

**LAPORAN PROYEK PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN DASAR 2024**

SIMULASI MANAJEMEN KRISIS KESEHATAN DI BUMI



Oleh Kelompok 27

Anggota:

Lalu Aqsha Nayaka Maulana (F1D02310069)

Muhammad Arafa Yahfazhka (F1D02310126)

Zainul Majdi (F1D02310028)

Zamzami Satria Tegar (F1D02310029)

**PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS MATARAM
2024**

**LEMBAR PENGESAHAN LAPORAN PROYEK PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN DASAR 2024**

1. Kelompok : 27
2. Judul Proyek : Simulasi Manajemen Krisis Kesehatan di Bumi
3. Anggota Kelompok : Lalu Aqsha Nayaka Maulana (F1D02310069)
Muhammad Arafa Yahfazhka (F1D02310126)
Zainul Majdi (F1D02310028)
Zamzami Satria Tegar (F1D02310029)

Laporan proyek ini disusun sesuai dengan kaidah penyusunan yang telah ditentukan dan dibuat sebagai syarat mata kuliah Algoritma dan Pemrograman Dasar 2024.

Mataram, 15 Juni 2024

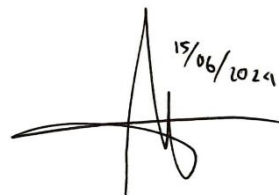
Telah diperiksa dan disahkan
oleh:

**Koordinator
Asisten**



Muhammad Kholilullah
F1D022014

**Asisten
Pembimbing**



Gusti Ayu Devi Anjani Putri
F1D022006

1.1 JUDUL

Simulasi Manajemen Krisis Kesehatan di Bumi

1.2 LATAR BELAKANG

Simulasi berbasis teks ini merupakan implementasi dari sebuah skenario di mana bumi menghadapi ancaman virus yang berbahaya dan para pasien harus dijaga kesehatannya selama periode waktu tertentu. Program ini bertujuan untuk memodelkan penyebaran virus, kondisi kesehatan pasien, serta upaya pemberian obat untuk menyelamatkan pasien yang terinfeksi virus. Dalam dunia yang semakin kompleks dan rentan terhadap krisis kesehatan, simulasi seperti ini memberikan gambaran tentang dinamika penanganan wabah serta keputusan penting yang harus diambil untuk menyelamatkan nyawa.

Simulasi ini tidak hanya berguna sebagai alat pendidikan, tetapi juga sebagai latihan praktis dalam manajemen krisis. Melalui simulasi ini, pengguna dapat mempelajari bagaimana virus menyebar, bagaimana menentukan prioritas dalam pemberian obat, dan bagaimana mengelola sumber daya yang terbatas. Selain itu, simulasi ini dapat membantu masyarakat memahami betapa pentingnya peran setiap individu dalam upaya pencegahan dan pengendalian penyakit menular.

Dalam konteks yang lebih luas, simulasi ini dapat digunakan untuk mendidik masyarakat tentang pentingnya manajemen kesehatan dalam situasi pandemi dan juga untuk melatih pengambilan keputusan dalam krisis kesehatan. Dengan demikian, masyarakat dapat lebih siap menghadapi situasi darurat kesehatan dengan pengetahuan yang lebih baik dan respons yang lebih cepat dan efektif. Hal ini juga membantu meningkatkan kesadaran tentang pentingnya tindakan preventif dan pengobatan yang tepat waktu dalam menghadapi penyakit menular.

Simulasi ini dirancang untuk memberikan pengalaman yang realistis dan mendidik, menggabungkan elemen-elemen strategi dan keputusan kritis. Dengan mensimulasikan skenario di mana pasien terinfeksi harus dirawat dan diselamatkan, pengguna dapat melihat dampak langsung dari setiap keputusan yang mereka buat. Hal ini diharapkan dapat meningkatkan kemampuan analitis dan pemecahan masalah, yang sangat penting dalam situasi krisis kesehatan.

Melalui simulasi ini, diharapkan pengguna dapat memahami lebih dalam tentang mekanisme penyebaran virus, pentingnya tindakan medis yang tepat, dan bagaimana kebijakan kesehatan dapat mempengaruhi hasil akhir dari krisis kesehatan. Dengan demikian, simulasi ini bukan hanya alat pembelajaran, tetapi juga platform untuk melatih keterampilan manajemen krisis yang esensial.

1.3 DESKRIPSI PROGRAM

Program ini adalah sebuah simulasi berbasis teks yang memodelkan skenario penyebaran virus di bumi dan upaya untuk menjaga kesehatan pasien selama sepuluh hari. Setiap hari dalam simulasi, kondisi kesehatan pasien diperbarui dan pemain harus mengambil keputusan yang tepat untuk menjaga pasien tetap hidup.

Pada hari pertama, semua pasien memiliki *health point* (HP) penuh yakni 100 dan tidak ada ancaman virus dalam tubuh pasien. Namun, mulai hari kedua, virus mulai menyebar dan mengacak nilai HP pasien, membuat kondisi mereka semakin buruk jika tidak segera diberikan obat. Pemain harus memprioritaskan pasien yang akan diberikan obat setiap harinya, karena obat hanya dapat digunakan satu kali per hari. Setiap keputusan yang diambil akan mempengaruhi kelangsungan hidup pasien dan kondisi keseluruhan bumi.

Program ini dirancang untuk menghadirkan tantangan yang realistis dan menguji kemampuan pemain dalam mengambil keputusan yang cepat dan tepat. Dengan setiap hari yang berlalu, HP pasien akan berkurang sebesar 20, dan jika HP mereka mencapai nol, pasien tersebut akan mati. Hal ini menambahkan elemen urgensi dan tekanan bagi pemain untuk selalu waspada dan strategis dalam pengelolaan kesehatan pasien.

Selain itu, program ini menampilkan kondisi bumi setiap harinya, memberikan informasi tentang status virus dan jumlah pasien yang masih hidup. Jika semua pasien mati sebelum hari ketujuh, permainan berakhir dengan kegagalan. Namun, jika pemain berhasil menjaga setidaknya satu pasien tetap hidup hingga hari ketujuh, permainan berakhir dengan keberhasilan. Ini menciptakan tujuan yang jelas dan menantang bagi pemain untuk dicapai.

Secara keseluruhan, program ini menawarkan pengalaman simulasi yang mendidik dan interaktif, membantu pemain memahami kompleksitas manajemen krisis kesehatan dan pentingnya pengambilan keputusan yang tepat waktu. Dengan menggabungkan elemen-elemen edukatif dan strategis, simulasi ini diharapkan dapat meningkatkan kesadaran dan pengetahuan tentang penyebaran penyakit menular serta upaya pengendaliannya.

