

**LAPORAN PROGRAM
WORLD SEARCH PUZZLE
MENGUNAKAN MULTI LINKED-LIST**

Dokumen Tugas Besar MK Struktur Data dan Algoritma (Praktik)



Disusun Oleh :

Kelompok 1, Kelas 1-A

Fitri Salwa

231524009

Zahra Hilyatul Jannah

231524031

**JURUSAN TEKNIK KOMPUTER DAN INFORMATIKA
PROGRAM STUDI D4 TEKNIK INFORMATIKA POLITEKNIK
NEGERI BANDUNG**

2024

DAFTAR ISI

DAFTAR ISI.....	i
DAFTAR GAMBAR	iii
BAB I ANALISIS DAN PERANCANGAN PROGRAM WORD SEARCH PUZZLE.	1
1.1 Definisi Program Word Search Puzzle.....	1
1.2 Fitur Program	2
1.3 Perancangan Tampilan (<i>User Interface</i>)	6
1.4 Perancangan Struktur Data.....	8
1.5 Contoh Instansiasi Data.....	10
BAB II PEMBUATAN KODE PROGRAM WORD SEARCH PUZZLE	13
2.1 Source Code Program	13
2.1.1 display.cpp	13
2.1.2 node.cpp	26
2.1.3 wordSearch.cpp.....	30
2.1.4 leaderboard.....	45
2.1.5 rules.cpp	51
2.2 Hasil <i>Output</i> Program	56
2.2.1 Bermain Game	56
2.2.2 Leaderboard.....	61
2.2.3 Aturan Bermain.....	61
BAB III PENGUJIAN	63
3.1 Hasil Pengujian Program	63

3.2	Kekurangan Program	71
BAB IV KESIMPULAN		72
DAFTAR PUSTAKA		73

DAFTAR GAMBAR

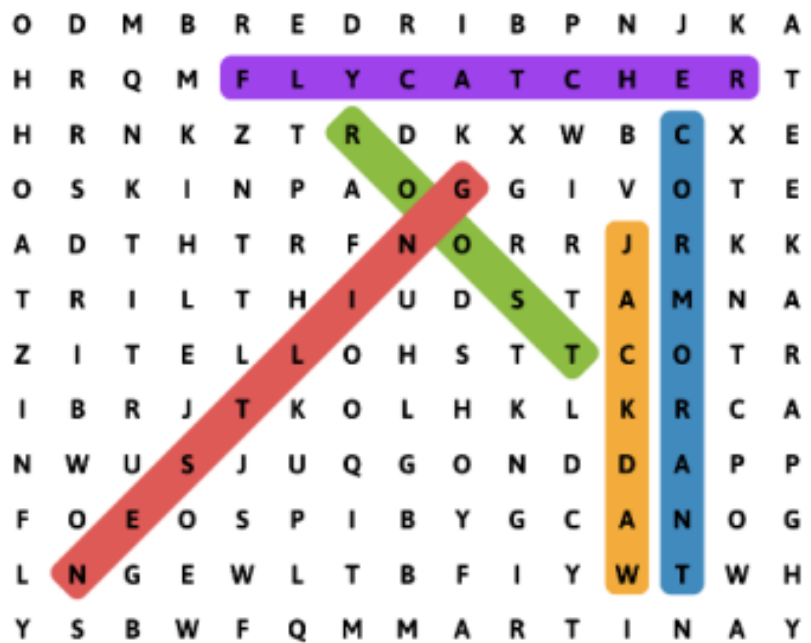
Gambar 1 Game Word Search Puzzle.....	1
Gambar 2 Rancangan Tampilan Main Menu	6
Gambar 3 Rancangan Tampilan Pemilihan Bahasa	6
Gambar 4 Rancangan Tampilan Pemilihan Mode	6
Gambar 5 Rancangan Tampilan Pemilihan Tema	7
Gambar 6 Rancangan Tampilan Papan	7
Gambar 7 Rancangan Tampilan Leaderboard.....	8
Gambar 8 Rancangan Tampilan Aturan Bermain	8
Gambar 9 Tampilan program input nama user	56
Gambar 10 Tampilan program pemilihan bahasa	56
Gambar 11 Tampilan program pemilihan mode	57
Gambar 12 Tampilan program pemilihan tema	57
Gambar 13 Tampilan papan mode easy	58
Gambar 14 Tampilan papan mode medium	58
Gambar 15 Tampilan papan mode hard	59
Gambar 16 Tampilan skor saat permainan dimulai	59
Gambar 17 Tampilan ketika kata yang dimasukkan benar	60
Gambar 18 Tampilan saat poin sudah bertambah	60
Gambar 19 Tampilan Leaderboard	61
Gambar 20 Tampilan pilihan bahasa untuk aturan permainan.....	61
Gambar 21 Tampilan aturan dalam bahasa Indonesia	62
Gambar 22 Tampilan aturan dalam bahasa Inggris	62

BAB I

ANALISIS DAN PERANCANGAN PROGRAM WORD SEARCH PUZZLE

1.1 Definisi Program Word Search Puzzle

Word Search Puzzle atau pencarian kata adalah permainan dimana sejumlah kata disembunyikan dan ditempatkan secara acak dalam sebuah grid kotak. Kata-kata tersebut ditempatkan secara horizontal, vertikal, dan diagonal. Tujuan dari teka teki ini adalah untuk menemukan dan menandai semua kata yang tersembunyi dalam papan.



Gambar 1 Game Word Search Puzzle

1.2 Fitur Program

Program Word Search Puzzle ini memiliki beragam fitur yang menyediakan pengalaman bermain yang menarik dan menantang sebagai berikut :

1. Bermain Game

Fitur bermain game memungkinkan pemain untuk menikmati aktivitas pencarian kata-kata dengan berbagai fitur tambahan yang tersedia.

a. Input Nama Pemain

Fitur input nama pemain mengharuskan setiap pemain untuk memasukkan nama mereka sebelum memulai permainan. Nama ini digunakan untuk mencatat skor dan menampilkan nama pemain di leaderboard, yang memungkinkan pemain untuk melihat peringkat mereka di antara pemain lain. Dengan mengidentifikasi pemain secara unik, fitur ini memastikan bahwa setiap skor yang diperoleh dicatat dengan benar dan membantu dalam membangun kompetisi yang sehat di antara para pemain.

b. Pemilihan Bahasa

Fitur pemilihan bahasa memberikan pemain opsi untuk memilih bahasa yang akan digunakan dalam permainan, biasanya mencakup Bahasa Indonesia dan Bahasa Inggris. Setelah memilih bahasa, semua instruksi, tema kata, dan antarmuka permainan akan disesuaikan dengan bahasa yang dipilih, sehingga pemain dapat bermain dengan kenyamanan maksimal sesuai preferensi bahasa mereka. Ini memungkinkan pemain dari berbagai latar belakang bahasa untuk menikmati permainan dengan mudah.

c. Mode Permainan

Fitur mode permainan menyediakan tingkat kesulitan game yang berbeda-beda disetiap tingkatnya, ada tiga mode permainan yaitu Mode Mudah (Easy), Sedang (Medium) dan Sulit (Hard), hal ini menawarkan

pengalaman bermain sesuai dengan tingkat keahlian dan preferensi pemain. Mode mudah cocok untuk pemula dengan kisi huruf yang kecil dan kata yang mudah untuk ditemukan, mode sedang menawarkan tantangan dengan kisi huruf lebih besar dan kata yang susah untuk ditemukan, sementara mode sulit menantang dengan kisi huruf yang besar dan kata yang semakin susah untuk ditemukan.

d. Pemilihan Tema Kata

Fitur pemilihan tema kata memungkinkan pemain untuk memilih kategori kata-kata yang ingin mereka cari dalam permainan. Tema kata dapat mencakup berbagai topik seperti buah-buahan, hewan, teknologi, elektronik, atau topik lainnya. Selain itu, tema permainan tersedia dalam 2 bahasa, bahasa Inggris dan bahasa Indonesia. Pilihan dari tema kata ini memengaruhi kata-kata yang tersembunyi dalam kisi huruf, menambah variasi dan keunikan dalam pengalaman bermain Word Search Puzzle ini.

e. Papan Puzzle setiap Level

i. Easy

Mode mudah menyediakan papan puzzle dengan ukuran grid kecil yaitu (8x8) dan kata-kata yang mudah ditemukan. Fitur ini dirancang untuk pemula atau pemain yang mencari pengalaman bermain yang ringan dan cepat. Grid yang lebih kecil dan kata-kata yang sederhana membuatnya mudah diakses dan dimengerti oleh semua pemain, terutama mereka yang baru memulai atau ingin bermain secara santai tanpa terlalu banyak tantangan.

ii. Medium

Mode sedang menawarkan papan puzzle dengan ukuran grid

sedang yaitu (10x10) dan kata-kata dengan tingkat kesulitan menengah. Fitur ini memberikan tantangan yang lebih besar daripada mode mudah, cocok untuk pemain yang memiliki sedikit pengalaman dan mencari permainan yang lebih menarik dan menantang. Ukuran grid yang lebih besar dan kata-kata yang lebih kompleks membuat pemain harus berpikir lebih keras untuk menemukan semua kata.

iii. Hard

Mode sulit menghadirkan papan puzzle dengan ukuran grid besar yaitu (12x12) dan kata-kata yang sulit ditemukan. Fitur ini menantang pemain dengan pengalaman bermain yang paling sulit di antara semua mode, cocok untuk pemain berpengalaman yang mencari tantangan maksimal. Grid besar dan kata-kata yang sangat tersembunyi membuat permainan lebih menantang, menguji keterampilan dan kesabaran pemain dalam menemukan semua kata yang ada.

f. Scoring

Fitur score dalam permainan Word Search Puzzle memberikan 100 poin kepada pemain setiap mereka berhasil menemukan satu kata. Skor pemain akan terakumulasi seiring dengan penemuan kata-kata dalam satu permainan, memberikan penghargaan atas upaya yang mereka lakukan untuk menyelesaikan permainan kata ini. Skor total yang diperoleh akan ditampilkan pada akhir permainan, dan dapat digunakan untuk mengukur kinerja pemain serta memberikan dorongan untuk mencapai skor yang lebih tinggi di sesi permainan berikutnya.

2. Leaderboard

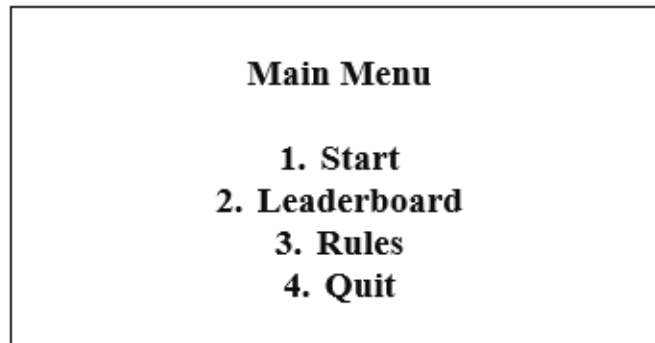
Fitur Leaderboard menampilkan 10 pemain dengan total skor tertinggi, memberikan tantangan tambahan kepada pemain untuk bersaing dan mencapai peringkat tertinggi dalam permainan. Leaderboard diperbarui setiap kali pemain mendapatkan skor baru, mencatat nama dan skor mereka secara berurutan berdasarkan skor tertinggi. Ini menciptakan elemen kompetitif dalam permainan, mendorong pemain untuk terus bermain dan meningkatkan skor mereka dengan harapan bisa mencapai posisi teratas di leaderboard.

