

Math Games Adventure

Disusun oleh Kelompok 8

Anggota Kelompok



- 1. Farrell Luthfi Adriansyah L0125098**
- 2. Muhammad Haidar Amru L0125025**
- 3. Satrio Unggul Prayogo L0125114**

BAB II

Pendahuluan

Penjelasan Umum

Permainan ini bertemakan petualangan di gua misterius. Pengguna diceritakan masuk ke dalam gua misterius dan bertemu dengan jin. Apabila ingin keluar dari gua misterius, maka harus menyelesaikan tantangan matematika yang diberikan. Pemain harus menjawab seluruh soal matematika yang diberikan oleh jin dalam 3 babak. Pemain diberikan 3 nyawa dan akan berkurang apabila pemain salah menjawab soal.

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <string.h>
#include <windows.h>
```

Library

#include <stdio.h>

Sebagai standar input output, wajib ada agar fungsi seperti printf, scanf, fopen, dll. bisa digunakan.

#include <string.h>

Sebagai string handling, digunakan untuk manipulasi teks seperti strcpy.

#include <stdlib.h>

Standard library, digunakan untuk manajemen memori (malloc) dan kontrol sistem.

Library

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <string.h>
#include <windows.h>
```

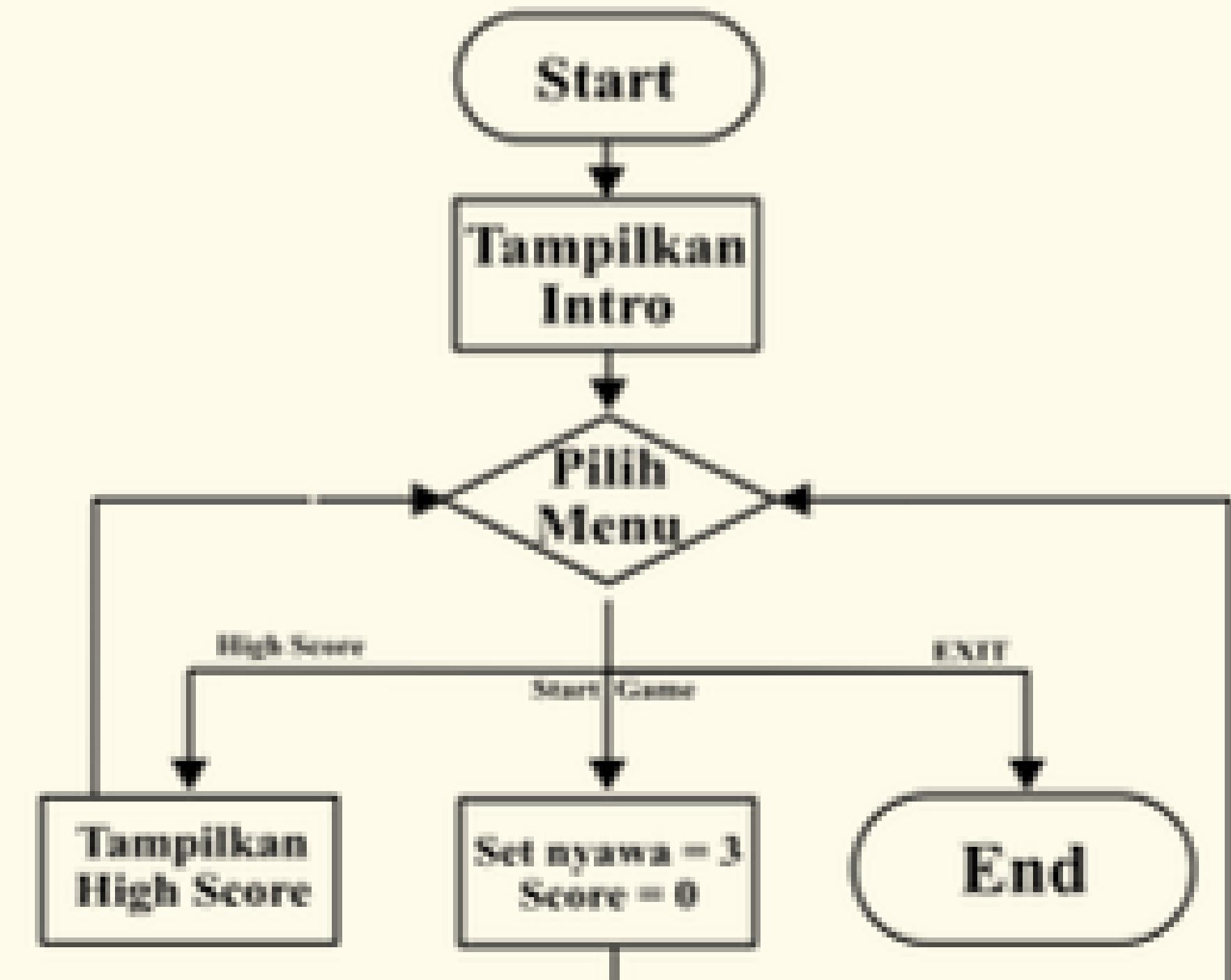
#include <time.h>

Library untuk memanipulasi tanggal dan waktu sistem, akan digunakan untuk **srand(time(NULL))**

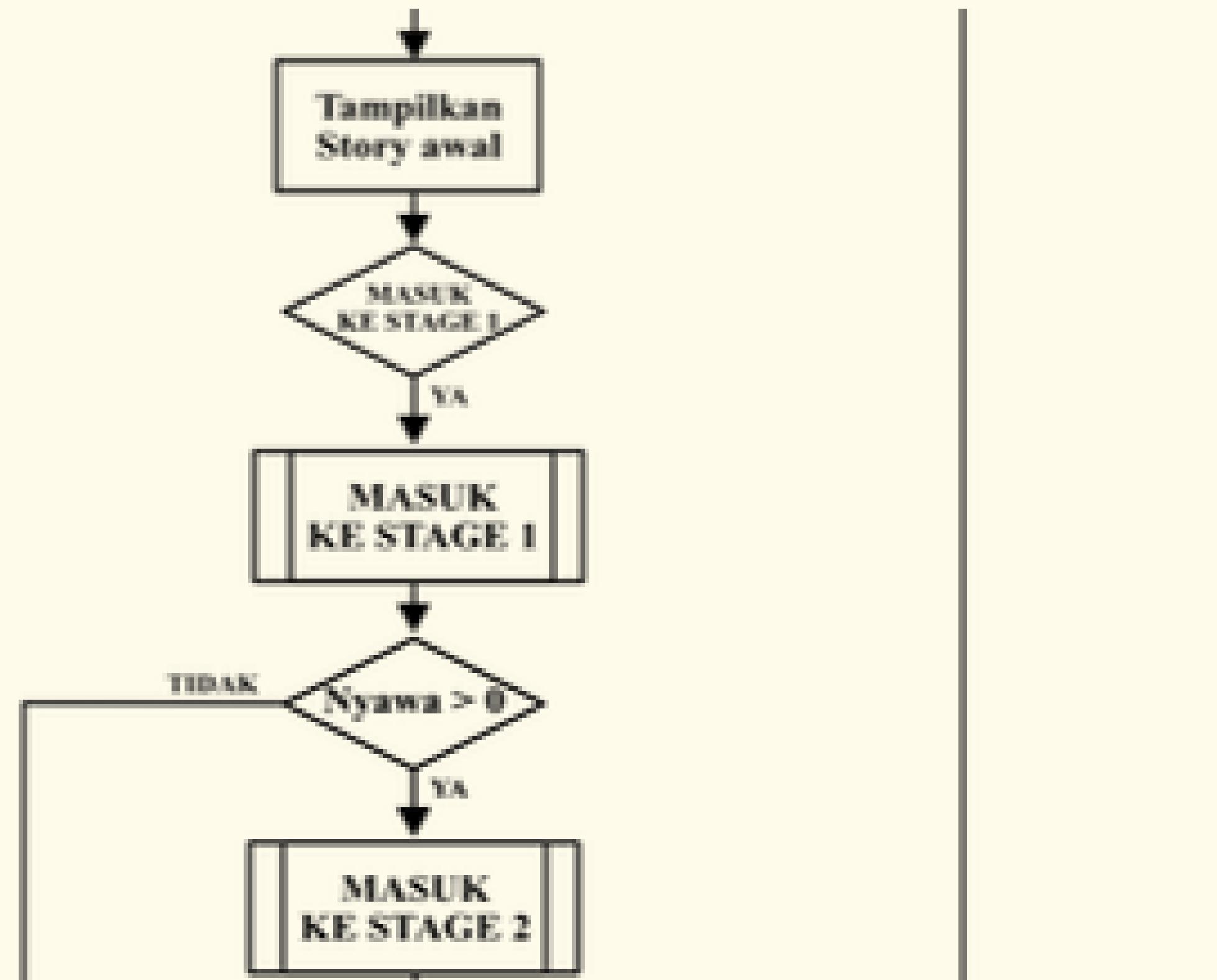
#include <windows.h>

Windows API, library khusus untuk OS Windows. Digunakan di program ini untuk mengakses fungsi **Sleep()**.

