potion yang ingin dibeli dan jumlah potion tersebut. bila OC mencukupi, barang akan dimasukkan ke inventory

bila dipilih opsi untuk keluar:

akan keluar dari menu laboratory

12. F11 - Laboratory

Masukkan:

> index monster

> konfirmasi

Keluaran:

Bila user memiliki monster akan diminta index monster yang ingin di level up dan konfirmasi ingin di level up. Ketika berhasil, program akan mencetak:

> "**Selamat, {name} berhasil di-upgrade ke level {level + 1}!"**

Jika user ingin level up namun tidak memiliki coin yang cukup, akan dicetak:

> "**OC tidak cukup untuk melakukan upgrade.**"

Bila user tidak memiliki monster, akan langsung keluar dari menu laboratory dan diminta untuk membeli monster terlebih dahulu

13. F12 - Shop Management

Masukan:

> Masukkan aksi (lihat, tambah, hapus, keluar)

> Masukkan id monster atau potion

- > Masukkan harga monster atau potion
- > Masukkan stok monster atau potion

Keluaran:

Bila aksi yang dimasukkan adalah "lihat":

- **Jika pengguna memilih untuk melihat monster, fungsi akan mencetak informasi monster yang tersedia di toko.**
- **Jika pengguna memilih untuk melihat potion, fungsi akan mencetak informasi potion yang tersedia di toko.**

Bila aksi yang dimasukkan adalah "tambah":

- **Jika pengguna memilih untuk menambah monster, fungsi akan meminta pengguna untuk memasukkan ID, harga, dan stok untuk monster yang baru ditambahkan ke toko.**
- **Jika pengguna memilih untuk menambah potion, fungsi akan meminta pengguna untuk memasukkan ID, harga, dan stok untuk potion yang baru ditambahkan ke toko.**

Bila aksi yang dimasukkan adalah "hapus":

- **Jika pengguna memilih untuk menghapus monster, fungsi akan meminta pengguna untuk memasukkan ID monster yang ingin dihapus.**
- **Jika pengguna memilih untuk menghapus potion, fungsi akan meminta pengguna untuk memasukkan ID potion yang ingin dihapus.**

Bila aksi yang dimasukkan adalah "keluar":

Program akan keluar dari Shop Management.

14. F13 - Monster Management

Masukan:

- > "1" untuk menampilkan semua monster
- > "2" untuk menambahkan monster baru

Keluaran:

Bila aksi yang dipilih adalah "1":

Program akan menampilkan tabel yang berisi informasi tentang semua monster yang ada.

Bila aksi yang dipilih adalah "2":

Program akan meminta Anda untuk memasukkan informasi tentang monster baru yang akan ditambahkan ke database.

- 1. Masukkan nama/type monster.**
- 2. Masukkan attack power monster.**
- 3. Masukkan defend power monster (dalam rentang 0-50).**
- 4. Masukkan HP monster.**

Setelah memasukkan informasi, program akan menampilkan detail monster yang akan ditambahkan.

Program akan meminta konfirmasi untuk menambahkan monster ke database.

Jika Anda menyetujui (Y), monster akan ditambahkan ke database dan tabel Monster akan ditampilkan kembali.

Jika Anda menolak (N), monster tidak akan ditambahkan ke database.

15. F14 - Load
- animate():
- **Menampilkan animasi loading di terminal.**
- load():
- **Memvalidasi apakah folder yang diberikan pengguna ada, dan membuka file CSV dalam folder tersebut jika ada.**
 - **Menggunakan argparse untuk menerima nama folder dari argumen command line.**
 - **Menjalankan animasi loading.**
 - **Memeriksa apakah nama folder diberikan dan ada di direktori yang diharapkan.**
 - **Membaca file CSV dalam folder dan mengubahnya menjadi dictionary menggunakan fungsi dictionary_csv.**
 - **Jika folder tidak ditemukan, memberikan pesan kesalahan.**
- fungsisplit(bariscsv):
- **Membagi baris CSV menjadi list berdasarkan karakter titik koma.**
- dictionary_csv(file_csv, nama_folder):
- **Mengubah file CSV menjadi dictionary.**
- sambutan():
- **Menampilkan pesan sambutan di terminal.**

```
~$ python F14.py nama_folder

# parent folder dari nama_folder akan sama seperti dalam fungsi save

Loading \

Selesai!

----- ----- -
/ _ \ \    / / ___| / \
| | | \ \ / \ / / | / _ \
| | -| | \ V  V /| |___ / ___ \
\___/ \_/\_/\ \___/ \_/
selamat datang di OWCA!!

# meminta perintah berikutnya..
```

```
# user tidak memberikan nama folder

~$ python F14.py

Loading \

Selesai!

Tidak ada nama folder yang diberikan!

# program keluar
```

```
# user memberikan folder yang tidak ada  
  
~$ python main.py tes  
  
loading \  
  
Selesai!  
  
folder tes tidak ditemukan  
  
# program keluar
```

16. F15 - Save

save(user, monster, monster_shop, monster_inventory, item_shop, item_inventory):

- **Meminta pengguna memasukkan nama folder**
- **Memeriksa apakah folder ada, jika tidak, membuat folder baru**
- **Menyimpan data dalam bentuk CSV ke folder tersebut dengan memanggil fungsi makecsvfile untuk setiap dataset (user, monster, monster_shop, dll.)**
- **Menampilkan animasi penyimpanan**

dicttocsv(dict):

- **Mengubah dictionary menjadi list yang disusun menyerupai struktur data CSV**
- **Menyusun list sesuai dengan format CSV (memisahkan nilai dengan titik koma dan menambahkan baris baru)**

makecsvfile(nama_folder, dictionary, nama_file):

- **Membuat file CSV di folder yang diberikan**
- **Menulis data dari dictionary ke file CSV sesuai dengan format yang dihasilkan oleh dicttocsv**

animate_saving(a):

- **Menampilkan animasi "saving" dengan titik-titik yang bergerak di terminal untuk menunjukkan proses penyimpanan**

```
# nama folder tidak ditemukan, folder belum dibuat  
>>> SAVE  
Masukkan nama folder: tes  
# Folder tes belum ada  
Membuat folder tes  
Saving...  
Berhasil menyimpan data di folder tes
```

```
# nama folder ditemukan  
>>> SAVE  
Masukkan nama folder:tes  
#folder sudah ada  
Saving.....  
Berhasil menyimpan data di folder tes  
# apabila program melakukan overwrite/replace, tidak diperlukan pesan tambahan
```

```
>>> SAVE

Masukkan nama folder: tes

# Folder ./save belum ada

Masukkan nama folder:tes

Saving.....

Berhasil menyimpan data di folder tes
```

17. F16 - Exit

Masukan:

- > Pengguna dapat memasukkan 'y' atau 'Y' jika ingin menyimpan perubahan.
- > Pengguna dapat memasukkan 'n' atau 'N' jika tidak ingin menyimpan perubahan.
- > Jika pengguna memasukkan input lain, program akan mengulangi pertanyaan hingga mendapatkan input yang valid ('y', 'Y', 'n', atau 'N').

Keluaran:

Jika Pengguna Memilih 'y' atau 'Y':

1. **Program akan menjalankan fungsi save untuk menyimpan data yang sudah diubah.**
2. **Program akan menampilkan pesan: "Anda telah keluar dari permainan."**

Jika Pengguna Memilih 'n' atau 'N':

1. **Program akan menampilkan pesan: "Data tidak disimpan."**
2. **Program akan menampilkan pesan: "Anda telah keluar dari permainan."**

Jika Pengguna Memasukkan Input Tidak Valid:

- > **Program akan mengulangi pertanyaan: "Apakah Anda mau melakukan penyimpanan file yang sudah diubah? (y/n)"**

```
>>> EXIT
```

```
Apakah Anda mau melakukan penyimpanan file yang sudah diubah? (y/n) n
```

```
Anda telah keluar dari permainan
```

```
# Keluar program
```

```
>>> EXIT
```

```
# jika input tidak valid program akan terus meminta masukkan dari pengguna
```

```
Apakah Anda mau melakukan penyimpanan file yang sudah diubah? (y/n) a
```

```
Apakah Anda mau melakukan penyimpanan file yang sudah diubah? (y/n) Y
```

```
# Menjalankan prosedur save (F15) dan keluar program
```

Desain Kamus Data

1. F00 - Random Number Generator

KAMUS LOKAL

```
procedure linear_congruential_generator(m: int, a: int, c: int,
seed: int) => Iterator array of int
    x : integer
function random(lower_limit : real, upper_limit : real) -> real
    rand : real
    seed_random : integer
    gen : int
```

2. F01 - Register

KAMUS LOKAL

```
function valid_username(username: str) -> bool
    valid_characters : str
    char : str
function exist_username(user: dict, username: str) -> bool
    i : integer
function register_account():
    username, password : str
    index, choice, chosen monster: int
    monsters : array of any

type user : < id:string,
        username : string,
        password : string,
        role : string
        oc : string >
```

3. F02 - Login

KAMUS LOKAL

```
function wrong_password(user: dict, username: str, password: str)
-> bool
    i, IDUser: integer
function exist_username(user: dict, username: str) -> bool
    i : integer

function login(user: dict) -> int
    username, password : str
    i, IDUser: integer
```

```

type user : < id:string,
        username : string,
        password : string,
        role : string
        oc : string >

```

4. F03 - Logout

KAMUS LOKAL

```
procedure Logout(masuk: bool)
```

5. F04 - Menu & Help

KAMUS LOKAL

```
procedure help_before_login()
```

```
procedure help_admin()
```

```
procedure help_agent(username: str)
```

6. F05 - Monster

KAMUS LOKAL

```
function ListToDict_monster (monster_list: dictionary of list) ->
dictionary of dictionary
```

```
monster_dict = dictionary of string {Name, ATK Power, DEF Power,
HP, MaxHP, Level}
```

```
dictOfDict = dictionary of dictionary {id:{monster_dict}}
```

```
function create_monster (monster: dictionary, lvl: integer) ->
dictionary
```

```
    name = string
```

```
    attack, defense, hp, level = integer
```

```
    wild monster = monster_dict
```

```
type monster_list < id:stringe,
```

```
        type : string,
```

```
        atk_power : string
```

```
        def_power : string
```

```
        hp : string >
```

```
type monster_dict : < type : string,
```

```

    atk_power : string

    def_power : string

    hp : string >

type dictOfDict < 'id': monster_dict >

```

7. F06 - Potion

KAMUS LOKAL

```

procedure take_potion (monster: dictionary, potion: string)

type monster_list < id:string,
        type : string,
        atk_power : string

        def_power : string

        hp : string >

```

8. F07 - Inventory

KAMUS LOKAL

```

function organize_user_info(user_info: dictionary of array) ->
dictionary of dictionary
    user_data = dictionary

function ListToList_monster_inventory(monster_inventory:
dictionary of array of any) -> dictionary of dictionary
    user_monsters = dictionary

function ListToList(item_inventory: dictionary of array of any) ->
dictionary of dictionary
    user_items = dictionary

function inventory (user: dictionary, list_item: dictionary,
list_mons: dictionary, id: integer) -> array of dictionary
    list_of_inv = array of any
    i, j = integer

procedure show_detail_inventory (inventory: array of any)
    list_everything = array of any
    i, j = integer

procedure inventory_to_info(inv_player: dictionary)

```

```

i, j = integer

type item_inventory_csv < user_id : char,
                                type : string,
                                quantity : char >

type user_info : < id:string,
                                username : string,
                                password : string,
                                role : string
                                oc : string >

type monster_inventory < id:string,
                                type : string,
                                atk_power : string
                                def_power : string
                                hp : string >

type user_data < 'id' : user_info>

type user_item < 'id': item_inventory>

type user_monster < 'id': monster_inventory>

type inventory < 'oc', user_monster, user_item, 'id' >