3 Aturan Bermain

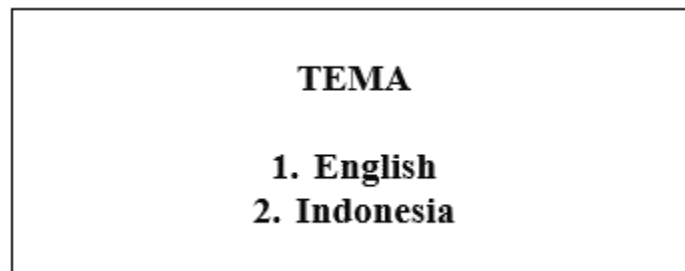
Fitur Aturan Bermain dalam permainan Word Search Puzzle memberikan pemahaman yang lengkap tentang bagaimana permainan dimainkan. Fitur aturan permainan tersedia dalam dua bahasa, yaitu bahasa Indonesia dan bahasa Inggris Dengan menampilkan aturan dalam bahasa yang dipilih oleh pemain, fitur ini memastikan bahwa semua pemain, terlepas dari latar belakang bahasa mereka, memahami cara bermain dan mengikuti aturan dengan benar. Panduan ini penting untuk memberikan pengalaman bermain yang adil dan menyenangkan bagi semua pemain.

Dengan kombinasi fitur-fitur ini, program Word Search Puzzle menawarkan pengalaman bermain yang seru dan menantang bagi para pemain.

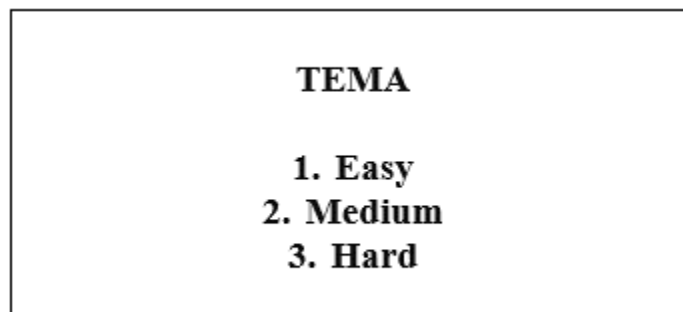
1.3 Perancangan Tampilan (*User Interface*)



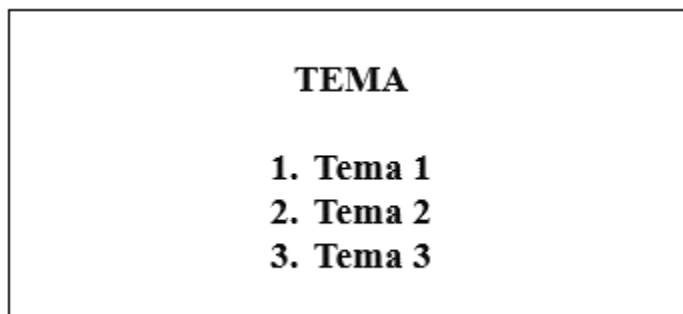
Gambar 2 Rancangan Tampilan Main Menu



Gambar 3 Rancangan Tampilan Pemilihan Bahasa



Gambar 4 Rancangan Tampilan Pemilihan Mode



Gambar 5 Rancangan Tampilan Pemilihan Tema

X	X	X	X	X	X	X	X	X	X
X	X	B	U	N	G	A	X	X	X
X	K	X	X	X	X	X	X	X	X
X	A	X	X	H	X	X	X	X	X
X	T	X	A	X	U	X	X	X	X
X	A	X	P	X	X	K	X	X	X
X	K	X	E	X	X	X	U	X	X
X	X	X	L	X	X	X	X	M	X
X	X	X	X	X	X	X	X	X	X
D	U	R	I	A	N	X	X	X	X

Gambar 6 Rancangan Tampilan Papan

LEADERBOARD		
No	Nama	Skor
1	Player5	11000
2	Player9	7600
3	Player7	5300
4	Player20	3900
5	Player2	1700
6	Player13	800

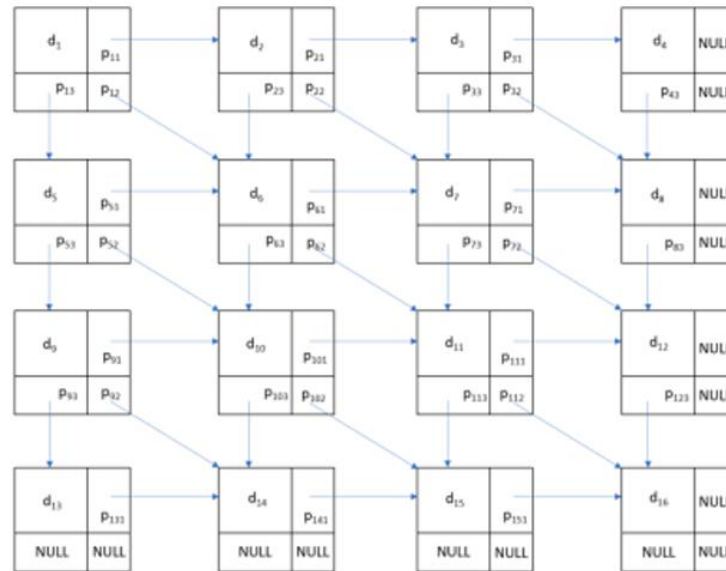
Gambar 7 Rancangan Tampilan Leaderboard

ATURAN	
1.	Aturan permainan yang pertama
2.	Aturan permainan yang kedua
3.	Aturan permainan yang ketiga
4.	Aturan permainan yang keempat
5.	Aturan permainan yang kelima
6.	dst

Gambar 8 Rancangan Tampilan Aturan Bermain

1.4 Perancangan Struktur Data

Struktur data yang digunakan dalam implementasi ini adalah multi linked list. Multi linked list adalah variasi dari linked list di mana setiap node memiliki lebih dari satu pointer yang menunjuk node lain. Dalam aplikasi Word Search Puzzle, setiap node dalam multi-linked list mewakili huruf dalam grid sedangkan pointer pada node menunjuk ke node-node tetangga dalam grid (horizontal, vertikal, dan diagonal). Berikut adalah ilustrasi dari struktur multi linked list yang akan digunakan pada aplikasi Word Search Puzzle.(Patel & Mahariba, 2020)



Berdasarkan hasil identifikasi terhadap data, berikut adalah rancangan struktur data pada program Word Search Puzzle ini.

1. Struktur data untuk Multi Linked List

```
typedef struct Node {
    char data;
    Node* right;
    Node* bottom;
    Node* cross;
} Node;
```

2. Struktur data untuk Papan

```
typedef struct puzzle {
    Node* head;
    int size;
} puzzle;
```

3. Struktur data untuk Scoring

```
typedef struct Player {
    char name[50];
    int correctWord;
    int score;
} Player;
```

1.5 Contoh Instansiasi Data

Berdasarkan rancangan struktur data, berikut adalah contoh instansiasi data pada program Word Search Puzzle ini.

1. Instansiasi struktur data untuk Multi Linked List

```
Node* allocation (){
    Node* node = (Node*) malloc(sizeof(Node));

    if (node == NULL) {
        printf ("\tMemory Penuh");
        exit(EXIT_FAILURE);
    }

    node->data = NULL;
    node->right = NULL;
    node->bottom = NULL;
    node->cross = NULL;

    return node;
}
```

```

void addNode(puzzle* board) {
    Node* node = allocation ();

    if (board->head == NULL){ //jika tidak ada elemen
        board->head = node;
    } else {
        node->right = board->head;
        board->head = node;
    }
}

```

2. Instansiasi Struktur data untuk Papan

```

puzzle board;
board.head = NULL;
board.size = 10;

```

```

switch (themeChoice) {
    case 1:
        if (languageChoice == 1)
            selectedKamus = easyKamusEN;
        else
            selectedKamus = easyKamusID;
            kamusSize = sizeof(easyKamusEN) /
sizeof(easyKamusEN[0]);
        board.size = 8;
        break;
    case 2:
        if (languageChoice == 1)
            selectedKamus = mediumKamusEN;
        else
            selectedKamus = mediumKamusID;
            kamusSize = sizeof(mediumKamusEN) /
sizeof(mediumKamusEN[0]);
        board.size = 10;
        break;
    case 3:
        if (languageChoice == 1)
            selectedKamus = hardKamusEN;
        else
            selectedKamus = hardKamusID;
            kamusSize = sizeof(hardKamusEN) /
sizeof(hardKamusEN[0]);
}

```

```

        board.size = 12;
        break;
    default:
        printf("Invalid choice.\n");
        return; // Keluar dari fungsi playGame

```

3. Instansiasi Struktur data untuk Scoring

```

player->score = 0; // Inisialisasi total skor
player->correctWord = 0; // Inisialisasi jumlah kata yang
benar

```

```

if (!found) {
    printCentered("AWESOME ^^", lastpuzzle + 4);
    getch();

    // Menambahkan jumlah kata yang benar
    player->correctWord++;
    // Menambahkan skor jika kata ditemukan
    player->score = player->score + 100;
    strcpy(findedWord[k], input_word);
    k++;
}

```


BAB II

PEMBUATAN KODE PROGRAM WORD SEARCH PUZZLE

2.1 Source Code Program

Source code kami dapat di akses pada link google drive dan link github berikut:

Gdrive : [Tubes_SDA_2024_Fitri_Zahra - Google Drive](#)

Github : <https://github.com/pworrins/Struktur-Data>

2.1.1 display.cpp

Nama Modul	gotoxy(int x, int y)
Fungsi	Fungsi untuk mengatur posisi kursor di konsol
Penanggung Jawab	Fitri Salwa
Parameter	x: Posisi horizontal. y: Posisi vertikal.
Source Code	
<pre>void gotoxy(int x, int y) { COORD coord; coord.X = x; coord.Y = y; SetConsoleCursorPosition(GetStdHandle(STD_OUTPUT_HANDLE), coord); }</pre>	

Nama Modul	getConsoleSize(int *width, int *height)
Fungsi	Fungsi untuk mendapatkan ukuran konsol
Penanggung Jawab	Fitri Salwa
Parameter	width: Pointer untuk menyimpan lebar konsol. height: Pointer untuk menyimpan tinggi konsol.
Source Code	
<pre>void getConsoleSize(int *width, int *height) { CONSOLE_SCREEN_BUFFER_INFO csbi; int columns, rows; GetConsoleScreenBufferInfo(GetStdHandle(STD_OUTPUT_HANDLE), &csbi); columns = csbi.srWindow.Right - csbi.srWindow.Left + 1; rows = csbi.srWindow.Bottom - csbi.srWindow.Top + 1; *width = columns; *height = rows; }</pre>	

Nama Modul	void printCentered(char *text, int y)
Fungsi	Menempatkan output berada di tengah konsol
Penanggung Jawab	Zahra Hilyatul Jannah
Parameter	text : pointer untuk menyimpan text yang akan di print y: posisi text secara vertikal
Source Code	
<pre>void printCentered(char *text, int y) { int consoleWidth, consoleHeight; getConsoleSize(&consoleWidth, &consoleHeight);</pre>	

```
// Hitung posisi horizontal untuk teks
int posX = (consoleWidth - strlen(text)) / 2;

// Menampilkan teks di tengah konsol
gotoxy(posX, y);
printf("%s", text);
}
```