1.4 ALGORITMA

1. Program mulai.
2. Program akan menampilkan krisis di bumi, ikon bumi serta pesan karakteristik bumi yang begitu damai.
3. Program akan menampilkan hari ke-1 dan menu utama yakni lihat kondisi pasien, hari selanjutnya serta menu berikan obat.
4. Ketika pengguna meng-*input* menu 1, maka program akan menampilkan status kondisi pasien pada hari ke-1. Selain itu, program akan menampilkan kondisi *health point* dari tubuh pasien dan menampilkan pesan bumi baik baik saja.
5. Ketika pengguna kembali ke halaman menu utama, program akan memasuki hari ke-2. Kemudian akan menampilkan sebuah halaman menu utama.
6. Apabila pengguna meng-*input*-kan menu berikan obat maka program akan memerintahkan kepada pengguna untuk memilih pasien yang akan di berikan obat.
7. Setelah pengguna memilih Pasien yang diberikan obat tersebut maka program akan menampilkan status pasien (nama pasien) yang dimana sebelumnya telah diberikan obat oleh dokter serta *health point* tersebut meningkat.
8. Ketika pengguna mengecek menu kondisi pasien pada hari ke-2, maka program akan menampilkan status pasien dan *health point*-nya serta program akan menampilkan pesan bahwa bumi sudah terkena virus.
9. Ketika pengguna kembali ke menu utama program. Kemudian, program akan meminta pengguna untuk meng-*input*-kan menu ke-2. Sehingga, program tersebut akan memasuki hari ke-3 yang dimana pada hari ini program akan menampilkan halaman menu utama.
10. Apabila pengguna meng-*input*-kan menu berikan obat maka program akan memerintahkan kepada pengguna untuk memilih pasien yang akan di berikan obat.
11. Setelah pengguna memilih Pasien yang diberikan obat tersebut maka program akan menampilkan status pasien (nama pasien) yang dimana sebelumnya telah diberikan obat oleh dokter serta *health point* tersebut meningkat dan kembali ke menu utama.
12. Ketika pengguna mengecek menu kondisi pasien pada hari ke-3, maka program akan menampilkan status pasien dan *health point*-nya serta program akan menampilkan pesan bahwa bumi sudah terkena virus.
13. Ketika pengguna kembali ke menu utama program. Kemudian, program akan meminta pengguna untuk meng-*input*-kan menu ke-2. Sehingga, program akan menampilkan

pesan (nama pasien) telah mati dan program tersebut akan memasuki hari ke-4 yang dimana pada hari ini program akan menampilkan halaman menu utama.

14. Apabila pengguna meng-*input*-kan menu berikan obat maka program akan memerintahkan kepada pengguna untuk memilih pasien yang akan di berikan obat.
15. Setelah pengguna memilih pasien yang diberikan obat tersebut maka program akan menampilkan status pasien (nama pasien) yang dimana sebelumnya telah diberikan obat oleh dokter serta *health point* tersebut meningkat dan kembali ke menu utama.
16. Ketika pengguna mengecek menu status pasien pada hari ke-4, maka program akan menampilkan status pasien serta *health point*-nya. Setelah itu, program akan menampilkan pesan kepada pengguna untuk cepat menyembuhkan bumi dari virus tersebut.
17. Ketika pengguna kembali ke menu utama program. Kemudian, program akan meminta pengguna untuk meng-*input*-kan menu ke-2. Sehingga, program tersebut akan memasuki hari ke-5 yang dimana pada hari ini program akan menampilkan halaman menu utama.
18. Apabila pengguna meng-*input*-kan menu berikan obat maka program akan memerintahkan kepada pengguna untuk memilih pasien yang akan di berikan obat.
19. Setelah pengguna memilih pasien yang diberikan obat tersebut maka program akan menampilkan status pasien (nama pasien) yang dimana sebelumnya telah diberikan obat oleh dokter serta *health point* tersebut meningkat dan kembali ke menu utama.
20. Ketika pengguna mengecek menu status pasien pada hari ke-5, maka program akan menampilkan status pasien serta *health point*-nya. Setelah itu, program akan menampilkan pesan kepada pengguna untuk cepat menyembuhkan bumi dari virus tersebut.
21. Ketika pengguna bisa meminimalisir menyelamatkan 2 orang pasien, maka akan program akan menampilkan sebuah apresiasi “Selamat! Anda Menang” kepada pengguna.
22. Program selesai.

1.5 PENJELASAN CODE

1. Pendeklarasian Variabel

```
const int MAX_PASIEN = 5;
const int MAX_HP = 100;

struct Entity {
    bool pasien;
    int day;
    bool virus;
    int* hp;
    int num_pasien;
    bool obat_used;
    string names[MAX_PASIEN];
};
Entity earth;
```

Script diatas merupakan pendeklarasian beberapa variabel dan juga struktur yang penting dalam penggunaan program yang akan dipanggil di *line code* selanjutnya seperti struktur “Entity” memiliki fungsi yang spesifik dalam permainan. Variabel global “const int MAX_PASIEN” dan “const int MAX_HP” menentukan jumlah maksimal pasien dan HP maksimum masing-masing, sementara “Entity earth” adalah struktur yang menyimpan semua data dan status terkait bumi dan pasien.

Variabel dalam Entity seperti “bool pasien”, “int day”, “bool virus”, “int* hp”, “int num_pasien”, “bool obat_used”, dan “string names[MAX_PASIEN]” digunakan untuk melacak status umum pasien, jumlah hari, keberadaan virus, kesehatan setiap pasien, jumlah pasien yang hidup, penggunaan obat, dan nama-nama pasien.

2. Main Menu

```
int main() {
    int choose; earth.day = 0;
    earth.pasien = true; earth.virus = false;

    day_one();
    system("cls");
    display();
    cin.get();
    system("cls");
    narasi();
    cin.get();
    system("cls");
    while (earth.day <= 10) {
        cout << "======" << endl;
        cout << "    Hari ke-" << earth.day << endl;
        cout << "======" << endl;
        cout << "1. Lihat kondisi pasien" << endl;
        cout << "2. Hari selanjutnya" << endl;
        cout << "3. Berikan obat" << endl;
        cout << "Masukkan Pilihan: ";
        cin >> choose;
        system("cls");
        switch (choose) {
```

```

        case 1:
            tampilkanKondisiBumi();
            pause();
            system("cls");
            break;
        case 2:
            day();
            break;
        case 3:
            berikanObat();
            pause();
            system("cls");
            break;
        default:
            cout << "Pilihan tidak valid. Silakan pilih lagi." << endl;
            break;
    }
    if (earth.day >= 10) {
        if (checkWinCondition()) {
            cout << "Selamat! anda menang karena telah
menyelamatkan minimal 2 pasien" << endl;
        }else{
            cout << "Semua Pasien mati anda kalah!" << endl;
        }
        break;
    }
    if (checklosecondition()){
        cout<<"Anda Kalah! Karena anda tidak bisa menyelamatkan
minimal 2 pasien"<<endl;
        pause();
        exit(0);
    }
    if (earth.num_pasien == 0) {
        cout << "Semua pasien telah mati. Permainan selesai."
<< endl;
        break;
    } }
    deallocateMemory();
    return 0;}

```

Script di atas merupakan sebuah pengimplementasian dari menu utama *game* yang dibuat. Pada awalnya terdapat pendeklarasian berupa "int choose", "earth.day = 0", "earth.pasien = true" dan "earth.virus = false". Lalu terdapat beberapa *input function* yang dimasukkan ke dalam *code* yang akan digunakan sebagai jalannya program.

