Nama : Muhammad Iqbal Maulana
NIM : 1203230066
Kelas : IF-03-03

TUGAS ASD PRAKTIKUM ARRAY, POINTER DAN FUNGSI

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

1. Refan Judi

1.1 Screenshot Code

refanJudi.c > ...

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

int mapInput(char input)
{
    switch (input)
    {
    case 'J':
        return 11;

    case 'Q':
        return 12;

    case 'K':
        return 13;

    default:
        return -1;
    }
}

void printOutput(int output)
{
    switch (output)
    {
    case 11:
        printf("J");
        break;

    case 12:
        printf("Q");
        break;

    case 13:
        printf("K");
        break;

    default:
        printf("%d", output);
        break;
    }
}

```

```c

void swap(int *xp, int *yp, int swap_count, int count, int *numbers)
{
    int temp = *xp;
    *xp = *yp;
    *yp = temp;

    printf("Pertukaran %d: ", swap_count);
    for (size_t i = 0; i < count; i++)
    {
        printOutput(*(numbers + i));
        printf(" ");
    }

    printf("\n");
}

void sort(int *numbers, int count)
{
    int swap_count = 0;
    for (size_t i = 0; i < count - 1; i++)
    {
        int minIndex = i;
        for (size_t j = i; j < count; j++)
        {
            if (numbers[j] < numbers[minIndex])
            {
                minIndex = j;
            };
        }

        if (minIndex != i)
        {
            swap_count++;
            swap(numbers + minIndex, numbers + i, swap_count, count, numbers);
        }
    }

    printf("%d\n", swap_count);
}

```

```c
int main()
{
    int count;
    scanf("%d", &count);

    int *numbers = (int *)malloc(count * sizeof(int));
    if (numbers == NULL)
    {
        printf("Memory allocation failed.\n");
        return 1;
    }

    for (int i = 0; i < count; i++)
    {
        char input;
        scanf(" %c", &input);

        int digit;
        if (isdigit(input))
        {
            digit = input - '0';
            if (digit < 0 || digit > 10)
            {
                printf("Invalid input!\n");
                return 1;
            }
        }
        else
        {
            digit = mapInput(input);
            if (digit == -1)
            {
                printf("Invalid input!\n");
                return 1;
            }
        }

        numbers[i] = digit;
    }

    sort(numbers, count);

    return 0;
}
```

1.2 Penjelasan Kode

```
int mapInput(char input)
{
    switch (input)
    {
    case 'J':
        return 11;

    case 'Q':
        return 12;

    case 'K':
        return 13;

    default:
        return -1;
    }
}
```

Baris kode diatas adalah fungsi untuk memetakan karakter kartu J, Q dan K ke dalam integer untuk memudahkan proses pengurutan.

```
void printOutput(int output)
{
    switch (output)
    {
    case 11:
        printf("J");
        break;

    case 12:
        printf("Q");
        break;

    case 13:
        printf("K");
        break;
```

```
    default:
        printf("%d", output);
        break;
    }
}
```

Baris kode diatas adalah fungsi untuk memetakan kembali integer ke bentuk karakter kartu J, Q dan K yang mana dipakai ketika mencetak hasil pengurutan ke terminal.

```
void swap(int *xp, int *yp, int swap_count, int count, int *numbers)
{
    int temp = *xp;
    *xp = *yp;
    *yp = temp;

    printf("Pertukaran %d: ", swap_count);
    for (size_t i = 0; i < count; i++)
    {
        printOutput(*(numbers + i));
        printf(" ");
    }

    printf("\n");
}

void sort(int *numbers, int count)
{
    int swap_count = 0;
    for (size_t i = 0; i < count - 1; i++)
    {
        int minIndex = i;
        for (size_t j = i; j < count; j++)
        {
            if (numbers[j] < numbers[minIndex])
            {
                minIndex = j;
            };
```