```

9. F08 - Battle

KAMUS LOKAL

```

procedure take_damage (monster: dictionary, damage: integer)

procedure attack_opponent (attacker: dictionary ,opponent:
                                dictionary)
                                damage = integer
function is_alive (monster: dictionary) -> boolean

function battle (inventory: array of any, monster_dict: dictionary
                                of

```

```

        dictionary) -> array of any
            wild_monster, monster = monster
            used_potion = array of boolean
            run_away = boolean
            turn, i, chosen_monster, success = integer
            action = string

type monster < id:stringe,
                    type : string,
                    atk_power : string
                    def_power : string
                    hp : string >

```

10. F09 - Arena

KAMUS LOKAL

```

procedure take_damage (monster: dictionary, damage: integer)

procedure attack_opponent (attacker: dictionary ,opponent:
dictionary)
damage = integer

function is_alive (monster: dictionary) -> boolean

function arena (inventory: array of any, monster_dict: dictionary
of
        dictionary) -> array of any
            wild_monster, monster = dictionary
            used_potion = array of boolean
            run_away = boolean
            turn, stage, i, chosen_monster, success = integer
            action = string

type monster < id:stringe,
                    type : string,
                    atk_power : string
                    def_power : string
                    hp : string >

```

11. F10 - Shop & Currency

KAMUS LOKAL

```
function spend_currency(inventory :array, amount: integer) -> bool

procedure buy_item(item_type: str, item_id: str, inventory: array,
quantity: int)

procedure display_shop(inventory: array)
    action, item_type, item_id : str
    quantity: int

type inventory < 'oc', user_monster, user_item, 'id' >
```

12. F11 - Laboratory

KAMUS LOKAL

```
function get_value(dictionary: dict, key: str or int ,default: any
) -> dict or none

procedure display_monsters(monsters: array of monster)
    monster: dict

procedure display_upgrade_prices(upgrade_prices: dict)
    level, price: int

function choose_monster(monsters: array of monster)-> int
    choice: int

function upgrade_monster(monsters: array of monster, index: int,
upgrade_prices: dict, oc_balance: int)-> int
    monster: dict
    name, confirm: str
    level, upgrade_cost: int

function laboratory(inventory: list)-> int, array of any
    index, oc_balance: int

type inventory < 'oc', user_monster, user_item, 'id' >

type monster < id:stringe,
    type : string,
    atk_power : string
    def_power : string
    hp : string >
```

```
type upgrade_prices < id : price >
```

13. F12 - Shop Management

KAMUS LOKAL

```
function ubahmonster(monster: dict, monster_shop: dict) -> None
    monster: dict
    monster_shop: dict
    id: string
    index: int
    harga: string
    stok: string

function remove_by_index(dictionary: dict, key: string, index: int) -> None
    dictionary: dict
    key: string
    index: int

function fungsisplit(bariscsv: string) -> list of string
    bariscsv: string
    fields: list of string
    field: string
    karakter: string

function ubah_potion(potion: dict, potion_shop: dict) -> None
    potion: dict
    potion_shop: dict
    potion_id: int
    indeks_potion: int
    indeks_shop: int
    stok_baru: int
    harga_baru: int
function hapus_potion(potion: dict, potion_shop: dict) -> None
    potion: dict
    potion_shop: dict
    potion_id: int
    indeks_potion: int
    indeks_shop: int
    konfirmasi: string

function match_monsters(monster: dict, monster_shop: dict) -> None
    monster: dict
        id: list of string
        type: list of string
        atk_power: list of string
        def_power: list of string
        hp: list of string
```

```

monster_shop: dict
i: int
j: int

function hapus_item(lst: list, index: int) -> tuple
    lst: list
    index: int
    removed_item: any

function match_items(item_inventory: dict, item_shop: dict) ->
None
    item_inventory: dict
    item_shop: dict
    i: int
    j: int
    printed_types: list of string

function tambah_monster(monster: dict, monster_shop: dict) -> None
    monster: dict
    monster_shop: dict
    monster_id: string
    indeks_monster: int
    stok_awal: int
    harga: int
function tambah_potion(potion: dict, potion_shop: dict) -> None
    potion: dict
    potion_shop: dict
    potion_id: int
    indeks_potion: int
    stok_awal: int
    harga: int

function print_monster_tersedia(monster: dict, monster_shop: dict)
-> None
    monster: dict
    monster_shop: dict
    i: int

function print_potion_tersedia(potion: dict, item_shop: dict) ->
None
    potion: dict
    item_shop: dict
        type: list of string
    i: int

function hapus_monster(monster: dict, monster_shop: dict) -> None
    monster: dict
    monster_shop: dict

```

```

        id_monster: string
        monster_type: string
        confirmation: string
        index: int
        removed_item: any

function shop_management() -> None
    aksi: string
    opsi_lihat: string
    opsi_tambah: string
    opsi_ubah: string
    opsi_hapus: string

type item_shop_csv < type : string,
        stock : string,
        price : string >

type is : < artype : array[0..f-1] of type,
        arstock : array[0..f-1] of stock
        arprice : array[0..f-1] of price >

array_item_shop : array[0] of is
monster_inventory : SEQFILE OF
    (*)rekitem_shop : item_shop_csv
(1)<"", "", "">

```

14. F13 - Monster Management

```

function print_table
KAMUS LOKAL
    monster : m

function validate_alpha_input
KAMUS LOKAL
    prompt : string
    result : string

function validate_integer_input
KAMUS LOKAL

```

```

        prompt : string
        result : integer

function validate_integer_input_2
KAMUS LOKAL
    max_value : integer
    prompt : string
    result : integer

    ascii_monster : string
    aksi : string
    new_id : integer
    new_type : string
    new_atk_power : integer
    new_def_power : integer
    new_hp : integer
    confirmation : string

    type monster_csv : < id : integer,
                           type : string,
                           atk_power : integer,
                           def_power : integer,
                           hp : integer >

    type m : < arid : array[0..n-1] of integer,
                artype : array[0..n-1] of string,
                aratk_power : array[0..n-1] of integer,
                ardef_power : array[0..n-1] of integer,
                arhp : array[0..n-1] of integer >

    monster : m
    monster <- {
        'arid' : [],
        'artype' : [],
        'aratk_power' : [],
        'ardef_power' : [],
        'arhp' : []
    }

```

15. F14 - Load

KAMUS LOKAL

```

function load () -> array of any
a : integer

b : integer

c : integer

```

```

d : integer

e : integer

f : integer

type user_csv : < id:string,
           username : string,
           password : string,
           role : string
           oc : string >

type u : < arid : array[0..a-1] of id,
           arusername : array[0..a-1] of username
           arpASSWORD : array[0..a-1] of password
           arrole : array[0..a-1] of role
           aroc : array[0..a-1] of oc

user : SEQFILE OF
       (*)rekuser : user_csv
       (1)<"","","","","","","">

array_user : array[0] of u

type monster_csv < id:stringe,
           type : string,
           atk_power : string
           def_power : string
           hp : string >

type m : < arid : array[0..b-1] of id,
           artype : array[0..b-1] of type
           aratk : array[0..b-1] of atk_power
           ardef : array[0..b-1] of def_power

```

```

arhp : array[0..b-1] of hp

array_monster : array[0] of m

monster : SEQFILE OF

(*) rekmonster : monster_csv

(1)<"", "", "", "", "">

type monster_shop_csv < monsterid:stringe,
stock : string,
price : string >

type ms : < armonsterid : array[0..c-1] of monsterid,
arstock : array[0..c-1] of stock

arprice : array[0..c-1] of price >

array_monster_shop : array[0] of ms

monster_shop : SEQFILE OF

(*) rekmonster_shop : user_csv

(1)<"", "", "">

type monster_inventory_csv < user_id:char,
monster_id : char,
price : char >

type mi : < aruser_id : array[0..d-1] of user_id,
armonster_id : array[0..d-1] monster_id

arprice : array[0..d-1] of price >

array_monster_inventory : array[0] of mi

monster_inventory : SEQFILE OF

(*) rekmonster_inventory : monster_inventory_csv

(1)<"", "", "">

```

```

type item_inventory_csv < user_id : char,
                                type : string,
                                quantity : char >

type ii : < aruser_rid : array[0..e-1] of user_id,
                                artype : array[0..e-1] of type
                                arquantity : array[0..e-1] of quantity >

array_item_inventory : array[0] of ii
item_inventory : SEQFILE OF
(*) rekitem_inventory : item_inventory_csv
(1)<"", "", "">

type item_shop_csv < type : stringe,
                                stock : string,
                                price : string >

type is : < artype : array[0..f-1] of type,
                                arstock : array[0..f-1] of stock
                                arprice : array[0..f-1] of price >

array_item_shop : array[0] of is
monster_inventory : SEQFILE OF
(*) rekitem_shop : item_shop_csv
(1)<"", "", "">

```

16. F15 - Save

KAMUS LOKAL

```
procedure save(array_user : array of u, array_monster : array of
m, array_monster_inventory : array of mi, array_monster_shop :
array of ms, array_item_inventory : array of ii, array_item_shop
: array of is)

a : integer
b : integer
c : integer
d : integer
e : integer
f : integer

type user_csv : < id:string,
        username : string,
        password : string,
        role : string
        oc : string >

type u : < arid : array[0..a-1] of id,
        arusername : array[0..a-1] of username
        arpssword : array[0..a-1] of password
        arrole : array[0..a-1] of role
        aroc : array[0..a-1] of oc

user : SEQFILE OF
        (*) rekuser : user_csv
        (1)<"", "", "", "", "">

array_user : array[0] of u

type monster_csv < id:stringe,
        type : string,
```

```

    atk_power : string

    def_power : string

    hp : string >

type m : < arid : array[0..b-1] of id,
        artype : array[0..b-1] of type
        aratk : array[0..b-1] of atk_power
        ardef : array[0..b-1] of def_power
        arhp : array[0..b-1] of hp

array_monster : array[0] of m

monster : SEQFILE OF

(*) rekmonster : monster_csv

(1)<"", "", "", "", "">

type monster_shop_csv < monsterid:stringe,
        stock : string,
        price : string >

type ms : < armonsterid : array[0..c-1] of monsterid,
        arstock : array[0..c-1] of stock
        arprice : array[0..c-1] of price >

array_monster_shop : array[0] of ms

monster_shop : SEQFILE OF

(*) rekmonster_shop : user_csv

(1)<"", "", "">

type monster_inventory_csv < user_id:char,
        monster_id : char,
        price : char >

```

```

type mi : < aruser_id : array[0..d-1] of user_id,
          armonster_id : array[0..d-1] monster_id
          arprice : array[0..d-1] of price >

array_monster_inventory : array[0] of mi

monster_inventory : SEQFILE OF

(*) rekmonster_inventory : monster_inventory_csv

(1)<"", "", "">

type item_inventory_csv < user_id : char,
          type : string,
          quantity : char >

type ii : < aruser_rid : array[0..e-1] of user_id,
          artype : array[0..e-1] of type
          arquantity : array[0..e-1] of quantity >

array_item_inventory : array[0] of ii

item_inventory : SEQFILE OF

(*) rekitem_inventory : item_inventory_csv

(1)<"", "", "">

type item_shop_csv < type : stringe,
          stock : string,
          price : string >

type is : < artype : array[0..f-1] of type,
          arstock : array[0..f-1] of stock
          arprice : array[0..f-1] of price >

array_item_shop : array[0] of is

```

```
monster_inventory : SEQFILE OF  
(*) rekitem_shop : item_shop_csv  
(1)<"""",""","">
```

17. F16 - Exit

```
procedure exit(array_user:array of any, array_monster:array of  
any, array_monster)  
{menerima input folder dari pengguna dan mengakses file csv dan  
mengubah ke dalam bentuk array}
```