Dalam fungsi "main()", variabel "choose" digunakan untuk menyimpan pilihan input dari pemain dalam menu utama permainan. Program dimulai dengan inisialisasi hari pertama menggunakan "day_one()", diikuti oleh tampilan layar pembuka dan narasi.

Loop utama permainan berjalan hingga hari ke-10, menampilkan menu pilihan kepada pemain, dan mengolah input pemain untuk melihat kondisi pasien, melanjutkan ke hari berikutnya, atau memberikan obat. Program juga mengecek kondisi kemenangan

setelah 10 hari menggunakan kondisi *if* “`earth.day >= 10`” dan kondisi kekalahan setiap hari, serta menghentikan permainan jika semua pasien mati. Terakhir, memori yang dialokasikan dibebaskan menggunakan “`deallocateMemory()`”

3. Game

```
void display() {
    cout << "
    cout << " | / / | \ \ | / / | \ \ | / / | \ \ | \n";
    cout << " | / / | / | \ \ | \ \ | \ \ | \ \ | \ \ | \n";
    cout << " | \ \ | / | \ \ | \ \ | \ \ | \ \ | \ \ | \n";
    cout << " | \ \ | \ \ | \ \ | \ \ | \ \ | \ \ | \ \ | \n";
    cout << " | \ \ | \ \ | \ \ | \ \ | \ \ | \ \ | \ \ | \n";
    cout << "
    cout << " | \ \ | \ \ | \ \ | \ \ | \ \ | \ \ | \ \ | \n";
    cout << " | \ \ | \ \ | \ \ | \ \ | \ \ | \ \ | \ \ | \n";
    cout << " | \ \ | \ \ | \ \ | \ \ | \ \ | \ \ | \ \ | \n";
    cout << " | \ \ | \ \ | \ \ | \ \ | \ \ | \ \ | \ \ | \n";
    cout << " | \ \ | \ \ | \ \ | \ \ | \ \ | \ \ | \ \ | \n";
    cout << endl;
    cout << "Press any key to continue....." <<
endl;
}
```

Script di atas merupakan tampilan untuk awal program dimulai sebagai bentuk judul yang akan diangkat berdasarkan program yang telah dibuat dimana dalam “`void display()`” ini terdapat suatu *output* yang berbentuk sebuah seni kata yang disusun menggunakan ASCII untuk menampilkan sebuah judul yang tidak membosankan dengan *style font* judul dari sebuah video game DOOM.

```
void narasi() {
    cout << "===== \n";
    cout << "- - - - - \n"
    cout << " - - - - - \n"
    cout << " - - - - - \n"
    cout << " - - - - - \n"
    cout << " - - - - - \n"
    cout << " - - - - - \n"
    cout << " - - - - - \n"
    cout << " - - - - - \n"
    cout << " - - - - - \n"
    cout << "===== \n";
    cout << "Bumi yang dulu damai dan harmonis kini terguncang oleh serangan \n"
    cout << "virus mematikan. Dalam kepanikan global ini, Anda, seorang \n"
    cout << "perawat berbakat, diberi tugas krusial: merawat lima pasien \n"
    cout << "penting yang memegang kunci untuk menyelamatkan dunia. \n"
    cout << " \n"
    cout << "1. Dr. Majdi - ilmuwan yang sedang mengembangkan vaksin. \n"
    cout << "Tanpa perawatannya, riset vaksin bisa terhenti. \n"
```

```

"2. Jendral Tegar - pemimpin militer yang
mengoordinasikan\n"
"    upaya pengendalian virus. Kehadirannya penting
untuk\n"
"    menjaga ketertiban dan keamanan.\n"
arahan\n"
"    dan harapan kepada rakyat. Stabilitas
pemerintahan\n"
"    bergantung padanya.\n"
"4. Mentri Faza - seorang menteri yang berpegang
teguh arahan\n"
"    Presiden jika jabatan tersebut diserahkan ke
dirinya.\n"
"5. Prof. Ayu - penemu yang membantu Dr. Majdi dalam
mencari\n"
"    obat vaksin yang berguna sebagai penyelamat bumi
ini.\n"
"\n"
"Tugas Anda adalah memastikan 5 pasien tersebut
tetap sehat\n"
"dan mampu melanjutkan misi mereka.>";

```

Script diatas merupakan *function* yang berguna untuk memberi latar belakang terhadap narasi program yang akan disampaikan guna mendapat pandangan yang dapat mempermudah *user* dalam mengerti alur cerita yang akan disampaikan oleh program.

```

void pause() {
    cin.get();
    cin.fail();
    cin.ignore();
}

```

Bagian ini merupakan bentuk *function* yang akan mengantisipasi jika terdapat kesalahan dalam peng-*input*-an yang dilakukan oleh *user* yang mana jika terdapat kesalahan maka program tidak mengeluarkan *output error* melainkan akan terus mengulang hingga *user* memasukkan *input* yang tepat.

```

void allocateMemory()
{
    earth.hp = new int[MAX_PASIEN];
}
void deallocateMemory() {
    delete[] earth.hp;
}

```

Bagian ini digunakan sebagai pengalokasian terhadap memori *integer* dari sebuah array "hp" yang terdapat pada struktur "Entity". *Pointer* dari "hp" pertama kali akan di *set* sebagai bentuk NULL. *Function* dari "allocateMemory" berguna untuk mengalokasi ukuran memori dari "MAX_PASIEN" dan akan mengantarkannya terhadap variabel "hp" hal ini berguna untuk mengatasi jumlah pasien dan poin darah yang

mereka milikk secara efisien. Lalu untuk *function* “*deallocateMemory*” digunakan untuk menghapus memori yang telah di inisialisasikan menggunakan operator “*delete[]*” yang akan dipanggil di akhir *game*.

```
void randomizeHP() {
    int base_hp[MAX_PASIEN] = {70, 50, 30, 30, 35};
    for (int i = 0; i < MAX_PASIEN; i++) {
        int rand_index = (i + 1) % MAX_PASIEN;
        swap(base_hp[i], base_hp[rand_index]);
    }
    for (int i = 0; i < MAX_PASIEN; i++) {
        earth.hp[i] = base_hp[i];
    }
}
```