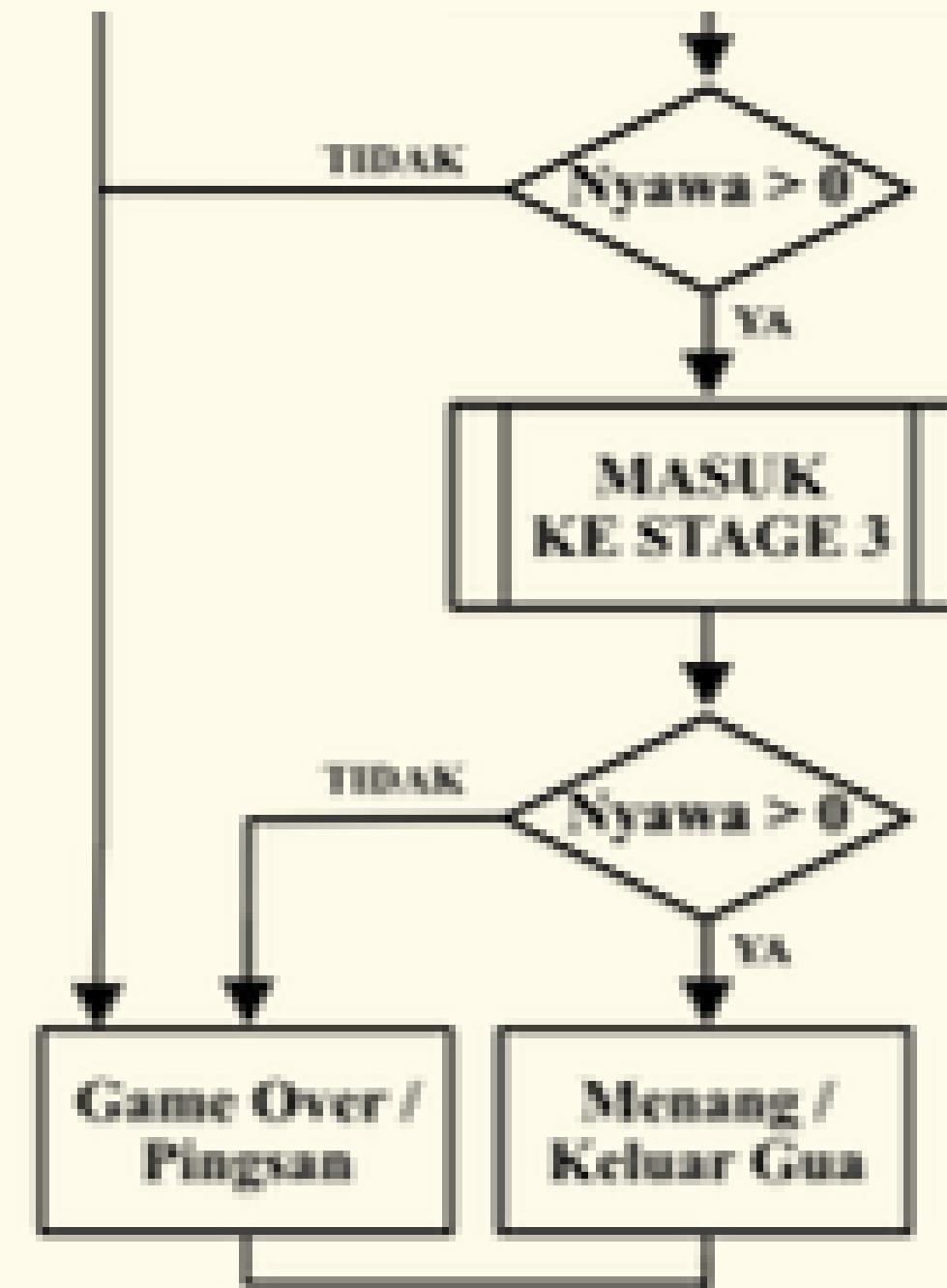
Alur Kerja Program



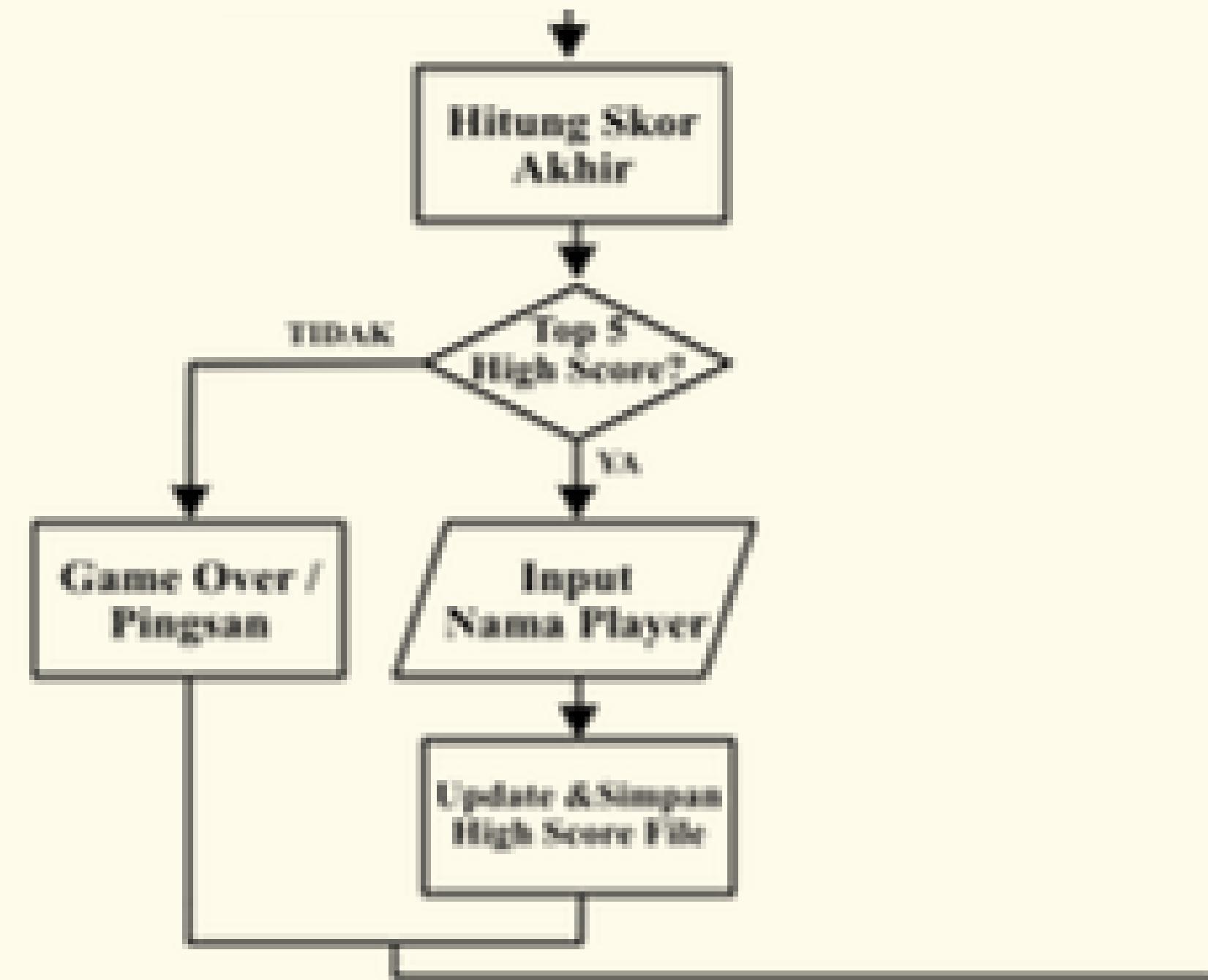
Alur Kerja Program



Alur Kerja Program



Alur Kerja Program



BAB II

Source Code

2.1.1

Header, Konstanta, Dan Variabel Global

Konstanta

```
#define FILE_HIGHSCORE "highscore.txt"  
#define MAX_SOAL_2 8  
#define MAX_SOAL_3 5  
#define PANJANG_SOAL 512  
#define LEBAR_KOTAK 70
```

#define FILE_HIGHSCORE

Mendefinisikan nama file database sebagai konstanta untuk mempermudah perubahan nama file tanpa perlu mengedit seluruh kode.

Variabel Global

```
// Variabel global untuk skor dan nyawa  
int lives = 3;  
int score = 0;  
int death = 0;
```

1. **int lives** Menyimpan sisa nyawa pemain (awal 3) yang berkurang saat jawaban salah dan otomatis di-reset saat permainan baru dimulai.
2. **int score** Menyimpan akumulasi poin pemain yang nilainya fluktuatif tergantung ketepatan jawaban dan di-reset menjadi 0 di awal permainan.
3. **int death** Berfungsi sebagai flag (penanda) untuk mendeteksi apakah pemain sudah pernah mengalami kekalahan sebelumnya dalam sesi tersebut.

Struct

```
struct highScore {  
    char username[50];  
    int score;  
}; //Struct untuk highscore  
  
struct highScore topLeaderboard[5];
```

1. struct highScore Mendefinisikan format struktur data pemain yang terdiri dari komponen username dan nilai skor.

2. topLeaderboard[5] Array global yang berfungsi menampung 5 data skor tertinggi yang dikelola dan diurutkan secara dinamis.

2.1.2

Fungsi Utility & Interface

Manajemen Waktu

```
//Fungsi deLay  
void delay(int milliseconds) {  
    Sleep(milliseconds);  
}
```

Menyimpan fungsi sleep() bawaan dari Windows. Fungsi ini menjeda program selama beberapa milidetik, memberikan efek transisi yang halus.