Nama Modul	speed(float seconds);
Fungsi	Fungsi untuk mengatur kecepatan dalam game berdasarkan waktu yang diberikan.
Penanggung Jawab	Fitri Salwa
Parameter	seconds: Waktu dalam detik untuk menentukan kecepatan.
Source Code	
<pre>void speed(float seconds) { clock_t endwait; endwait = clock() + seconds * CLOCKS_PER_SEC; while (clock() < endwait); }</pre>	

Nama Modul	loading()
Fungsi	Fungsi untuk menampilkan animasi loading.
Penanggung Jawab	Fitri Salwa
Parameter	-
Source Code	
<pre>void loading() { system("cls");</pre>	

```

// loading
printCentered("Loading", 14);
printCentered("=====", 15);
printCentered("|", 16);
printCentered("=====", 17);

int consoleWidth, consoleHeight;
getConsoleSize(&consoleWidth, &consoleHeight);

int startPos = ((consoleWidth - 30) / 2) + 1; // Menghitung
posisi awal loop
int endPos = startPos + 26; // Menghitung posisi akhir loop

for (int i = startPos; i <= endPos; i++) {
    gotoxy(i, 16);
    printf("\xdb");
    Sleep(100); // Waktu jeda (ms)
}
}

```

Nama Modul	table()
Fungsi	Fungsi untuk menggambar tabel di tengah konsol
Penanggung Jawab	Zahra Hilyatul Jannah
Parameter	-
Source Code	
<pre> void table() { system("cls"); int tableWidth = 65; // Lebar tabel int tableHeight = 13; // Tinggi tabel pertama </pre>	

```

int consoleWidth, consoleHeight;

getConsoleSize(&consoleWidth, &consoleHeight);

    // Hitung offset untuk menempatkan tabel di tengah konsol
    secara horizontal
    int offsetX = (consoleWidth - tableWidth) / 2;
    int offsetY = 3; // Posisi vertikal tetap (misalnya, 3 baris
    dari atas)

    // Membuat kotak pertama (atas)
    // Garis atas
    gotoxy(offsetX, offsetY); printf("%c", 201);
    for (int i = 1; i < tableWidth - 1; i++) {
        printf("%c", 205);
    }
    printf("%c\n", 187);

    // Garis samping dan isi kotak pertama
    for (int i = 1; i < tableHeight; i++) {
        gotoxy(offsetX, offsetY + i); printf("%c", 186);
        gotoxy(offsetX + tableWidth - 1, offsetY + i); printf("%c",
186);
    }

    // Garis bawah kotak pertama (bagian atas kotak kedua)
    gotoxy(offsetX, offsetY + tableHeight); printf("%c", 204);
    for (int i = 1; i < tableWidth - 1; i++) {
        printf("%c", 205);
    }
    printf("%c\n", 185);

```

```

// Membuat kotak kedua (bawah)
// Garis samping kotak kedua
tableHeight = 10;
for (int i = 1; i < tableHeight; i++) {
    gotoxy(offsetX, offsetY + tableHeight + i); printf("%c",
186);
    gotoxy(offsetX + tableWidth - 1, offsetY + tableHeight +
i); printf("%c", 186);
}

// Garis bawah kotak kedua
gotoxy(offsetX, offsetY + 2 * tableHeight); printf("%c", 200);
for (int i = 1; i < tableWidth - 1; i++) {
    printf("%c", 205);
}
printf("%c\n", 188);
}

```

Nama Modul	header()
Fungsi	Fungsi untuk menampilkan header di konsol
Penanggung Jawab	Fitri Salwa
Parameter	-
Source Code	
<pre> void header() { table(); printCentered(" _ _ _ \n", 4); printCentered(" \ \ / / \n", 5); printCentered(" \ \ /\ \ / / _ _ _ \n", 6); printCentered(" \ \ \ \ \ / / _ \ \ ' _ / _ ` \n", 7); printCentered(" \ \ /\ \ / (_ (_ \n", 8); </pre>	

```
printCentered("  _\\/_\\/_\\_/_|_|_  _\\_,_|_  _  \\n", 9);
printCentered(" / ____|          | |  \\n", 10);
printCentered(" | (____ _ _ _ _ _|_|_  \\n", 11);
printCentered("  _\\_  _\\_/_ _\\_ _`|'_/_|'_ _\\_\\n", 12);
printCentered(" ____)|_/_/(_|_|_|(_|_|_|\\n", 13);
printCentered(" |____/_\\_\\_|\\_\\_,_|_|_  _\\_\\_|_|_|_\\n", 14);
}
```

Nama Modul	inputPlayerName(char* playerName)
Fungsi	Fungsi untuk meminta nama pemain
Penanggung Jawab	Fitri Salwa
Parameter	playerName: Pointer ke array karakter yang akan menyimpan nama pemain. */
Source Code	
<pre>void inputPlayerName(char* playerName) { int x = 48; gotoxy(40, 17); printf("WELCOME TO WORD SEARCH PUZZLE GAME "); gotoxy(x, 19); printf("Input your name: "); scanf("%s", playerName); }</pre>	

Nama Modul	BannerTheme()
Fungsi	Fungsi untuk menampilkan banner theme di konsol
Penanggung Jawab	Fitri Salwa
Parameter	-
Source Code	
<pre>void BannerTheme(){ printCentered(" _____ \n", 7);</pre>	

```

printCentered("|_ _| | \n", 8);
printCentered(" | | | |_ _ _ _ _ _ \n", 9);
printCentered(" | | | '_ \\ / _ \\ ' _ ` _ \\ / _ \\ \n", 10);
printCentered(" | | | | | | _/ | | | | | _/\n", 11);
printCentered(" \\ \\ / | | | | \\ \\ _ | | | | | \\ \\ _ |\n", 12);
printCentered(" \n", 13);
}

```

Nama Modul	displayTheme()
Fungsi	Fungsi untuk menampilkan menu pilihan tema dan mengembalikan pilihan pengguna
Penanggung Jawab	Fitri Salwa
Parameter	-

Source Code

```

int displayTheme(){
    table();
    BannerTheme();
    printCentered("1. Easy\n", 19);
    printCentered("2. Medium\n", 20);
    printCentered("3. Hard\n", 21);
    printCentered("Your Input: ", 18);

    int levelChoice;
    scanf("%d", &levelChoice);
    return levelChoice;
}

```

Nama Modul	displayLanguage()
Fungsi	Fungsi untuk menampilkan pilihan bahasa Mengembalikan nilai integer yang menunjukkan

	pilihan bahasa.
Penanggung Jawab	Fitri Salwa
Parameter	-
Source Code	
<pre>int displayLanguage(){ table(); BannerTheme(); printCentered("1. English\n", 19); printCentered("2. Indonesia\n", 20); printCentered("Your Input: ", 18); int LanguageChoice; scanf("%d", &LanguageChoice); return LanguageChoice; }</pre>	

Nama Modul	displayList(puzzle* board)
Fungsi	Fungsi untuk menampilkan list secara horizontal menggunakan pointer right
Penanggung Jawab	Zahra Hilyatul Jannah
Parameter	board: Pointer ke struktur puzzle yang berisi board.
Source Code	
<pre>void displayList(puzzle* board) { Node* currHead = board->head; Node* curr = currHead; printf("\n = = = = = B O A R D = = = = = \n "); printf("\n"); for (int i = 1; i <= board->size; i++){ curr = currHead; for (int j = 1; j <= board->size; j++){</pre>	

```

        printf("%c ", curr->data);
        curr = curr->right;
    }
    currHead = currHead->bottom;
    printf("\n");
}
printf("\n\n");
}

```

Nama Modul	displayPuzzle(puzzle* board, Player *player)
Fungsi	Fungsi untuk menampilkan puzzle di konsol
Penanggung Jawab	Zahra Hilyatul Jannah
Parameter	board: Pointer ke struktur puzzle yang berisi board.
Source Code	
<pre> void displayPuzzle(puzzle* board, Player *player){ system("cls"); Node* currHead = board->head; Node* curr = currHead; int consoleWidth, consoleHeight; getConsoleSize(&consoleWidth, &consoleHeight); int boxWidth = 4; // Lebar setiap kotak (termasuk garis vertikal) int boxHeight = 2; // Tinggi setiap kotak (termasuk garis horizontal) int size = board->size; // Hitung offset untuk menempatkan papan di tengah konsol int offsetX = (consoleWidth - (size * boxWidth)) / 2; </pre>	

```

int offsetY = (consoleHeight - (size * boxHeight)) / 2;

char puzzleTitle[] = " W O R D   S E A R C H   P U Z Z L E ";

// Hitung posisi horizontal untuk teks
int titlePosX = (consoleWidth - strlen(puzzleTitle)) / 2;

// Menampilkan teks di tengah konsol
gotoxy(titlePosX, offsetY-2);
printf("%s", puzzleTitle);

gotoxy(offsetX + size * boxWidth + 3, offsetY+1);
printf("%s's game ^^", player->name);
gotoxy(offsetX + size * boxWidth + 3, offsetY+3);
printf("You found %d words", player->correctWord);
gotoxy(offsetX + size * boxWidth + 3, offsetY+4);
printf("Total Score: %d", player->score);

gotoxy(offsetX + size * boxWidth + 3, offsetY+7);
printf("Press 'Q' to quit the game");

int i, j, k;

// Membuat papan dengan kotak 10x10
for (i = 0; i < size; i++) {
    for (j = 0; j < size; j++) {
        gotoxy(offsetX + boxWidth * j, offsetY + boxHeight *
i);

        if (i < size) {
            // Garis horizontal
            printf("%c", 197); // Titik persilangan
            for (k = 1; k < boxWidth; k++) {

```

```

        gotoxy(offsetX + boxWidth * j + k, offsetY +
boxHeight * i);
        printf("%c", 196); // Garis horizontal
    }
}

if (j < size) {
    // Garis vertikal
    for (k = 1; k < boxHeight; k++) {
        gotoxy(offsetX + boxWidth * j, offsetY +
boxHeight * i + k);
        printf("%c", 179); // Garis vertikal
    }
}

if (i < size && j < size) {
    // Huruf di dalam kotak
    gotoxy(offsetX + boxWidth * j + 1, offsetY +
boxHeight * i + 1); // Menyesuaikan posisi huruf di dalam kotak
    printf(" %c", curr->data);
    if (curr->right != NULL){
        curr = curr->right;
    } else {
        curr = currHead->bottom;
        currHead = currHead->bottom;
    }
}
}

// Membuat garis tepi kanan dan bawah papan
for (i = 0; i <= size * boxHeight; i++) {

```

```

        gotoxy(offsetX + size * boxWidth, offsetY + i);
        printf("%c", 186); // Garis vertikal tepi kanan
    }
    for (i = 0; i <= size * boxWidth; i++) {
        gotoxy(offsetX + i, offsetY + size * boxHeight);
        printf("%c", 205); // Garis horizontal tepi bawah
    }

    // Membuat garis tepi kiri dan atas papan
    for (i = 0; i <= size * boxHeight; i++) {
        gotoxy(offsetX, offsetY + i);
        printf("%c", 186); // Garis vertikal tepi kiri
    }
    for (i = 0; i <= size * boxWidth; i++) {
        gotoxy(offsetX + i, offsetY);
        printf("%c", 205); // Garis horizontal tepi atas
    }

    // Membuat sudut-sudut papan
    gotoxy(offsetX, offsetY);
    printf("%c", 201); // Sudut kiri atas
    gotoxy(offsetX + size * boxWidth, offsetY);
    printf("%c", 187); // Sudut kanan atas
    gotoxy(offsetX, offsetY + size * boxHeight);
    printf("%c", 200); // Sudut kiri bawah
    gotoxy(offsetX + size * boxWidth, offsetY + size * boxHeight);
    printf("%c", 188); // Sudut kanan bawah
}