Script di atas merupakan sebuah *function* yang akan memberi suatu nilai acak terhadap *array* dari HP yang ditetapkan dengan syarat sebagai “70, 50, 30, 30, 35”. Terdapat sebuah perulangan untuk *array* dari “*base_hp*” dengan cara menggunakan rumus “ $(i + 1) \% \text{MAX_PASIEN}$ ” ini digunakan agar indeks tidak dipilih dua kali sebagai nilai HP yang akan diterima oleh pasien. Lalu untuk fungsi dari “*swap*” yaitu untuk menukar elemen “*base_hp*” saat ini dengan elemen acak dari indeks yang akan memungkinkan agar nilai yang diterima pasien akan acak dari syarat yang telah di set di dalam *array*. Terdapat perulangan lagi yang mana nilai dari “*base_hp*” diberikan ke variabel “*earth.hp*”.

```
void tampilkanPasien() {
    if (earth.num_pasien == 0) {
        cout << "Semua pasien telah mati." << endl;
        return;
    }
    cout<<"======"<<endl;
    cout << "Status pasien hari ke-" << earth.day << ":" << endl;
    cout<<"======"<<endl;
    for (int i = 0; i < earth.num_pasien; i++) {
        for (int i = 0; i < earth.num_pasien - 1; i++) {
            for (int j = 0; j < earth.num_pasien - i - 1; j++) {
                if (earth.hp[j] > earth.hp[j + 1]) {
                    swap(earth.hp[j], earth.hp[j + 1]);
                    swap(earth.names[j], earth.names[j + 1]);
                }
            }
        }
    }
}
```

```

    }

    cout << "Pasien " << earth.names[i] << ": HP = " <<
earth.hp[i] << endl;

    }

}

```

Script “void tampilkanPasien()” merupakan sebuah perintah untuk menampilkan kondisi HP dari setiap pasien yang masih hidup pada hari tertentu dalam simulasi. Jika semua pasien telah mati, fungsi akan menampilkan pesan yang sesuai dan keluar. Jika ada pasien yang masih hidup, fungsi akan mencetak status HP dari setiap pasien tersebut. Terdapat dua iterasi perulangan yang digunakan sebagai proses mengurutkan nama pasien yang memiliki darah yang lebih sedikit dibandingkan pasien lainnya.

```

void tampilkanKondisiBumi() {
    if (earth.num_pasien == 0) {
        cout << "Semua pasien telah mati." << endl;
        return;
    }
    tampilkanPasien();
    if (earth.day == 1) {
        cout<<"===== "<<endl;
        cout<< " Bumi Baik Baik Saja " << endl;
        cout<<"===== "<<endl;
    } else if (earth.day == 2 || earth.day == 3) {
        cout<<"===== "<<endl;
        cout << " Bumi Terkena Virus " << endl;
        cout<<"===== "<<endl;
    } else if (earth.day >= 4) {
        cout<<"===== "<<endl;
        cout << "Cepat Sembuhkan Bumi" << endl;
        cout<<"===== "<<endl;
    }
}

```

Dalam *function* “tampilkanKondisiBumi()” digunakan sebagai pemberi tanda bahwa keadaan bumi yang telah terkena oleh virus. “earth.num_pasien == 0” yang berarti jika pasien tidak ada maka terdapat indikasi pesan bahwa semua pasien mati, lalu untuk indikasi “earth.day” yaitu jika terdapat syarat lebih dari beberapa hari maka terdapat *output* yang memberi tahu tentang kondisi bumi saat ini yang akan memberi gambaran untuk *user* agar segera membantu pasien yang terkena virus agar dapat sembuh dan melawan virus yang mneyerang.

```

void kondisi() {
    if (earth.day == 2) {
        earth.virus = true;
        randomizeHP();
    } else if (earth.day >= 4) {

```

```

        earth.pasien = false;
    }
}

```

Script ini digunakan sebagai cara untuk memberi aktivasi terhadap virus yang akan dimulai pada hari kedua dengan menggunakan penggunaan *if* “`earth.day == 2`” maka variabel “`earth.virus`” di-*set* menjadi *true* lalu untuk “`earth.day >= 4`” untuk mengindikasikan jika tidak ada pasien yang hidup untuk menentukan menang atau kalahnya dari game ini.

```

void berikanObat() {
    if (earth.obat_used) {
        cout << "Obat sudah digunakan hari ini. Tunggu hari
berikutnya." << endl;
        return;
    }
    if (earth.num_pasien == 0) {
        cout << "Tidak ada pasien yang bisa diberi obat." << endl;
        return;
    }
    string pasien_name;
    cout<<"=====\n";
    cout << "Pilih pasien untuk diberi obat\n";
    cout<<"=====\n";
    cout<<"- Dokter\n";
    cout<<"- Jendral\n";
    cout<<"- Presiden\n";
    cout<<"- Mentri\n";
    cout<<"- Professor\n";
    cout<<"Masukkan nama pasien yang akan diobati : ";
    cin >> pasien_name;
    bool valid_choice = false;
    for (int i = 0; i < earth.num_pasien; i++) {
        if (earth.names[i] == pasien_name) {
            int& hp = earth.hp[i];
            if (hp < MAX_HP) {
                hp = min(hp + 35, MAX_HP);
                cout << "Pasien " << pasien_name << " telah diberikan
obat. HP sekarang = " << hp << endl;
            } else {
                cout << "Pasien " << pasien_name << " sudah memiliki
HP maksimum." << endl;
            }
            earth.obat_used = true;
            valid_choice = true;
            break;}
        }
    if (!valid_choice) {
        cout << "Pasien tidak valid." << endl;
    }
}

```

Script “`void berikanObat()`” merupakan sebuah perintah untuk memberikan obat kepada salah satu pasien yang masih hidup, meningkat HP pasien sebanyak 35, dan memastikan bahwa obat hanya bisa diberikan sekali perhari. “`if (earth.obat_used)`”

fungsi untuk mengecek apakah obat sudah digunakan hari ini. “if (earth.num_pasien == 0)” fungsi untuk mengecek apakah tidak ada pasien yang hidup “earth.num_pasien == 0” jika semua pasien telah mati, maka fungsi akan menampilkan pesan dan keluar dari fungsi dengan “return;”.