Animasi Teks

```
// Fungsi untuk mencetak teks per karakter (animasi)
void ketikPelan(char *teks, int kecepatan) {
    // Loop sampai akhir string (karakter '\0')
    while (*teks != '\0') {
        printf("%c", *teks);
        fflush(stdout);
        delay(kecepatan);
        teks++;
    }
}
```

Fungsi untuk memberikan efek animasi mengetik huruf per huruf. Menggunakan pointer `*teks` untuk melihat string dari karakter demi karakter. `fflush(stdout)` digunakan agar fungsi memaksa huruf segera muncul di layar tanpa menunggu baris baru.

Pembersihan Layar

```
// Fungsi untuk mencetak teks per karakter (animasi)
void ketikPelan(char *teks, int kecepatan) {
    // Loop sampai akhir string (karakter '\0')
    while (*teks != '\0') {
        printf("%c", *teks);
        fflush(stdout);
        delay(kecepatan);
        teks++;
    }
}
```

Menyimpan perintah `system("cls")` yang berfungsi agar membersihkan layar terminal agar tampilan menu selalu rapi.

Pembersihan Buffer Input

```
// Fungsi untuk mencetak teks per karakter (animasi)
void ketikPelan(char *teks, int kecepatan) {
    // Loop sampai akhir string (karakter '\0')
    while (*teks != '\0') {
        printf("%c", *teks);
        fflush(stdout);
        delay(kecepatan);
        teks++;
    }
}
```

- Menutupi masalah yang sering terjadi pada scanf, yakni meninggalkan karakter newline di dalam buffer input setelah membaca angka, jika tidak dibersihkan maka fungsi input berikutnya akan membaca sisa enter tersebut.
- Fungsi ini melakukan perulangan sederhana, ia akan terus menerus mengambil getchar dan membuang karakter yang tersisa di memori sementara sampai ditemukan karakter baris baru atau akhir file.

2.1.3

Sistem High Score & Manajemen File

Fungsi Save High Score

```
//Fungsi menyimpan skor
void saveHighScore(){
    FILE *fp = fopen(FILE_HIGHSCORE, "w");
    if (fp == NULL) {
        printf("Error opening file for writing high scores.\n");
        return;
    }

    for (int i = 0; i < 5; i++) {
        if (strlen(topLeaderboard[i].username) == 0) {
            fprintf(fp, "EMPTY %d\n");
        }
        else {
            fprintf(fp, "%s %d\n", topLeaderboard[i].username, topLeaderboard[i].score);
        }
    }

    fclose(fp);
}
```

Fungsi Save High Score

- **Tujuan Fungsi:** Fungsi bertugas memindahkan data skor dari memori program ke file eksternal `highscore.txt` agar tersimpan permanen.
- **Pembukaan File:** File dibuka dalam mode write ("w") dan pointer divalidasi untuk memastikan proses pembukaan berhasil tanpa error.
- **Perulangan:** Program menjalankan looping sebanyak 5 kali menyesuaikan kapasitas maksimal data pada leaderboard.
- **Penanganan Data Kosong:** Jika slot kosong, program menuliskan "EMPTY 0" untuk menjaga konsistensi format file agar tetap berjumlah 5 baris.
- **Penulisan & Penutupan:** Data pemain yang valid ditulis sesuai format nama dan skor, kemudian file ditutup menggunakan `fclose` untuk keamanan data.

Fungsi Load High Score

```
void loadHighScore(){
    FILE *fp = fopen(FILE_HIGHSCORE, "r");

    // Jika file tidak ada, isi dengan data kosong (EMPTY)
    if (fp == NULL) {
        for(int i=0; i<5; i++){
            strcpy(topLeaderboard[i].username, "EMPTY");
            topLeaderboard[i].score = 0;
        }
        return;
    }

    for (int i = 0; i < 5; i++) {
        fscanf(fp, "%s %d", topLeaderboard[i].username, &topLeaderboard[i].score);
    }
    fclose(fp);
}
```

Fungsi Load High Score

- **Tujuan Fungsi:** Fungsi ini bertugas memuat data skor dari file eksternal `highscore.txt` ke dalam memori program agar bisa diproses.
- **Validasi File:** File dibuka dalam mode read, dan jika tidak ditemukan, sistem mengisi array dengan data default ("EMPTY") untuk mencegah program crash.
- **Proses Pembacaan:** Program membaca data nama dan skor baris demi baris menggunakan `fscanf` ke dalam struct, kemudian menutup akses file setelah selesai.

Fungsi Update High Score

```
//Fungsi update high score
void updateHighScore(int skorBaru) {
    loadHighScore();

    if (skorBaru > topLeaderboard[4].score || strcmp(topLeaderboard[4].username, "EMPTY") == 0) {
        printf("Selamat! Anda mendapatkan High Score baru!\n");

        char namaBaru[50];
        printf("Masukkan username Anda (satu kata): ");
        scanf("%s", namaBaru);

        strcpy(topLeaderboard[4].username, namaBaru);
        topLeaderboard[4].score = skorBaru;

        struct highScore temp;
        for (int i = 0; i < 5 - 1; i++) {
            for (int j = 0; j < 5 - i - 1; j++) {
                if (topLeaderboard[j].score < topLeaderboard[j + 1].score) {
                    temp = topLeaderboard[j];
                    topLeaderboard[j] = topLeaderboard[j + 1];
                    topLeaderboard[j + 1] = temp;
                }
            }
        }
    }
}
```

Fungsi Update High Score

- **Tujuan Fungsi:** Fungsi ini memvalidasi kelayakan skor terbaru pemain untuk masuk ke daftar 5 besar, diikuti dengan proses memasukkan data, mengurutkan, dan menyimpan.
- **Sinkronisasi & Cek Awal:** Diawali dengan sinkronisasi data menggunakan `loadHighScore()`, kemudian program memeriksa kelayakan skor baru terhadap skor terendah yang ada di array (indeks ke-4).
- **Kriteria Kelayakan:** Skor baru diterima jika nilainya lebih besar dari Juara 5 ATAU jika slot peringkat ke-5 tersebut masih kosong.
- **Penyisipan Data Baru:** Jika lolos kualifikasi, program meminta input username dan langsung menimpa data yang ada pada indeks ke-4.
- **Pengurutan Ulang:** Setelah data baru masuk, array diurutkan kembali menggunakan Bubble Sort Descending (Besar ke Kecil) dengan membandingkan dan menukar posisi elemen yang bersebelahan hingga skor terbesar naik ke indeks 0.

Fungsi Update High Score

```
    saveHighScore();
    printf("High Score berhasil diperbarui!\n");
} else {
    printf("Maaf, Anda tidak masuk dalam Top 5 High Score.\n");
}
}
```

Fungsi Update High Score

Setelah array terurut rapi, maka fungsi akan memanggil `saveHighScore`. Dan kondisi else akan mencetak pesan bahwa user tidak layak masuk ke top 5 highscore.

Fungsi Show Leaderboard

```
//Fungsi menampilkan Leaderboard
void showLeaderboard() {
    loadHighScore();

    printf("===== TOP 5 HIGH SCORE =====\n");
    printf("Rank  %-15s Score\n", "Username");
    printf("-----");

    for (int i = 0; i < 5; i++)
    {
        printf("\n%d.  %-15s %d\n", i + 1, topLeaderboard[i].username, topLeaderboard[i].score);
    }
    printf("-----");
}
```

Fungsi Show Leaderboard

- **Tujuan Visualisasi:** Fungsi ini bertugas memvisualisasikan isi array `topLeaderboard` ke layar terminal dalam format tabel agar mudah dibaca.
- **Persiapan & Formatting:** Diawali dengan pemanggilan `loadHighScore` dan pencetakan header tabel yang menggunakan padding 15 karakter untuk menjaga kerapian kolom username dan skor.
- **Pencetakan Data:** Program melakukan perulangan sebanyak 5 kali untuk mencetak nomor peringkat, nama pemain, dan skor secara berurutan baris demi baris.

2.1.4

Logika Level 1

Fungsi Soal Penjumlahan

```
//Fungsi soal penjumlahan
void soalPenjumlahan() {
    if (lives <= 0) {
        return;
    }
    int x, y, jawaban, kunciJawaban;
    x = (rand()% 201 )- 100;
    y = (rand()% 201 )- 100;

    printf ("Hasil dari %d + %d adalah : ",x, y);
    scanf("%d", &jawaban);

    clearInputBuffer();

    kunciJawaban = x + y;
    if (jawaban == kunciJawaban) {
        printf("Selamat jawaban Anda benar!\n");
        score += 1;
    }
    else {
        printf("Maaf jawaban Anda salah. Jawaban yang benar adalah %d\n", kunciJawaban);
        score -= 1;
        lives--;
    }
}
```

Fungsi Soal Penjumlahan

- **Pemeriksaan Awal:** Di awal fungsi, dilakukan pengecekan nyawa; jika `lives <= 0`, fungsi segera berhenti (`return`) untuk mencegah eksekusi soal.
- **Inisialisasi & Generator:** Setelah lolos cek nyawa, program menyiapkan variabel lokal untuk soal (`x`, `y`, `jawaban`, `kunciJawaban`), di mana `x` dan `y` diisi menggunakan generator angka acak `rand()`.
- **Logika Batasan Angka:** Generator `rand()` dibatasi dengan operator modulo `% 201` untuk menghasilkan angka 0 hingga 200, kemudian dikurangi 100 untuk menggeser rentang menjadi bilangan positif dan negatif antara `-100` sampai `100`.
- **Validasi Jawaban:** Program menghitung `kunciJawaban` (`x + y`), membandingkannya dengan input user. Jika cocok, skor bertambah 1; jika salah, skor berkurang 1 dan nyawa pemain berkurang 1.

