LAPORAN PRAKTIKUM

OTH ASD WEEK 4

Dibuat oleh

Ananda Bintang Saputra (1203230040)

Fakultas Informatika

Prodi Informatika

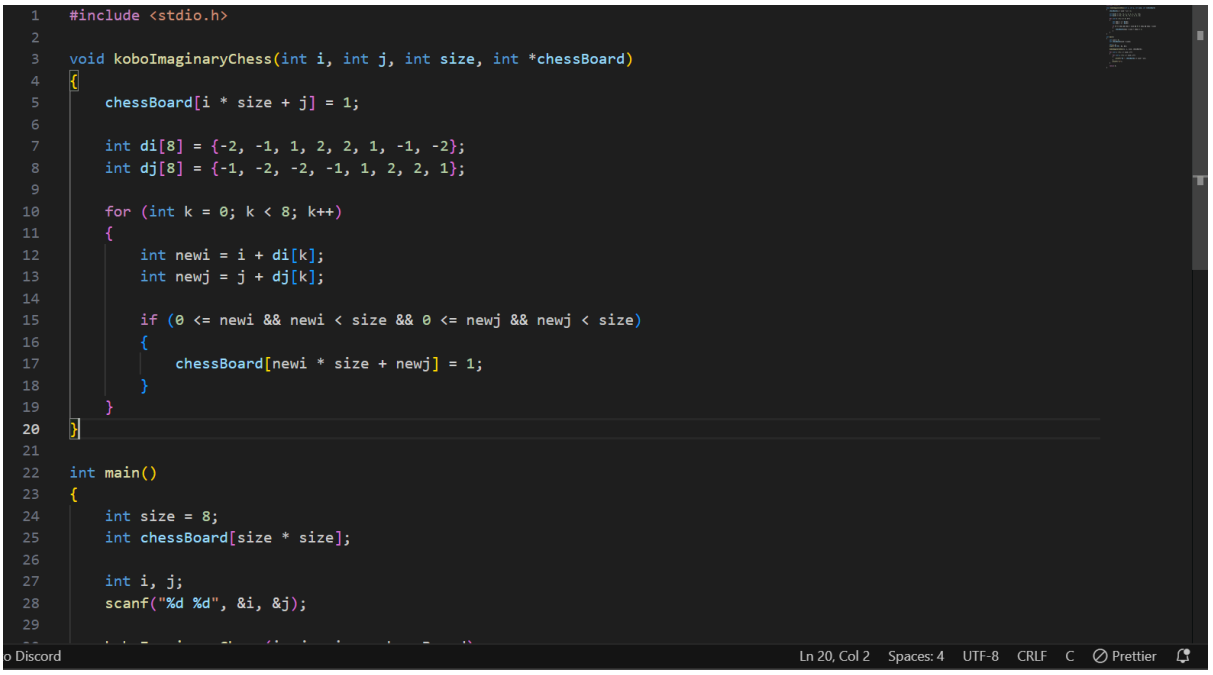
Telkom University Surabaya 2023/2024

TABEL PENILAIAN

Komponen Penilaian	Ya	Tidak
Soal 1 sesuai dengan output yang diinginkan		
Soal 2 sesuai dengan output yang diinginkan		
Bonus soal 1 dikerjakan		

SCREENSHOT CODE

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

int card_value(char card)
{
    if (card >= '2' && card <= '9')
    {
        return card - '0';
    }
    else if (card == 'J' || card == 'j')
    {
        return 11;
    }
    else if (card == 'Q' || card == 'q')
    {
        return 12;
    }
    else if (card == 'K' || card == 'k')
    {
        return 13;
    }
    return 0;
}