```
        }

        if (minIndex != i)
        {
            swap_count++;
            swap(numbers + minIndex, numbers + i, swap_count, count,
numbers);
        }
    }

    printf("%d\n", swap_count);
}
```

Baris kode diatas ada 2 fungsi terkait yang melakukan tugas pengurutan kartu. Fungsi sorting yang dipakai adalah selection sort dengan modifikasi fungsi swap, yaitu penambahan perintah untuk mencetak pertukaran elemen yang terjadi ke terminal.

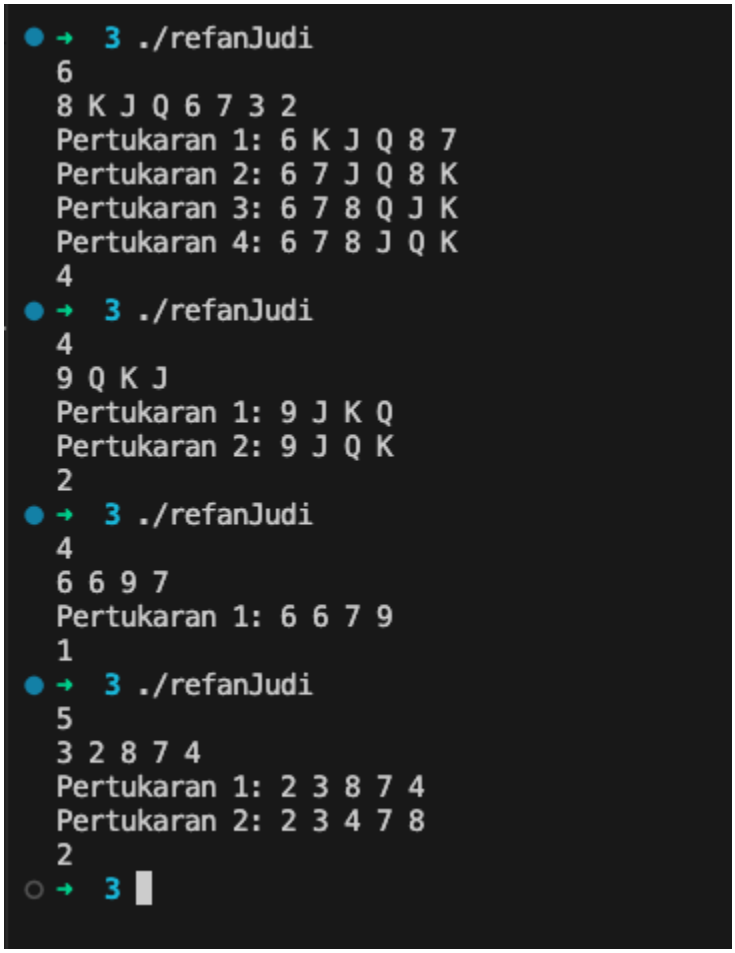
```
int main()
{
    int count;
    scanf("%d", &count);

    int *numbers = (int *)malloc(count * sizeof(int));
    if (numbers == NULL)
    {
        printf("Memory allocation failed.\n");
        return 1;
    }

    for (int i = 0; i < count; i++)
    {
        char input;
        scanf(" %c", &input);

        int digit;
        if (isdigit(input))
        {
            digit = input - '0';
```

```
            if (digit < 0 || digit > 10)
            {
                printf("Invalid input!\n");
                return 1;
            }
        }
        else
        {
            digit = mapInput(input);
            if (digit == -1)
            {
                printf("Invalid input!\n");
                return 1;
            }
        }

        numbers[i] = digit;
    }

    sort(numbers, count);