KAMUS LOKAL

```
a : char
```

Desain Dekomposisi Algoritmik dan Fungsional Program

1. F00 - Random Number Generator

```
procedure linear_congruential_generator()
{Membuat generator kongruensial linear untuk menghasilkan angka
pseudo-random
I.S. Parameter m, a, c, dan seed terdefinisi
F.S. Menghasilkan deret angka pseudo-random}

procedure random(lower_limit, upper_limit)
{Menghasilkan angka floating-point acak dalam rentang tertentu
menggunakan LCG
I.S. Batas bawah dan batas atas (lower_limit dan upper_limit)
terdefinisi
F.S. Menghasilkan angka acak dalam rentang [lower_limit,
upper_limit)}
```

2. F01 - Register

```
procedure register ()
{Mendaftarkan pengguna baru dengan memasukkan username dan
password.
I.S. Data user terdefinisi
F.S. Data user baru berhasil ditambahkan dalam variabel user}
```

3. F02 - Login

```
procedure login ()
{Memasukkan user ke dalam akun yang terdaftar dengan username dan
password dari input user yang sesuai.
I.S. User berhasil login
F.S. User melakukan login ulang}
```

4. F03 - Logout

```
function Logout ()
{Mengeluarkan user dari akun yang digunakan dengan mengembalikan
boolean False. User hanya dapat melakukan logout jika sudah
melakukan login ke sebuah akun yang tersedia. }
I.S. User berhasil logout
F.S. User dapat melakukan logout ketika sudah login}
```

5. F04 - Menu & Help

```
procedure menu & help_sebelum_login()
{Menerima input command, lalu mengembalikan nilai command yang
diinput, serta menampilkan daftar command jika user menginput
"HELP".
I.S. user memanggil menu HELP
F.S command menampilkan menu HELP}
procedure menu & help_setelah_login ()
```

{ Prosedur menampilkan daftar command yang bisa dipanggil oleh Agent.

I.S. command menampilkan menu help
F.S user memilih menu HELP yang bisa dipanggil }

6. F05 - Monster

```
procedure ListToDict_monster ()  
{Masukkan data atribut monster-monster berupa type, ATK Power,  
DEF Power, HP, MaxHP, dan Level  
I.S. monster_list terdefinisi dengan elemen-elemen 'id', 'type',  
'atk_power', 'def_power', dan 'hp'  
F.S. Menghasilkan dictionary di mana setiap id monster menjadi  
kunci, dan nilai adalah dictionary atribut monster}  
  
function create_monster(monster, lvl)  
{Membuat monster baru dengan level tertentu berdasarkan data  
monster yang diberikan  
I.S. Data monster dan level (lvl) terdefinisi  
F.S. Menghasilkan dictionary yang merepresentasikan monster dengan  
level yang ditentukan}
```

7. F06 - Potion

```
procedure take_potion()  
{Menambahkan ATK Power, DEF Power, atau HP tergantung potion yang  
digunakan  
I.S user memasukkan jenis potion  
F.S perhitungan jenis potion yang sesuai dengan level}
```

8. F07 - Inventory

```
function inventory()  
{Menggabungkan data oc, monster, dan item user tertentu di dalam  
satu list  
I.S kepemilikan user terhadap (coins, monsters, items)  
F.S list yang terdiri dari 3 index (coins, monsters, items) }
```

9. F08 - Battle

```
procedure battle()  
{Menampilkan tampilan battle pengguna dan memberikan return hasil  
battle dan coin yang diperoleh  
I.S. Monster yang muncul , monster pengguna, opsi pengguna tiap  
turn  
F.S. Hasil battle, coin gain}
```

10. F09 - Arena

```
procedure arena()
```

- {Menampilkan tampilan arena oleh pengguna dan memberikan return hasil arena beserta jumlah stage yang dicapai dan coin yang diperoleh
I.S. Monster yang muncul , monster pengguna, opsi pengguna tiap turn
F.S. Hasil battle, coin gain, informasi monster yang hidup}
11. F10 - Shop & Currency
procedure shop & currency () :
{Prosedur untuk user dapat melakukan pembelian dari shop yang menjual monster dan item.
I.S. user menginput command yang di inginkan
F.S command akan memanggil sesuai fungsi yang di pilih}
12. F11 - Laboratory
procedure laboratory
{Mengelola inventori monster dan melakukan upgrade level monster dalam game
I.S. Inventori monster terdefinisi dalam bentuk dictionary
F.S. Level monster dapat berubah dengan mengurangi OC yang dimiliki pemain}
13. F12 - Shop Management
procedure shop_management ()
{Mengelola penambahan stok dari shop and currency
I.S data stok pada shop and currency ter inisiasi
F.S prosedur pengaturan stok lainnya tereksekusi oleh user dengan role "Admin"}
14. F13 - Monster Management
procedure monster_management
I.S data monster ter inisialisasi
F.S prosedur pengaturan monster lainnya tereksekusi oleh user dengan role "Admin"}
15. F14 - Load
function load ()-> array of any
{fungsi yang meng open file csv dan menyimpan data pada csv ke array
I.S memuat data dari csv ke dalam kamus,serta menampilkan animasi pemuatan
F.S pencetakan dan pemuatan data }
16. F15 - Save
procedure save ()
{Menyimpan data ke dalam file di folder yang diinputkan
I.S. semua data yang akan disimpan telah terdefinisikan dalam variabel
user,monster,monster_inventory,monster_shop,item_inventory,item_shop
F.S. jika nama folder ada, maka program akan menyimpan data dari program ke folder tersebut, jika folder tidak ada maka program

akan membuat folder terlebih dahulu lalu menyimpan data ke folder tersebut}

17. F16 - Exit

```
procedure exit()
{prosedur yang akan dijalankan ketika pengguna ingin keluar,
program akan meminta pengguna apakah ingin menyimpan perubahan
yang terjadi
I.S. pengguna telah menjalankan beberapa bagian program dan telah
terjadi perubahan pada data yang telah tersimpan pada load
F.S. program berhenti dan keluar dari program}
```

Spesifikasi dalam Notasi Algoritmik

1. F00 - Random Number Generator

```
procedure linear_congruential_generator()
procedure random(lower_limit, upper_limit)

ALGORITMA

Seed ← int(time.time() + os.getpid() + iteration
While len(result) do:
    Seed ← (a * seed + c) % m
        random_number ← ((seed % (max_value - min_value + 1)) +
+min_value)
            if (unique = y) then
                if (random_number) not in result then

result.append(random_number)
else
result.append(random_number)

→result
```

2. F01 - Register

```
procedure RegisterUser (input UserData : UserDataStruct, output Message
: string)
{ Proses registrasi pengguna baru }

KAMUS LOKAL

UsernameExists : boolean
PasswordStrength : boolean

ALGORITMA

{ Validasi data pendaftar}
UsernameExists <- CheckUsernameExists(UserData.username)
PasswordStrength <- CheckPasswordStrength(UserData.password)

{Jika data pendaftar valid}
if (UsernameExists = False and PasswordStrength = True) then
    SaveUserData(UserData) { Menyimpan data pengguna ke dalam basis
data}
```

```

        output( "Registrasi berhasil. Anda sekarang dapat masuk." )
    else
        ("Ada kesalahan dalam proses registrasi. Mohon coba lagi.")

function CheckUsernameExists (input username : string) -> boolean
{ Memeriksa apakah username sudah ada dalam basis data }
function CheckPasswordStrength (input password : string) -> boolean
{ Memeriksa kekuatan kata sandi }

KAMUS LOKAL
    PasswordStrength : boolean

ALGORITMA
    PasswordStrength := EvaluatePasswordStrength(password) // Menilai
kekuatan kata sandi
    return PasswordStrength

KAMUS LOKAL
    UserExists : boolean

ALGORITMA
    UserExists <- QueryDatabaseForUsername(username) // Melakukan query
pada basis data untuk mencari username
    return UserExists

procedure SaveUserData (input UserData : UserDataStruct)
{ Menyimpan data pengguna ke dalam basis data }

Kamus Lokal
    // Tidak ada

Algoritma
    InsertUserDataIntoDatabase(UserData) // Memasukkan data pengguna ke
dalam basis data

function QueryDatabaseForUsername (input username : string) : boolean
{ Melakukan query pada basis data untuk memeriksa keberadaan username }

Kamus Lokal
    UserExists : boolean
Algoritma
    // Implementasi query pada basis data untuk mencari username
    // Jika username ditemukan, set UserExists menjadi true
    // Jika tidak ditemukan, set UserExists menjadi false

```

```
    return UserExists
```

3. F02 - Login

ALGORITMA

```
FUNCTION exist_username(user: DICTIONARY, username: STRING) -> BOOLEAN
    FOR each id IN user
        IF id ≠ "Atribut" AND user[id][0] = username THEN
            RETURN TRUE
        RETURN FALSE
    END FUNCTION

FUNCTION wrong_password(user: DICTIONARY, username: STRING, password: STRING) -> BOOLEAN
    FOR each id IN user
        IF id ≠ "Atribut" AND user[id][0] = username THEN
            RETURN user[id][1] ≠ password
        RETURN TRUE
    END FUNCTION

FUNCTION login(user: DICTIONARY) -> INTEGER
    username ← INPUT("Username: ")
    password ← INPUT("Password: ")

    IF NOT exist_username(user, username) THEN
        PRINT "Username tidak terdaftar!"
        RETURN -1
    END IF

    IF wrong_password(user, username, password) THEN
        PRINT "Password salah!"
        RETURN -1
    END IF

    FOR each id IN user
        IF id ≠ "Atribut" AND user[id][0] = username THEN
            IDUser ← id
            BREAK
        END IF
    END FOR

    IF IDUser IS UNDEFINED THEN
        PRINT "User not found!"
        RETURN -1
    END IF

    user_role ← user[IDUser][2]
    IF user_role = "Agent" THEN
        PRINT "Selamat datang, Agent " + username + "!"
    ELSE
        PRINT "Selamat datang, Admin " + username + "!"
    END IF
```

```
    RETURN CONVERT TO INTEGER(IDUser)
END FUNCTION
```

4. F03 - Logout

ALGORITMA

```
FUNCTION logout(masuk: BOOLEAN)
IF masuk THEN
    PRINT "Logout Berhasil."
ELSE
    PRINT "Logout gagal!"
    PRINT "Login dulu mas baru logout."
END IF
END FUNCTION
```

5. F04 - Menu & Help

{Help_before_login function: menampilkan pesan bantuan untuk pengguna yang belum melakukan login}

ALGORITMA

```
FUNCTION help_before_login()
PRINT "===== HELP ====="
PRINT "Login belum dilakukan, silahkan login terlebih dahulu."
PRINT "1. Login: Masuk ke dalam akun yang sudah terdaftar"
PRINT "2. Register: Membuat akun baru"

PRINT "\nFootnote:"
PRINT "Silahkan masukan fungsi yang terdaftar dalam menggunakan
aplikasi"
PRINT "Pastikan input yang dimasukkan valid"
END FUNCTION
```

{Help_admin function: menampilkan pesan bantuan untuk pengguna dengan peran admin}

ALGORITMA

```
FUNCTION help_admin()
PRINT "===== HELP ====="
PRINT "Selamat datang, Admin. Berikut adalah hal-hal yang dapat kamu
lakukan:"
PRINT "1. Logout: Keluar dari akun yang sedang digunakan"
PRINT "2. Shop Management: Melakukan manajemen pada SHOP sebagai
tempat jual beli peralatan Agent"
PRINT "3. Monster Management: Melakukan manajemen Monster"
PRINT "4. Save: Menyimpan progress Anda"
PRINT "5. Exit: Keluar dari game"

PRINT "\nFootnote:"
PRINT "Silahkan masukan fungsi yang terdaftar dalam menggunakan
aplikasi"
```

```

    PRINT "Pastikan input yang dimasukkan valid"
END FUNCTION