int bubble_sort(char cards[], int n)
{
    int i, j;
    int steps = 0;
```

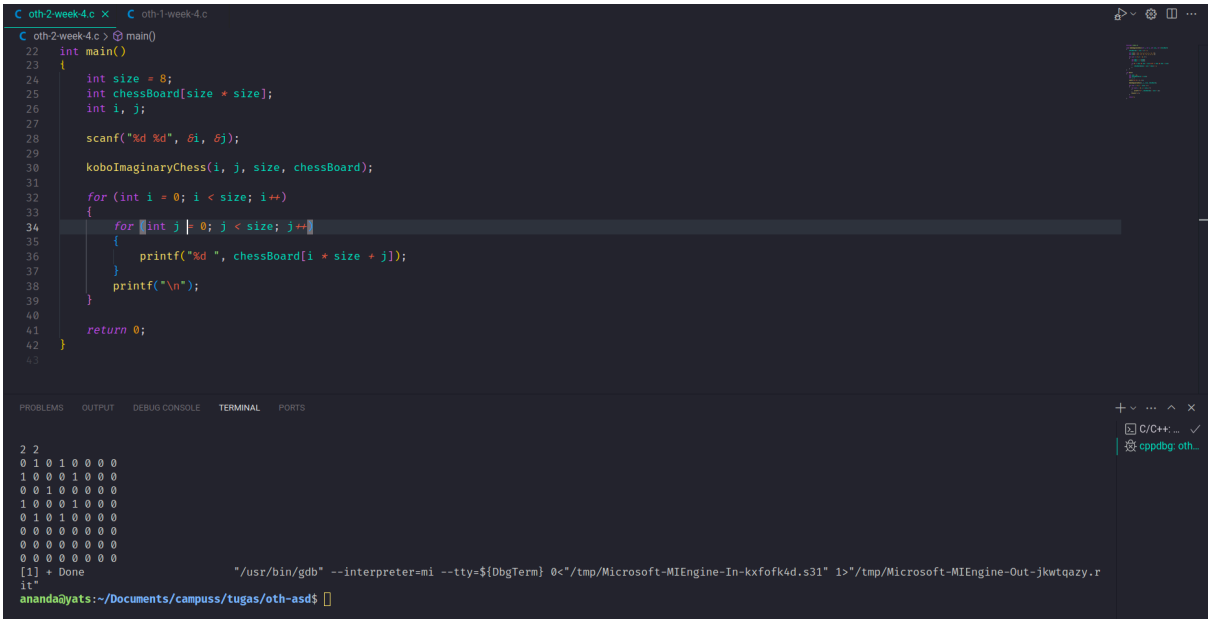



```c
#include <stdio.h>

void koboImaginaryChess(int i, int j, int size, int *chessBoard)
{
    chessBoard[i * size + j] = 1;

    int di[8] = {-2, -1, 1, 2, 2, 1, -1, -2};
    int dj[8] = {-1, -2, -2, -1, 1, 2, 2, 1};

    for (int k = 0; k < 8; k++)
    {
        int newi = i + di[k];
        int newj = j + dj[k];

        if (0 <= newi && newi < size && 0 <= newj && newj < size)
        {
            chessBoard[newi * size + newj] = 1;
        }
    }
}

int main()
{
    int size = 8;
    int chessBoard[size * size];

    int i, j;
    scanf("%d %d", &i, &j);
```

OUTPUT PROGRAM

```
int bubble_sort(char cards[], int n)
{
    int i, j;
    int steps = 0;
    int swapped;
    for (i = 0; i < n - 1; i++)
    {
        swapped = 0;
        for (j = 0; j < n - i - 1; j++)
        {
            if (card_value(cards[j]) > card_value(cards[j + 1]))
            {
                char temp = cards[j];
                cards[j] = cards[j + 1];
                cards[j + 1] = temp;
                steps++;
                swapped = 1;
                printf("Swap %d = ", steps);
                for (int k = 0; k < n; k++)
                {
                    printf("%c ", cards[k]);
                }
                printf("\n");
            }
```