```

2.1.2 node.cpp

Nama Modul	Node* allocation()
Fungsi	Fungsi untuk mengalokasikan memori untuk node baru dan menginisialisasinya
Penanggung Jawab	Zahra Hilyatul Jannah
Parameter	-
Source Code	
<pre>Node* allocation (){ Node* node = (Node*) malloc(sizeof(Node)); if (node == NULL) { printf ("\tMemory Penuh"); exit(EXIT_FAILURE); } node->data = NULL; node->right = NULL; node->bottom = NULL; node->cross = NULL; return node; }</pre>	

Nama Modul	addNode(puzzle* board)
Fungsi	Fungsi untuk menambahkan node ke board
Penanggung Jawab	Zahra Hilyatul Jannah
Parameter	board: Pointer ke struktur puzzle yang berisi board.
Source Code	
<pre>void addNode(puzzle* board) { Node* node = allocation ();</pre>	

```

    if (board->head == NULL){ //jika tidak ada elemen
        board->head = node;
    } else {
        node->right = board->head;
        board->head = node;
    }
}

```

Nama Modul	setRightLinks(puzzle* board)
Fungsi	Fungsi untuk mengatur tautan horizontal antar node
Penanggung Jawab	Zahra Hilyatul Jannah
Parameter	board: Pointer ke struktur puzzle yang berisi board.
Source Code	
<pre> void setRightLinks(puzzle* board) { int n = (board->size) * (board->size); for (int i = 1; i <= n; i++){ addNode(&*board); } } </pre>	

Nama Modul	setBottomLinks(puzzle* board)
Fungsi	Fungsi untuk mengatur tautan vertikal antar node
Penanggung Jawab	Zahra Hilyatul Jannah
Parameter	board: Pointer ke struktur puzzle yang berisi board.
Source Code	
<pre> void setBottomLinks(puzzle* board) { Node* currHead = board->head; Node* curr = currHead; </pre>	

```

int n = (board->size) * ((board->size) - 1);
for (int i = 1; i <= n; i++){
    curr = currHead;
    for (int j = 1; j <= board->size; j++){
        if (curr->right != NULL){
            curr = curr->right;
        }
    }
    currHead->bottom = curr;
    currHead = currHead->right;
}
}

```

Nama Modul	setDiagonalLinks(puzzle* board)
Fungsi	Fungsi untuk mengatur tautan diagonal antar node
Penanggung Jawab	Zahra Hilyatul Jannah
Parameter	board: Pointer ke struktur puzzle yang berisi board.
Source Code	
<pre> void setDiagonalLinks(puzzle* board) { Node* curr = board->head; Node* currHead = board->head; int n = (board->size) * ((board->size) - 1); for (int i = 1; i <= n; i++){ curr = currHead; for (int j = 1; j <= (board->size) + 1; j++){ if (curr->right != NULL){ curr = curr->right; } } } currHead->cross = curr; } </pre>	


```

        currHead = currHead->right;
    }

    currHead = board->head;
    curr = currHead;

    for (int i = 1; i < board->size; i++){
        curr = currHead;
        for (int j = 1; j < board->size; j++){
            if (curr != NULL){
                curr = curr->right;
            }
        }
        curr->cross = NULL;
        curr->right = NULL;
        currHead = currHead->bottom;
    }
}

```

Nama Modul	makePuzzle(puzzle* board)
Fungsi	ungsi untuk membuat puzzle dengan mengatur semua tautan node
Penanggung Jawab	Zahra Hilyatul Jannah
Parameter	board: Pointer ke struktur puzzle yang berisi board.
Source Code	
<pre> void makePuzzle (puzzle* board) { setRightLinks(*&board); setBottomLinks(*&board); setDiagonalLinks(*&board); } </pre>	

2.1.3 wordSearch.cpp

Nama Modul	playGame(Player *player, Player leaderboard[], int *leaderboardSize)
Fungsi	Fungsi untuk memulai permainan, memperbarui status pemain, dan memperbarui leaderboard.
Penanggung Jawab	Fitri Salwa
Parameter	<p>player: Pointer ke struktur Player yang mewakili pemain saat ini.</p> <p>leaderboard: Array dari struktur Player yang berisi pemain dalam leaderboard.</p> <p>leaderboardSize: Pointer ke integer yang menyimpan ukuran leaderboard.</p>
Source Code	
<pre>void playGame(Player *player, Player leaderboard[], int *leaderboardSize) { // Inisialisasi Struct Puzzle puzzle board; board.head = NULL; // Pilih bahasa int languageChoice = displayLanguage(); char kamusPath[MAX_FILENAME_LENGTH]; // Path untuk kamus kata berdasarkan pilihan bahasa switch (languageChoice) { case 1: strcpy(kamusPath, "Kamus/EN/"); // Untuk bahasa Inggris break; case 2: strcpy(kamusPath, "Kamus/ID/"); // Untuk bahasa Indonesia break; default:</pre>	

```

        printf("Invalid choice.\n");
        return; // Keluar dari fungsi playGame
    }

    // Pilih tema
    int themeChoice = displayTheme();
    char (*selectedKamus)[MAX_FILENAME_LENGTH]; // Pointer untuk
    memilih kamus kata berdasarkan tingkat kesulitan
    int kamusSize; // Ukuran kamus kata yang dipilih

    // Memilih kamus kata berdasarkan tingkat kesulitan yang
    dipilih
    switch (themeChoice) {
        case 1:
            if (languageChoice == 1)
                selectedKamus = easyKamusEN;
            else
                selectedKamus = easyKamusID;
            kamusSize = sizeof(easyKamusEN) /
sizeof(easyKamusEN[0]);
            board.size = 8;
            break;
        case 2:
            if (languageChoice == 1)
                selectedKamus = mediumKamusEN;
            else
                selectedKamus = mediumKamusID;
            kamusSize = sizeof(mediumKamusEN) /
sizeof(mediumKamusEN[0]);
            board.size = 10;
            break;
        case 3:
            if (languageChoice == 1)

```

```

        selectedKamus = hardKamusEN;
    else
        selectedKamus = hardKamusID;
        kamusSize = sizeof(hardKamusEN) /
sizeof(hardKamusEN[0]);
        board.size = 12;
        break;
    default:
        printf("Invalid choice.\n");
        return; // Keluar dari fungsi playGame
    }

    system("cls");
    table();
    int x = 50;
    int y = 19; // Mulai dari baris ke-19
    BannerTheme();
    for (int i = 0; i < kamusSize; ++i) {
        char buffer[50];
        sprintf(buffer, "%d. %s", i + 1, selectedKamus[i]);
        printCentered(buffer, y); // Mencetak nomor dan kata dengan
rata tengah
        y++; // Pindah ke baris berikutnya
    }
    printCentered("Your Choice: ", 18);
    int choice;
    scanf("%d", &choice);

    // Validasi pilihan tema
    if (choice < 1 || choice > kamusSize) {
        printf("Invalid theme choice.\n");
    }

```

```

    // Memuat tema dari file
    char filename[MAX_FILENAME_LENGTH]; // Nama file tema yang
    dipilih
    sprintf(filename, "%s\\%s.txt", kamusPath, selectedKamus[choice
- 1]);
    FILE *file;
    // Membuka file
    file = fopen(filename, "r");
    if (file == NULL) {
        printf("\n\n\n\tFailed to open the file.\n");
    }

    // Membaca kata-kata dari file dan menyimpannya dalam array
    char words[MAX_WORDS][50];
    int scores[MAX_WORDS] = {0}; // Array untuk menyimpan skor
    setiap kata
    int wordCount = 0;
    while (wordCount < MAX_WORDS && fscanf(file, "%49s",
words[wordCount]) == 1) {
        wordCount++;
    }

    // Menutup file
    fclose(file);

    // Mengacak urutan kata-kata dalam array
    srand((unsigned int)time(NULL)); // Menggunakan nilai hash dari
waktu saat ini sebagai bibit
    for (int i = wordCount - 1; i > 0; i--) {
        int j = rand() % (i + 1);
        if (i != j) {
            char temp[50];

```

```

        strcpy(temp, words[i]);
        strcpy(words[i], words[j]);
        strcpy(words[j], temp);
    }
}

// Menampilkan lima kata yang telah diacak ke dalam array baru
int numWordsToDisplay = 5; // Jumlah kata yang akan ditampilkan
char *searchWord[5];
for (int i = 0; i < numWordsToDisplay && i < wordCount; i++) {
    searchWord[i] = words[i];
}
printf("\n");

makePuzzle(&board);

if (board.size <= 10){
    for (int i = 0; i < 5; i++) {
        insertWordRandom(&board, searchWord[i]);
    }
} else {
    for (int i = 0; i < 5; i++) {
        insertWordRandom(&board, searchWord[i]);
    }
}

fillEmptySpacesWithRandomLetters(&board);

// Loop untuk mencari kata-kata
player->score = 0; // Inisialisasi total skor
player->correctWord = 0; // Inisialisasi jumlah kata yang benar

```

```

    while (player->correctWord < 5) { // Loop berakhir setelah lima
kata yang benar dimasukkan
        // Mencari kata yang dimasukkan oleh pengguna dalam array
kata
        displayPuzzle(&board, &*player);

        char input_word[50];
        char findedWord[5][50];

        int consoleWidth, consoleHeight;
        getConsoleSize(&consoleWidth, &consoleHeight);

        int boxWidth = 4; // Lebar setiap kotak (termasuk garis
vertikal)
        int boxHeight = 2; // Tinggi setiap kotak (termasuk garis
horizontal)
        int size = board.size;

        // Hitung offset untuk menempatkan papan di tengah konsol
        int offsetX = (consoleWidth - (size * boxWidth)) / 2;
        int offsetY = (consoleHeight - (size * boxHeight)) / 2;
        int lastpuzzle = offsetY + size * boxHeight;

        // Menampilkan teks di tengah konsol
        gotoxy(offsetX + size * boxHeight / 2 - 4, lastpuzzle+2);
        printf("Input Word : ");

        scanf("%s", input_word);
        int found = 0;

        // Keluar dari loop jika pengguna mengetik "q"
        if (strcmp(input_word, "Q") == 0)
            break;

```

```

        int titlePosX;
        // Mencari kata dalam array kata dan memperbarui skor
        int k = 0;
        if ((searchAndUpdateScore(words, scores, wordCount,
input_word)) && (findWord(&board, input_word))) {
            for (int i = 0; i < 5; i++) {
                if (strcmp(input_word, findedWord[i]) == 0) {
                    printCentered("You have found it O_O",
lastpuzzle + 4);

                    getch();
                    found = 1;
                }
            }
            if (!found) {
                printCentered("AWESOME ^^", lastpuzzle + 4);
                getch();

                player->correctWord++; // Menambahkan jumlah kata
yang benar
                player->score = player->score + 100; //
Menambahkan skor jika kata ditemukan
                strcpy(findedWord[k], input_word);
                k++;
            }
        } else {
            printCentered("TRY AGAIN O_O", lastpuzzle + 4);
            getch();
        }
    }

    int totalScore = player->score;
    // Menambahkan pemain ke leaderboard

```



```

        addToLeaderboard(leaderboard, leaderboardSize, player->name,
totalScore);
    }