    return 0;
}
```

Baris kode diatas adalah fungsi main dari program refan judi. Pada bagian awal fungsi main, user diminta memasukkan jumlah kartu yang akan diurutkan, kemudian dialokasikan array dinamis sesuai jumlah kartu dari input user. Selanjutnya, program akan membaca input masing-masing posisi kartu sebagai tipe data karakter.

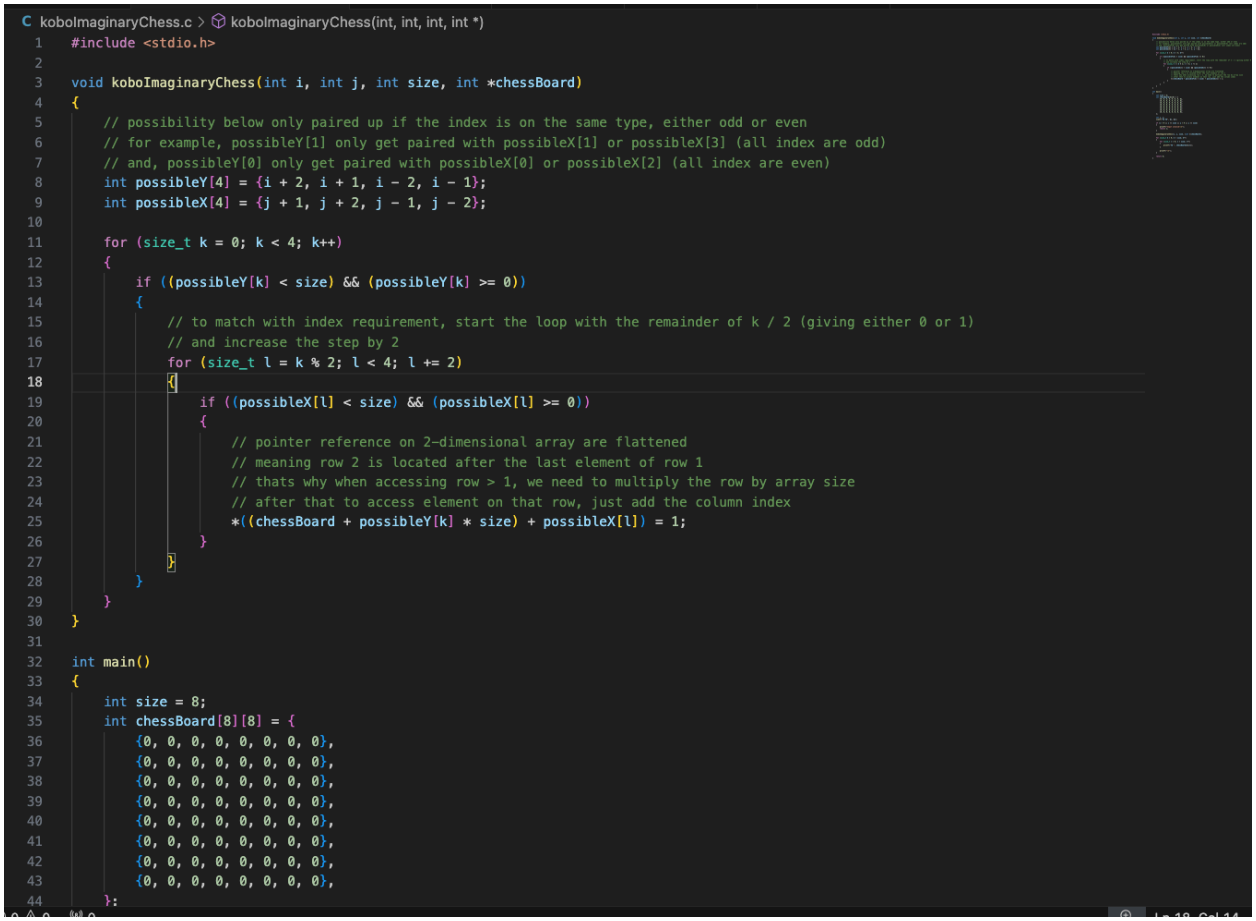
Setelah membaca jenis kartu, program akan mengkonversi tipe data karakter menjadi integer, untuk kartu berupa gambar, maka konversi menggunakan fungsi mapInput() sebelumnya. Setelah konversi selesai fungsi sort() dipanggil untuk melakukan fungsi utama dari program untuk mengurutkan kartu.

1.3 Output


3 ./refanJudi
6
8 K J Q 6 7 3 2
Pertukaran 1: 6 K J Q 8 7
Pertukaran 2: 6 7 J Q 8 K
Pertukaran 3: 6 7 8 Q J K
Pertukaran 4: 6 7 8 J Q K
4
3 ./refanJudi
4
9 Q K J
Pertukaran 1: 9 J K Q
Pertukaran 2: 9 J Q K
2
3 ./refanJudi
4
6 6 9 7
Pertukaran 1: 6 6 7 9
1
3 ./refanJudi
5
3 2 8 7 4
Pertukaran 1: 2 3 8 7 4
Pertukaran 2: 2 3 4 7 8
2
3

2. Kobo Imaginary Chess

2.1 Screenshot Code



```c

int main()
{
    int size = 8;
    int chessBoard[8][8] = {
        {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, 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},
        {0, 0, 0, 0, 0, 0, 0, 0},
    };