Fungsi Soal Pengurangan

```
//Fungsi soal pengurangan
void soalPengurangan() {
    if (lives <= 0) {
        return;
    }
    int x, y, jawaban, kunciJawaban;
    x = (rand()% 201 )- 100;
    y = (rand()% 201 )- 100;

    printf ("Hasil dari %d - %d adalah : ",x, y);
    scanf("%d", &jawaban);

    clearInputBuffer();

    kunciJawaban = x - y;
    if (jawaban == kunciJawaban) {
        printf("Selamat jawaban Anda benar!\n");
        score += 1;
    }
    else {
        printf("Maaf jawaban Anda salah. Jawaban yang benar adalah %d\n", kunciJawaban);
        score -= 1;
        lives--;
    }
}
```

Fungsi Soal Pengurangan

Konsep dasarnya sama dengan soal penjumlahan, hanya saja bedanya di sini adalah variabel kunci Jawaban berisi pengurangan dari x dan y.

Fungsi Soal Perkalian

```
//Fungsi soal Perkalian
void soalPerkalian() {
    if (lives <= 0) {
        return;
    }
    int x, y, jawaban, kunciJawaban;
    x = (rand()% 21 )- 10;
    y = (rand()% 21 )- 10;

    printf ("Hasil dari %d x %d adalah : ",x, y);
    scanf("%d", &jawaban);

    clearInputBuffer();

    kunciJawaban = x * y;
    if (jawaban == kunciJawaban) {
        printf("Selamat jawaban Anda benar!\n");
        score += 2;
    }
    else {
        printf("Maaf jawaban Anda salah. Jawaban yang benar adalah %d\n", kunciJawaban);
        score -= 1;
        lives--;
    }
}
```

Fungsi Soal Perkalian

- Konsep dasar juga masih mirip dengan 2 fungsi sebelumnya. Hanya saja x dan y yang dirandom dimod 21 dan dikurangi 10, sehingga angkanya hanya berkisar dari -10 sampai 10.
- Variabel kunci jawaban akan berisi perkalian dari x dan y. Untuk soal perkalian skor akan bertambah 2 jika jawaban benar.

Fungsi Soal Pembagian

```
//Fungsi soal pembagian
void soalPembagian() {
    if (lives <= 0) {
        return;
    }
    int x, y, jawaban, kunciJawaban;
    do{
        y = (rand()% 21 )- 10;
    } while (y == 0);

    kunciJawaban = (rand()% 21 )- 10;
    x = y * kunciJawaban;

    printf ("Hasil dari %d : %d adalah : ",x, y);
    scanf("%d", &jawaban);

    clearInputBuffer();
```

Fungsi Soal Pembagian

Pada dasarnya konsep masih sama, namun di sini agar soal tidak menghasilkan pembagian dengan 0 program akan menggunakan looping dan logika terbalik. Di mana, selama $y == 0$ maka nilai y akan dirandom terus menerus. Untuk kunci jawaban juga akan dirandomisasi sehingga nanti, nilai x adalah y dikali kunci jawaban. Hal ini akan membuat kunci jawaban bernilai x bagi y .

Fungsi Soal Pembagian

```
if (jawaban == kunciJawaban) {
    printf("Selamat jawaban Anda benar!\n");
    score += 2;
}
else {
    printf("Maaf jawaban Anda salah. Jawaban yang benar adalah %d\n", kunciJawaban);
    score -= 1;
    lives--;
}
```

Fungsi Soal Pembagian

Kemudian seperti soal perkalian, akan dicek apakah jawaban cocok dengan kunci jawaban, jika iya maka skor bertambah 2.

2.1.5

Logika Level 2

Deklarasi Variabel

```
int stage2() {
    // Array untuk menyimpan soal yang dibaca dari file
    char questions[MAX_SOAL_2][PANJANG_SOAL];
```

```
char jawabanUser[50];
int i;
```

Deklarasi Variabel

Hal yang pertama kali dilakukan adalah deklarasi variabel yang akan digunakan dalam logika stage 2 ini. Yang pertama ada deklarasi array `questions` bertipe `char`, dengan ukuran hasil perkalian definisi `MAX_SOAL_2` dan `PANJANG_SOAL`. Selanjutnya diperlukan deklarasi array `jawabanUser` bertipe `char` dengan ukuran 50 karakter. Array ini akan menjadi tempat penampung input jawaban yang dimasukkan pemain. Selanjutnya, diperlukan deklarasi variabel `i` yang akan digunakan untuk iterasi.

Array Jawaban

```
// Array untuk menyimpan jawaban yang benar
char *answers[MAX_SOAL_2] = {
    "4",      // Jawaban Soal 1
    "-13",    // Jawaban Soal 2
    "12",     // Jawaban Soal 3
    "66",     // Jawaban Soal 4
    "550",    // Jawaban Soal 5
    "342",    // Jawaban Soal 6
    "60",     // Jawaban Soal 7
    "25",     // Jawaban Soal 8
};
```

Array Jawaban

Selanjutnya, dilakukan deklarasi dan inisialisasi pada array answers bertipe char dengan pointer dan berukuran sesuai dengan definisi **MAX_SOAL_2**. Array ini akan digunakan sebagai kunci jawaban dari pertanyaan pada stage 2. Array ini akan menjadi pembanding antara nilai jawaban yang diinput oleh pengguna dengan nilai jawaban pada array answers. Hasil perbandingan ini akan menentukan skor yang didapat oleh pemain.

Logika Membaca File Soal

```
// --- BACA SOAL DARI FILE ---
FILE *file = fopen("STAGE 2.txt", "r");

if (file == NULL) {
    printf("Error: Tidak bisa membuka file soal.txt.\n");
    printf("Pastikan file soal.txt ada di folder yang sama dengan program ini.\n");
    return 1;
}

for (i = 0; i < MAX_SOAL_2; i++) {
    if (fgets(questions[i], PANJANG_SOAL, file) == NULL) {
        break;
    }
}
fclose(file);
// --- SELESAI BACA FILE ---
```

Logika Membaca File Soal

Selanjutnya, untuk menampilkan soal, program harus membaca dari file txt yang telah dibuat. Oleh karena itu, mula-mula dideklarasikan file pointer dan membuka file STAGE 2.txt dengan perintah read. Kemudian untuk mencetak soal pada layar. digunakan perulangan for untuk mencetak soal dari file baris per baris. Setelah selesai mencetak soal, maka file STAGE 2.txt akan ditutup

Logika Menjawab Soal

```
for (i = 0; i < MAX_SOAL_2; i++) {
    if (lives <= 0) return 0;

    printf("Soal no. %d\n", i + 1);

    // Cetak soal ke dalam interface kotak
    printf("%s", questions[i]);

    printf("Jawaban Anda: ");
    scanf("%s", jawabanUser);
    clearInputBuffer();
    printf("\n");

    if (strcmp(jawabanUser, answers[i]) == 0) {
        printf("Selamat jawaban Anda benar!\n\n");
        score += 3;
    } else {
        printf("Maaf jawaban Anda salah. Jawaban yang benar adalah %s\n\n", answers[i]);
        score -= 1;
        lives--;
    }
}
```

Logika Menjawab Soal

Pertama-tama, soal akan dicetak dalam sebuah batas dan diberi nomor soal. Selanjutnya, untuk menjawab pertanyaan, maka pemain akan memasukkan input jawaban melalui `scanf`. Input pemain akan masuk ke dalam array `jawabanUser`. Setelah jawaban disimpan, maka buffer input akan dibersihkan dan disiapkan untuk input selanjutnya.

Logika Pencocokan Jawaban

```
if (strcmp(jawabanUser, answers[i]) == 0) {  
    printf("Selamat jawaban Anda benar!\n\n");  
    score += 3;  
} else {  
    printf("Maaf jawaban Anda salah. Jawaban yang benar adalah %s\n\n", answers[i]);  
    score -= 1;  
    lives--;
```

Logika Pencocokan Jawaban

Ketika array jawabanUser telah terisi oleh input jawaban pada pertanyaan tersebut, maka input tersebut akan dibandingkan dengan array answers sesuai dengan nomor soal. Ketika input dibandingkan dengan kunci jawaban, apabila sesuai, maka score akan bertambah 3. Apabila tidak sesuai, maka score akan berkurang 1 dan lives juga berkurang 1

2.1.6

Logika Level 3

Deklarasi Variabel

```
int stage3() {  
    // Array untuk menyimpan soal yang dibaca dari file  
    char questions[MAX_SOAL_3][PANJANG_SOAL];  
  
    char jawabanUser[50];  
    int i;
```

Deklarasi Variabel

Hal pertama yang perlu dilakukan yaitu deklarasi variabel yang akan digunakan dalam logika stage 3 kali ini. Yang pertama adalah deklarasi array untuk pertanyaan bertipe char, dengan ukuran hasil perkalian definisi **MAX_SOAL_3** dan **PANJANG_SOAL**. Untuk berikutnya diperlukan deklarasi array jawabanUser bertipe char dengan ukuran 50 karakter. Array ini berfungsi untuk menyimpan input jawaban yang dimasukkan player / user. Selanjutnya, diperlukan deklarasi variabel i yang akan digunakan untuk iterasi.