```

{Help_agent function: menampilkan pesan bantuan dengan peran agent dan menyedialan tindakan yang dapat dilakukan oleh agent}

ALGORITMA

```

FUNCTION help_agent(username: STRING)
    PRINT "===== HELP ====="
    PRINT "Halo Agent " + username + ". Kamu memanggil command HELP."
    Kamu memilih jalan yang benar, semoga kamu tidak sesat kemudian. Berikut adalah hal-hal yang dapat kamu lakukan sekarang:
        PRINT "1. Logout: Keluar dari akun yang sedang digunakan"
        PRINT "2. Inventory: Melihat owca-dex yang dimiliki oleh Agent"
        PRINT "3. Shop: Membeli potion atau monster"
        PRINT "4. Battle: Bertarung melawan monster!"
        PRINT "5. Arena: Latih dan tingkatkan kemampuan agen dan para monstermu!"
        PRINT "6. Laboratory: Upgrade monster yang kamu miliki"
        PRINT "7. Jackpot: Uji keberuntunganmu di sini!"
        PRINT "8. Legenda: Melihat peta kota Denville"
        PRINT "9. Save: Menyimpan progress Anda"
        PRINT "10. Exit: Keluar dari game"

    PRINT "\nFootnote:"
    PRINT "Silahkan masukan fungsi yang terdaftar dalam menggunakan aplikasi"
    PRINT "Pastikan input yang dimasukkan valid"
END FUNCTION

```

{menu function: menampilkan pesan sambutan dan deskripsi singkat tentang kota Monster Topia dan menyediakan menu help}

```

FUNCTION menu()
    PRINT """
        ====== KOTA MONSTERTOPIA
=====
        || -----
        ||           Selamat datang di kota MonsterTopia
        || -----
        ||       ~~ mulai petualangan dan kalahkan musuh musuh     ~~
        || -----
        ======
        (Ketik HELP untuk mendapat bantuan)

"""
END FUNCTION

```

6. F05 - Monster

```
procedure ListToDict_monster (monster_list: Dictionary ('id', 'type',
'atk_power', 'def_power', 'hp'): integer) → dictOfDict: Dictionary
```

KAMUS LOKAL

```
monster_dict = dictionary of string {Name, ATK Power, DEF Power, HP,
MaxHP, Level}
dictOfDict = dictionary of dictionary {id:{monster_dict}}
```

ALGORITMA

```
monster_list <- {}
dictOfDict <- {}

i traversal [0..len(monster_list['id'])]
    monster_dict['Name'] <- monster_list['type'][i]
    monster_dict['ATK Power'] <- monster_list['atk_power'][i]
    monster_dict['DEF Power'] <- monster_list['def_power'][i]
    monster_dict['HP'] <- monster_list['hp'][i]
    monster_dict['MaxHP'] <- monster_list['hp'][i]
    monster_dict['Level'] <- 1
    dictOfDict[monster_list['id'][i]] <- monster_dict

-> dictOfDict
```

```
function create_monster (monster: Dictionary ('Name', 'ATK Power', 'DEF
Power', 'HP', 'MaxHP', 'Level'): integer)→ wild_monster: Dictionary
(Name, ATK Power, DEF Power, HP, MaxHP, Level)
```

KAMUS LOKAL

```
name = string
attack, defense, hp, level = integer
wild_monster = dictionary
```

ALGORITMA

```
name <- monster['Name']
attack <- monster['ATK Power']
defense <- monster['DEF Power']
hp <- monster['HP']
level <- int(lvl)
wild_monster <- {
    'Name' <- name,
    'ATK Power' <- attack * (1 + ((level - 1) * 0.1)),
    'DEF Power' <- defense * (1 + ((level - 1) * 0.1)),
    'HP' <- hp * (1 + ((level - 1) * 0.1)),
    'MaxHP' <- hp * (1 + ((level - 1) * 0.1)),
    'Level' <- level}

-> wild_monster
```

7. F06 - Potion

```
procedure take_potion(monster: Dictionary ( 'ATK Power', 'DEF Power', 'HP',
'MaxHP'): integer, potion: ('Strength Potion', 'Resilience Potion',
'Healing Potion'): String) → (monster: Dictionary ( 'ATK Power', 'DEF
Power', 'HP', 'MaxHP'): integer) ###hanya perubahan atribut
```

KAMUS LOKAL

```

monster = dictionary
potion = String

ALGORITMA

depend on potion

    potion == 'Strength Potion' :
        monster['ATK Power'] <- monster['ATK Power'] * 1.05
    potion == 'Resilience Potion' :
        monster['DEF Power'] <- monster['DEF Power'] * 1.05
    potion == 'Healing Potion' :
        monster['HP'] <- ( monster['HP'] ) + ( 0.25 * monster['MaxHP'] )
    )

        if monster['HP'] > monster['MaxHP'] then
            monster['HP'] = monster['MaxHP']

```

8. F07 - Inventory

```

function organize_user_info(user_info: dictionary of array) -> dictionary
of dictionary

ALGORITMA

user_info <-
user_data <- {}

(id, username, password, role, oc) traversal [user_info]
    user_data[id] <- {
        'username'<- username,
        'password'<- password,

```

```

    'role' <- role,
    'oc' <- oc }

-> user_data

function ListToList_monster_inventory(monster_inventory: dictionary of
array of any) -> dictionary of dictionary

```

ALGORITMA

```

monster_inventory <-
user_monsters <- {}
(user_id, monster_id, level) traversal [monster_inventory]
    if user_id not in user_monsters then
        user_monsters[user_id] <- []
        user_monsters[user_id].append({'monster_id': monster_id, 'level':
level})
-> user_monsters

function ListToList(item_inventory: dictionary of array of any) ->
dictionary of dictionary

```

ALGORITMA

```

item_inventory <- {}
user_items <- {}
(user_id, item_type, quantity) traversal [item_inventory]
    if user_id not in user_items then
        user_items[user_id] <- []
        user_items[user_id].append({'type': item_type, 'quantity':
quantity})

-> user_items

function inventory (user: Dictionary (user), list_item: List of dictionary
(item), list_mons: List of dictionary (monster), id: String) → list_of_inv

```

ALGORITMA

```
list_of_inv <- [i traversal [i traversal [0..4]]]
list_of_inv[0] <- int(user[id]['oc'])

list_of_inv_mons <- [i traversal [i traversal [0..(len(list_mons[id]))]]]

i traversal [0..(len(list_mons[id]))]
    list_of_inv_mons[i] <-
create_monster(monster_dict[list_mons[id][i]['monster_id']],
list_mons[id][i]['level'])

list_of_inv[1] <- list_of_inv_mons

list_of_inv_item <- [{type: 'strength', quantity: 0}, {type: 'resilience', quantity: 0}, {'type': 'healing', 'quantity': 0}]
i traversal [0..(len(list_item[id]))]
    j traversal [0..3]
        if list_item[id][i]['type'] == list_of_inv_item[j]['type'] then
            list_of_inv_item[j]['quantity'] <-
int(list_item[id][i]['quantity'])

list_of_inv[2] <- list_of_inv_item
list_of_inv[3] <- id
-> list_of_inv

procedure show_detail_inventory (inventory>List of string) → String
### mengeluarkan tampilan inventory pengguna
```

ALGORITMA

```
list_everything <- [i traversal [ i traversal
[0..(len(inventory[1])+(len(inventory[2]))+1))]]
list_everything[0] <- inventory[0]
i traversal [0..(len(inventory[1]))]
    list_everything[i+1] <- inventory[1][i]
j traversal [0..(len(inventory[2]))]
```

```

list_everything[j+len(inventory[1])+1] <- inventory[2][j]

count <- 1

output("you have " + str({list_everything[0]})) + " coins")
i traversal [1..len(list_everything)]
    if len(list_everything[i]) == 6 then
        output(str({count}). Monster (Name: {list_everything[i]['Name']},
Lvl: {list_everything[i]['Level']}, HP: {list_everything[i]['HP']}))

    if len(list_everything[i]) == 2 and list_everything[i]['quantity'] != '0' then
        output(str(({count}). Potion           (Type:
{list_everything[i]['type']}), Qty: {list_everything[i]['quantity']})))
        count <- count + 1

index <- validate_integer_input_2(len(list_everything)-1,'Index: ')

depend on len(list_everything[index])

len(list_everything[index]) == 6 :
    output('Monster')
    output("Name      : " + str({list_everything[index]['Name']}))
    output("ATK Power : " + str({list_everything[index]['ATK
Power']}))
    output("DEF Power : " + str({list_everything[index]['DEF
Power']}))
    output("HP      : " + str({list_everything[index]['HP']}))
    output("Level     : " + str({list_everything[index]['Level']}))

len(list_everything[index]) == 2 :
    output('Potion')
    output("Type      : " + str({list_everything[index]['type']}))
    output("Quantity  : " +
str({list_everything[index]['quantity']}))

procedure inventory_to_info(inv_player: dictionary)

```

ALGORITMA

```
inv_player <- inventory(user_info_org, inventory_of_items,
inventory_of_monsters, '3')
user_info['oc'][int(inv_player[3])-1] <- str(int(inv_player[0]))
ind <- []
i traversal [0..(len(item_inventory['user_id']))]
    if item_inventory['user_id'][i] == str(inv_player[3]) then
        ind.append(i)
j traversal [0..(len(ind))]
    k traversal [0..(len(inv_player[2]))]
        if item_inventory['type'][ind[j]] == inv_player[2][k]['type']
then
    item_inventory['quantity'][ind[j]] <-
inv_player[2][k]['quantity']
```

9. F08 - Battle

procedure take_damage (monster: dictionary, damage: integer)**ALGORITMA**

```
monster['HP'] <- (monster['HP']) - damage
if monster['HP'] < 0 then
    monster['HP'] <- 0
```

procedure attack_opponent (attacker: dictionary ,opponent: dictionary)**ALGORITMA**

```
damage <- (attacker['ATK Power'] * ((random(7, 14)) / 10)) * ((1 -
(opponent['DEF Power'] / 100)))
```

```
take_damage(opponent, damage)
output((str({attacker['Name']})) + str( attacks {opponent['Name']})) + "
and deals " + str( {damage}) +" damage!")
-> damage
```

```
function is_alive (monster: dictionary) -> boolean
```

ALGORITMA

```
-> (monster['HP'] > 0)
```

```
function battle (inventory: List, monster_dict: Dictionary, action:
integer) → String ### hasil battle, integer ### jumlah koin yang didapat
```

KAMUS LOKAL

```
wild_monster, monster = dictionary
used_potion = array of boolean
run_away = boolean
turn, i, chosen_monster, success = integer
action = string
```

ALGORITMA

```
wild_monster <- monster_dict[str(int(random(1,6)))]
used_potion <- [False, False, False]
run_away <- False
turn <- 1
output(sprite_goblin)
output("A wild " + str( {wild_monster['Name']}) + " appears!")
output("Name      : " + str({wild_monster['Name']}))
output("ATK Power : " + str({wild_monster['ATK Power']}))
output("DEF Power : " + str({wild_monster['DEF Power']}))
output("HP       : " + str({wild_monster['HP']}))
output("Level     : " + str({wild_monster['Level']}))
output("==== MONSTER LIST ====")
i traversal [0..int((len(inventory[1])))]
```

```

        output(str({i+1}. {inventory[1][i]['Name']}))

output("")

chosen_monster <- validate_integer_input_2(len(inventory[1]), str('Pilih
Monster: '))

monster <- inventory[1][int(chosen_monster)-1]

monster['HP'] <- monster['MaxHP']

output(sprite_monster)

output("Agent chose " + str({monster['Name']}))

output("Name      : " + str({monster['Name']}))

output("ATK Power : " + str({monster['ATK Power']}))

output("DEF Power : " + str({monster['DEF Power']}))

output("HP       : " + str({monster['HP']}))

output("Level     : " + str({monster['Level']}))

while (is_alive(monster) and is_alive(wild_monster)) do
    output("===== TURN " + str ({turn} {monster['Name']}) + " =====")
    output ("1. Attack")
    output ("2. Use potion")
    output ("3. Quit")
    action <- str(validate_integer_input_2(3, str("Pilih Perintah: ")))

    depend on action
    action == '1':
        attack_opponent(monster,wild_monster)
        output("Name      : " + str({wild_monster['Name']}))
        output("ATK Power : " + str({wild_monster['ATK Power']}))
        output("DEF Power : " + str({wild_monster['DEF Power']}))
        output("HP       : " + str({wild_monster['HP']}))
        output("Level     : " + str({wild_monster['Level']}))

    action == '2':
        output("===== POTION LIST =====")
        output("1. Strength Potion (Qty: " +
str({inventory[2][0]['quantity']})) + " - Increases ATK Power")
        output("2. Resilience Potion (Qty: " +
str({inventory[2][1]['quantity']})) + " - Increases DEF Power")