```

Nama Modul	checkHorizontal(Node* node, const char* word)
Fungsi	Fungsi untuk mengecek kata secara horizontal dari node awal
Penanggung Jawab	Zahra Hilyatul Jannah
Parameter	node: Pointer ke node awal. word: Kata yang akan dimasukkan
Source Code	
<pre> int checkHorizontal(Node* node, const char* word) { while (*word && node) { if (node->data != *word) { return 0; } word++; node = node->right; } return *word == '\0'; } </pre>	

Nama Modul	checkVertical(Node* node, const char* word)
Fungsi	Fungsi untuk mengecek kata secara vertikal dari node awal
Penanggung Jawab	Zahra Hilyatul Jannah
Parameter	node: Pointer ke node awal. word: Kata yang akan dimasukkan
Source Code	
<pre> int checkVertical(Node* node, const char* word) { </pre>	

```

while (*word && node) {
    if (node->data != *word) {
        return 0;
    }
    word++;
    node = node->bottom;
}
return *word == '\0';
}

```

Nama Modul	checkDiagonal(Node* node, const char* word)
Fungsi	Fungsi untuk mengecek kata secara diagonal dari node awal
Penanggung Jawab	Zahra Hilyatul Jannah
Parameter	node: Pointer ke node awal. word: Kata yang akan dimasukkan

Source Code

```

int checkDiagonal(Node* node, const char* word) {
    while (*word && node) {
        if (node->data != *word) {
            return 0;
        }
        word++;
        node = node->cross;
    }
    return *word == '\0';
}

```

Nama Modul	findWord(puzzle* board, const char* word)
Fungsi	Fungsi untuk mencari kata di dalam board
Penanggung Jawab	Zahra Hilyatul Jannah

Parameter	board: Pointer ke struktur puzzle yang berisi board. word: Kata yang akan dicari.
Source Code	
<pre> int findWord(puzzle* board, const char* word) { Node* row = board->head; bool find = false; for (int i = 0; i < board->size; i++){ Node* current = row; while (current) { if (checkHorizontal(current, word) checkVertical(current, word) checkDiagonal(current, word)) { find = true; return find; } current = current->right; } row = row->bottom; } return find; } </pre>	

Nama Modul	checkSpaceHorizontal(Node* node, const char* word)
Fungsi	Fungsi untuk mengecek ruang kosong secara horizontal dari node awal
Penanggung Jawab	Zahra Hilyatul Jannah
Parameter	node: Pointer ke node awal. word: Kata yang akan dimasukkan.
Source Code	
<pre> int checkSpaceHorizontal(Node* node, const char* word) { while (*word && node) { </pre>	

```

        if (node->data != '\0') {
            return 0;
        }
        word++;
        node = node->right;
    }
    return *word == '\0';
}

```

Nama Modul	checkSpaceVertical(Node* node, const char* word)
Fungsi	Fungsi untuk mengecek ruang kosong secara vertikal dari node awal
Penanggung Jawab	Zahra Hilyatul Jannah
Parameter	node: Pointer ke node awal. word: Kata yang akan dimasukkan.

Source Code

```

int checkSpaceVertical(Node* node, const char* word) {
    while (*word && node) {
        if (node->data != '\0') {
            return 0;
        }
        word++;
        node = node->bottom;
    }
    return *word == '\0';
}

```

Nama Modul	checkSpaceDiagonal(Node* node, const char* word)
Fungsi	Fungsi untuk mengecek ruang kosong secara diagonal dari node awal
Penanggung Jawab	Zahra Hilyatul Jannah

Parameter	node: Pointer ke node awal. word: Kata yang akan dimasukkan.
Source Code	
<pre>int checkSpaceDiagonal(Node* node, const char* word) { while (*word && node) { if (node->data != '\0') { return 0; } word++; node = node->cross; } return *word == '\0'; }</pre>	

Nama Modul	insertWordHorizontal(Node* node, const char* word)
Fungsi	Fungsi untuk memasukkan kata secara horizontal dari node awal
Penanggung Jawab	Zahra Hilyatul Jannah
Parameter	node: Pointer ke node awal. word: Kata yang akan dimasukkan.
Source Code	
<pre>void insertWordHorizontal(Node* node, const char* word) { while (*word && node) { node->data = *word; word++; node = node->right; } }</pre>	

Nama Modul	insertWordVertical(Node* node, const char* word)
-------------------	---

Fungsi	Fungsi untuk memasukkan kata secara vertikal dari node awal
Penanggung Jawab	Zahra Hilyatul Jannah
Parameter	node: Pointer ke node awal. word: Kata yang akan dimasukkan.
Source Code	
<pre>void insertWordVertical(Node* node, const char* word) { while (*word && node) { node->data = *word; word++; node = node->bottom; } }</pre>	

Nama Modul	insertWordDiagonal(Node* node, const char* word)
Fungsi	Fungsi untuk memasukkan kata secara diagonal dari node awal
Penanggung Jawab	Zahra Hilyatul Jannah
Parameter	node: Pointer ke node awal. word: Kata yang akan dimasukkan.
Source Code	
<pre>void insertWordDiagonal(Node* node, const char* word) { while (*word && node) { node->data = *word; word++; node = node->cross; } }</pre>	

Nama Modul	insertWordRandom(puzzle* board, const char* word)
-------------------	--

Fungsi	Fungsi untuk memasukkan kata secara acak ke dalam board
Penanggung Jawab	Zahra Hilyatul Jannah
Parameter	node: Pointer ke struktur puzzle yang berisi board. word: Kata yang akan dimasukkan.
Source Code	
<pre> void insertWordRandom(puzzle* board, const char* word) { int row, col; Node* node; int direction; srand((unsigned int)time(NULL)); do { row = rand() % (board->size); col = rand() % (board->size - strlen(word) + 1); direction = ((rand()) * row) % 3; node = board->head; for (int i = 0; i < row; i++) { node = node->bottom; } for (int j = 0; j < col; j++) { node = node->right; } } while ((direction == 0 && !checkSpaceVertical (node, word)) (direction == 1 && !checkSpaceDiagonal (node, word)) (direction == 2 && !checkSpaceHorizontal(node, word))); if (direction == 0) { insertWordVertical (node, word); } else if (direction == 1) { </pre>	

```

        insertWordDiagonal (node, word);
    } else {
        insertWordHorizontal(node, word);
    }
}

```

Nama Modul	fillEmptySpacesWithRandomLetters(puzzle* board)
Fungsi	Fungsi untuk mengisi ruang kosong di board dengan huruf acak
Penanggung Jawab	Zahra Hilyatul Jannah
Parameter	board: Pointer ke struktur puzzle yang berisi board
Source Code	
<pre> void fillEmptySpacesWithRandomLetters(puzzle* board) { Node* currHead = board->head; Node* curr = currHead; for (int i = 0; i < board->size; i++) { curr = currHead; for (int j = 0; j < board->size; j++) { if (curr->data == '\0') { curr->data = 'A' + rand() % 26; // Mengisi dengan huruf acak dari 'A' hingga 'Z' } curr = curr->right; } currHead = currHead->bottom; } } </pre>	


```

    for (int i = 0; i < leaderboardSize; ++i) {
        gotoxy(40, 11 + i); printf("%d.", i + 1);
        gotoxy(55, 11 + i); printf("%s", leaderboard[i].name);
        gotoxy(75, 11 + i); printf(" %d", leaderboard[i].score);
    }

    // Tampilkan informasi terupdate
    time_t rawtime;
    struct tm *info;
    char buffer[80];
    time(&rawtime);
    info = localtime(&rawtime);
    strftime(buffer, 80, "%Y-%m-%d", info);
    gotoxy(x, 12 + leaderboardSize); printf("Leaderboard updated
on: %s", buffer);
}

```

Nama Modul	searchAndUpdateScore(char words[][50], int scores[], int wordCount, char* target)
Fungsi	Fungsi untuk mencari kata dalam array kata dan memperbarui skornya
Penanggung Jawab	Fitri Salwa
Parameter	words: Array kata yang dicari. scores: Array skor yang akan diperbarui. wordCount: Jumlah kata dalam array words. target: Kata yang dicari.
Source Code	
<pre> int searchAndUpdateScore(char words[][50], int scores[], int wordCount, char* target) { for (int i = 0; i < wordCount; ++i) { if (strcmp(words[i], target) == 0) { // Kata ditemukan, tambahkan skor </pre>	

```

        scores[i]++;
        return 1; // Mengembalikan 1 jika kata ditemukan
    }
}
return 0; // Mengembalikan 0 jika kata tidak ditemukan
}

```

Nama Modul	calculateTotalScore(int scores[], int wordCount)
Fungsi	Fungsi untuk menghitung total skor dari semua kata
Penanggung Jawab	Fitri Salwa
Parameter	scores: Array skor kata. wordCount: Jumlah kata dalam array scores.
Source Code	
<pre> int calculateTotalScore(int scores[], int wordCount) { int totalScore = 0; for (int i = 0; i < wordCount; ++i) { totalScore += scores[i]; } return totalScore; } </pre>	

Nama Modul	addToLeaderboard(Player leaderboard[], int* leaderboardSize, char* name, int score)
Fungsi	Fungsi untuk menambahkan pemain ke leaderboard
Penanggung Jawab	Fitri Salwa
Parameter	leaderboard: Array dari struktur Player yang berisi leaderboard. leaderboardSize: Pointer ke jumlah pemain dalam leaderboard. name: Nama pemain yang akan ditambahkan. score: Skor pemain yang akan ditambahkan.

Source Code

```

void addToLeaderboard(Player leaderboard[], int* leaderboardSize,
char* name, int score) {
    // Cek apakah pemain sudah ada di leaderboard
    for (int i = 0; i < *leaderboardSize; ++i) {
        if (strcmp(leaderboard[i].name, name) == 0) {
            // Update skor jika pemain sudah ada
            leaderboard[i].score += score;
            return;
        }
    }
    // Jika pemain belum ada, tambahkan ke leaderboard
    if (*leaderboardSize < MAX_LEADERBOARD_SIZE) {
        strcpy(leaderboard[*leaderboardSize].name, name);
        leaderboard[*leaderboardSize].score = score;
        (*leaderboardSize)++;
    } else {
        // Jika leaderboard penuh, gantikan pemain dengan skor
        // terendah jika perlu
        int minIndex = 0;
        for (int i = 1; i < MAX_LEADERBOARD_SIZE; ++i) {
            if (leaderboard[i].score < leaderboard[minIndex].score)
            {
                minIndex = i;
            }
        }
        if (score > leaderboard[minIndex].score) {
            strcpy(leaderboard[minIndex].name, name);
            leaderboard[minIndex].score = score;
        }
    }
    // Urutkan leaderboard berdasarkan skor (dari yang terbesar ke
    // terkecil)

```

```

        for (int i = 0; i < *leaderboardSize - 1; ++i) {
            for (int j = i + 1; j < *leaderboardSize; ++j) {
                if (leaderboard[j].score > leaderboard[i].score) {
                    Player temp = leaderboard[i];
                    leaderboard[i] = leaderboard[j];
                    leaderboard[j] = temp;
                }
            }
        }
    }
}

```

Nama Modul	saveLeaderboard(Player leaderboard[], int leaderboardSize)
Fungsi	Fungsi untuk menyimpan leaderboard ke file teks
Penanggung Jawab	Fitri Salwa
Parameter	leaderboard: Array dari struktur Player yang berisi leaderboard. leaderboardSize: Jumlah pemain dalam leaderboard.
Source Code	
<pre> void saveLeaderboard(Player leaderboard[], int leaderboardSize) { FILE *file = fopen("Kamus/leaderboard.txt", "w"); if (file == NULL) { printf("\n\n\n Gagal membuka file leaderboard.\n"); return; } time_t rawtime; struct tm *info; char buffer[80]; </pre>	

```

    time(&rawtime);
    info = localtime(&rawtime);
    strftime(buffer, 80, "%Y-%m-%d", info);
    fprintf(file, "Leaderboard updated on: %s\n", buffer);
    for (int i = 0; i < leaderboardSize; ++i) {
        fprintf(file, "%s %d\n", leaderboard[i].name,
leaderboard[i].score);
    }
    fclose(file);
}

