LAPORAN TUGAS
SHUFFLE KARTU & KOBO CHESS
(SOAL No. 1 & 2)

Laporan ini disusun dalam rangka

Penugasan Algoritma dan Struktur Data

INFORMATIKA
IF 03-01

DANI ADINUGROHO
(1203230119)

FAKULTAS INFORMATIKA (FIF)
TELKOM UNIVERSITY SURABAYA

MARET 2024

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	

SHUFFLE KARTU

Refan sedang bermain judi kartu pada sebuah kasino, ia sedang memegang N kartu yang tidak terurut. Refan ingin mengurutkan kartu yang ia miliki dari terkecil hingga terbesar untuk mempermudah dalam permainan.

Program menerima 2 baris input, baris pertama berupa jumlah kartu dan baris kedua berupa angka atau nilai dari kartu yang dipisahkan dengan spasi. Nilai kartu yang dapat diinput adalah 1-10 dan J, Q, K. Dengan ketentuan 10<J<Q<K.

```c
int main(){
    int jumlahKartu;

    printf("Jumlah Kartu yang ada ditangan: ");
    scanf("%d", &jumlahKartu);

    char nilaiKartu[jumlahKartu][3];

    // Memasukkan input per array dari nilaiKartu
    printf("Cards on Deck: ");
    for (int i = 0; i < jumlahKartu; i++){
        scanf(" %s", nilaiKartu[i]);
    }
    printf("\n");

    // Memberikan tampilan Jumlah kartu dan Kartu apa saja yang di input
    printf("Jumlah Kartu: %d\n", jumlahKartu);
    for (int i = 0; i < jumlahKartu; i++) {
        printf("%s ", nilaiKartu[i]);
    }
    printf("\n");

    // Konversi array nilaiKartu[][] menjadi array nilaiKartu[].
    char *ptr[jumlahKartu];
    for (int i = 0; i < jumlahKartu; i++) {
        ptr[i] = nilaiKartu[i];
    }

    sortKartu(jumlahKartu, ptr);
}
```

(Tampilan main code soal No.1)

- Terdapat **INT** jumlahKartu dan **Char** nilaiKartu 2D yang dimana jumlahKartu sebagai acuan banyak kartu yang kita masukkan, lalu [*][3] itu maksudnya batasan “String” yang bisa di-inputkan.
- **Char** *ptr[jumlahKartu], itu sebagai konversi array 2D dari nilaiKartu, agar tidak kesulitan membaca array 1D.
- Dan ada **sortKartu**(jumlahKartu, ptr), jumlahKartu sebagai size dari value yang dimasukkan dan ptr sebagai array pointer **Char** yang digunakan untuk pengecekan pada dalam fungsi.

```
void sortKartu(int jumlahKartu, char *ptr[]){
    int count = 0, min;
    for (int i = 0; i < jumlahKartu - 1; i++){
        // Inisialisasi value min = i
        min = i;
        for (int j = i + 1; j < jumlahKartu; j++){
            // Mengecek kondisi bahwa ptr[j] < ptr[min] atau ptr[min] == "K"
            // Akan mengubah nilai min = j karena ptr[min] lebih besar atau sama dengan "K"
            if (cardValueByINT(ptr[j]) < cardValueByINT(ptr[min]) || ptr[min] == "K") {
                min = j;
            }
        }
```

(Fungsi sortKartu)

- Terdapat variabel count = 0, min, min = i, dan min = j. Saat melakukan perulangan pada iterasi ke-i, min akan mengubah value-nya ke value perulangan ke-i.
- Pada iterasi ke-j, terdapat **IF** statement dengan kondisi, Fungsi **cardValueByINT**(ptr[j]) < **cardValueByINT**(ptr[min]) **OR**(||) ptr[min] == “K”, agar membedakan Value berdasarkan nilai string dari kartu ke nilai integer pada fungsi **cardValueByINT**. Seperti pada gambar dibawah ini:

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

int cardValueByINT(char *value1){
    if (strcmp(value1, "J") == 0) return 11;
    if (strcmp(value1, "Q") == 0) return 12;
    if (strcmp(value1, "K") == 0) return 13;
    if (strcmp(value1, "10") == 0) return 10;
    return (*value1 - '0'); // Mengubah karakter menjadi bilangan bulat
}
```

(Fungsi cardValueByINT)

- Fungsi ini sebagai membandingkan input yang kita masukkan menggunakan strcmp(pastinya ada library **string.h**) berdasarkan **ASCII** characters termasuk angka, simbol, dsbg.
- Tetapi kita membandingkan string-nya dengan input yang kita masukkan, kecuali (***value1 – ‘0’**), yang mengurangi nilai dari angka 0 – 9 dengan **ASCII**.