Array Jawaban

```
char *answers[MAX_SOAL_3] = {  
    "3",           // Jawaban Soal 1  
    "105000",      // Jawaban Soal 2  
    "175",          // Jawaban Soal 3  
    "90",           // Jawaban Soal 4  
    "1400"          // Jawaban Soal 5  
};
```

Array Jawaban

lalu, dilakukan deklarasi dan inisialisasi pada array `answers` bertipe `char` dengan pointer dan berukuran sesuai dengan definisi `MAX_SOAL_3`. Array ini akan menjadi kunci jawaban dari pertanyaan yang ada pada stage 3. Array ini akan menjadi pembanding antara nilai jawaban yang diinput oleh pengguna dengan nilai jawaban pada array `answers` apakah nilai jawaban dari user sesuai dengan array `answers` atau tidak. Hasil perbandingan ini akan menentukan skor yang didapat oleh player.

Logika Membaca File Soal

```
// --- BACA SOAL DARI FILE ---
FILE *file = fopen("STAGE 3.txt", "r");

if (file == NULL) {
    printf("Error: Tidak bisa membuka file soal.txt.\n");
    printf("Pastikan file soal.txt ada di folder yang sama dengan program ini.\n");
    return 1;
}

for (i = 0; i < MAX_SOAL_3; i++) {
    if (fgets(questions[i], PANJANG_SOAL, file) == NULL) {
        break;
    }
}
fclose(file);
// --- SELESAI BACA FILE ---
```

Logika Membaca File Soal

berikutnya, agar program dapat menampilkan soal, program harus membaca dari file txt yang telah dibuat terlebih dahulu. Oleh karena itu, pertama-tama file pointer akan dideklarasikan lalu membuka file **STAGE 3.txt** yang ada di folder yang sama dengan program tersebut dengan perintah `read`. Kemudian agar program dapat mencetak soal pada layar, digunakan perulangan `for` untuk mencetak soal dari file baris per baris. Setelah selesai mencetak soal, maka file **STAGE 2.txt** akan ditutup

Logika Permainan / Menjawab Soal Pada stage 3

```
for (i = 0; i < MAX_SOAL_3; i++) {  
    // Cek nyawa  
    if (lives <= 0) return 0;  
  
    printf("Soal no. %d\n", i + 1);  
  
    //untuk merapikan di terminal  
    for (int k = 0; questions[i][k] != '\0'; k++) {  
        if (questions[i][k] == '|') {  
            printf("\n");  
        } else {  
            printf("%c", questions[i][k]);  
        }  
    }  
    printf("\n");  
  
    printf("Jawaban Anda: ");  
    scanf("%s", jawabanUser);  
    clearInputBuffer();  
    printf("\n");
```

Logika Permainan / Menjawab Soal Pada stage 3

Sebelum user dapat mengerjakan soal. Soal akan dicetak terlebih dahulu dalam sebuah batas dan diberi nomor soal. Selanjutnya, untuk menjawab pertanyaan, maka pemain akan memasukkan input jawaban melalui scanf. Setelah player mamasukkan input, input dari player akan masuk ke dalam array jawabanUser. Setelah jawaban disimpan, maka buffer input akan dibersihkan dan disiapkan untuk input selanjutnya.

Logika Pencocokan Jawaban Penampil Skor Akhir

```
printf("Jawaban Anda: ");
scanf("%s", jawabanUser);
clearInputBuffer();
printf("\n");

if (strcmp(jawabanUser, answers[i]) == 0) {
    printf("Selamat jawaban Anda benar!\n\n");
    score += 5;
} else {
    printf("Maaf jawaban Anda salah. Jawaban yang benar adalah %s\n\n", answers[i]);
    score -= 1;
    lives--;
}

delay(1000);

if (lives > 0) {
    printf("\nSelamat! Kamu berhasil menyelesaikan semua stage dengan skor: %d\n", score);
}

return 0;
}
```

Logika Pencocokan Jawaban Penampil Skor Akhir

Ketika array jawabanUser telah terisi oleh input jawaban dari player pada pertanyaan tersebut, maka input tersebut akan dibandingkan dengan array answers sesuai dengan nomor soal. Ketika input dibandingkan dengan kunci jawaban, apabila sesuai, maka score akan bertambah 3. Apabila tidak sesuai, maka score akan berkurang 1 dan lives juga berkurang 1. Lalu program akan menampilkan output berisi ucapan selamat serta skor akhir player ketika nyawa player masih tersisa.

2.1.7

Main Program

Loading Screen

```
int main() {  
  
    ketikPelan(" _ \W | / \W _ \W | \W | \W | / \W _ \W _ \W | \n", 1);  
    ketikPelan(" | \W | / _ \W | | | \W | / _ \W | \W | | | / \W | \n", 1);  
    ketikPelan(" | | | / _ \W | | | | \W | | | / _ \W | | | | \W | \n", 1);  
    ketikPelan(" | | | | / _ \W | | | | | \W | | | | | \W | | | \n", 1);  
    ketikPelan(" | | | | / / \W \W | | | | \W | | | / / \W \W | | | | \W | | | / \n", 1);  
    ketikPelan(" | | | | / / \W \W | | | | \W | | | / / \W \W | | | | \W | | | / \n", 1);  
    ketikPelan(" | | | | / / \W \W | | | | \W | | | / / \W \W | | | | \W | | | / \n", 1);  
    ketikPelan(" | | | | / / \W \W | | | | \W | | | / / \W \W | | | | \W | | | / \n", 1);  
    ketikPelan(" | | | | / / \W \W | | | | \W | | | / / \W \W | | | | \W | | | / \n", 1);  
    printf("\n"); // Baris kosong biasa (tidak perlu animasi)  
  
    ketikPelan("                                BY Kelompok 8                                \n", 20);  
    ketikPelan("                                Version 1.0.0 - December 2025                                \n", 20);  
    printf("\n");  
  
    ketikPelan("Press Enter to continue...", 50);  
    getchar();  
    clearScreen();  
  
    int pilihan;  
    char pilUlang;  
    srand(time(NULL));
```

Loading Screen

- **Intro Visual:** Program membuka tampilan dengan judul ASCII Art "**MATH GAMES ADVENTURE**" serta identitas kelompok yang ditampilkan menggunakan efek animasi teks bertahap (ketikPelan).
- **Transisi Menu:** Sistem menunggu input tombol "Enter" dari pengguna untuk masuk ke menu utama, kemudian layar dibersihkan (clearScreen) untuk persiapan tampilan antarmuka berikutnya.
- **Persiapan Variabel & Randomizer:** Dilakukan deklarasi variabel kontrol menu serta inisialisasi seed pengacak angka (srand) berbasis waktu sistem untuk menjamin variasi soal yang berbeda setiap sesi permainan.

Main Menu

```
do
{
    ketikPelan("=====GAME STARTED=====\n", 10);
    ketikPelan("Silakan pilih menu dibawah ini:\n");
    printf("1. Mulai Game\n");
    printf("2. High Score\n");
    printf("3. Exit\n");

    printf("\nPilihan Anda: ");
    scanf("%d", &pilihan);

    clearScreen();
}
```

Main Menu

- **Menu akan ditampilkan dengan headernya diberi animasi ketikpelan lalu ditampilkan pilihan 1–3.**
- **User akan diminta untuk menginputkan pilihannya yang akan disimpan dalam variabel pilihan. Terakhir, fungsi clearScreen dijalankan.**
- **Ada looping di menu ini, di mana selama pilihan tidak sama dengan 3 maka menu akan terus dicetak sampa user memilih menu 3 (exit).**

Alur Game

```
switch (pilihan)
{
case 1:
    // Memulai game
    lives = 3;
    score = 0;

    ketikPelan("Suatu hari anda terbangun di Gua misterius...\n", 10);
    delay(1000);
    ketikPelan("Tiba-tiba seorang Jin mendatangi anda.\n", 10);
    ketikPelan("\nPress Enter to continue...", 50);
    while(getchar() != '\n');
    getchar();
    clearScreen();
```

Alur Game

Ketika user memilih 1, maka game akan dimulai. Game dimulai dengan nyawa user diatur ke 3 dan skor 0. Lalu program akan menampilkan teks cerita sebelum masuk ke level 1 (ada delay(1000) untuk memberi jeda 1 detik). Setelah user menekan enter maka clearScreen akan dipanggil.

Alur Game

```
//New game
if(death == 0) {

    ketikPelan("Jin : Selamat datang di Gua Misterius!\n", 10);
    ketikPelan("Untuk keluar dari sini, kamu harus menyelesaikan tantangan matematika yang aku berikan.\n", 10);
    ketikPelan("Setiap jawaban yang benar akan memberimu poin, \n", 10);
    ketikPelan("tapi hati-hati, jawaban yang salah akan mengurangi nyawamu! Kamu punya 3 nyawa.\n", 10);

    printf("\nJawabanmu? (y/n): ");
    scanf(" %c", &pilUlang);
    clearScreen();
    if(pilUlang == 'n' || pilUlang == 'N') {
        ketikPelan("Jin: Baiklah, sampai jumpa lain waktu!\n", 10);
        delay(1000);
        clearScreen();
        break;
    }
}
```

Alur Game

Kemudian akan dicek apabila user pernah kalah sebelumnya, jika tidak (`death == 0`) maka cerita awal akan dimulai. Jika user menjawab N ketika ditanya sang Jin, maka program akan menampilkan pesan perpisahan dan berhenti. Jika user memilih Y maka program akan lanjut ke level 1

Alur Game

```
//Apabila sudah pernah mati
else {
    ketikPelan("Jin: Kita bertemu lagi nak, apakah kamu siap untuk mengulang tantanganku?\n", 10);
    printf("Jawabanmu? (y/n): ");
    scanf(" %c", &pilUlang);
    clearScreen();
    if(pilUlang == 'n' || pilUlang == 'N') {
        ketikPelan("Jin: 'Baiklah, sampai jumpa lain waktu! '\n", 10);
        delay(1000);
        clearScreen();
        break;
    }
}
```

Alur Game

Jika user pernah kalah sebelumnya, maka akan ada pesan unik di mana kita bertemu lagi dengan Jin. Untuk pertanyaan Jin, konsepnya sama seperti sebelumnya,

Alur Game

```
//level 1 start
ketikPelan("Game dimulai! Semoga beruntung!\n", 10);
printf("Anda memiliki %d nyawa dan skor awal %d.\n\n", lives, score);
delay(1000);
clearScreen();

printf("Soal No. 1 :\n");
soalPenjumlahan();
delay(1000);

if(lives <= 0) {
    goto cekGameOver;
    delay(1000);
}
```

Alur Game

Ketika game dimulai, user akan diberitahu dulu berapa nyawa dan skor awal mereka. Kemudian soal no.1 akan dicetak dengan memanggil fungsi `soalPenjumlahan`. Setiap soal akan dicek, apabila nyawanya kurang dari sama dengan 0 (kalah) maka akan menjalankan `goto cekGameOver` (akan dibahas nanti).