```

Nama Modul	loadLeaderboard(Player leaderboard[], int* leaderboardSize)
Fungsi	Fungsi untuk memuat leaderboard dari file teks
Penanggung Jawab	Fitri Salwa
Parameter	leaderboard: Array dari struktur Player yang berisi leaderboard. leaderboardSize: Pointer ke jumlah pemain dalam leaderboard

Source Code

```

void loadLeaderboard(Player leaderboard[], int* leaderboardSize) {
    FILE *file = fopen("Kamus/leaderboard.txt", "r");
    if (file == NULL) {
        printf("\n\n\n          File leaderboard tidak ditemukan,
leaderboard kosong.\n");
        return;
    }
    char buffer[100];
    fgets(buffer, sizeof(buffer), file); // baca tanggal terupdate
    while (*leaderboardSize < MAX_LEADERBOARD_SIZE && fscanf(file,
"%s %d",
leaderboard[*leaderboardSize].name,
&leaderboard[*leaderboardSize].score) == 2) {
        (*leaderboardSize)++;
    }
}

```

```

    }
    fclose(file);
}

```

2.1.5 rules.cpp

Nama Modul	BannerRules()
Fungsi	Fungsi untuk menampilkan banner yang akan digunakan di leaderboard
Penanggung Jawab	Fitri Salwa
Parameter	-
Source Code	
<pre> void BannerRules() { printCentered(" _____ \n", 4); printCentered(" _ \ \ \n", 5); printCentered(" \ \ / _ _ _ _ _ \n", 6); printCentered(" _ / _ ` ' _ ` _ \ \ / _ \ \n", 7); printCentered(" \ \ \ \ (_ _ / \n", 8); printCentered(" \ \ _ / \ \ _ , _ \ \ _ \n", 9); printCentered(" _____ _ \n", 10); printCentered(" _ \ \ \n", 11); printCentered(" / / _ _ _ \n", 12); printCentered(" _ / / _ \ \ / _ \n", 13); printCentered(" \ \ _ _ / \ \ _ \ \n", 14); printCentered(" \ \ \ \ \ \ _ , _ \ \ _ _ / \n", 15); } </pre>	

Nama Modul	Rules()
Fungsi	Fungsi untuk menampilkan banner bertuliskan “Rules”
Penanggung Jawab	Zahra Hilyatul Jannah

Parameter	-
Source Code	
<pre> void Rules () { printCentered("_____ _ \n", 1); printCentered(" __ \ \ \n", 2); printCentered(" _/ / _ __ _ \n", 3); printCentered(" / /_ _ \ \/_ \n", 4); printCentered(" \ \ \ _ _/\ _ \ \n", 5); printCentered("\ _ \ \ \ _, _ \ _ _/\n", 6); printCentered(" = = = = = \n", 7); } </pre>	

Nama Modul	displayRulesIndonesian()
Fungsi	Fungsi untuk menampilkan aturan dalam bahasa Indonesia
Penanggung Jawab	Fitri Salwa
Parameter	-
Source Code	
<pre> void displayRulesIndonesian() { system("cls"); Rules(); printCentered("Selamat datang di Word Search Puzzle!\n", 11); printCentered("Word Search Puzzle adalah permainan di mana Anda mencari kata yang tersembunyi di dalam grid huruf.\n", 12); printCentered("Berikut adalah aturan permainan:\n", 13); printCentered("1. Pilih Tema Bahasa.\n", 14); printCentered("2. Pilih tingkat kesulitan: Mudah, Sedang, atau Sulit\n", 15); printCentered("3. Pilih tema permainan.\n", 16); printCentered("4. Cari kata-kata yang tersembunyi di dalam grid huruf.\n", 17); } </pre>	


```

        printCentered("5. Ketikkan kata yang ditemukan untuk mencari
        tahu apakah itu benar.\n", 18);

        printCentered("6. Jika kata yang dimasukkan benar dan ditemukan
        di dalam grid, Anda akan mendapatkan skor 100.\n", 19);

        printCentered("7. Permainan berakhir ketika anda menekan huruf
        q pada pilihan atau\n", 20);

        printCentered("8. Permainan berakhir ketika semua kata telah
        ditemukan.\n", 21);

        printCentered("9. Skor Anda akan ditampilkan di
        leaderboard.\n", 22);

        printCentered("10. Nikmati bermain Word Search Puzzle!\n", 23);
    }

```

Nama Modul	displayRulesEnglish()
Fungsi	Fungsi untuk menampilkan aturan dalam bahasa Inggris
Penanggung Jawab	Fitri Salwa
Parameter	-
Source Code	
<pre> void displayRulesEnglish() { system("cls"); Rules(); printCentered("Welcome to Word Search Puzzle!\n", 11); printCentered("Word Search Puzzle is a game where you search for hidden words in a grid of letters.\n", 12); printCentered("Here are the game rules:\n", 13); printCentered("1. Choose the Language Theme.\n", 14); printCentered("2. Choose the difficulty level: Easy, Medium, or Hard.\n", 15); printCentered("3. Select the game theme.\n", 16); printCentered("4. Find the hidden words in the grid of letters.\n", 17); printCentered("5. Type the found words to check if they are correct.\n", 18); } </pre>	

```

        printCentered("6. If the entered word is correct and found in
the grid, you will earn a score of 100.\n", 19);

        printCentered("7. The game ends when you press the letter 'q'
on the selection or\n", 20);

        printCentered("8. The game ends when all words have been
found.\n", 21);

        printCentered("9. Your score will be displayed on the
leaderboard.\n", 22);

        printCentered("10. Enjoy playing Word Search Puzzle!\n", 23);
    }

```

Nama Modul	mainRules()
Fungsi	Fungsi utama untuk menjalankan aturan Mengembalikan nilai integer yang menunjukkan hasil eksekusi.
Penanggung Jawab	Fitri Salwa
Parameter	-

Source Code

```

int mainRules() {
    system("cls");
    table();
    BannerRules();
    int languageChoice;
    int x = 46;
    // Pilihan bahasa
    printCentered("Pilih bahasa / Choose language:\n", 19);
    printCentered("1. Bahasa Indonesia\n", 20);
    printCentered("2. English\n", 21);
    printCentered("Your choice: ", 18);
    scanf("%d", &languageChoice);
    system("cls");
    // Menampilkan aturan permainan sesuai dengan bahasa yang

```

```
dipilih
    switch(languageChoice) {
        case 1:
            system("cls");
            displayRulesIndonesian();
            break;
        case 2:
            system("cls");
            displayRulesEnglish();
            getch();
            break;
        default:
            printf("Pilihan bahasa tidak valid.\n");
    }

    return 0;
}
```

2.2 Hasil *Output* Program

2.2.1 Bermain Game

a. Input Nama Pemain



Gambar 9 Tampilan program input nama user

b. Pemilihan Bahasa



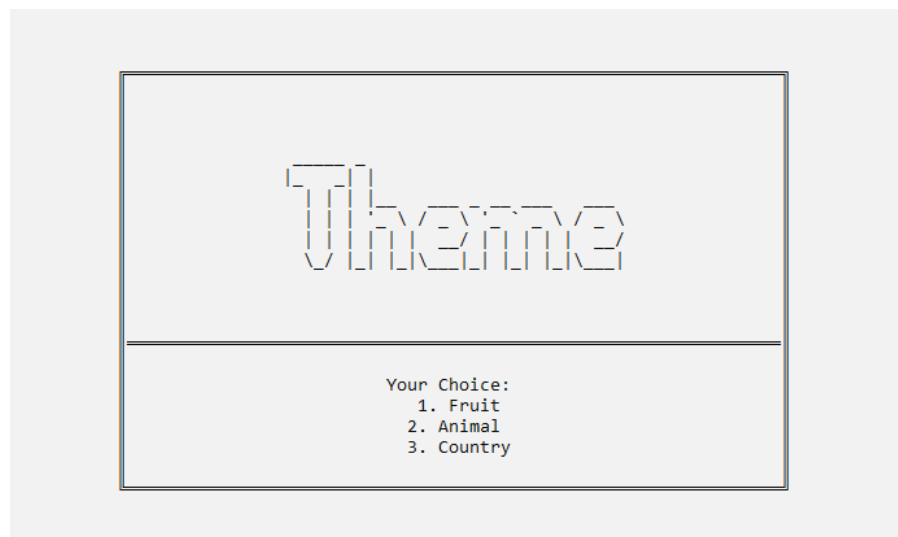
Gambar 10 Tampilan program pemilihan bahasa