```

```

        output("3. Healing Potion (Qty: " +
str({inventory[2][2]['quantity']})) + " - Restores Health")
            potion <- (str(validate_integer_input_2(3, 'Pilih
Perintah: ')))

            depend on potion, inventory, used_potion
            potion == '1' and inventory[2][0]['quantity'] !=
0 and used_potion[0] == False:
                take_potion(monster, 'Strength Potion')
                inventory[2][0]['quantity'] <-
inventory[2][0]['quantity'] - 1
                used_potion[0] <- True
                output(str({monster['Name']})) + " has been
strengthened"
            potion == '2' and inventory[2][1]['quantity'] !=
0 and used_potion[1] == False:
                take_potion(monster, 'Resilience Potion')
                inventory[2][1]['quantity'] <-
inventory[2][1]['quantity'] - 1
                used_potion[1] <- True
                output(str({monster['Name']})) + " has
gained resilience"
            potion == '3' and inventory[2][2]['quantity'] !=
0 and used_potion[2] == False:
                take_potion(monster, 'Healing Potion')
                inventory[2][2]['quantity'] <-
inventory[2][2]['quantity'] - 1
                used_potion[2] <- True
                output(str({monster['Name']})) + " has been
healed")
            else:
                output("Out of potion!")

action == '3':
    success <- random(0,2)
    if success == 1 then
        output("You have successfully escaped!")

```

```

        run_away <- True
        break
    else:
        output("You failed to run away!")

    if is_alive(wild_monster):
        output("===== TURN " + str({turn}) +
str({wild_monster['Name']})) + "====")
        attack_opponent(wild_monster,monster)
        output("Name : " + str({monster['Name']}))
        output("ATK Power : " + str({monster['ATK Power']})))
        output("DEF Power : " + str({monster['DEF Power']})))
        output("HP : " + str({monster['HP']})))
        output("Level : " + str({monster['Level']})))

    turn += 1

depend on is_alive(monster) and run_away
is_alive(monster) and not run_away == True:
    output(str({monster['Name']}) + " wins!")
    output("Name : " + str({wild_monster['Name']}))
    output("ATK Power : " + str({wild_monster['ATK Power']})))
    output("DEF Power : " + str({wild_monster['DEF Power']})))
    output("HP : " + str({wild_monster['HP']})))
    output("Level : " + str({wild_monster['Level']})))
    coin_gain <- random(5,31)
    inventory[0] <- inventory[0] + coin_gain
    output("Congrats, You got " + str({coin_gain}) +" coins!")

not run_away == True:
    output(str({wild_monster['Name']})+ " wins!")
    output("Name : " + str({monster['Name']}))
    output("ATK Power : " + str({monster['ATK Power']})))
    output("DEF Power : " + str({monster['DEF Power']})))
    output("HP : " + str({monster['HP']})))
    output("Level : " + str({monster['Level']})))
    coin_gain <- random(5,31)
    inventory[0] <- inventory[0] - coin_gain
    output("You fainted, you lost " + str ({coin_gain}) + " coins...")

-> inventory

```

10. F09 - Arena

```
procedure take_damage (monster: dictionary, damage: integer)

ALGORITMA

monster['HP'] <- (monster['HP']) - damage
if monster['HP'] < 0 then
    monster['HP'] <- 0

procedure attack_opponent (attacker: dictionary ,opponent: dictionary)

ALGORITMA

damage <- (attacker['ATK Power'] * ((random(7, 14)) / 10)) * ((1 -
(opponent['DEF Power'] / 100)))
take_damage(opponent, damage)
output((str({attacker['Name']})) + str( attacks {opponent['Name']})) + "
and deals " + str( {damage}) +" damage!")
-> damage

function is_alive (monster: dictionary) -> boolean

ALGORITMA

-> (monster['HP'] > 0)

function arena (inventory: List, monster_dict: Dictionary, action: integer)
→ String ### hasil arena, integer ### jumlah koin yang didapat

KAMUS LOKAL

wild_monster, monster = dictionary
```

```

used_potion = array of boolean
run_away = boolean
turn, stage, i, chosen_monster, success = integer
action = string

```

ALGORITMA

```

used_potion <- [False, False, False]
run_away <- False
turn <- 1
stage <- 1
total_damage <- 0
total_received <- 0
output("==== MONSTER LIST ====")
i traversal [0..(len(inventory[1]))]
    output(str({i+1}. {inventory[1][i]['Name']}))
output("")

chosen_monster <- validate_integer_input_2(len(inventory[2]), 'Pilih
Monster: ')
monster <- (inventory[1][int(chosen_monster)-1])
monster['HP'] <- monster['MaxHP']
output(sprite_monster)
output("Agent chose " + str({monster['Name']}))
output("Name      : " + str({monster['Name']}))
output("ATK Power : " + str({monster['ATK Power']}))
output("DEF Power : " + str({monster['DEF Power']}))
output("HP       : " + str({monster['HP']}))
output("Level     : " + str({monster['Level']}))

while stage < 6 and is_alive(monster) and not run_away do
    output("==== STAGE " + str({stage}) +" ====")
    wild_monster <- monster_dict[str(int(random(1,6)))]
    wild_monster <- create_monster(wild_monster, stage)
    monster['HP'] <- monster['MaxHP']
    wild_monster['HP'] <- wild_monster['MaxHP']

```

```

        output(sprite_goblin)

        output("A wild " + str( {wild_monster['Name']} ) + " appears!")
        output("Name      : " + str({wild_monster['Name']}))
        output("ATK Power : " + str({wild_monster['ATK Power']}))
        output("DEF Power : " + str({wild_monster['DEF Power']}))
        output("HP       : " + str({wild_monster['HP']}))
        output("Level     : " + str({wild_monster['Level']}))

        while is_alive(monster) and is_alive(wild_monster) do
            output("==== TURN "+ str({turn}) + str({monster['Name']})) + "
====")
            output ("1. Attack")
            output ("2. Use potion")
            output ("3. Quit")
            action <- str(validate_integer_input_2(3, str("Pilih Perintah:
")))
    
```

depend on action

```

            action == '1':
                total_damage <- total_damage +
                (attack_opponent(monster,wild_monster))
                output("Name      : " + str({wild_monster['Name']}))
                output("ATK Power : " + str({wild_monster['ATK
Power']}))
                output("DEF Power : " + str({wild_monster['DEF
Power']}))
                output("HP       : " + str({wild_monster['HP']}))
                output("Level     : " + str({wild_monster['Level']}))

            action == '2':
                output("==== POTION LIST ====")
                output("1. Strength Potion (Qty: " +
str({inventory[2][0]['quantity']})) + " - Increases ATK Power")
                output("2. Resilience Potion (Qty: " +
str({inventory[2][1]['quantity']})) + " - Increases DEF Power")
                output("3. Healing Potion (Qty: " +
str({inventory[2][2]['quantity']})) + " - Restores Health")
    
```

```

        potion <- (str(validate_integer_input_2(3, 'Pilih
Perintah: ')))

            depend on potion, inventory, used_potion
            potion == '1' and
            inventory[2][0]['quantity'] != 0 and used_potion[0] == False:
                take_potion(monster, 'Strength
Potion')
                    inventory[2][0]['quantity'] <-
inventory[2][0]['quantity'] - 1
                    used_potion[0] <- True
                    output(str({monster['Name']})) + " has
been strengthened"
            potion == '2' and
            inventory[2][1]['quantity'] != 0 and used_potion[1] == False:
                take_potion(monster, 'Resilience
Potion')
                    inventory[2][1]['quantity'] <-
inventory[2][1]['quantity'] - 1
                    used_potion[1] <- True
                    output(str({monster['Name']})) + " has
gained resilience"
            potion == '3' and
            inventory[2][2]['quantity'] != 0 and used_potion[2] == False:
                take_potion(monster, 'Healing Potion')
                inventory[2][2]['quantity'] <-
inventory[2][2]['quantity'] - 1
                used_potion[2] <- True
                output(str({monster['Name']})) + " has
been healed"
            else:
                output("Out of potion!")

        action == '3':
            success <- random(0,2)
            if success == 1 then
                output("You have successfully escaped!")

```

```

        run_away <- True
        break
    else:
        output("You failed to run away!")

    if is_alive(wild_monster) then
        output("==== TURN " + str({turn}
{wild_monster['Name']})) + " ====")
        total_received <- total_received +
attack_opponent(wild_monster,monster)
        output("Name      : " + str({monster['Name']}))
        output("ATK Power : " + str({monster['ATK Power']})))
        output("DEF Power : " + str({monster['DEF Power']})))
        output("HP       : " + str({monster['HP']})))
        output("Level     : " + str({monster['Level']})))
        turn += 1

    if is_alive(monster) and not run_away == True then
        output(str({monster['Name']})) + " wins!")
        output("Name      : " + str({wild_monster['Name']}))
        output("ATK Power : " + str({wild_monster['ATK Power']})))
        output("DEF Power : " + str({wild_monster['DEF Power']})))
        output("HP       : " + str({wild_monster['HP']})))
        output("Level     : " + str({wild_monster['Level']})))

        stage <- stage + 1

    else :
        output(str({wild_monster['Name']})) + " wins!")
        output("Name      : " + str({wild_monster['Name']}))
        output("ATK Power : " + str({wild_monster['ATK Power']})))
        output("DEF Power : " + str({wild_monster['DEF Power']})))
        output("HP       : " + str({wild_monster['HP']})))
        output("Level     : " + str({wild_monster['Level']}))

inventory[0] <- inventory[0] + (30*(stage-1))
output("==== STATS ====")

```

```

output("Total hadiah      : " + str({(30*(stage-1))}) + " OC ")
output("Jumlah stage       : " + str({stage-1}) )
output("Damage diberikan   : " + str({total_damage}))
output("Damage diterima     : " + str({total_received}))

-> inventory

```

11. F10 - Shop & Currency

```

{Inisialisasi list}

monsters <- [ {'id': monster_shop['monster_id'][i], 'stock':
int(monster_shop['stock'][i]), 'price': int(monster_shop['price'][i]),
'type': monster['type'][i], 'atk_power': int(monster['atk_power'][i]),
'def_power': int(monster['def_power'][i]), 'hp': int(monster['hp'][i])}
for i in range(len(monster_shop['monster_id']))]

potions <- [ {'type': potion_shop['type'][i], 'stock':
int(potion_shop['stock'][i]), 'price': int(potion_shop['price'][i])} for
i in range(len(potion_shop['type']))]

function spend_currency(inventory, amount)

ALGORITMA

if inventory[0] >= amount then
    inventory[0] <- inventory[0] - amount
    -> True
else:
    output("OC anda tidak cukup.")
    ->False

procedure buy_item(item_type, item_id, inventory, quantity=1

```

ALGORITMA

```
if item_type == "monster" then
    item <- None
    monster traversal [0.. monsters]
        if monster['id'] == item_id then
            item <- monster
            break
    if not item then
        output("Monster dengan ID tersebut tidak ada.")
        ->
    if item['stock'] < quantity then
        output("Stok monster tidak mencukupi.")
        ->
    price <- item['price'] * quantity
    if not spend_currency(inventory, price) then
        ->
    if any(monster['Name'] == item['type'] for monster in inventory[1])
then
    output("Monster tersebut sudah ada dalam inventory-mu!")
    ->
    item['stock'] <- item['stock'] - quantity
    new_monster <-{'Name': item['type'], 'ATK Power': item['atk_power'],
'DEF Power': item['def_power'], 'HP': item['hp'], 'MaxHP': item['hp'],
'Level': 1}
    inventory[1].append(new_monster)
    output("Berhasil membeli item: Monster " + str({item_id}) + ". Item
sudah masuk ke inventory-mu!")

elif item_type == "potion" then
    item <- None
    potion traversal [0..potions]
        if potion['type'] == item_id then
            item <- potion
            break
    if not item then
```

```

        output("Potion dengan ID tersebut tidak ada.")
    ->

    if item['stock'] < quantity then
        output("Stok potion tidak mencukupi.")
    ->

    price <- item['price'] * quantity
    if not spend_currency(inventory, price) then
        ->
        item['stock'] <- item['stock'] - quantity
        inv_item traversal [0..(inventory[2])]
        if inv_item['type'] == item_id then
            inv_item['quantity'] <- inv_item['quantity'] - quantity
            break
        else:
            inventory[2].append({'type': item_id, 'quantity':
quantity})

        output("Berhasil membeli item: Potion " + str(item_id)) + ".
Item sudah masuk ke inventory-mu!")