Untuk *code* “string pasien_name” digunakan sebagai pendeklarasian untuk memanggil nama pasien seperti Dokter, Jendral, Presiden dan lain lain yang akan dimasukkan saat memberikan obat yang dipakai dalam *code* “cin >> pasien_name”. “valid_choice” di-set menjadi “false” agar memeriksa jika nama pasien yang dimasukkan benar lalu hal itu berguna sebagai iterasi *array* “earth.names” lalu jika hal itu sesuai dengan elemen nama pasien yang berada dalam *array* maka hal itu merespon nilai HP yang terdapat di dalam *array* lalu menambahkan poin HP tersebut sesuai dengan nama pasien yang telah dipilih namun sebagai syarat yang mana pasien tidak dapat menerima obat jika HP pasien telah dinilai maksimum. Lalu “earth.obat_use” di-set menjadi “true” yang berarti penggunaan obat telah dipakai untuk hari ini dan “valid_choice” juga di-set menjadi “true” yang berarti menandakan untuk keluar dari iterasi pengulangan dan jika masih di-set sebagai *false* maka nama pasien yang dimasukkan salah.

```
void hapusPasien(int index) {
    for (int i = index; i < earth.num_pasien - 1; i++) {
        earth.hp[i] = earth.hp[i + 1];
        earth.names[i] = earth.names[i + 1];
    }
    earth.num_pasien--;
}
```

Script ini digunakan untuk mengetahui pasien mana yang poin HP nya menyentuh angka 0 atau dapat dibilang pasien tersebut mati. *Script* ini menggunakan perulangan dengan cara mengambil parameter “index” yang merepresentasikan posisi dari “earth.names” dan “earth.hp” lalu mulai dengan perulangan “earth.num_pasien - 1” yang berguna menggeser elemen ke kanan sebanyak 1 posisi hal ini digunakan untuk memberi nilai ulang pada indeks selanjutnya. Setelah menggeser nilai, fungsi dari “earth.num_pasien--” sebagai tanda untuk mengurangi nilai dari jumlah pasien yang mengindikasikan bahwa pasien berkurang 1

```
void day()
{
    earth.day++;
    earth.obat_used = false;
    for (int i = 0; i < earth.num_pasien; i++)
    {
```

```

        if (earth.hp[i] > 0)
        {
            earth.hp[i] -= 20;
            if (earth.hp[i] <= 0) {
                cout << "Pasien " << earth.names[i] << " telah mati."
<< endl;
                hapusPasien(i);
                i--;
            }
        }
        kondisi();
    }
}

```

Script “void day()” merupakan sebuah fungsi untuk mensimulasikan berjalannya satu hari dalam program. “earth.day++;” code yang berfungsi untuk meningkatkan nilai sebesar satu yang menandakan satu hari telah berlalu. “earth.obat_used = false;” code yang berfungsi untuk mengatur nilai menjadi false yang menandakan bahwa obat belum digunakan pada hari yang baru dimulai. “for(int i = 0; i < earth.num_pasien; i++)” code yang berfungsi untuk looping ke iterasi melalui semua pasien yang masih hidup. “if (earth.hp[i] > 0)” code yang berfungsi untuk mengecek apakah HP pasien di indeks “i” lebih besar dari 0. Jika iya, maka akan melanjutkan ke langkah berikutnya. “earth.hp[i] -= 20;” code yang berfungsi untuk mengurangi nilai HP pasien sebesar 20 untuk menandakan bahwa pasien tersebut kehilangan HP hari ini. “if (earth.hp[i] <= 0)” code yang mengecek apakah setelah pengurangan HP, nilai HP pasien menjadi kurang dari atau sama dengan 0. Jika iya, maka lanjutkan ke langkah berikutnya. “cout << “Pasien ” << earth.names[i] << “ telah mati.” << endl;” code yang berfungsi untuk menampilkan pesan bahwa pasien yang bersangkutan telah mati. “hapusPasien(i);” code yang berfungsi untuk memanggil fungsi untuk menghapus pasien yang telah mati dari daftar pasien yang hidup. “i--;” code yang berfungsi untuk mengurangi nilai “i” sebesar satu untuk menyesuaikan indeks. “kondisi();” code yang berfungsi untuk memanggil fungsi untuk memperbarui kondisi bumi.

```

void day_one() \
{
    earth.day = 1;
    earth.pasien = true;
    earth.virus = false;
    earth.obat_used = false;
    earth.num_pasien = MAX_PASIEN;
    allocateMemory();
    string default_names[MAX_PASIEN] = {"Dokter", "Jendral",
    "Presiden", "Mentri", "Professor"};
    for (int i = 0; i < MAX_PASIEN; i++)

```

```

{
    earth.hp[i] = MAX_HP;
    earth.names[i] = default_names[i];
}
}

```

Bagian ini merupakan pendeklarasian dari semua struktur awal dari game yang dibuat yang mana bagian ini mengubah seluruh nilai variabel menjadi *default* agar proses game dapat berjalan sesuai yang diinginkan. “`earth.day = 1`” mendeklarasikan hari yang dimulai yaitu hari pertama. “`earth.pasien = true`” mengindikasikan bahwa ada pasien dalam simulasi ini, “`earth.virus = false`” dan “`earth.obat_used = false`” diatur agar virus dan obat tidak dapat digunakan terlebih dahulu. “`earth.num_pasien = MAX_PASIEN`” yaitu men-*set* banyak pasien yang akan diuji yaitu sebanyak 5 pasien. “`allocateMemory();`” *code* ini digunakan untuk mengalokasikan jumlah HP kepada *array* “`hp`”. “`string default_names[MAX_PASIEN] = {"Dokter", "Jendral", "Presiden", "Mentri", "Professor"}`” yaitu nama pasien yang harus disembuhkan dan iterasi pengulangan di sini yaitu memeriksa pasien. “`earth.hp[i] = MAX_HP`” yaitu mengatur darah dari masing masing pasien yaitu 100 “`earth.names[i] = default_names[i];`” yaitu memberi nama *default* pada setiap pasien.

