Gabriel Chrisoberryl Guevara Kaawoan
1203230041
Informatika 03-03

1. **Source Code :**

```c
#include <stdio.h>
#include <stdbool.h>
int value(char card) {
    if (card >= '2' && card <= '9')
        return card - '0';
    else if (card == 'J')
        return 14; // J memiliki nilai 14 agar berada di belakang K
    else if (card == 'Q')
        return 15; // Q memiliki nilai 15 agar berada di belakang J
    else if (card == 'K')
        return 16; // K memiliki nilai 16 agar berada di belakang Q
    else // Assuming 'A' represents 1
        return 1;
}

void bubble_sort(char cards[], int n) {
    int i, j, steps = 0;
    bool swapped;
    char temp;

    printf("Initial order: %s\n", cards);

    for (i = 0; i < n; i++) {
        swapped = false;
        for (j = 0; j < n - i - 1; j++) {
            if (value(cards[j]) > value(cards[j+1])) {
                // Swap cards
                temp = cards[j];
                cards[j] = cards[j+1];
                cards[j+1] = temp;
                swapped = true;
                steps++;
                printf("Pertukaran %d: %s\n", steps, cards);
            }
        }
        if (!swapped)
            break;
    }
```

```c
    for (i = 0; i < n; i++) {
        swapped = false;
        for (j = 0; j < n - i - 1; j++) {
            if (value(cards[j]) > value(cards[j+1])) {
                // Swap cards
                temp = cards[j];
                cards[j] = cards[j+1];
                cards[j+1] = temp;
                swapped = true;
                steps++;
                printf("Pertukaran %d: %s\n", steps, cards);
            }
        }
        if (!swapped)
            break;
    }
    printf("Jumlah minimal langkah pertukaran: %d\n", steps);
}

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

    char cards[n+1]; // +1 for '\0'
    printf("Masukkan nilai kartu: ");
    for (i = 0; i < n; i++) {
        scanf(" %c", &cards[i]); // Read cards one by one
    }
    cards[n] = '\0'; // Null-terminate the string

    bubble_sort(cards, n);

    return 0;
}
```

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

Penjelasan Source Code :

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

- **#include <stdio.h>** : Library standar untuk fungsi input-output (I/O) dalam bahasa C.
- **#include <stdbool.h>** : Library standar yang menyediakan tipe data boolean dan nilai-nilainya true dan false.

```c
int value(char card) {
    if (card >= '2' && card <= '9')
        return card - '0';
    else if (card == 'J')
        return 14; // J memiliki nilai 14 agar berada di belakang K
    else if (card == 'Q')
        return 15; // Q memiliki nilai 15 agar berada di belakang J
    else if (card == 'K')
        return 16; // K memiliki nilai 16 agar berada di belakang Q
    else // Assuming 'A' represents 1
        return 1;
}
```

- **value(char card)** : Fungsi ini mengembalikan nilai numerik dari sebuah kartu. Misalnya, untuk kartu '2', nilainya adalah 2, untuk 'J' nilainya adalah 14, untuk 'Q' nilainya adalah 15, dan seterusnya. Ini membantu dalam pengurutan kartu berdasarkan nilai numeriknya.

```c
void bubble_sort(char cards[], int n) {
    int i, j, steps = 0;
    bool swapped;
    char temp;

    printf("Initial order: %s\n", cards);

    for (i = 0; i < n; i++) {
        swapped = false;
        for (j = 0; j < n - i - 1; j++) {
            if (value(cards[j]) > value(cards[j+1])) {
                // Swap cards
                temp = cards[j];
                cards[j] = cards[j+1];
                cards[j+1] = temp;
                swapped = true;
                steps++;
                printf("Pertukaran %d: %s\n", steps, cards);
            }
        }
        if (!swapped)
            break;
    }
    printf("Jumlah minimal langkah pertukaran: %d\n", steps);
}
```

- **bubble_sort(char cards[], int n)** : Fungsi ini menerima array dari kartu dan jumlah kartu, dan mengurutkan kartu tersebut menggunakan algoritma Bubble Sort.
- **swapped** : Variabel boolean untuk menandai apakah ada pertukaran pada iterasi tertentu.
- **temp** : Variabel sementara untuk menukar kartu.