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

    if (i < 0 || i >= size || j < 0 || j >= size)
    {
        printf("Input invalid!\n");
        return 1;
    }

    koboImaginaryChess(i, j, size, (int *)chessBoard);

    for (size_t k = 0; k < size; k++)
    {
        for (size_t l = 0; l < size; l++)
        {
            printf("%d ", chessBoard[k][l]);
        }

        printf("\n");
    }

    return 0;
}

```

2.2 Penjelasan Kode

```
void koboImaginaryChess(int i, int j, int size, int *chessBoard)
{
    // possibility below only paired up if the index is on the same
type, either odd or even
    // for example, possibleY[1] only get paired with possibleX[1] or
possibleX[3] (all index are odd)
    // and, possibleY[0] only get paired with possibleX[0] or
possibleX[2] (all index are even)
    int possibleY[4] = {i + 2, i + 1, i - 2, i - 1};
    int possibleX[4] = {j + 1, j + 2, j - 1, j - 2};

    for (size_t k = 0; k < 4; k++)
    {
        if ((possibleY[k] < size) && (possibleY[k] >= 0))
        {
            // to match with index requirement, start the loop with
the remainder of k / 2 (giving either 0 or 1)
            // and increase the step by 2
            for (size_t l = k % 2; l < 4; l += 2)
            {
                if ((possibleX[l] < size) && (possibleX[l] >= 0))
                {
                    // pointer reference on 2-dimensional array are
flattened
                    // meaning row 2 is located after the last
element of row 1
                    // thats why when accessing row > 1, we need to
multiply the row by array size
                    // after that to access element on that row, just
add the column index
                    *((chessBoard + possibleY[k] * size) +
possibleX[l]) = 1;
                }
            }
        }
    }
}
```

Baris kode diatas merupakan fungsi utama (bukan main function) yang memetakan seluruh kemungkinan letak bidak kuda pada papan catur berdasarkan posisi awalnya. Teknik yang dipakai adalah dengan memetakan 4 kemungkinan arah gerakan bidak kuda, yaitu ke atas, ke bawah, ke kiri dan ke kanan. Untuk setiap arah gerakan, masing-masing memiliki 2 kombinasi gerakan vertikal dan horizontal. Sebagai contoh untuk gerakan bidak keatas, bidak dapat bergerak antara 2 unit vertikal ditambah 1 unit horizontal ke kiri atau ditambah 1 unit ke kanan.

Fungsi ini memetakan semua kemungkinan koordinat final pergerakan bidak dalam array possibleX dan possibleY, dimana pasangan koordinat (x, y) didapat dari ganjil-genapnya indeks array tersebut. Sebagai contoh koordinat x indeks ke 1 (ganjil) memiliki kemungkinan pasangan koordinat y indeks ke 1 atau 3 (ganjil). Sementara, koordinat x indeks ke 2 (genap) memiliki kemungkinan pasangan koordinat y indeks ke 2 atau 4 (genap).

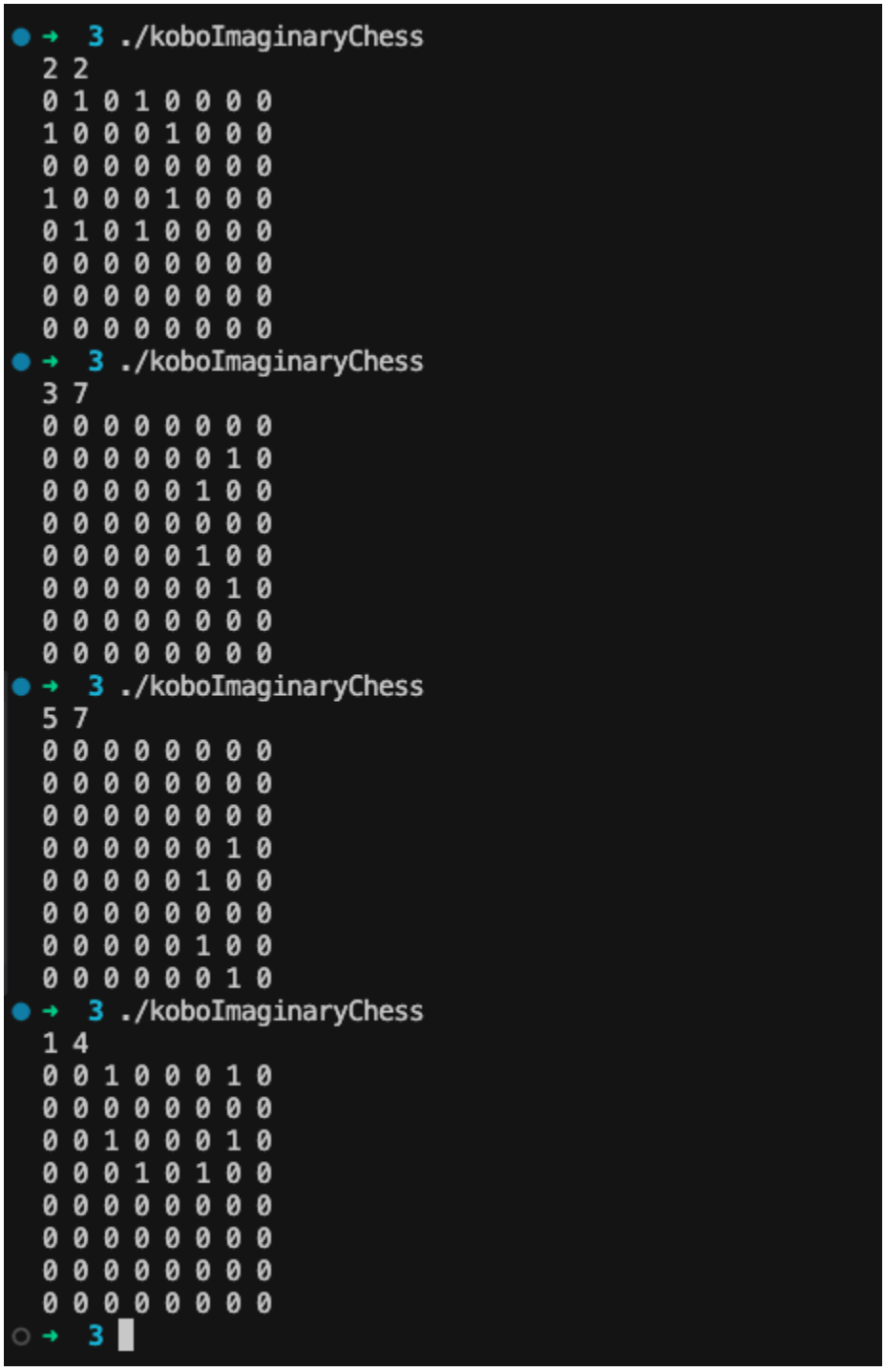
Semua kemungkinan koordinat (x. y) tersebut kemudian dievaluasi terhadap batas papan catur, yaitu 0 - 7 (inclusive). Koordinat yang melebihi batas ini tidak akan digunakan. Koordinat yang valid akan membalik nilai 0 pada papan catur menjadi 1.