```

procedure display_shop(inventory)

ALGORITMA

```

output("Irasshaimase! Selamat datang di SHOP!!")
while True do
    output(">>> Pilih aksi (lihat/beli/keluar):")
    input(action)
    depend on action
    action == "lihat" then
        output(">>> Mau lihat apa? (monster/potion):")
        input(item_type)
        depend on item_type
        item_type == "monster" :
            output("{'ID':<3} | {'Type':<10} | {'ATK
Power':<9} | {'DEF Power':<9} | {'HP':<4} | {'Stock':<5} | {'Price':<6}")
            output("-" * 61)

```

```

        monster traversal [0..monsters]
        output("{monster['id']:<3} |
{monster['type']:<10} | {monster['atk_power']:<9} |
{monster['def_power']:<9} | {monster['hp']:<4} | {monster['stock']:<5} |
{monster['price']:<6}")

        item_type == "potion":
            output("{'Type':<10} | {'Stock':<5} |
{'Price':<6}")

            output(( "-" * 23))
            potion traversal [0..potions]
            output("{potion['type']:<10} |
{potion['stock']:<5} | {potion['price']:<6}")

        else:
            output("Tipe item tidak valid.")

action == "beli":
    output("Jumlah O.W.C.A. Coin-mu sekarang + +
str({inventory[0]}) + ".")
    output(">>> Mau beli apa? (monster/potion):")
    input(item_type)
    depend on item_type
    item_type == "monster":
        output(">>> Masukkan ID monster:")
        input(item_id)
        output(">>> Masukkan jumlah:")
        input(quantity)
        buy_item("monster", item_id, inventory,
quantity)

    item_type == "potion":
        output(">>> Masukkan type potion:")
        input(item_id)
        output(">>> Masukkan jumlah:")
        input( quantity )
        buy_item("potion", item_id, inventory, quantity)

    else:
        output("Tipe item tidak valid.")

action == "keluar":

```

```

        output("Mr. Yanto bilang makasih, belanja lagi ya nanti
:")
    break
else:
    output("Aksi tidak valid.")

```

12. F11 - Laboratory

```

function get_value(dictionary, key, default=None)

ALGORITMA
if key in dictionary then
    -> dictionary[key]
else:
    ->default

procedure laboratory(inventory)

    monsters <- inventory[1]
    upgrade_prices <- {
        1 <- 300,
        2 <- 500,
        3 <- 800,
        4 <- 1000
    }

    procedure display_monsters(monsters)
        output("===== MONSTER LIST =====")
        i traversal [0..(len(monsters))]
        monster <- monsters[i]
        output(str({i+1}. {monster['Name']} ) (Level:
{monster['Level']})))

    function display_upgrade_prices(upgrade_prices)
        output("===== UPGRADE PRICE =====")

```

```

        (level, price) traversal [0..(upgrade_prices.items())]
            output("Level "+ str({level})+" -> Level " + str({level +
1}) + " : " + str({price}) + " O")

function choose_monster(monsters)
    output(">>> Pilih monster: ")
    choice <- validate_integer_input("Masukkan nomor monster: ")
    choice <- int(choice) - 1
    if 0 <= choice < len(monsters) then
        -> choice
    output("Pilihan tidak valid. Coba lagi.")
        -> choose_monster(monsters)

procedure upgrade_monster(monsters, index, upgrade_prices,
oc_balance)
    monster <- monsters[index]
    name, level <- monster['Name'], monster['Level']
    if level >= 5 then
        output("Maaf, monster yang Anda pilih sudah memiliki level
maksimum.")
        -> oc_balance

    upgrade_cost <- get_value(upgrade_prices, level, 0)
    output(str({name}) + " akan di-upgrade ke level " + str(
{level + 1}) + ".")
    output("Harga untuk melakukan upgrade " +str({name}) + " adalah
" + str({upgrade_cost}) + " OC.")
    output(">>> Lanjutkan upgrade (y/n): ")
    input(confirm)
    if confirm == 'y' then
        if oc_balance >= upgrade_cost then
            oc_balance <- oc_balance - upgrade_cost
            inventory[1][index]['Level'] <-
inventory[1][index]['Level'] + 1
            output("Selamat, " +str({name})+ " berhasil
di-upgrade ke level "+ str({level + 1}) + " !")
        else:

```

```

        output("OC tidak cukup untuk melakukan upgrade.")
    else:
        output("Upgrade dibatalkan.")
    -> oc_balance

output("Selamat datang di Lab Dokter Asep !!!")
display_monsters(monsters)
display_upgrade_prices(upgrade_prices)

index <- choose_monster(monsters)
oc_balance <- upgrade_monster(monsters, index, upgrade_prices,
inventory[0])
inventory[0] <- oc_balance
output("OC Balance:" + str(oc_balance) )
display_monsters(monsters)
-> inventory[0], inventory[1]

```

13. F12 - Shop Management

```

procedure take_potion(monster: Dictionary ( 'ATK Power', 'DEF Power', 'HP',
'MaxHP'): integer, potion: ('Strength Potion', 'Resilience Potion',
'Healing Potion'): String) → (monster: Dictionary ( 'ATK Power', 'DEF
Power', 'HP', 'MaxHP'): integer) ###hanya perubahan atribut

```

KAMUS LOKAL

```

monster = dictionary
potion = String

```

ALGORITMA

```

depend on potion

    potion == 'Strength Potion' :
        monster['ATK Power'] <- monster['ATK Power'] * 1.05
    potion == 'Resilience Potion' :
        monster['DEF Power'] <- monster['DEF Power'] * 1.05
    potion == 'Healing Potion' :
        monster['HP'] <- ( monster['HP'] ) + ( 0.25 * monster['MaxHP'] )
)

    if monster['HP'] > monster['MaxHP'] then
        monster['HP'] = monster['MaxHP']

```

14. F13 - Monster Management

```

procedure print_table(data)

ALGORITMA

headers <- []
key traversal [0..data]
    headers.append(key)
rows <- []
values traversal [0.. (data.values())]
    rows.append(values)

max_lengths <- []
i traversal [0..(len(headers))]
    max_len <- len(headers[i])
    j <- 0
    while j < len(rows[i]) do
        if len(str(rows[i][j])) > max_len then

```

```

        max_len <- len(str(rows[i][j]))
        j <- j + 1
        max_lengths.append(max_len)

i <- 0
while i < len(headers) do
    output(str({headers[i]} + ":" + str({max_lengths[i]})) + " |")
    output("")
    i <- i + 1
output("")

longest_row <- 0
i <- 0
while i < len(rows) do
    if len(rows[i]) > longest_row then
        longest_row <- len(rows[i])
    i <- i + 1

i <- 0
while i < longest_row do
    j <- 0
    while j < len(headers) do
        if i < len(rows[j]) then
            value <- rows[j][i]
        else:
            value <- "None"
        output(str({value}:{max_lengths[j]})) + " |")
        output("")
        j <- j + 1
    print("")
    i <- i + 1

procedure monster_management()

ALGORITMA
output (ascii_monster)

```

```

while True do
    output("">>>> \033[1m Monster\033[0m")
    output("Selamat datang di database Monster, silahkan pilih menu yang
diinginkan:")
    output("1. Tampilkan semua Monster")
    output("2. Tambahkan Monster baru")
    input(aksi)

    while aksi != "1" and aksi != "2" and aksi != "keluar" do
        output("Aksi tidak valid! Silahkan pilih aksi yang valid.")
        input(aksi)

    depend on aksi
    aksi == "1":
        print_table(monster)

    aksi == "keluar":
        break
    aksi == "2":
        new_id <- len(monster['id']) + 1

        new_type <- validate_alpha_input("Masukkan nama/type Monster: ")
        while new_type in monster["type"] do
            output("Nama/Type Monster sudah ada di database! Silahkan
masukkan monster type yang berbeda.")
            new_type <- validate_alpha_input("Enter new monster type: ")

        new_atk_power <- validate_integer_input("Masukkan attack power: ")
        new_def_power <- validate_integer_input_2(50, "Masukkan defend power
(0-50): ")
        new_hp <- validate_integer_input("Masukkan HP Monster: ")
        output("Nama/type Monster : " + str({new_type}))
        output("Attack Power : " + str({new_atk_power}))
        output("Defense Power : " + str({new_def_power}))
        output("HP Monster : " + str({new_hp}))

        input(confirmation)

```

```

depend on confirmation

confirmation == "Y":
    new_monster <- {
        'id': new_id,
        'type': new_type,
        'atk_power': new_atk_power,
        'def_power': new_def_power,
        'hp': new_hp
    }

    key traversal [0..new_monster]
        monster[key].append(new_monster[key])
        output("New monster added successfully!")
        print_table(monster)

confirmation == "N":
    output("Monster tidak ditambahkan ke database.")