```
Masukkan jumlah kartu: 8
Masukkan nilai kartu (dipisahkan dengan spasi): 5 9 2 6 J 1 K Q
Swap 1 = 5 2 9 6 J 1 K Q
Swap 2 = 5 2 6 9 J 1 K Q
Swap 3 = 5 2 6 9 1 J K Q
Swap 4 = 5 2 6 9 1 J Q K
Swap 5 = 2 5 6 9 1 J Q K
Swap 6 = 2 5 6 1 9 J Q K
Swap 7 = 2 5 1 6 9 J Q K
Swap 8 = 2 1 5 6 9 J Q K
Swap 9 = 1 2 5 6 9 J Q K

Jumlah minimal langkah pertukaran: 9
```



```
int main()
{
    int size = 8;
    int chessBoard[size * size];
    int i, j;

    scanf("%d %d", &i, &j);

    koboImaginaryChess(i, j, size, chessBoard);

    for (int i = 0; i < size; i++)
    {
        for (int j = 0; j < size; j++)
        {
            printf("%d ", chessBoard[i * size + j]);
        }
        printf("\n");
    }

    return 0;
}
```

```
2 2
0 1 0 1 0 0 0 0
1 0 0 0 1 0 0 0
0 0 1 0 0 0 0 0
1 0 0 0 1 0 0 0
0 1 0 1 0 0 0 0
0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0
[1] + Done                       "/usr/bin/gdb" --interpreter=mi --tty=${DbgTerm} 0<"/tmp/Microsoft-MIEngine-In-kxfofk4d.s31" 1>"/tmp/Microsoft-MIEngine-Out-jkwtqazy.r
it"
ananda@yats:~/Documents/campuss/tugas/oth-asd$
```

PENJELASAN PROGRAM

1.==========

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
```

Mengimport library yang akan digunakan.

```c
int card_value(char card)
{
    if (card >= '2' && card <= '9')
    {
        return card - '0';
    }
    else if (card == 'J' || card == 'j')
    {
        return 11;
    }
    else if (card == 'Q' || card == 'q')
    {
        return 12;
    }
    else if (card == 'K' || card == 'k')
    {
        return 13;
    }
    return 0;
}
```

Fungsi card_value menerima karakter card yang mewakili nilai sebuah kartu. Dalam fungsi ini, dilakukan pengecekan terhadap nilai karakter card untuk menentukan nilai

numeriknya. Karakter '2' hingga '9' diubah menjadi bilangan bulat dengan mengurangkan nilai ASCII karakter '0'. Karakter 'J', 'Q', 'K' atau 'j', 'q', 'k' diatur menjadi masing-masing 11, 12, dan 13. Jika karakter tidak sesuai dengan kriteria di atas, fungsi return 0.

```
int bubble_sort(char cards[], int n)
{
    int i, j;
    int steps = 0;
    int swapped;
    for (i = 0; i < n - 1; i++)
    {
        swapped = 0;
        for (j = 0; j < n - i - 1; j++)
        {
            if (card_value(cards[j]) > card_value(cards[j + 1]))
            {
                char temp = cards[j];
                cards[j] = cards[j + 1];
                cards[j + 1] = temp;
                steps++;
                swapped = 1;
                printf("Swap %d = ", steps);
                for (int k = 0; k < n; k++)
                {
                    printf("%c ", cards[k]);
                }
                printf("\n");
            }
        }

        if (swapped == 0)
            break;
    }
    return steps;
}
```

Fungsi bubble_sort menerima array karakter cards yang berisi kartu-kartu dan ukuran array n. Fungsi ini mengurutkan kartu-kartu menggunakan algoritma Bubble Sort. Dalam iterasi pertama, ia membandingkan setiap elemen dengan elemen berikutnya, dan jika ditemukan bahwa elemen pertama lebih besar dari elemen kedua, mereka

ditukar. Selama proses pengurutan, setiap kali pertukaran dilakukan, langkah-langkahnya dicetak menggunakan printf. Fungsi mengembalikan jumlah langkah yang dilakukan selama proses pengurutan.

```c
int main()
{
    int n;
    printf("Masukkan jumlah kartu: ");
    scanf("%d", &n);

    char *cards = (char *)malloc(n * sizeof(char));

    printf("Masukkan nilai kartu (dipisahkan dengan spasi): ");
    for (int i = 0; i < n; i++)
    {
        scanf(" %c", &cards[i]);
    }

    int steps = bubble_sort(cards, n);
    printf("\nJumlah minimal langkah pertukaran: %d\n", steps);

    free(cards);

    return 0;
}
```