c. Mode Permainan



Gambar 11 Tampilan program pemilihan mode

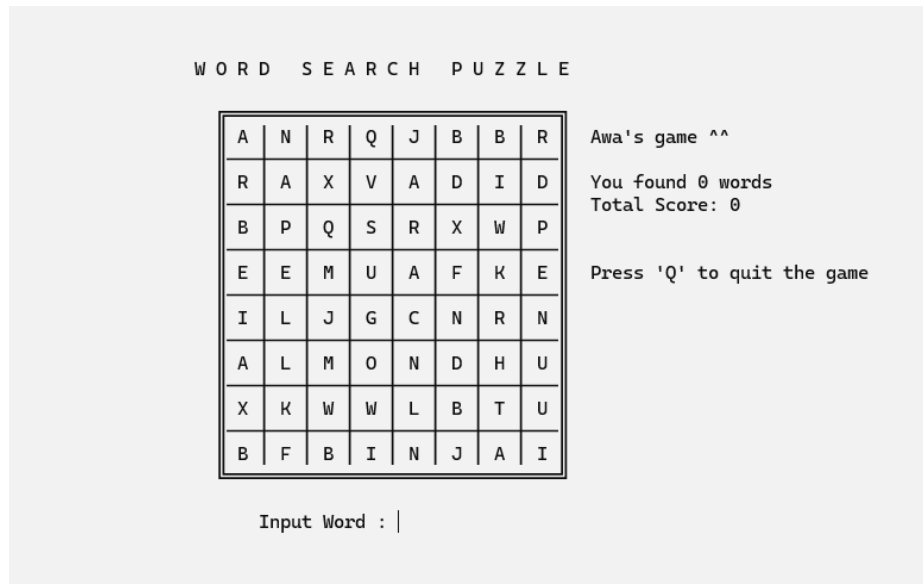
d. Pemilihan Tema



Gambar 12 Tampilan program pemilihan tema

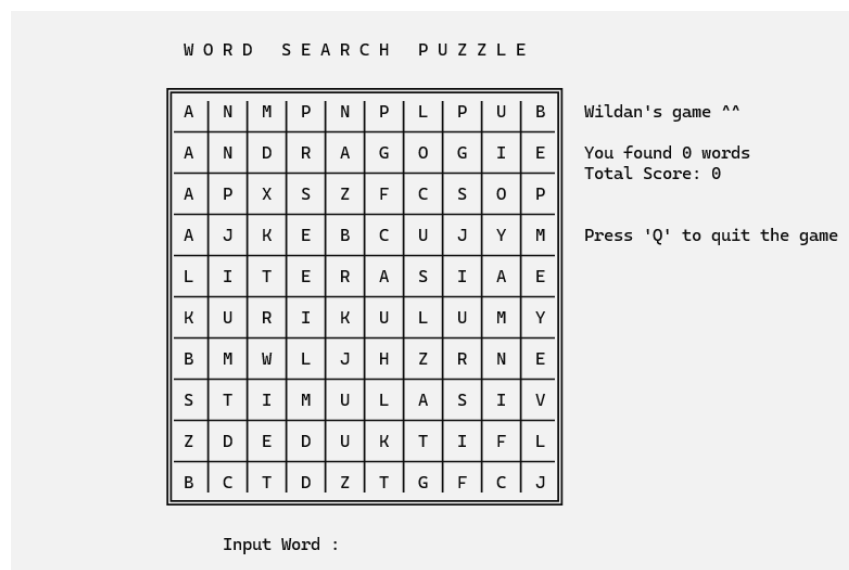
e. Papan Puzzle setiap Level

i. Easy



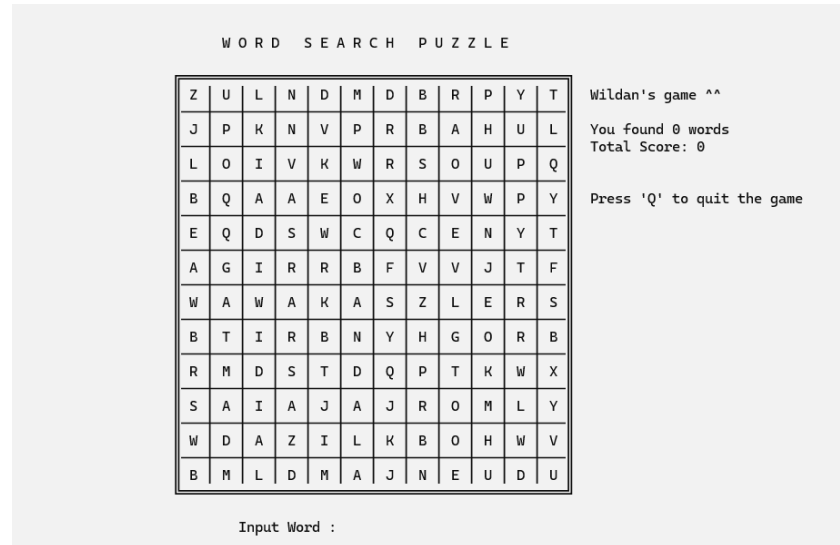
Gambar 13 Tampilan papan mode easy

ii. Medium



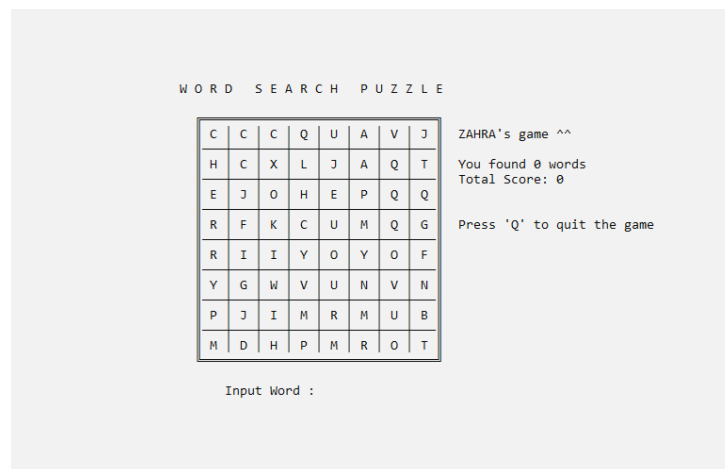
Gambar 14 Tampilan papan mode medium

iii. Hard

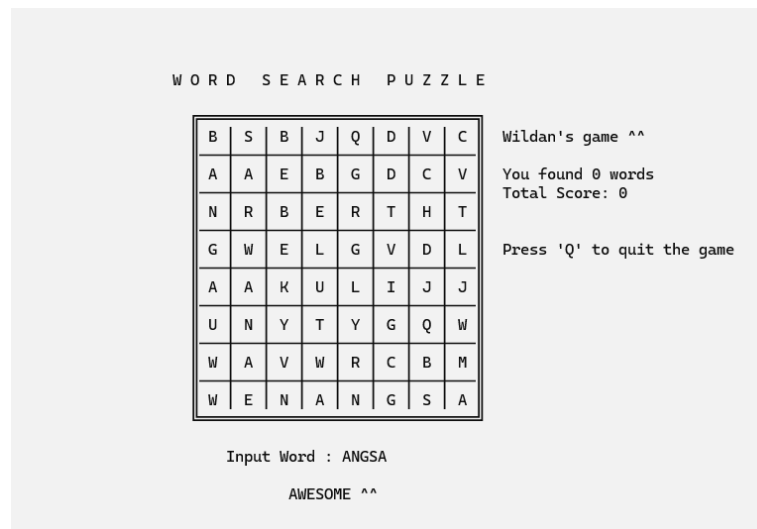


Gambar 15 Tampilan papan mode hard

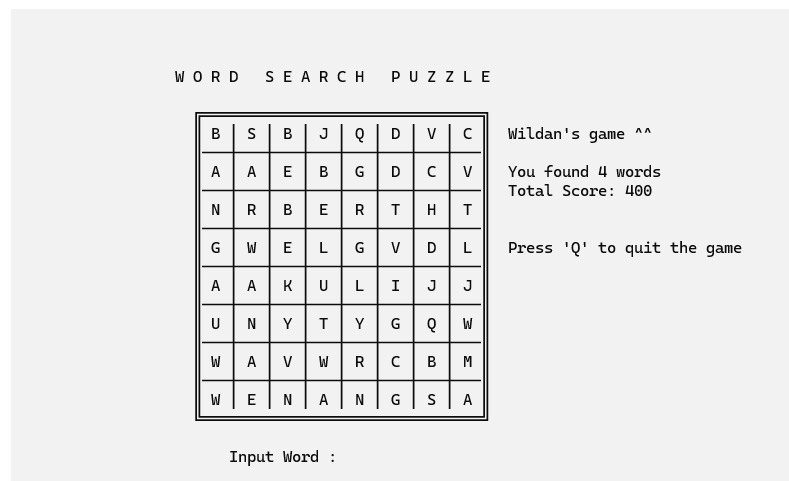
f. Scoring



Gambar 16 Tampilan skor saat permainan dimulai

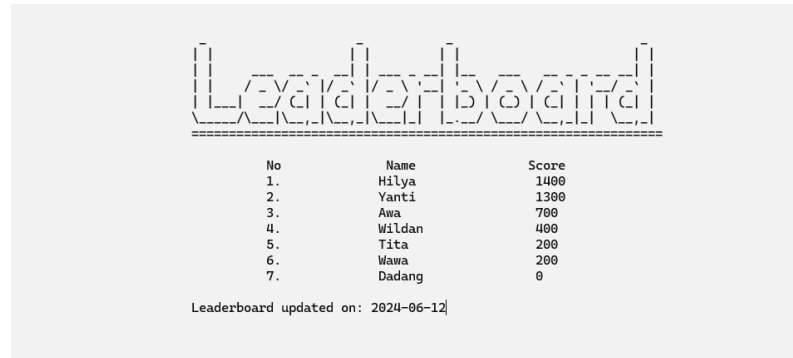


Gambar 17 Tampilan ketika kata yang dimasukkan benar



Gambar 18 Tampilan saat poin sudah bertambah

2.2.2 Leaderboard

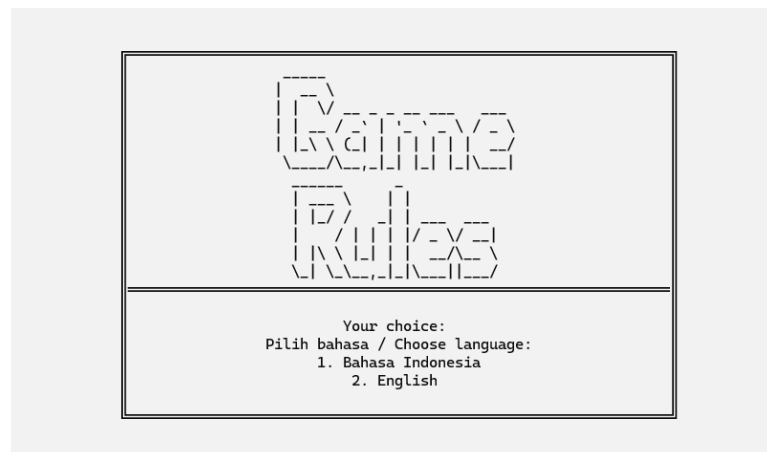


No	Name	Score
1.	Hilya	1400
2.	Yanti	1300
3.	Awa	700
4.	Wildan	400
5.	Tita	200
6.	Wawa	200
7.	Dadang	0

Leaderboard updated on: 2024-06-12

Gambar 19 Tampilan Leaderboard

2.2.3 Aturan Bermain



Game Rules

Your choice:
Pilih bahasa / Choose language:
1. Bahasa Indonesia
2. English

Gambar 20 Tampilan pilihan bahasa untuk aturan permainan



Gambar 21 Tampilan aturan dalam bahasa Indonesia



Gambar 22 Tampilan aturan dalam bahasa Inggris

BAB III

PENGUJIAN

3.1 Hasil Pengujian Program

Berisi laporan kegiatan pengujian program (berisi skenario pengujian) yang terdiri dari fitur program, hasil yang diharapkan (expected result), dan hasil output program (actual result).

No	1
Skenario	<ul style="list-style-type: none">• Memastikan antarmuka pengguna mudah dipahami.• Memastikan bahwa permainan berjalan lancar dan responsif terhadap input pengguna.• Memeriksa fitur-fitur tambahan seperti pilihan tema kata, mode permainan, dan scoring.
Fitur Program	Bermain Game
Deskripsi Fitur	Fitur ini memungkinkan pemain untuk menikmati aktivitas pencarian kata-kata dalam permainan Word Search Puzzle. Fitur ini menyediakan antarmuka yang ramah pengguna dan berbagai fitur tambahan yang memperkaya pengalaman bermain.
Target	<ul style="list-style-type: none">• Antarmuka pengguna yang ramah dengan navigasi yang mudah dipahami.• Permainan berjalan lancar dan responsif terhadap input pengguna.• Fitur tambahan seperti pemilihan tema kata dan mode permainan tersedia dan berfungsi sesuai dengan

	deskripsi.
Ouput	Setelah pengujian, fitur bermain game berjalan sesuai dengan harapan. Antarmuka pengguna mudah dipahami, permainan berjalan lancar, dan semua fitur tambahan bekerja dengan baik.
Kesimpulan	<ul style="list-style-type: none"> • Antarmuka pengguna didesain dengan baik, memberikan pengalaman yang intuitif dan menarik bagi pemain. • Permainan berjalan lancar dan responsif, memberikan pengalaman bermain yang menyenangkan. • •Semua fitur tambahan seperti pemilihan tema kata, mode permainan, dan scoring berfungsi dengan baik dan menambah keunikan permainan.