```

15. F14 - Load

Function load () -> array of any

{menerima input folder dari pengguna dan mengakses file csv dan mengubah ke dalam bentuk array}

Kamus lokal

```
a : integer  
b : integer  
c : integer  
d : integer  
e : integer  
f : integer  
folder : string  
type user_csv : < id:string,  
                  username : string,  
                  password : string,  
                  role : string  
                  oc : string >  
type u : < arid : array[0..a-1] of id,  
                  arusername : array[0..a-1] of username  
                  arpassword : array[0..a-1] of password  
                  arrole : array[0..a-1] of role  
                  aroc : array[0..a-1] of oc  
user : SEQFILE OF  
       (*)rekuser : user_csv  
       (1)<"", "", "", "", "", "">  
array_user : array[0] of u  
type monster_csv < id:stringe,  
                  type : string,
```

```

    atk_power : string

    def_power : string

    hp : string >

type m : < arid : array[0..b-1] of id,
        artype : array[0..b-1] of type
        aratk : array[0..b-1] of atk_power
        ardef : array[0..b-1] of def_power
        arhp : array[0..b-1] of hp

array_monster : array[0] of m

monster : SEQFILE OF

    (*)rekmonster : monster_csv

    (1)<"", "", "", "", "">

type monster_shop_csv < monsterid:stringe,
        stock : string,
        price : string >

type ms : < armonsterid : array[0..c-1] of monsterid,
        arstock : array[0..c-1] of stock
        arprice : array[0..c-1] of price >

array_monster_shop : array[0] of ms

monster_shop : SEQFILE OF

    (*)rekmonster_shop : user_csv

    (1)<"", "", "">

type monster_inventory_csv < user_id:char,
        monster_id : char,

```

```

                price : char >

type mi : < aruser_id : array[0..d-1] of user_id,
          armonster_id : array[0..d-1] monster_id
          arprice : array[0..d-1] of price >

array_monster_inventory : array[0] of mi

monster_inventory : SEQFILE OF

(*) rekmonster_inventory : monster_inventory_csv

(1)<"","","">

type item_inventory_csv < user_id : char,
          type : string,
          quantity : char >

type ii : < aruser_rid : array[0..e-1] of user_id,
          artype : array[0..e-1] of type
          arquantity : array[0..e-1] of quantity >

array_item_inventory : array[0] of ii

item_inventory : SEQFILE OF

(*) rekitem_inventory : item_inventory_csv

(1)<"","","">

type item_shop_csv < type : stringe,
          stock : string,
          price : string >

type is : < artype : array[0..f-1] of type,
          arstock : array[0..f-1] of stock
          arprice : array[0..f-1] of price >

```

```

array_item_shop : array[0] of is

monster_inventory : SEQFILE OF

(*)rekitem_shop : item_shop_csv

(1)< "", "", "">

```

Algoritma

```

open(user,rekuser)

Open(monster,rekmonster)

Open(monster_shop,rekmonster_shop)

Open(monster_inventory,rekmonster_inventory)

Open(item_shop,rekitem_shop)

Open(item_inventory,rekitem_inventory)

a <- 0

b <- 0

c <- 0

d <- 0

e <- 0

f <- 0

while rekuser.id != "" do

    a <- a + 1

    read(user,rekuser)

while rekmonster.id != "" do

    b <- b + 1

    read(monster,rekmonster)

while rekmonster_shop.monsterid != "" do

```

```

c <- c + 1

read(monster_shop,rekmonster_shop)

while rekmonster_inventory.user_id != "" do

d <- d + 1

read(monster_inventory,rekmonster_inventory)

while rekitem_inventory.user_id != "" do

e <- e + 1

read(item_inventory,rekitem_inventory)

while reitem_shop.type != "" do

f <- f + 1

read(item_shop,rekitem_shop)

i tranversal [0..a-1]

read(user,rekuser)

array_user[0].arid[i] <- rekuser.id

array_user[0].arusername[i] <- rekuser.username

array_user[0].arpassword[i] <- rekuser.password

array_user[0].arrole[i] <- rekuser.role

array_user[0].aroc[i] <- rekuser.oc

i tranversal [0..b-1]

read(monster,rekmonster)

array_monster[0].arid[i] <- rekmonster.id

array_monster[0].artype[i] <- rekmonster.type

array_monster[0].aratk[i] <- rekmonster.atk_power

array_monster[0].ardef[i] <- rekuser.def_power

```

```

array_monster[0].arhp[i] <- rekmonster.hp

i tranversal [0..c]

    read(monster_shop,rekmonster_shop)

array_monster_shop[0].armonster[i] <- rekmonster_shop.monster_id

array_monster_shop[0].arstock[i] <- rekmonster_shop.stock

array_monster_shop[0].arprice[i] <- rekmonster_shop.price

i tranversal [0..d-1]

    read(monsterinventory,rekmonster_inventory)

array_monster_shop[0].aruser_id[i] <- rekmonster_inventory.user_id

array_monster_shop[0].armonster_id[i] <-
rekmonster_inventory.monster_id

array_monster_inventory[0].arlevel[i] <- rekmonster_inventory.level

i tranversal [0..f-1]

    read(item_shop,rekitem_shop)

array_item_shop[0].artype[i] <- rekitem_shop.type

array_item_shop[0].arstock[i] <- rekitem_shop.stock

array_item_shop[0].arprice[i] <- rekitem_shop.price

i tranversal [0..e-1]

write(item_inventory,<array_item_inventory[0].aruser_id[i],array_item_i
nventory[0].artype[i],array_item_inventory[0].arquantity[i]>)

close(user)

close(monster)

close(monster_inventory)

```

```
close(monster_shop)  
close(item_inventory)  
close(item_shop)
```

16. F15 - Save

```
Procedure save(array_user:array of any, array_monster:array of any,  
array_monster_inventory : array of any, array_monster_shop: array_item_inventory :  
array of any, array_item_shop : array of any)  
  
{menerima input folder dari pengguna dan mengakses file csv dan mengubah ke dalam  
bentuk array}
```

Kamus lokal

```
a : integer  
b : integer  
c : integer  
d : integer  
e : integer  
f : integer  
type user_csv : < id:string,  
                  username : string,  
                  password : string,  
                  role : string  
                  oc : string >  
type u : < arid : array[0..a-1] of id,  
                  arusername : array[0..a-1] of username  
                  arpassword : array[0..a-1] of password  
                  arrole : array[0..a-1] of role  
                  aroc : array[0..a-1] of oc  
  
user : SEQFILE OF  
       (* )rekuser : user_csv  
       (1)<"", "", "", "", "">  
  
array_user : array[0] of u  
  
type monster_csv < id:stringe,  
                  type : string,  
                  atk_power : string
```

```

        def_power : string

        hp : string >

type m : < arid : array[0..b-1] of id,
          artype : array[0..b-1] of type
          aratk : array[0..b-1] of atk_power
          ardef : array[0..b-1] of def_power
          arhp : array[0..b-1] of hp

array_monster : array[0] of m

monster : SEQFILE OF

(*)rekmonster : monster_csv

(1)<"", "", "", "", "">

type monster_shop_csv < monsterid:stringe,
          stock : string,
          price : string >

type ms : < armonsterid : array[0..c-1] of monsterid,
          arstock : array[0..c-1] of stock
          arprice : array[0..c-1] of price >

array_monster_shop : array[0] of ms

monster_shop : SEQFILE OF

(*)rekmonster_shop : user_csv

(1)<"", "", "">

type monster_inventory_csv < user_id:char,
          monster_id : char,
          price : char >

```

```

type mi : < aruser_id : array[0..d-1] of user_id,
          armonster_id : array[0..d-1] monster_id
          arprice : array[0..d-1] of price >

array_monster_inventory : array[0] of mi

monster_inventory : SEQFILE OF

(*)rekmonster_inventory : monster_inventory_csv

(1)<"", "", "">

type item_inventory_csv < user_id : char,
          type : string,
          quantity : char >

type ii : < aruser_rid : array[0..e-1] of user_id,
          artype : array[0..e-1] of type
          arquantity : array[0..e-1] of quantity >

array_item_inventory : array[0] of ii

item_inventory : SEQFILE OF

(*)rekitem_inventory : item_inventory_csv

(1)<"", "", "">

type item_shop_csv < type : stringe,
          stock : string,
          price : string >

type is : < artype : array[0..f-1] of type,
          arstock : array[0..f-1] of stock
          arprice : array[0..f-1] of price >

array_item_shop : array[0] of is

```

```
monster_inventory : SEQFILE OF  
    (*) rekitem_shop : item_shop_csv  
    (1)<"", "", "">
```

Algoritma

```
rewrite(user)  
  
rewrite(monster)  
  
rewrite(monster_shop)  
  
rewrite(monster_inventory)  
  
rewrite(item_shop)  
  
rewrite(item_inventory)  
  
a <- 0  
  
b <- 0  
  
c <- 0  
  
d <- 0  
  
e <- 0  
  
f <- 0  
  
i tranversal [array_user[0].arid]  
    a <- a+1  
  
i tranversal [array_monster[0].arid]  
    b <- b+1  
  
i tranversal [array_monster_shop[0].armonster_id]  
    c <- c+1  
  
i tranversal [array_monster_inventory[0].aruser_id]  
    d <- d+1
```

```

i tranversal [array_item_inventory[0].aruser_id

e <- e+1

i tranversal [array_item_shop[0].artype

f <- f+1

i tranversal [0..a-1]

write(user,<array_user[0].arid[i],array_user[0].arusername
e[i],array_user[0].arpassword[i],array_user[0].arrole[i],array_user[0].
aroc[i]>

i tranversal [0..b-1]

write(monster,<array_monster[0].arid[i],array_monster[0].artype[i],arra
y_monster[0].aratk[i],array_monster[0].ardef[i],array_monster[0].arhp[i]
]>

i tranversal [0..c-1]

write(monster_shop,<array_monster_shop[0].armonster[i],array_monster_sh
op[0].arstock[i],array_monster_shop[0].arprice[i]>

i tranversal [0..d-1]

write(monster_inventory,<array_monster_shop[0].aruser_id[i],array_monst
er_shop[0].armonster_id[i],array_monster_inventory[0].arlevel[i]>

i tranversal [0..f-1]

write(item_shop,<array_item_shop[0].artype[i],array_item_shop[0].arstoc
k[i],array_item_shop[0].arprice[i]>

i tranversal [0..f-1]

read(item_inventory,rekitem_inventory)

array_item_inventory[0].aruser_id[i] <- rekitem_inventory.user_id

array_item_inventory[0].artype[i] <- rekitem_inventory.type

array_item_inventory[0].arquantity[i] <- rekitem_inventory.

```

17. F16 - Exit

```
Procedure exit (array_user:array of any, array_monster:array of any,  
array_monster_inventory : array of any, array_monster_shop: array_item_inventory :  
array of any, array_item_shop : array of any)
```

Kamus lokal

a : string

Algoritma

Input(a)

While a != 'Y' or a!='y' or a!='N' or a!='n' do

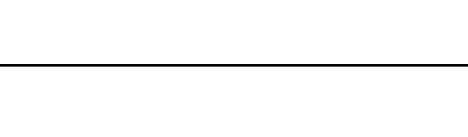
 Input(a)

 If a='Y' or a='y' then

 Save((array_user, array_monster,
array_monster_inventory, array_monster_shop, array_item_inventory, array_item_shop))

Lampiran Hasil Pengujian Program

Fitur	Hasil Pengujian
F01 - Register	<pre>>>> register Silahkan Masukkan Username: user1 Silahkan Masukkan Password: pass1 1. Pikachuow 2. Bulbu 3. Zeze 4. Zuko 5. Chacha 6. Chadrizzard 7. Squirtlo Monster Pilihanmu: 1 Selamat datang Agent user1. Bersama Pikachuow, ayo kita lawan musuh musuh itu!</pre>
F02 - Login	<pre>>>> login Username: user1 Password: pass1 Selamat datang, Agent user1! ===== KOTA MONSTERTOPIA ===== ----- Selamat datang di kota MonsterTopia ----- ~~ mulai petualangan dan kalahkan musuh musuh ~~ ----- ----- (Ketik HELP untuk mendapat bantuan)</pre>
F03 - Logout	<pre>>>> logout Logout Berhasil. Anda telah Logout dari akun</pre>
F04 - Menu & Help	<pre>>>> help ===== HELP ===== Login belum dilakukan,silahkan login terlebih dahulu. 1. Login: Masuk ke dalam akun yang sudah terdaftar 2. Register: Membuat akun baru Footnote: Silahkan masukan fungsi yang terdaftar dalam menggunakan aplikasi Pastikan input yang dimasukkan valid</pre>