Alur Game

```
printf("\nSoal No. 2 :\n");
soalPenjumlahan();
delay(1000);

if(lives <= 0) {
    goto cekGameOver;
    delay(1000);
}

printf("\nSoal No. 3 :\n");
soalPengurangan();
delay(1000);

if(lives <= 0) {
    goto cekGameOver;
    delay(1000);
}

printf("\nSoal No. 4 :\n");
soalPengurangan();
delay(1000);

if(lives <= 0) {
    goto cekGameOver;
    delay(1000);
}

printf("\nSoal No. 5 :\n");
soalPerkalian();
delay(1000);

if(lives <= 0) {
    goto cekGameOver;
    delay(1000);
}

printf("\nSoal No. 6 :\n");
soalPerkalian();
delay(1000);

if(lives <= 0) {
    goto cekGameOver;
    delay(1000);
}

printf("\nSoal No. 7 :\n");
soalPerkalian();
delay(1000);

if(lives <= 0) {
    goto cekGameOver;
    delay(1000);
}

printf("\nSoal No. 8 :\n");
soalPembagian();
delay(1000);

if(lives <= 0) {
    goto cekGameOver;
    delay(1000);
}

printf("\nSoal No. 9 :\n");
soalPembagian();
delay(1000);

if(lives <= 0) {
    goto cekGameOver;
    delay(1000);
}

printf("\nSoal No. 10 :\n");
soalPembagian();
delay(1000);

if(lives <= 0) {
    goto cekGameOver;
    delay(1000);
}
```

Alur Game

User harus menjawab soal nomor 1–10 dengan nomor 1 dan 2 berupa penjumlahan, nomor 3 dan 4 pengurangan, nomor 5 sampai 7 perkalian, dan 8 sampai 10 pembagian. Setiap soal dijawab akan dicek terus apabila nyawa user telah habis.

Alur Game

```
clearScreen();
printf("\nSelamat! Level 1 Selesai.\n");
printf("\nPress enter to continue...");
getchar();
clearScreen();

ketikPelan("Jin : Selamat sudah menyelesaikan level pertama!\n", 10);
ketikPelan("Namun perjalananmu belum selesai, masih ada tantangan yang lebih sulit\n", 10);
ketikPelan("Apakah kamu siap untuk melanjutkan?", 10);
printf("\nPress enter to continue...");
getchar();
clearScreen();

//Level 2 start
ketikPelan("Level 2 dimulai!\n", 10);
printf("Anda memiliki %d nyawa dan skor %d.\n", lives, score);
delay(2000);
clearScreen();
```

Alur Game

Jika user menyelesaikan level 1 tanpa kehabisan nyawa, maka cerita akan berlanjut dan user harus menghadapi level 2. Sebelum dimulai, akan ditampilkan lagi nyawa dan skor user saat ini.

Alur Game

```
stage2();
delay(2000);

if(lives <= 0) {
    goto cekGameOver;
    delay(1000);
}

clearScreen();
printf("\nSelamat! Level 2 Selesai.\n");
printf("\nPress enter to continue...");
getchar();
clearScreen();

ketikPelan("Jin : Hebat, kamu sudah menyelesaikan level kedua!\n", 10);
ketikPelan("Sampailah kamu di level terakhir, semangat!\n", 10);
printf("\nPress enter to continue...");
getchar();
clearScreen();
```

Alur Game

- Fungsi stage2 akan dipanggil, dan jika user akan dicek jika nyawanya habis atau belum di soal ke-berapapun itu.
- Setelah menyelesaikan soal level 2 tanpa kehabisan nyawa, maka story akan berlanjut ke babak final, level 3.

Alur Game

```
//Level 3 start  
ketikPelan("Level 3 dimulai!\n", 10);  
printf("Anda memiliki %d nyawa dan skor %d.\n", lives, score);  
delay(2000);  
clearScreen();  
  
stage3();  
delay(2000);  
clearScreen();
```

Alur Game

Level 3 dimulai dengan menampilkan skor dan nyawa user. Kemudian fungsi stage3 akan dipanggil.

Alur Game

```
cekGameOver:  
delay(1000);  
clearScreen();  
  
if(lives > 0) {  
    ketikPelan("Jin : Nampaknya kamu telah menyelesaikan seluruh tantanganku.\n", 5);  
    ketikPelan("Selamat! Pintu keluar Gua ada di belakangku.", 5);  
    delay(2000);  
    clearScreen();  
  
    printf("=====\\n");  
    printf(" ANDA KELUAR DARI GUA DENGAN SELAMAT \\n");  
    printf("=====\\n");  
} else {  
    printf("Kamu pingsan dan terbangun kembali di awal gua.\n");  
    death = 1;  
}  
  
printf("Skor Akhir Anda: %d\\n", score);  
updateHighScore(score);  
  
printf("\\nPress enter to return to menu...");  
getchar();  
clearScreen();  
break;
```

Alur Game

- Setelah menyelesaikan level 3, maka akan memasuki cekGameOver. Jika user sempat kalah di level 1, 2, ataupun 3, maka program akan melompat ke bagian ini.
- Dijalankan logika sederhana, di mana jika nyawa user masih lebih dari 0, maka pesan selamat karena telah menyelesaikan game ditampilkan. Jika nyawa user kurang dari 0, maka akan muncul pesan kalah dan variabel death menjadi 1.
- Kemudian akan dicetak skor akhir kita, lalu dipanggil fungsi updateHighScore untuk mengecek apakah user layak masuk ke top 5. Game berhenti dan user bisa menekan enter untuk kembali ke menu.

Alur Game

```
case 2:  
    clearScreen();  
    showLeaderboard();  
    printf("\nPress enter to return to menu...");  
    while(getchar() != '\n');  
    getchar();  
    clearScreen();  
  
    break;
```

Alur Game

Ketika user memilih menu 2 maka akan clearScreen akan dipanggil lalu fungsi showLeaderBoard akan dipanggil juga. Jika sudah melihat highscorenya, maka user bisa menekan enter untuk kembali ke menu

Alur Game

```
case 3:  
    ketikPelan("Terima kasih telah bermain! Sampai jumpa lagi!\n", 10);  
    printf("\nPress enter to continue...");  
    while(getchar() != '\n');  
    getchar();  
    clearScreen();  
    exit(0);  
    break;  
  
default:  
    ketikPelan("Pilihan tidak valid. Silakan coba lagi.\n", 10);  
    break;  
}  
  
} while (pilihan != 3);  
return 0;  
}
```

Alur Game

- Ketika user memilih menu 3, akan ditampilkan pesan perpisahan dan program akan selesai ketika user menekan enter.
- Jika user memasukkan angka 1 dan 2 maka do while akan terus berjalan. Jika user memasukkan selain 1, 2 dan 3, maka akan muncul pesan pilihan tidak valid.