No	2
Skenario	<ul style="list-style-type: none"> • Memastikan bahwa setiap mode permainan memberikan tingkat kesulitan yang sesuai dengan deskripsi. • Memeriksa ukuran kisi huruf dan kesulitan menemukan kata-kata dalam setiap mode permainan.
Fitur Program	Mode Permainan
Deskripsi Fitur	Fitur ini menyediakan tiga mode permainan berbeda: Mudah (Easy), Sedang (Medium), dan Sulit (Hard). Setiap mode permainan menawarkan tingkat kesulitan yang berbeda, sesuai dengan keahlian dan preferensi pemain. Mode Mudah cocok untuk pemula dengan kisi huruf kecil dan kata-kata

	yang mudah ditemukan, Mode Sedang menawarkan tantangan dengan kisi huruf yang lebih besar dan kata-kata yang lebih sulit ditemukan, sementara Mode Sulit menantang dengan kisi huruf yang besar dan kata-kata yang sangat sulit ditemukan.
Target	<ul style="list-style-type: none"> • Mode permainan sesuai dengan deskripsi masing-masing, dengan tingkat kesulitan yang meningkat secara proporsional sesuai dengan mode yang dipilih. • Pemain akan merasakan perbedaan signifikan antara mode mudah, sedang, dan sulit dalam hal ukuran kisi huruf dan kesulitan menemukan kata-kata.
Ouput	<ul style="list-style-type: none"> • Setelah pengujian, semua mode permainan berjalan sesuai dengan deskripsi. Mode mudah memiliki kisi huruf yang kecil dan kata-kata yang mudah ditemukan, mode sedang menawarkan tantangan yang lebih besar, dan mode sulit menantang pemain dengan kisi huruf yang besar dan kata-kata yang sulit ditemukan.
Kesimpulan	<ul style="list-style-type: none"> • Mode permainan dirancang untuk memberikan variasi tingkat kesulitan yang sesuai dengan preferensi pemain. • Mode mudah cocok untuk pemula dengan kisi huruf yang kecil dan kata yang mudah ditemukan, sedangkan mode sulit menantang pemain dengan kisi huruf yang besar dan kata yang lebih sulit ditemukan.

No	3
----	---

Skenario	<ul style="list-style-type: none"> • Memeriksa apakah pemilihan tema kata mempengaruhi kata-kata yang tersembunyi dalam kisi huruf. • Memastikan bahwa pilihan tema kata berfungsi dengan baik dan antarmuka pengguna mudah dipahami.
Fitur Program	Pemilihan Tema Kata
Deskripsi Fitur	Fitur ini memungkinkan pemain untuk memilih kategori kata-kata yang ingin mereka cari dalam permainan. Tema kata dapat mencakup berbagai topik seperti buah-buahan, hewan dan lain sebagainya. Selain itu, tema permainan tersedia dalam 2 bahasa, bahasa Inggris dan bahasa Indonesia. Pilihan dari tema kata ini memengaruhi kata-kata yang tersembunyi dalam kisi huruf, menambah variasi dan keunikan dalam pengalaman bermain Word Search Puzzle ini.
Target	<ul style="list-style-type: none"> • Setiap pemilihan tema kata menghasilkan permainan dengan kata-kata yang relevan dan sesuai dengan tema tersebut. • Tema kata memengaruhi kata-kata yang tersembunyi dalam kisi huruf, sehingga memberikan variasi dan keunikan dalam pengalaman bermain.
Ouput	<ul style="list-style-type: none"> • Setelah pengujian, pemilihan tema kata berfungsi dengan baik. Setiap tema menghasilkan permainan dengan kata-kata yang sesuai dan relevan, serta memengaruhi kata-kata yang tersembunyi dalam kisi huruf.
Kesimpulan	<ul style="list-style-type: none"> • Fitur pemilihan tema kata menambah variasi dalam permainan, memungkinkan pemain untuk memilih kategori

	<p>kata-kata yang sesuai dengan minat mereka.</p> <ul style="list-style-type: none"> • Setiap tema kata menghasilkan permainan dengan kata-kata yang relevan dan sesuai dengan tema yang dipilih, menambah keunikan dalam pengalaman bermain.
--	--

No	4
Skenario	<ul style="list-style-type: none"> • Memastikan bahwa setiap kata yang ditemukan memberikan skor 100 kepada pemain. • Memeriksa bahwa skor akhir pemain terkumpul dengan benar dan sesuai dengan jumlah kata yang ditemukan.
Fitur Program	Scoring
Deskripsi Fitur	Fitur score dalam permainan Word Search Puzzle memberikan 100 poin kepada pemain setiap mereka berhasil menemukan satu kata. Skor pemain akan terakumulasi seiring dengan penemuan kata-kata dalam satu permainan, memberikan penghargaan atas upaya yang mereka lakukan untuk menyelesaikan permainan kata ini.
Target	<ul style="list-style-type: none"> • Setiap kata yang ditemukan memberikan skor 100 kepada pemain. • Skor akhir pemain terkumpul dengan benar dan sesuai dengan jumlah kata yang ditemukan.
Ouput	<ul style="list-style-type: none"> • Setelah menjalankan program, leaderboard ditampilkan <p>Setelah pengujian, setiap kata yang ditemukan memberikan skor 100 kepada pemain. Skor akhir pemain terkumpul</p>

	dengan benar dan sesuai dengan jumlah kata yang ditemukan
Kesimpulan	<ul style="list-style-type: none"> • Sistem scoring memberikan skor yang memadai kepada pemain untuk setiap kata yang berhasil ditemukan. • Skor akhir pemain dihitung dengan benar, mendorong pemain untuk mencapai skor tertinggi dan meningkatkan kepuasan dalam bermain.

No	5
Skenario	<ul style="list-style-type: none"> • Menjalankan program untuk menampilkan aturan permainan. • Memilih bahasa yang ingin ditampilkan aturan permainannya (Bahasa Indonesia atau Bahasa Inggris).
Fitur Program	Menampilkan aturan permainan dalam dua bahasa (Indonesia dan Inggris).
Target	<ul style="list-style-type: none"> • Program berhasil menampilkan aturan permainan dalam dua bahasa (Bahasa Indonesia dan Bahasa Inggris). • Setelah memilih bahasa, aturan permainan sesuai dengan bahasa yang dipilih ditampilkan di layar.
Ouput	<ul style="list-style-type: none"> • Setelah menjalankan program, aturan permainan dalam dua bahasa ditampilkan dengan baik. • Program berhasil menampilkan aturan permainan dalam bahasa Indonesia dan bahasa Inggris sesuai dengan pilihan pengguna.
Kesimpulan	<ul style="list-style-type: none"> • Program berhasil melewati pengujian dengan hasil yang

	<p>sesuai dengan yang diharapkan.</p> <ul style="list-style-type: none"> • Fitur untuk menampilkan aturan permainan dalam dua bahasa (Indonesia dan Inggris) berjalan dengan baik.
--	---

No	6
Skenario	<ul style="list-style-type: none"> • Memastikan bahwa leaderboard ditampilkan dengan benar. • Memastikan bahwa pemain ditambahkan ke leaderboard dengan benar. • Memastikan bahwa leaderboard dapat disimpan dan dimuat dengan benar dari file.
Fitur Program	Menampilkan dan mengelola leaderboard permainan.
Target	<ul style="list-style-type: none"> • Setelah permainan selesai, leaderboard ditampilkan dengan benar. • Pemain ditambahkan ke leaderboard dengan benar, dengan skor yang sesuai. • Leaderboard dapat disimpan dan dimuat dengan benar dari file.
Ouput	<ul style="list-style-type: none"> • Setelah menjalankan program, leaderboard ditampilkan dengan benar di layar. • Pemain berhasil ditambahkan ke leaderboard dengan benar setelah memainkan permainan. • Leaderboard dapat disimpan dan dimuat dengan benar dari

	file
Kesimpulan	<ul style="list-style-type: none"> • Program berhasil melewati pengujian dengan hasil yang sesuai dengan yang diharapkan. • Fitur untuk menampilkan, menambahkan, menyimpan, dan memuat leaderboard berjalan dengan baik

No	6
Skenario	Buat puzzle dengan mengatur semua tautan node menggunakan fungsi <code>makePuzzle</code> .
Fitur Program	Membuat puzzle dengan mengatur semua tautan node.
Target	<ul style="list-style-type: none"> • Program berhasil membuat puzzle dengan mengatur semua tautan node secara horizontal, vertikal, dan diagonal. • Setiap node pada puzzle terhubung dengan tepat sesuai dengan aturan yang ditetapkan.
Ouput	Setelah memanggil fungsi <code>makePuzzle</code> , program membuat puzzle dengan mengatur semua tautan node secara horizontal, vertikal, dan diagonal.
Kesimpulan	<ul style="list-style-type: none"> • Program berhasil melewati pengujian dengan hasil yang sesuai dengan yang diharapkan. • Fungsi <code>makePuzzle</code> berhasil membuat puzzle dengan mengatur semua tautan node secara tepat.

3.2 Kekurangan Program

Berdasarkan struktur data yang digunakan (multi-linked list) terdapat kekurangan ketika di implementasikan. Kekurangan pertama adalah kompleksitas memori. Dengan setiap node memiliki beberapa pointer (right, bottom, cross), mengalokasikan dan membebaskan memori menjadi lebih rumit. Selanjutnya, implementasi dan debugging struktur multi linked list lebih sulit dibandingkan dengan struktur yang lebih sederhana seperti array karena pada multi-linked ini, papan puzzle harus dbibuat bertahap satu persatu ditambahkan. Dari segi efektivitas penyimpanan, multi linked list memerlukan tambahan memori untuk setiap pointer dalam setiap node, yang bisa menjadi signifikan jika ukuran puzzle besar, terutama pada perangkat dengan keterbatasan memori. Kemudian, akses dan traversal data melalui multi linked list cenderung lebih lambat dibandingkan dengan array yang memungkinkan akses langsung ke elemen berdasarkan indeks, yang dapat memperpanjang waktu pencarian kata dalam puzzle besar. Terakhir, kompleksitas algoritma pencarian juga meningkat karena algoritma perlu menangani traversal melalui berbagai pointer, membuat implementasi lebih kompleks dan rentan terhadap kesalahan.

BAB IV

KESIMPULAN

Program ini mencakup beberapa fitur utama seperti input nama pemain, pemilihan bahasa, mode permainan, pemilihan tema, sistem papan puzzle untuk berbagai tingkat kesulitan, scoring, dan leaderboard. Setiap fitur dirancang untuk memberikan pengalaman bermain yang menarik dan menantang sambil tetap mudah diakses oleh pemula maupun pemain berpengalaman. Pemilihan bahasa memastikan inklusivitas, memungkinkan pemain untuk bermain dalam bahasa yang mereka pilih, sementara variasi mode permainan dan tema kata menawarkan keunikan dan variasi dalam setiap sesi permainan.

Salah satu kekuatan utama dari program ini adalah sistem scoring dan leaderboard yang mendorong kompetisi sehat di antara pemain. Skor yang terakumulasi memberikan motivasi bagi pemain untuk terus bermain dan meningkatkan performa mereka, sementara leaderboard menampilkan peringkat pemain dengan skor tertinggi, menciptakan elemen kompetitif yang menarik. Ini tidak hanya menambah nilai replayability permainan tetapi juga mendorong pemain untuk mengasah keterampilan mereka dan mencapai skor yang lebih tinggi. Dengan sistem ini, program berhasil menciptakan lingkungan di mana pemain dapat melihat perkembangan mereka dan berusaha untuk mencapai posisi teratas.

DAFTAR PUSTAKA

Patel, S. H. K., & Mahariba, A. J. (2020). Word Search Puzzle using Multi-Linked Lists. *International Journal of Engineering and Advanced Technology*, 8(4s2), 12–18. <https://doi.org/10.35940/ijeat.d1005.0484s219>