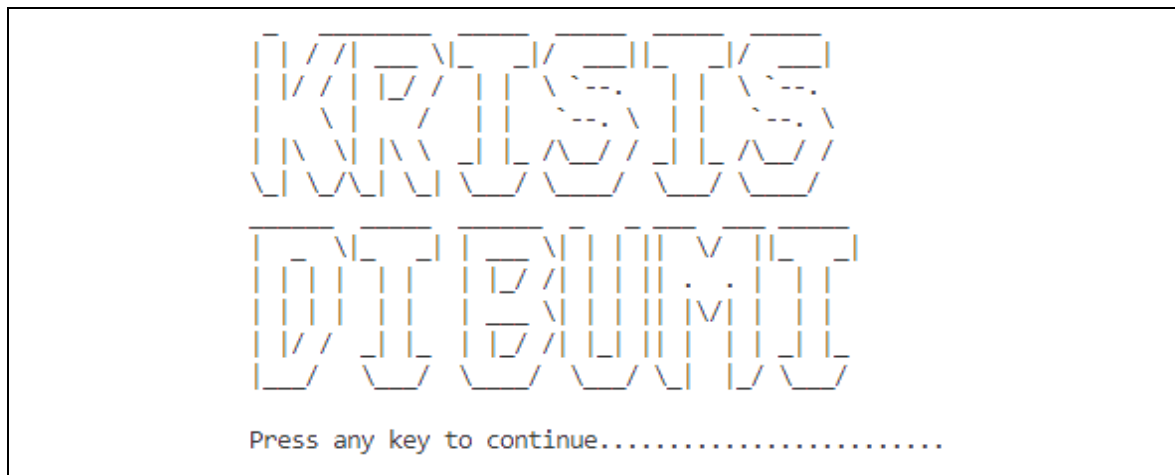
```

bool checkWinCondition() {
    return earth.num_pasien >= 2;
}
bool checklosecondition() {
    return earth.num_pasien < 2;
}

```

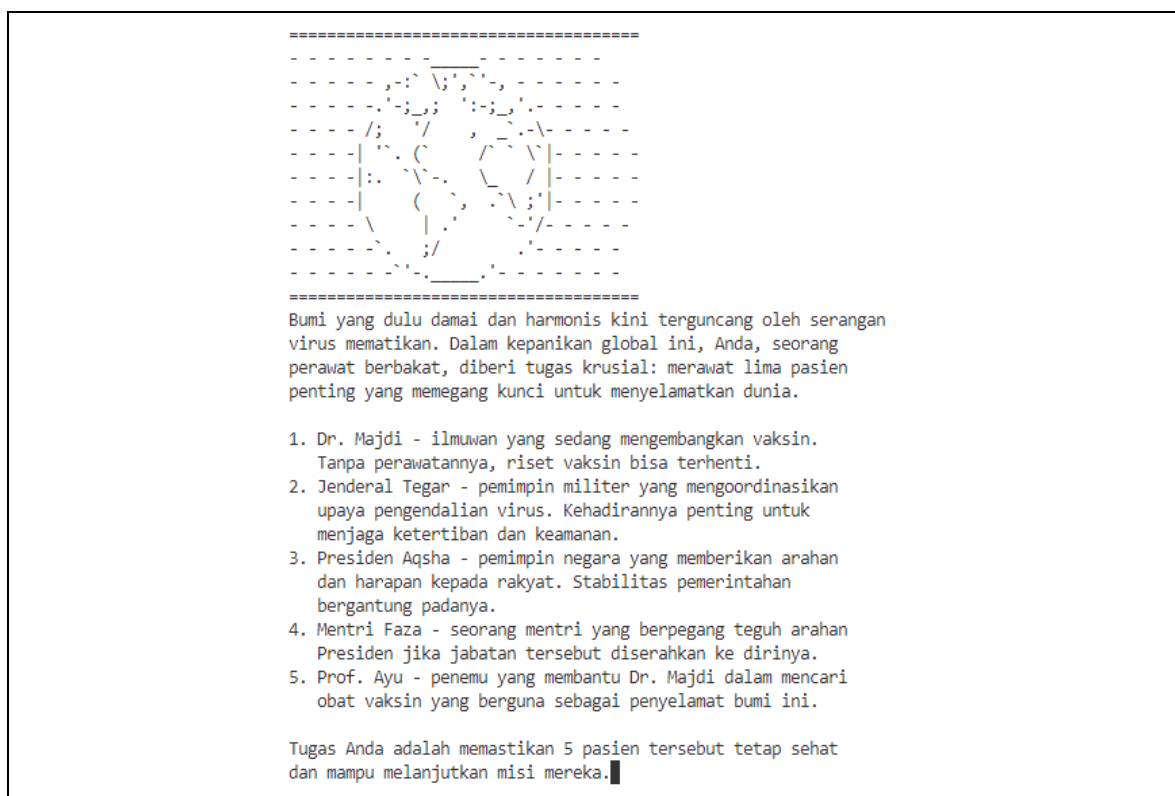
Guna *script* diatas adalah untuk memeriksa bahwa *user* telah menamatkan simulasi game ini dengan syarat bahwa pasien yang diselamatkan lebih dari sama dengan 2 pasien yang mana apabila *user* hanya mampu menyelamatkan kurang dari 2 maka game dinyatakan gagal dan *user* kalah dalam menyelamatkan bumi.

1.6 OUTPUT PROGRAM



Gambar 1.1 Tampilan Header

Berdasarkan **Gambar 1.1** di atas, program dapat menampilkan tampilan awal. Tampilan tersebut merupakan tampilan menu awal dari program simulasi manajemen krisis kesehatan di bumi yang menampilkan header dari program tersebut.



Gambar 1.2 Tampilan Ikon Bumi dan Skenario Program

Berdasarkan **Gambar 1.2** di atas, program tersebut menampilkan sebuah ikon kehidupan di bumi yang sedang tercemar oleh virus. Selain itu, program akan mencetak atau menampilkan sebuah skenario perjuangan seorang perawat ahli dalam mengatasi kepanikan global akan sebuah penyebaran virus yang ada di bumi ini.

```
=====
Hari ke-1
=====
1. Lihat kondisi pasien
2. Hari selanjutnya
3. Berikan obat
Masukkan Pilihan: █
```

Gambar 1.3 Tampilan Menu

Berdasarkan **Gambar 3.1** di atas, program dapat menampilkan sebuah menu utama dimana menu utama tersebut terdapat sebuah menu seperti melihat kondisi pasien, hari selanjutnya, dan masukkan pilihan.

```
=====
Status pasien hari ke-1:
=====
Pasien Dokter: HP = 100
Pasien Jendral: HP = 100
Pasien Presiden: HP = 100
Pasien Mentri: HP = 100
Pasien Professor: HP = 100
=====
Bumi Baik Baik Saja
=====
█
```

Gambar 1.4 Memasukkan pilihan ke-1

Berdasarkan **Gambar 1.4** di atas, ketika pengguna memasukkan pilihan ke-1, maka program akan menampilkan status pasien hari ke-1 beserta nama, *health point*, beserta pesan bahwa bumi dalam keadaan baik baik saja .

```
=====
Hari ke-2
=====
1. Lihat kondisi pasien
2. Hari selanjutnya
3. Berikan obat
Masukkan Pilihan: █
```