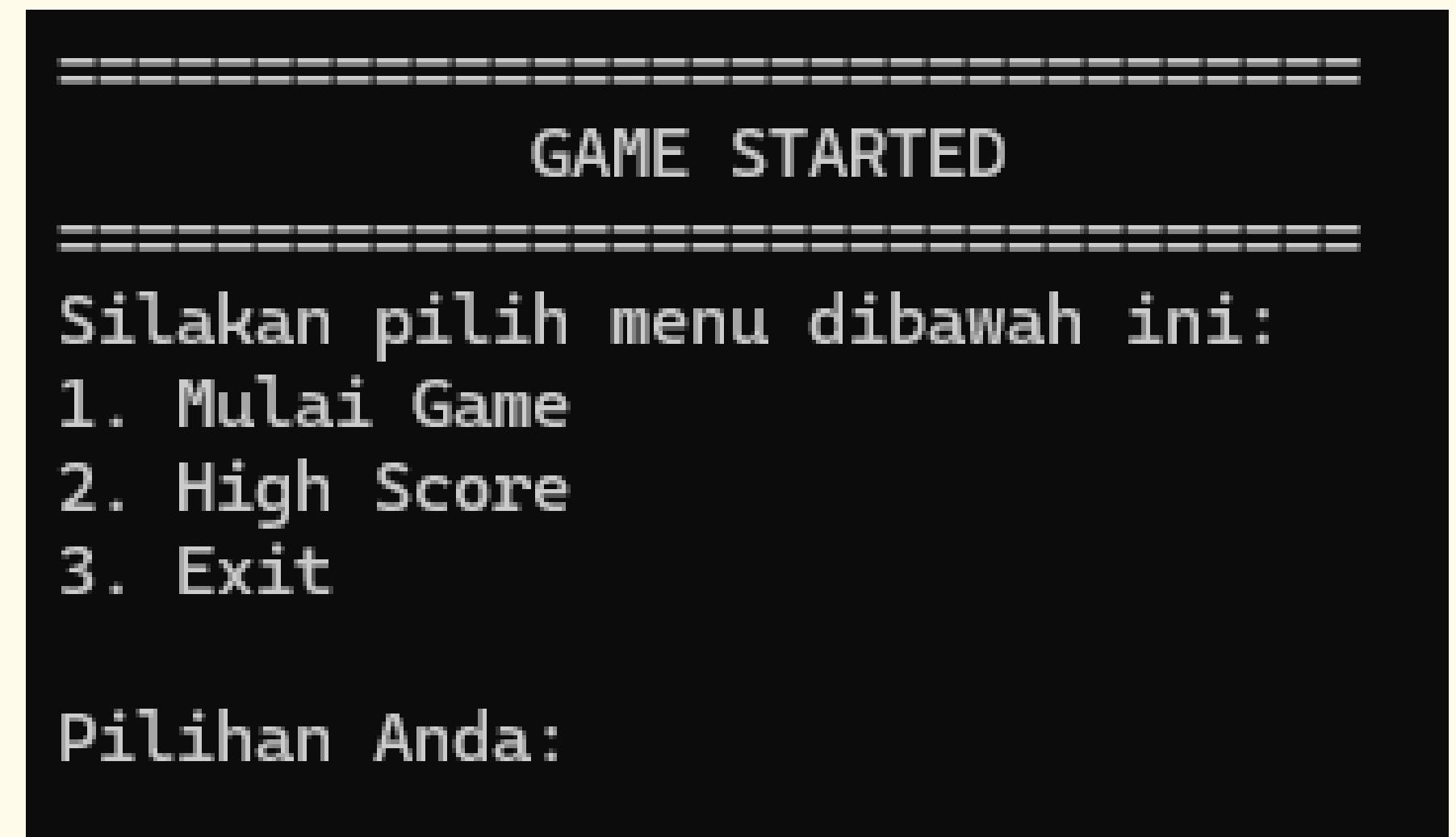
BAB III

Alur Program

3.1.1 Loading Screen



3.1.2 Menu Awal



3.1.3 High Score Saat Ini

A screenshot of a terminal window titled "highscore.txt". The window has a dark background and white text. It displays the following content:

```
File Edit View
=====
HAI DAR 46
Budyono 15
Empty 0
Empty 0
Empty 0
```

===== TOP 5 HIGH SCORE =====		
Rank	Username	Score
1.	HAI DAR	46
2.	Budyono	15
3.	Empty	0
4.	Empty	0
5.	Empty	0

Press enter to return to menu...

3.1.4 Mulai Game

Suatu hari anda terbangun di Gua misterius...
Tiba-tiba seorang Jin mendatangi anda.

Press Enter to continue...|

3.1.5 Alur Cerita Level 1

Jin : Selamat datang di Gua Misterius!
Untuk keluar dari sini, kamu harus menyelesaikan tantangan matematika yang aku berikan.
Setiap jawaban yang benar akan memberimu poin,
tapi hati-hati, jawaban yang salah akan mengurangi nyawamu! Kamu punya 3 nyawa.

Jawabanmu? (y/n): |

3.1.6 Soal Level 1

Game dimulai! Semoga beruntung!
Anda memiliki 3 nyawa dan skor awal 0.

Soal No. 1 :

Hasil dari $24 + -25$ adalah : -1

Selamat jawaban Anda benar!

Soal No. 2 :

Hasil dari $-38 + -83$ adalah : -121

Selamat jawaban Anda benar!

Soal No. 3 :

Hasil dari $-8 - -45$ adalah : 37

Selamat jawaban Anda benar!

Soal No. 4 :

Hasil dari $-85 - -73$ adalah : -12

Selamat jawaban Anda benar!

Soal No. 5 :

Hasil dari -7×4 adalah : -28

Selamat jawaban Anda benar!

Soal No. 6 :

Hasil dari -1×3 adalah : -3

Selamat jawaban Anda benar!

Soal No. 7 :

Hasil dari -3×-9 adalah : 27

Selamat jawaban Anda benar!

Soal No. 8 :

Hasil dari $8 : 8$ adalah : 1

Selamat jawaban Anda benar!

Soal No. 9 :

Hasil dari $0 : -4$ adalah : 0

Selamat jawaban Anda benar!

Soal No. 10 :

Hasil dari $-6 : 3$ adalah : -2

Selamat jawaban Anda benar!

3.1.7 User Menjawab Seluruh Level 1 Dengan Benar

Selamat! Level 1 Selesai.

Press enter to continue...|

3.1.8 Transisi Ke Level 2

Jin : Selamat sudah menyelesaikan level pertama!
Namun perjalananmu belum selesai, masih ada tantangan yang lebih sulit
Apakah kamu siap untuk melanjutkan?
Press enter to continue...

Level 2 dimulai!
Anda memiliki 3 nyawa dan skor 16.

3.1.9 Level 2 Dimulai

Soal no. 1
tentukan nilai x dari $2X - 6 = 2$
Jawaban Anda: 4

Selamat jawaban Anda benar!

Soal no. 2
tentukan nilai x dari $7X + 91 = 0$
Jawaban Anda: -13

Selamat jawaban Anda benar!

Soal no. 3
tentukan nilai x dari $5x + 27 = 9X - 21$
Jawaban Anda: 12

Selamat jawaban Anda benar!

Soal no. 4
tentukan volume kerucut dengan jari-jari = 3 meter,tinggi = 7 meter
Jawaban Anda: 66

Selamat jawaban Anda benar!

Soal no. 5
tentukan volume tabung dengan tinggi = 7 meter dan diameter = 10 meter
Jawaban Anda: 550

Selamat jawaban Anda benar!

Soal no. 6
tentukan luas permukaan balok dengan Panjang = 13 meter,tinggi = 7 meter, dan lebar = 4 meter
Jawaban Anda: 342

Selamat jawaban Anda benar!

Soal no. 7
tentukan luas segitiga siku siku dengan alas = 8 meter dan sisi miring = 17 meter
Jawaban Anda: 60

Selamat jawaban Anda benar!

Soal no. 8
tentukan luas persegi dengan panjang sisi = 5 meter
Jawaban Anda: 25

Selamat jawaban Anda benar!

3.1.10 User Menjawab Seluruh Level 2 Dengan Benar

Selamat! Level 2 Selesai.

Press enter to continue...|

3.1.11 Transisi Ke Level 3

Jin : Hebat, kamu sudah menyelesaikan level kedua!
Sampailah kamu di level terakhir, semangat!

Press enter to continue...

Level 3 dimulai!
Anda memiliki 3 nyawa dan skor 40.

3.1.12 Level 3 Dimulai

Soal no. 1

Jarak rumah Candra ke kampus adalah 2 km.

Setiap akan ke kampus, Candra mengendarai sepeda motor dengan kecepatan rata-rata 40 km/jam.

Waktu yang dibutuhkan Candra untuk sampai ke kampus adalah ... menit.

Jawaban Anda: 3

Selamat jawaban Anda benar!

Soal no. 2

Ani pergi ke suatu pusat perbelanjaan untuk mencari baju baru.

Ia tertarik pada sebuah kemeja dengan harga Rp. 150.000.

Pada toko tersebut, terdapat diskon untuk pembelian sebuah baju, sebesar 30%.

Uang yang harus dikeluarkan ani untuk membeli kemeja tersebut adalah... (jawaban tanpa titik)

Jawaban Anda: 105000

Selamat jawaban Anda benar!

Soal no. 3

Pipa air di suatu tempat mempunyai 354 cabang saluran untuk keperluan sehari-hari.

Setiap rumah terpasang 1 saluran air.

Dalam satu hari, pemakaian air mencapai 61.950 liter.

Rata-rata air yang digunakan di setiap rumah per hari adalah

Jawaban Anda: 175

Selamat jawaban Anda benar!

3.1.12 Level 3 Dimulai

Soal no. 4

Jono mendapatkan nilai 78, 87, 82, dan 93 untuk 4 mata kuliah.

Nilai yang harus diperoleh mata kuliah yang kelima agar nilai rata-rata Jono pada semester tersebut 86 adalah...

Jawaban Anda: 90

Selamat jawaban Anda benar!

Soal no. 5

Dalam sekali panen, Ahmad mampu mendapatkan 2 ton udang yang baik.

30% dari hasil tangkapannya ternyata kurang baik untuk dijual.

Jumlah hasil panen udang Ahmad yang layak dijual adalah kg.

Jawaban Anda: 1400

Selamat jawaban Anda benar!