	<pre>>>> help ===== ===== HELP ===== Halo Agent Agen_P. Kamu memanggil command HELP. Kamu memilih jalan yang benar, semoga kamu tidak sesat kemudian . Berikut adalah hal-hal yang dapat kamu lakukan sekarang: 1. Logout: Keluar dari akun yang sedang digunakan 2. Inventory: Melihat owca-dex yang dimiliki oleh Agent 3. Shop: Membeli potion atau monster 4. Battle: Bertarung melawan monster! 5. Arena: Latih dan tingkatkan kemampuan agen dan para monstermu! 6. Laboratory: Upgrade monster yang kamu miliki 7. Save: Menyimpan progress Anda 8. Exit: Keluar dari game Footnote: Silahkan masukan fungsi yang terdaftar dalam menggunakan aplikasi Pastikan input yang dimasukkan valid</pre> <pre>>>> help ===== ===== HELP ===== Selamat datang, Admin. Berikut adalah hal- hal yang dapat kamu lakukan: 1. Logout: Keluar dari akun yang sedang digunakan 2. Shop Management: Melakukan manajemen pada SHOP sebagai tempat jual beli peralatan Agent 3. Monster Management: Melakukan manajemen Monster 4. Save: Menyimpan progress Anda 5. Exit: Keluar dari game Footnote: Silahkan masukan fungsi yang terdaftar dalam menggunakan aplikasi Pastikan input yang dimasukkan valid</pre>
F05 - Monster	
F06 - Potion	
F07 - Inventory	<pre>>>> inventory</pre>  <pre>you have 0 coins 1. Monster (Name: Bulbu, Lvl: 2, HP: 1144.5) 2. Monster (Name: Zeze, Lvl: 1, HP: 100.0) Index: 1 Monster Name : Bulbu ATK Power : 55.00000000000001 DEF Power : 55.00000000000001 HP : 1144.5 Level : 2</pre>

F08 - Battle

```
BATTLE !  
  
/----/\_  
| | o o | /| | |
| | .vvvv.| /|  
| | ^^^^^^ | /|  
| /| | | | /|  
| \ | | | | /|  
| \ | | | | /|  
| \ | | | | /|  
| \ | | | | /|  
  
Zeze liar muncul!  
  
Name: Zeze  
ATK Power : 300  
DEF Power : 10  
HP : 100  
Level : 1  
  
==== MONSTER LIST ====  
  
/----/\_  
| | o o | /| | |
| | .vvvv.| /|  
| | ^^^^^^ | /|  
| /| | | | /|  
| \ | | | | /|  
| \ | | | | /|  
| \ | | | | /|  
| \ | | | | /|  
  
Agent chose Bulbu  
  
Name: Bulbu  
ATK Power : 55.00000000000001  
DEF Power : 55.00000000000001  
HP : 1320.0  
Level : 2  
  
==== TURN 1 Bulbu ====  
1. Attack  
2. Use potion  
3. Quit  
Pilih Perintah: 1  
Bulbu attacks Zuko and deals 49.5 damage!  
Name: Zuko  
ATK Power : 100  
DEF Power : 25  
HP : 750.5  
Level : 1
```

F09 - Arena

```
>>> arena  
  
ARENA  
  
==== MONSTER LIST ====  
1. Bulbu  
2. Zeze  
  
Pilih Monster: 1  
  
/----/\_  
| | o o | /| | |
| | .vvvv.| /|  
| | ^^^^^^ | /|  
| /| | | | /|  
| \ | | | | /|  
| \ | | | | /|  
| \ | | | | /|  
  
Agen memilih Bulbu  
  
Name: Bulbu  
ATK Power : 55.00000000000001  
DEF Power : 55.00000000000001  
HP : 1320.0  
  
==== TURN 59 Pikachuow ====  
Pikachow attacks Bulbu and deals 70.875 damage!  
Name: Bulbu  
ATK Power : 55.00000000000001  
DEF Power : 55.00000000000001  
HP : 0  
Level : 2  
  
Pikachow wins!  
Name: Pikachuow  
ATK Power : 175.0  
DEF Power : 14.0  
HP : 154.1499999999998  
Level : 5  
  
==== STATS ====  
Total hadiah : 120 OC  
Jumlah stage : 4  
Damage diberikan : 2564.292500000002  
Damage diterima : 4009.950000000003
```

F10 - Shop &
Currency

```
>>> shop

SHOP

Irasshaimase! Selamat datang di SHOP!!

>>> Pilih aksi (lihat/beli/keluar):
lihat
>>> Mau lihat apa? (monster/potion):
monster
-----  

ID | Type      | ATK Power | DEF Power | HP    | Stock | Price
-----  

1  | Pikachuow | 125       | 10        | 600   | 10    | 500  

2  | Bulbu     | 50        | 50        | 1200  | 4     | 700  

3  | Zeze      | 300       | 10        | 100   | 3     | 1000  

4  | Zuko      | 100       | 25        | 800   | 8     | 550  

5  | Chacha    | 80        | 30        | 700   | 7     | 600
>>> Pilih aksi (lihat/beli/keluar):
lihat
>>> Mau lihat apa? (monster/potion):
potion
Type      | Stock | Price
-----  

strength | 10    | 50
resilience | 5    | 30
>>> Pilih aksi (lihat/beli/keluar):
```

F11 - Laboratory

```
>>> laboratory

LABORATORY

Selamat datang di Lab Dokter Asep !!!
===== MONSTER LIST =====
1. Bulbu (Level: 2)
2. Zeze (Level: 1)
===== UPGRADE PRICE =====
Level 1 -> Level 2 : 300 OC
Level 2 -> Level 3 : 500 OC
Level 3 -> Level 4 : 800 OC
Level 4 -> Level 5 : 1000 OC
>>> Pilih monster:
Masukkan nomor monster: 1
Bulbu akan di-upgrade ke level 3.
Harga untuk melakukan upgrade Bulbu adalah 500 OC.
>>> Lanjutkan upgrade (y/n):
y
OC tidak cukup untuk melakukan upgrade.
OC Balance: 0
===== MONSTER LIST =====
1. Bulbu (Level: 2)
2. Zeze (Level: 1)
```

F12 - Shop Management

```
>>> shop management

SHOP MANAGEMENT

Selamat datang di Management SHOP, silahkan pilih menu yang diinginkan (lihat/tambah/ubah/hapus/keluar):
>>> Pilih Aksi : lihat
Mau lihat apa? (monster/potion): monster
ID | Type | ATK Power | DEF Power | HP | Stock | Price
1  | Pikachu | 125 | 10 | 600 | 10 | 500
2  | Bulbu | 50 | 50 | 1200 | 4 | 700
3  | Zeze | 300 | 10 | 100 | 3 | 1000
4  | Zuko | 100 | 25 | 800 | 8 | 550
5  | Chacha | 80 | 30 | 700 | 7 | 600
Selamat datang di Management SHOP, silahkan pilih menu yang diinginkan (lihat/tambah/ubah/hapus/keluar):
>>> Pilih Aksi : lihat
Mau lihat apa? (monster/potion): potion
ID | Type | Stok | Price
1  | strength | 10 | 50
2  | resilience | 5 | 30
```

F13 - Monster Management	<pre>>>> monster management \ / / \ / / \ / / \ / / \ / / \ / / \ / / \ / / \ / [] [] [] [] [] [] [] [] [] [] [] [] [] [] [] [] [] \ / / \ / / \ / / \ / / \ / / \ / / \ / / \ / / \ / / \ / [] [] [] [] [] [] [] [] [] [] [] [] [] [] [] [] [] [] >>> Monster Selamat datang di database Monster, silahkan pilih menu yang diinginkan: 1. Tampilkan semua Monster 2. Tambahkan Monster baru >>> Pilih Aksi : 1 id type atk_power def_power hp 1 Pikachow 125 10 600 2 Bulbu 50 50 1200 3 Zeee 300 10 100 4 Zuko 100 25 800 5 Chacha 80 30 700 6 Chadrizzard 200 50 500 7 Squirtlo 150 35 600 Masukkan nama/type Monster: 1 Input invalid! Input berupa alphabet, silahkan coba lagi Masukkan nama/type Monster: Pikachu Nama/type Monster sudah ada di database! Silahkan masukkan monster type yang berbeda. Enter new monster type: arkeus Masukkan attack power: 350 Masukkan defend power (0-50): 50 Masukkan HP Monster: 1500 Nama/type Monster : arkeus Attack Power : 350 Defense Power : 50</pre>
F14 - Load	<pre>Selesai! \ / / \ / / \ / / \ / / \ / / \ / / \ / / \ / / \ / [] [] [] [] [] [] [] [] [] [] [] [] [] [] [] [] [] \ / / \ / / \ / / \ / / \ / / \ / / \ / / \ / / \ / / \ / [] [] [] [] [] [] [] [] [] [] [] [] [] [] [] [] [] [] selamat datang di OWCA!! >>> </pre>
F15 - Save	<pre>>>> save Masukkan nama folder:FolderFILE membuat folder data FolderFILE Saving Berhasil menyimpan data di folder FolderFILE</pre>
F16 - Exit	<pre>>>> exit Apakah Anda mau melakukan penyimpanan file yang sudah diubah? (y/n) n Data tidak disimpan Anda telah keluar dari permainan</pre>

Lampiran Form Asistensi

Form Asistensi 1

Form MoM Asistensi Tugas Besar
IFL210/Dasar Pemrograman
Sem. 2 2023/2024

Nomor Asistensi : 1
No. Kelompok/Kelas : K/K-01
Tanggal asistensi : 02 Mei 2024

Anggota kelompok

NIM / Nama (Hanya yang Hadir)	
1	16523031 / Haidar Rafli Octavio Ramadhan
2	19623131 / Michael Dimas Sarono
3	19623241 / Persada Ramiza Abiyudaya
4	16523171 / Kinan Athaya
5	16523101 / Galih Muhammad
6	16523241/Abdullah Al Ash

Asisten pembimbing

NIM / Nama

Catatan Asistensi:

Rangkuman Diskusi

1. Interface dari program,harus dalam terminal atau boleh pake grafis?
(jawaban) : dalam terminal,paling mempercantik bisa pake asi art.pake gambar gitu.
2. Minta saran kita mulai sebaiknya dari mana?.
(jawaban) : tergantung kalian bagi tugasnya gimana?. sebenarnya itu udh bagus pembagian tugasnya.Menurut aku kalau mulai dari mana nya,boleh aja sih pembagiannya sesuai dengan flow yang mirip.Bisa dimulai independen jadi bisa mulai masing masing.F00 harus jadi pertama kali,karena bakal dipake banyak di fungsi lainnya.CSV parser.
3. kita gak pake import CSV,
(jawaban) : kalian gak boleh pake import csv.kalian pakai fungsi dari python itu sendiri.Parser tadi di awal harus sudah jadi biar bisa memudahkan kalian.

SARAN:

1. kalau sudah satu fungsi kelar,langsung masukin saja ke laporan.biar di cicil,dan jangan underestimate.dan perhatikan ketentuan laporan.
2. kalau ada kendala,semuanya kalau bisa dikomunikasikan.
3. Kalian bikin workflow kayak setiap berapa hari sekali di track progress nya.

Tindak Lanjut

Dokumentasi

Form Asistensi 2

**Form MoM Asistensi Tugas Besar
IF1210/Dasar Pemrograman
Sem. 2 2023/2024**

Nomor Asistensi : 2
No. Kelompok/Kelas : K/K-01
Tanggal asistensi : 12 Mei 2024

Anggota kelompok	NIM / Nama (Hanya yang Hadir)
1	16523101 / Galih Muhammad Syah Athaya
2	16523031/ Haidar Rafli Octavio Ramadhan
3	19623131 / Michael Dimas Sarono
4	19623241 / Persada Ramiiza Abudaya
5	16523171/ Kinan Athaya
6	

Asisten pembimbing

NIM / Nama

Catatan Asistensi:

Rangkuman Diskusi
<p>1. Program yang membutuhkan data dari csv tidak perlu membaca lagi pada setiap programnya, sedangkan, gunakan dictionary yang sudah disiapkan pada program F14 (Load).</p> <p>2. jangan lupa untuk mempertimbangkan kasus salah input dan membuat program untuk memvalidasi input.</p> <p>3. import random tidak diperbolehkan dan harus digunakan random generator buatan kelompok sendiri</p>
Tindak Lanjut
Dokumentasi