Gambar 1.5 Tampilan Menu dengan Pilihan ke-2

Berdasarkan **Gambar 1.5** di atas, ketika pengguna memasukan pilihan ke-2, maka program akan menampilkan pesan bahwa hari ini hari ke-2 beserta menu seperti lihat kondisi pasien, hari selanjutnya, berikan obat, beserta menu memasukkan pilihan.

```
=====
Status pasien hari ke-2:
=====
Pasien Jendral: HP = 30
Pasien Presiden: HP = 30
Pasien Mentri: HP = 35
Pasien Professor: HP = 50
Pasien Dokter: HP = 70
=====
Bumi Terkena Virus
=====
_
```

Gambar 1.6 Memasukkan Pilihan ke-1

Berdasarkan **Gambar 1.6** di atas, ketika pengguna memasukkan pilihan ke-1, maka program akan menampilkan status pasien hari ke-2 beserta nama, *health point*, beserta pesan bahwa bumi dalam keadaan baik baik saja.

```
=====
Pilih pasien untuk diberi obat
=====
- Dokter
- Jendral
- Presiden
- Mentri
- Professor
Masukkan nama pasien yang akan diobati : Jendral
Pasien Jendral telah diberikan obat. HP sekarang = 65
_
```

Gambar 1.7 Tampilan Berikan Obat

Berdasarkan **Gambar 1.7** di atas, ketika pengguna memasukkan pilihan ke-3, maka program akan menampilkan pesan serta meminta pengguna diminta memasukkan nama pasien yang akan diberikan obat. Selanjutnya, program akan meminta pengguna untuk memasukkan nama pasien yang akan diberikan obat tersebut serta program akan menampilkan sebuah pesan bahwa status pasien (nama pasien) telah diberikan obat beserta kondisi *health point* pasien tersebut.

```
=====
Status pasien hari ke-2:
=====
Pasien Presiden: HP = 30
Pasien Mentri: HP = 35
Pasien Professor: HP = 50
Pasien Jendral: HP = 65
Pasien Dokter: HP = 70
=====
Bumi Terkena Virus
=====
```

Gambar 1.8 Tampilan Status Pasien Hari ke-2

Berdasarkan **Gambar 1.8** di atas, ketika pengguna memasukkan pilihan ke-1, maka program akan menampilkan status pasien pada hari ke-2 yang akan menampilkan sebuah nama pasien, *health point* beserta pesan bahwa bumi dalam keadaan sudah terkena virus.

```
Obat sudah digunakan hari ini. Tunggu hari berikutnya.
```

Gambar 1.9 Status Berikan Obat pada Hari ke-2

Berdasarkan **Gambar 1.9** di atas, ketika pengguna memasukkan pilihan ke-3, maka program akan menampilkan status obat sudah digunakan hari ini, tunggu hari berikut, yang berarti program ini hanya bisa mengasih obat 1 kali dalam sehari.

```
=====
Hari ke-3
=====
1. Lihat kondisi pasien
2. Hari selanjutnya
3. Berikan obat
Masukkan Pilihan: █
```

Gambar 1.10 Tampilan Menu pada Hari ke-3

Berdasarkan **Gambar 1.10** di atas, program akan menampilkan sebuah menu pada hari ke-3 yang dimana program akan menampilkan sebuah menu berupa menu lihat kondisi pasien, hari selanjutnya dan berikan obat beserta perintah untuk memasukkan pilihan menu tersebut.

```
=====
Status pasien hari ke-3:
=====
Pasien Presiden: HP = 10
Pasien Mentri: HP = 15
Pasien Professor: HP = 30
Pasien Jendral: HP = 45
Pasien Dokter: HP = 50
=====
Bumi Terkena Virus
=====
```

Gambar 1.11 Status Pasien Hari ke-3

Berdasarkan **Gambar 1.11** di atas, ketika pengguna memasukkan pilihan ke-1, maka program akan menampilkan status pasien pada hari ke-3 yang akan menampilkan sebuah nama pasien, *health point* beserta pesan bahwa bumi sudah terkena virus.

```
=====
Pilih pasien untuk diberi obat
=====
- Dokter
- Jendral
- Presiden
- Mentri
- Professor
Masukkan nama pasien yang akan diobati : Presiden
Pasien Presiden telah diberikan obat. HP sekarang = 45
=====
```

Gambar 1.12 Pilihan Pasien untuk Diberikan Obat

Berdasarkan **Gambar 1.12** di atas, ketika pengguna memasukkan pilihan ke-3, maka program akan menampilkan pesan serta meminta pengguna diminta memasukkan nama pasien yang akan diberikan obat. Selanjutnya, program akan meminta pengguna untuk memasukkan nama pasien yang akan diberikan obat tersebut serta program akan menampilkan sebuah pesan bahwa status pasien (nama pasien) telah diberikan obat beserta kondisi *health point* pasien tersebut.

```
Pasien Mentri telah mati.
=====
      Hari ke-4
=====
1. Lihat kondisi pasien
2. Hari selanjutnya
3. Berikan obat
Masukkan Pilihan: █
```

Gambar 1.13 Tampilan pada Hari ke-4

Berdasarkan **Gambar 1.13** di atas, ketika pengguna memasukkan pilihan ke-2, maka program akan menampilkan pesan bahwa pasien (nama pasien) telah mati beserta hari ke-4 sudah masuk. lalu program akan menampilkan menu seperti lihat kondisi pasien, hari selanjutnya, berikan obat beserta pesan memasukkan pilihan.

```
=====
Status pasien hari ke-4:
=====
Pasien Professor: HP = 10
Pasien Presiden: HP = 25
Pasien Jendral: HP = 25
Pasien Dokter: HP = 30
=====
Cepat Sembuhkan Bumi
=====
```

Gambar 1.14 Status Pasien Hari ke-4

Berdasarkan **Gambar 1.14** di atas, ketika pengguna memasukkan pilihan ke-1, maka program akan menampilkan status pasien pada hari ke-4 yang akan menampilkan sebuah nama pasien, *health point* beserta pesan bahwa cepat sembuhkan bumi.

```
=====
Pilih pasien untuk diberi obat
=====
- Dokter
- Jendral
- Presiden
- Mentri
- Professor
Masukkan nama pasien yang akan diobati : Professor
Pasien Professor telah diberikan obat. HP sekarang = 45
```

Gambar 1.15 Tampilan Pilih Pasien untuk Diberi Obat

Berdasarkan **Gambar 1.15** di atas, ketika pengguna memasukkan pilihan ke-3, maka program akan menampilkan pesan serta meminta pengguna diminta memasukkan nama pasien yang akan diberikan obat. Selanjutnya, program akan meminta pengguna untuk memasukkan nama pasien yang akan diberikan obat tersebut serta program akan menampilkan sebuah pesan bahwa status pasien tersebut (nama pasien) telah diberikan obat beserta kondisi *health point* pasien tersebut.

```
=====
      Hari ke-5
=====
1. Lihat kondisi pasien
2. Hari selanjutnya
3. Berikan obat
Masukkan Pilihan: █
```