```c
int main()
{
    int size = 8;
    int chessBoard[8][8] = {
        {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, 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},
        {0, 0, 0, 0, 0, 0, 0, 0},
    };

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

    if (i < 0 || i >= size || j < 0 || j >= size)
    {
        printf("Input invalid!\n");
        return 1;
    }
```

```
    koboImaginaryChess(i, j, size, (int *)chessBoard);

    for (size_t k = 0; k < size; k++)
    {
        for (size_t l = 0; l < size; l++)
        {
            printf("%d ", chessBoard[k][l]);
        }

        printf("\n");
    }

    return 0;
}
```

Baris kode diatas merupakan fungsi main dari program ini. Pada awal fungsi didefinisikan array 2D yang menampung 8x8 papan catur dengan inisialisasi nilai seluruhnya adalah 0. Kemudian program meminta input user sebagai koordinat awal bidak kuda, fungsi koboImaginaryChess() sebelumnya lalu dipanggil dan hasil update array oleh fungsi tersebut kemudian dicetak pada terminal.

2.3 Output


3 ./koboImaginaryChess
2 2
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 0 0
3 ./koboImaginaryChess
3 7
0 0 0 0 0 0 0 0
0 0 0 0 0 0 1 0
0 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
0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0
3 ./koboImaginaryChess
5 7
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 0 0 1 0
0 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
3 ./koboImaginaryChess
1 4
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
0 0 0 0 0 0 0 0
3