- Algoritma Bubble Sort digunakan untuk membandingkan dua kartu berdekatan dan menukar posisinya jika tidak terurut. Ini diulangi hingga seluruh array terurut.
- Setiap kali ada pertukaran, jumlah langkah (steps) ditambah satu, dan hasil pengurutan setiap langkah dicetak.

```
42  int main() {
43      int n, i;
44      printf("Masukkan jumlah kartu: ");
45      scanf("%d", &n);
46
47      char cards[n+1]; // +1 for '\0'
48      printf("Masukkan nilai kartu: ");
49      for (i = 0; i < n; i++) {
50          scanf(" %c", &cards[i]); // Read cards one by one
51      }
52      cards[n] = '\0'; // Null-terminate the string
53
54      bubble_sort(cards, n);
55
56      return 0;
57  }
```

- Fungsi main() merupakan titik awal program yang akan dieksekusi.
- Meminta pengguna untuk memasukkan jumlah kartu (n) dan nilai-nilai kartu.
- Nilai-nilai kartu dimasukkan ke dalam array cards.
- Memanggil fungsi bubble_sort() untuk mengurutkan kartu.
- Mengembalikan nilai 0, menandakan bahwa program telah berakhir dengan sukses.

Hasil Run Program :

```
PS C:\Users\varry> cd "c:\Users\varry\" ; if ($?) { gcc tempCodeRunnerFile.c -o tempCodeRunnerFile } ; if ($?) { .\tempCodeRunnerFile }
Masukkan jumlah kartu: 7
Masukkan nilai kartu: 247JK83
Initial order: 247JK83
Pertukaran 1: 247J8K3
Pertukaran 2: 247J83K
Pertukaran 3: 2478J3K
Pertukaran 4: 24783JK
Pertukaran 5: 24738JK
Pertukaran 6: 24378JK
Pertukaran 7: 23478JK
Jumlah minimal langkah pertukaran: 7
PS C:\Users\varry>
```

2. **Source Code :**

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

// Function to mark the positions reachable by the knight
void koboImaginaryChess(int i, int j, int size, int *chessBoard) {
    // Array to represent possible moves of a knight
    int moves[8][2] = {{-2, -1}, {-2, 1}, {-1, -2}, {-1, 2},
                       {1, -2}, {1, 2}, {2, -1}, {2, 1}};

    // Check and mark reachable positions
    for (int k = 0; k < 8; k++) {
        int ni = i + moves[k][0];
        int nj = j + moves[k][1];
        if (ni >= 0 && ni < size && nj >= 0 && nj < size) {
            chessBoard[ni * size + nj] = 1;
        }
    }
}

int main() {
    // Allocate memory for chess board
    int *chessBoard = (int *)malloc(8 * 8 * sizeof(int));
    if (chessBoard == NULL) {
        printf("Memory allocation failed\n");
        return 1;
    }

    // Initialize chess board with zeros
    for (int i = 0; i < 8 * 8; i++) {
        chessBoard[i] = 0;
    }

    // Input the position of the knight
    int i, j;
    printf("Enter the position of the knight (i j): ");
    scanf("%d %d", &i, &j);
```

```c
    // Initialize chess board with zeros
    for (int i = 0; i < 8 * 8; i++) {
        chessBoard[i] = 0;
    }

    // Input the position of the knight
    int i, j;
    printf("Enter the position of the knight (i j): ");
    scanf("%d %d", &i, &j);

    // Call function to mark reachable positions
    koboImaginaryChess(i, j, 8, chessBoard);

    // Mark the position of the knight
    chessBoard[i * 8 + j] = 0;

    // Print the chess board
    printf("Chess board:\n");
    for (int row = 0; row < 8; row++) {
        for (int col = 0; col < 8; col++) {
            printf("%d ", chessBoard[row * 8 + col]);
        }
        printf("\n");
    }

    // Free dynamically allocated memory
    free(chessBoard);

    return 0;
}
```

Penjelasan Source Code :

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

- **#include <stdio.h>** : Library standar untuk fungsi input-output (I/O) dalam bahasa C.
- **#include <stdlib.h>** : Library standar yang menyediakan fungsi-fungsi untuk alokasi dan dealokasi memori dinamis.