Gambar 1.16 Tampilan Menu pada Hari ke-5

Berdasarkan **Gambar 1.16** di atas, ketika pengguna memasukkan pilihan ke-2, maka program akan menampilkan pesan bahwa pasien (nama pasien) telah mati beserta hari ke-5 sudah masuk. lalu program akan menampilkan menu seperti lihat kondisi pasien, hari selanjutnya, berikan obat beserta pesan memasukkan pilihan.

```
=====
Pilih pasien untuk diberi obat
=====
- Dokter
- Jendral
- Presiden
- Mentri
- Professor
Masukkan nama pasien yang akan diobati : Presiden
Pasien Presiden telah diberikan obat. HP sekarang = 40
```

Gambar 1.17 Berikan Obat

Berdasarkan **Gambar 1.17** di atas, ketika pengguna memasukkan pilihan ke-3, maka program akan menampilkan pesan serta meminta pengguna diminta memasukkan nama pasien yang akan diberikan obat. Selanjutnya, program akan meminta pengguna untuk memasukkan nama pasien yang akan diberikan obat tersebut serta program akan menampilkan sebuah pesan bahwa status pasien (nama pasien) telah diberikan obat beserta kondisi *health point* pasien tersebut.

```
Pasien Jendral telah mati.
Pasien Dokter telah mati.
=====
      Hari ke-6
=====
1. Lihat kondisi pasien
2. Hari selanjutnya
3. Berikan obat
Masukkan Pilihan: █
```

Gambar 1.18 Tampilan Hari ke-6

Berdasarkan **Gambar 1.18** di atas, ketika pengguna memasukkan pilihan ke-2, maka program akan menampilkan pesan bahwa pasien (nama pasien) telah mati beserta hari ke-6 sudah masuk. lalu program akan menampilkan menu seperti lihat kondisi pasien, hari selanjutnya, berikan obat beserta pesan memasukkan pilihan.

```
=====
Status pasien hari ke-6:
=====
Pasien Professor: HP = 5
Pasien Presiden: HP = 20
=====
Cepat Sembuhkan Bumi
=====
```

Gambar 1.19 Tampilan Status Pasien Hari ke-6

Berdasarkan **Gambar 1.19** di atas, ketika pengguna memasukkan pilihan ke-1, maka program akan menampilkan status pasien pada hari ke-6 yang akan menampilkan sebuah nama pasien, *health point* beserta pesan bahwa cepat sembuhkan bumi.

```
=====
Pilih pasien untuk diberi obat
=====
- Dokter
- Jendral
- Presiden
- Mentri
- Professor
Masukkan nama pasien yang akan diobati : Professor
Pasien Professor telah diberikan obat. HP sekarang = 40
_
```

Gambar 1.20 Tampilan Berikan Obat

Berdasarkan **Gambar 1.20** di atas, ketika pengguna memasukkan pilihan ke-3, maka program akan menampilkan pesan serta meminta pengguna diminta memasukkan nama pasien yang akan diberikan obat. Selanjutnya, program akan meminta pengguna untuk memasukkan nama pasien yang akan diberikan obat tersebut serta program akan menampilkan sebuah pesan bahwa status pasien tersebut (nama pasien) telah diberikan obat beserta kondisi *health point* pasien tersebut.

```
=====
Status pasien hari ke-6:
=====
Pasien Presiden: HP = 20
Pasien Professor: HP = 40
=====
Cepat Sembuhkan Bumi
=====
```

Gambar 1.21 Tampilan Status Pasien Hari ke-6

Berdasarkan **Gambar 1.21** di atas, ketika pengguna memasukkan pilihan ke-1, maka program akan menampilkan status pasien pada hari ke-6 yang akan menampilkan sebuah nama pasien, *health point* beserta pesan bahwa cepat sembuhkan bumi.

```
Pasien Presiden telah mati.
Anda Kalah! Karena anda tidak bisa menyelamatkan minimal 2 pasien
_
```

Gambar 1.22 Tampilan kalah

Berdasarkan **Gambar 1.22** di atas, ketika pengguna memasukkan pilihan ke-2, maka program akan menampilkan status pasien presiden telah mati dan anda kalah! karena anda tidak bisa menyelamatkan minimal 2 pasien.

1.7 KESIMPULAN

Simulasi Manajemen Krisis Kesehatan di Bumi merupakan sebuah inovatif pendekatan dalam memodelkan dinamika penyebaran virus dan upaya menjaga kesehatan pasien dalam menghadapi ancaman pandemi. Dengan fokus pada pengambilan keputusan strategis, manajemen sumber daya, dan pengetahuan medis, program ini memberikan pengalaman mendalam yang memperkuat pemahaman tentang kompleksitas krisis kesehatan. Melalui simulasi ini, pengguna diberi kesempatan untuk merasakan tekanan dan urgensi dalam situasi darurat kesehatan, meningkatkan kemampuan analitis serta kemahiran dalam mengambil keputusan cepat dan tepat.

Tujuan simulasi ini tidak hanya sebatas pendidikan, tetapi juga melatih keterampilan praktis dalam manajemen krisis kesehatan. Dengan memahami secara langsung dampak dari setiap keputusan yang diambil, pengguna dapat mengembangkan wawasan yang lebih dalam tentang prioritas pemberian obat, pengelolaan sumber daya yang terbatas, dan pentingnya koordinasi dalam penanganan wabah. Dengan demikian, simulasi ini menjadi sebuah platform yang efektif dalam meningkatkan kesiapan masyarakat dalam menghadapi krisis kesehatan yang kompleks dan tidak terduga.

Secara keseluruhan, simulasi ini memainkan peran penting dalam meningkatkan kesadaran akan pentingnya tindakan preventif dan respons yang cepat dalam menghadapi penyakit menular. Dengan fokus pada pendidikan, latihan dan kesadaran masyarakat, simulasi ini bukan hanya sebuah permainan, tetapi juga sebuah alat yang kuat dalam mempersiapkan individu dan komunitas untuk menghadapi tantangan kesehatan yang mungkin timbul di masa depan.

1.8 REFRENSI

- Fachrurrozi, M & Rosa, D.I. (2006). Algoritma dan Pemrograman 1. Universitas Sriwijaya. Palembang.
- Firliana, R. & Kasih, P. (2018). Algoritma dan Pemrograman C++. Edisi pertama, Adjie Media Nusantara. Nganjuk, Jawa Timur.
- Suprpto, dkk. (2008). Bahasa Pemrograman untuk Sekolah Menengah Kejuruan. Departemen Pendidikan Nasional.