Fungsi main adalah program utama. Pada awalnya, pengguna diminta untuk memasukkan jumlah kartu. Memori dialokasikan secara dinamis untuk array cards dengan ukuran n. Kemudian, pengguna diminta untuk memasukkan nilai-nilai kartu. Setelah semua kartu dimasukkan, fungsi bubble_sort dipanggil untuk mengurutkan kartu. Jumlah langkah minimum yang diperlukan untuk pengurutan dicetak. Terakhir, memori yang dialokasikan untuk cards dibebaskan.

2.==========

```c
#include <stdio.h>
```

Mengimport library yang dibutuhkan.

```c
void koboImaginaryChess(int i, int j, int size, int *chessBoard)
{
    // Menandai posisi awal kuda
    chessBoard[i * size + j] = 1;

    // Delapan kemungkinan arah gerakan kuda
    int di[8] = {-2, -1, 1, 2, 2, 1, -1, -2};
    int dj[8] = {-1, -2, -2, -1, 1, 2, 2, 1};

    // Menjelajahi semua kemungkinan arah
    for (int k = 0; k < 8; k++)
    {
        // Menghitung posisi baru
        int newi = i + di[k];
        int newj = j + dj[k];

        // Memastikan posisi baru berada dalam papan catur
        if (0 <= newi && newi < size && 0 <= newj && newj < size)
        {
            // Menandai posisi baru
            chessBoard[newi * size + newj] = 1;
        }
    }
}
```

Fungsi ini mengambil koordinat posisi awal kuda (i, j), ukuran papan catur, dan array chessBoard yang mewakili papan catur. Fungsi ini menandai posisi awal kuda dan delapan kemungkinan arah gerakan kuda di papan catur. Posisi awal kuda ditandai dengan nilai 1 di papan catur. Koordinat papan catur (i, j) diubah menjadi indeks linear pada array chessBoard. Array di dan dj digunakan untuk menyimpan delapan kemungkinan perubahan koordinat baris dan kolom untuk gerakan kuda di papan catur. Melakukan iterasi melalui delapan kemungkinan arah gerakan kuda di papan catur. Di dalam iterasi, program menghitung koordinat baru untuk kemungkinan gerakan kuda berdasarkan perubahan koordinat di array di dan dj. Program memastikan bahwa

posisi baru kuda berada di dalam batas papan catur. Jika posisi baru kuda valid, posisi tersebut ditandai dengan nilai 1 di papan catur.

```c
int main()
{
    int size = 8;                // Ukuran papan catur
    int chessBoard[size * size]; // Array untuk mensimulasikan papan catur

    // Membaca posisi awal kuda
    int i, j;
    scanf("%d %d", &i, &j);

    // Menjalankan simulasi
    koboImaginaryChess(i, j, size, chessBoard);

    // Mencetak hasil simulasi
    for (int i = 0; i < size; i++)
    {
        for (int j = 0; j < size; j++)
        {
            printf("%d ", chessBoard[i * size + j]);
        }
        printf("\n");
    }

    return 0;
}
```

Fungsi main() bertanggung jawab untuk membaca input posisi awal kuda dari pengguna, menjalankan simulasi menggunakan koboImaginaryChess(), dan mencetak hasil simulasi ke layar. Setelah simulasi selesai, program mencetak papan catur yang sudah dimodifikasi ke layar. Setiap sel yang ditandai dengan nilai 1 menunjukkan posisi yang dapat dijangkau oleh kuda setelah gerakan simulasi.