```c
// Function to mark the positions reachable by the knight
void koboImaginaryChess(int i, int j, int size, int *chessBoard) {
    // Array to represent possible moves of a knight
    int moves[8][2] = {{-2, -1}, {-2, 1}, {-1, -2}, {-1, 2},
                       {1, -2}, {1, 2}, {2, -1}, {2, 1}};

    // Check and mark reachable positions
    for (int k = 0; k < 8; k++) {
        int ni = i + moves[k][0];
        int nj = j + moves[k][1];
        if (ni >= 0 && ni < size && nj >= 0 && nj < size) {
            chessBoard[ni * size + nj] = 1;
        }
    }
}
```

- Fungsi ini bertanggung jawab untuk menandai posisi yang dapat dicapai oleh kuda (knight) pada papan catur imajiner.
- i dan j adalah koordinat baris dan kolom awal kuda.
- size adalah ukuran papan catur (8x8 pada kasus ini).
- *chessBoard adalah pointer ke papan catur yang direpresentasikan sebagai larik satu dimensi.
- moves adalah array yang berisi semua kemungkinan gerakan kuda di papan catur.
- Fungsi ini memeriksa setiap kemungkinan gerakan kuda dari posisi awal (i, j) dan menandai posisi yang dapat dicapai oleh kuda di papan catur.

```c
int main() {
    // Allocate memory for chess board
    int *chessBoard = (int *)malloc(8 * 8 * sizeof(int));
    if (chessBoard == NULL) {
        printf("Memory allocation failed\n");
        return 1;
    }

    // Initialize chess board with zeros
    for (int i = 0; i < 8 * 8; i++) {
        chessBoard[i] = 0;
    }

    // Input the position of the knight
    int i, j;
    printf("Enter the position of the knight (i j): ");
    scanf("%d %d", &i, &j);

    // Call function to mark reachable positions
    koboImaginaryChess(i, j, 8, chessBoard);

    // Mark the position of the knight
    chessBoard[i * 8 + j] = 0;

    // Print the chess board
    printf("Chess board:\n");
    for (int row = 0; row < 8; row++) {
        for (int col = 0; col < 8; col++) {
            printf("%d ", chessBoard[row * 8 + col]);
        }
        printf("\n");
    }

    // Free dynamically allocated memory
    free(chessBoard);

    return 0;
}
```

- Fungsi main adalah fungsi utama dari program.
- Pertama-tama, program mengalokasikan memori untuk papan catur menggunakan malloc.
- Kemudian, inisialisasi semua elemen papan catur dengan 0.
- Pengguna diminta untuk memasukkan posisi awal kuda.
- Fungsi koboImaginaryChess dipanggil untuk menandai posisi yang dapat dicapai oleh kuda.
- Setelah menandai posisi yang dapat dicapai, posisi awal kuda ditetapkan kembali ke 0.
- Papan catur dicetak di layar menggunakan dua loop bersarang.
- Akhirnya, memori yang dialokasikan secara dinamis untuk papan catur dibebaskan menggunakan free.

Hasil Run Program :

```
PS C:\Users\varry> cd "c:\Users\varry\" ; if ($?) { gcc tempCodeRunnerFile.c -o tempCodeRunnerFile } ; if ($?) { .\tempCodeRunnerFile }
Enter the position of the knight (i j): 2 4
Chess board:
0 0 0 1 0 1 0 0
0 0 1 0 0 0 1 0
0 0 0 0 0 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
PS C:\Users\varry> cd "c:\Users\varry\" ; if ($?) { gcc tempCodeRunnerFile.c -o tempCodeRunnerFile } ; if ($?) { .\tempCodeRunnerFile }
Enter the position of the knight (i j): 4 6
Chess board:
0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0
0 0 0 0 0 1 0 1
0 0 0 0 1 0 0 0
0 0 0 0 0 0 0 0
0 0 0 0 1 0 0 0
0 0 0 0 0 1 0 1
0 0 0 0 0 0 0 0
PS C:\Users\varry>
```