Selamat! Kamu berhasil menyelesaikan semua stage dengan skor: 65

3.1.13 Ending Screen dan Pencatatan High Score

Jin : Nampaknya kamu telah menyelesaikan seluruh tantanganku.
Selamat! Pintu keluar Gua ada di belakangku.|

=====

ANDA KELUAR DARI GUA DENGAN SELAMAT

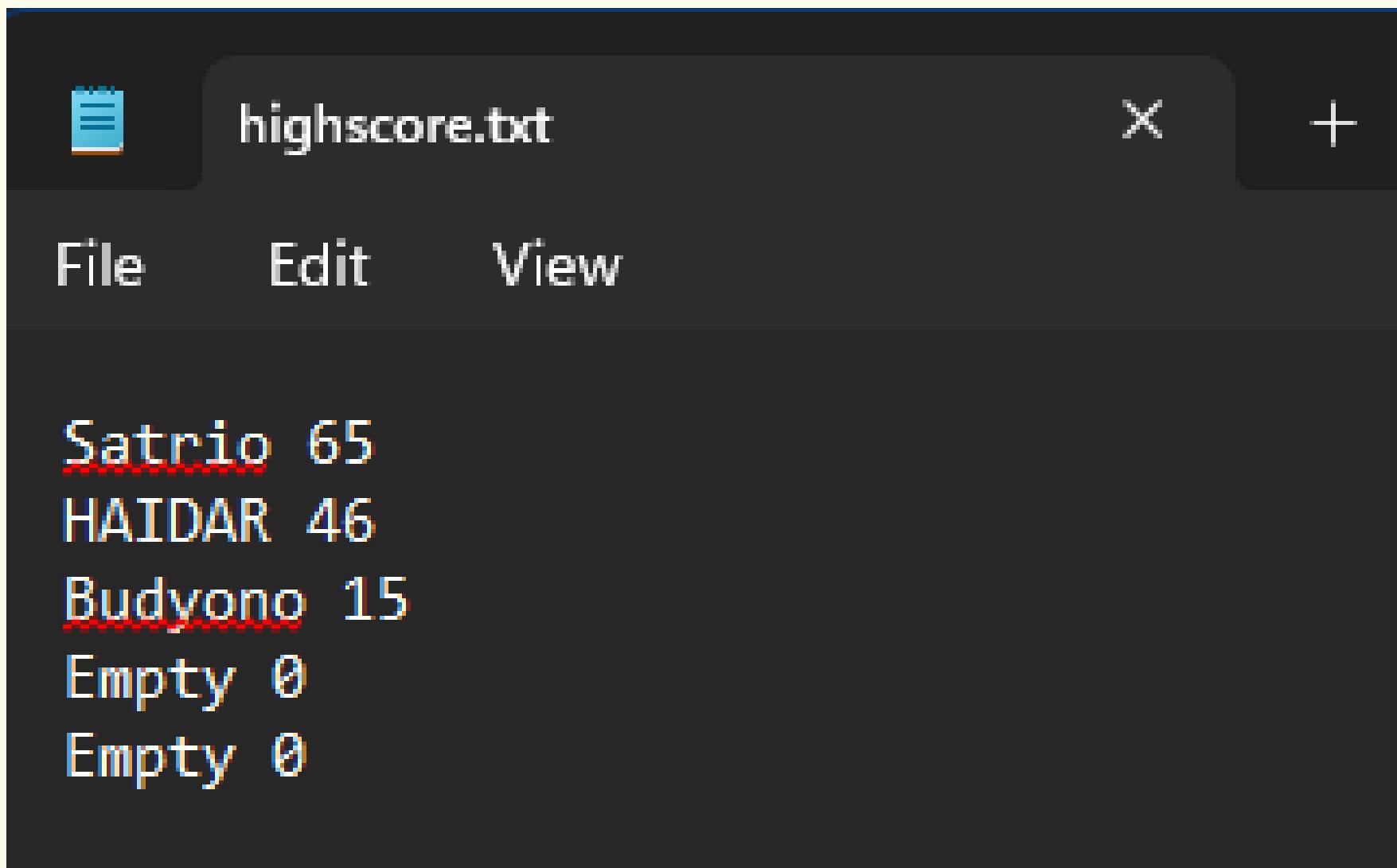
=====

Skor Akhir Anda: 65

Selamat! Anda mendapatkan High Score baru!

Masukkan username Anda (satu kata): Satrio|

3.1.14 High Score Baru



A screenshot of a terminal window titled "highscore.txt". The window has a dark background with light-colored text. At the top, there are icons for File, Edit, View, and a close button (X). Below the title, the text "===== TOP 5 HIGH SCORE =====" is displayed. The file content is as follows:

Rank	Username	Score
1.	Satrio	65
2.	HAIDAR	46
3.	Budyono	15
4.	Empty	0
5.	Empty	0

Press enter to return to menu...



A screenshot of a terminal window titled "===== TOP 5 HIGH SCORE =====". The window has a dark background with light-colored text. At the top, there are icons for File, Edit, View, and a close button (X). Below the title, the text "===== TOP 5 HIGH SCORE =====" is displayed. The file content is as follows:

Rank	Username	Score
1.	Satrio	65
2.	HAIDAR	46
3.	Budyono	15
4.	Empty	0
5.	Empty	0

Press enter to return to menu...

3.1.15 Jika User Kehabisan Nyawa

Kamu pingsan dan terbangun kembali di awal gua.
Skor Akhir Anda: 0
Maaf, Anda tidak masuk dalam Top 5 High Score.

Press enter to return to menu...

Kamu pingsan dan terbangun kembali di awal gua.
Skor Akhir Anda: 21
Selamat! Anda mendapatkan High Score baru!
Masukkan username Anda (satu kata):

3.1.16 Pesan Unik Ketika Mengulang Game Setelah Kalah

Jin: 'Kita bertemu lagi nak, apakah kamu siap untuk mengulang tantanganku?'
Jawabanmu? (y/n):

3.1.17 Menjawab N Ketika Ditanya Jin

Jin: 'Kita bertemu lagi nak, apakah kamu siap untuk mengulang tantanganku?'
Jawabanmu? (y/n): n
Jin: 'Baiklah, sampai jumpa lain waktu!'

3.1.18 Exit Program

Terima kasih telah bermain! Sampai jumpa lagi!

Press enter to continue...



BAB IV

Pembagian Tugas

4.1. Pembagian Tugas dalam Kelompok

1. Farrell Luthfi Adriansyah

Program :

- Merancang soal bertipe soal cerita pada stage 3**
- Merancang Logika Permainan dari Stage 3**
- Merancang logika penulisan soal cerita dengan file processing**
- Merancang logika file processing**

4.1. Pembagian Tugas dalam Kelompok

1. Farrell Luthfi Adriansyah

Makalah :

- Membuat bagian alur kerja dalam bentuk flowchart pada BAB I
- Menjelaskan bagian library yang digunakan pada program
- Menjelaskan bagian source code stage 3

4.1. Pembagian Tugas dalam Kelompok

1. Farrell Luthfi Adriansyah

PPT :

- Membuat bagian alur kerja dalam bentuk flowchart pada BAB I
- Menjelaskan bagian library yang digunakan pada program
- Menjelaskan bagian source code stage 3

4.1. Pembagian Tugas dalam Kelompok

2. Muhammad Haidar Amru

Program :

- Menyiapkan konstanta untuk menyimpan file highscore dengan #define FILE_HIGHSCORE.
- Menyiapkan variabel global nyawa, skor, dan death.
- Menyiapkan Struktur data high score beserta manajemen file untuk simpan, muat, dan update high score.
- Menyiapkan utility function untuk mengatur interface.
- Menyiapkan fungsi soal-soal untuk level 1.
- Menyiapkan main menu beserta storyline dari game.

4.1. Pembagian Tugas dalam Kelompok

2. Muhammad Haidar Amru

Makalah :

- Menjelaskan bagian source code dari konstanta, variabel global, struktur data, utility function, dan level 1.
- Menjelaskan Alur kerja program di BAB III

PPT :

- Menjelaskan bagian source code dari konstanta, variabel global, struktur data, utility function, dan level 1.
- Menjelaskan Alur kerja program di BAB III

4.1. Pembagian Tugas dalam Kelompok

3. Satrio Unggul Prayogo

Program :

- Merancang soal stage 2
- Merancang logika permainan stage 2
- Merancang logika file processing
- Merancang logika penulisan soal isian dengan file processing

4.1. Pembagian Tugas dalam Kelompok

3. Satrio Unggul Prayogo

Makalah :

- Menjelaskan bagian penjelasan umum
- Menjelaskan bagian source code stage 2
- Menjelaskan bagian kesimpulan
- Menjelaskan bagian saran
- Menjelaskan bagian refleksi

4.1. Pembagian Tugas dalam Kelompok

3. Satrio Unggul Prayogo

PPT :

- Menjelaskan bagian penjelasan umum
- Menjelaskan bagian source code stage 2
- Menjelaskan bagian kesimpulan
- Menjelaskan bagian saran
- Menjelaskan bagian refleksi

BAB V

Kesimpulan

5.1. Kesimpulan

Program permainan edukasi interaktif berjudul “Math Games Adventure” yang mengangkat tema petualangan matematika dan logika ini berhasil menggabungkan elemen pendidikan dan hiburan dengan mengembangkan metode pembelajaran yang menarik bagi pengguna.

5.2. Saran

Berdasarkan program yang telah kami buat, terdapat beberapa hal yang dapat dijadikan saran dalam pengembangan lebih lanjut :

- Pengelompokan peserta berdasarkan jenjang pendidikan.
- Pengembangan jenis soal yang lebih terstruktur dan sesuai dengan kurikulum sesuai dengan jenjang.
- Desain grafis permainan yang dapat dikembangkan menjadi berwarna dengan desain yang menarik sesuai cerita.

5.3. Refleksi

Dalam pengembangan program ini, metode sharing project yang kami gunakan masih kurang efektif karena terdapat beberapa kendala seperti koneksi yang terputus dan akses untuk percobaan program yang terbatas. Dengan pengembangan lebih lanjut, program ini dapat menjadi alat bantu pembelajaran yang menarik.



Terimakasih

Semoga bermanfaat dan dapat dipahami
dengan jelas