```
        if (min != i){
            //Perubahan tempat
            tukarKartu(ptr[min], ptr[i]);
            count++;

            // Menampilkan hasil kartu tiap pertukaran
            printf("Pertukaran %d: ", count);
            for (int k = 0; k < jumlahKartu; k++){
                printf("%s ", ptr[k]);
            }
            printf("\n");
        }
    }
    printf("%d\n", count);
}
```

(Fungsi sortKartu)

- Pada **IF** statement-nya terdapat (min != i), **min** ini berubah semenjak di looping iterasi j. Jika pada kondisinya value min ≠ value iterasi i, maka terjadi dimana fungsi tukarKartu(ptr[min], ptr[i]) akan ditukar. Jika kalian sadar bahwa ini memakai **Selection Sort.**

```
void tukarKartu(char *kartu1, char *kartu2) {
    char temp[3];
    strcpy(temp, kartu1);
    strcpy(kartu1, kartu2);
    strcpy(kartu2, temp);
}
```

(Fungsi tukarKartu)

- Dengan menggunakan **strcpy**(), kita menyalin string dari kartu1 yang akan ditaruh ke temp. Begitu seterusnya. (Fungsi sortKartu) setelah itu count++ akan bertambah, mengindikasi-kan bahwa pertukaran terjadi sebanyak **count**.
- Dan juga akan menampilkan hasil dari pertukaran **count** serta array yang di sort pada pertukaran itu.

```
PS C:\Users\[redacted]\projects> cd "c:\Us
Jumlah Kartu yang ada ditangan: 5
Cards on Deck: 3 2 8 7 4

Jumlah Kartu: 5
3 2 8 7 4
Pertukaran 1: 2 3 8 7 4
Pertukaran 2: 2 3 4 7 8
2
PS C:\Users\[redacted]projects\Semester 2
```

(Hasil dari pertukaran kartu)

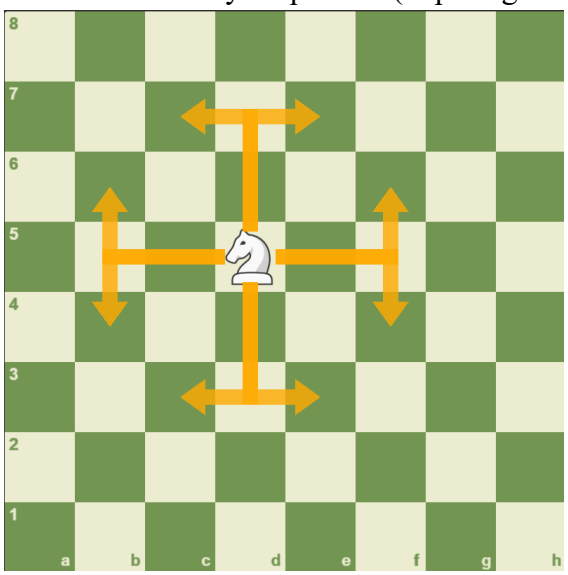
KOBO IMAGINARY CHESS

Kobo membayangkan suatu skenario dimana hanya ada satu buah bidak, yaitu kuda (knight), yang berada di dalam papan catur berukuran 8 × 8. Kobo penasaran dan ingin mengetahui posisi mana saja yang dapat dicapai oleh bidak kuda tersebut dalam sekali jalan apabila bidak tersebut berada pada posisi i, j dengan rincian $0 \leq i, j < 8$.

Dalam simulasinya, ia ingin punya array 2D dengan nilai awal 0 di setiap index sebagai bidak caturnya. Kemudian, Kobo ingin memberi nilai 1 pada setiap posisi yang mungkin dilalui oleh bidak kuda tersebut dalam sekali jalan apabila bidak tersebut berada pada posisi i, j.

Kesimpulan: Jadi kita akan membantu Kobo membuat Imaginary chess dengan ketentuan Bidak Kuda pada posisi **i** dan **j** adalah array 2D namaVariabel[i][j] dengan size-nya 8x8. Lalu Kobo ingin memeberi nilai 1 pada setiap posisi yang mungkin dilalui oleh bidak **Kuda** dalam sekali jalan.

Clue: Jalannya Bidak Kuda itu bentuknya seperti L. (Seperti gambar dibawah ini)



(Ilustrasi: Jalan Bidak Kuda berbentuk L)

```c
int main(){
    int Catur[8][8] = { {0} }, kudaX, kudaY;
    int sizeBidak = sizeof(Catur) / sizeof(Catur[0]);

    printf("Masukkan letak bidak Kuda: ");
    scanf("%d %d", &kudaX, &kudaY);
    printf("Bidak X dan Y: %d & %d\n", kudaX, kudaY);

    koboImaginaryChess(Catur, kudaX, kudaY, sizeBidak);
}
```

(Tampilan *main code* soal No.2)

- Terdapa variabel int dengan Array 2D size 8x8 dan value untuk 2-Dimensi array nya adalah **0**.
- **INT** kudaX sebagai letak [8][kudaX] secara Horizon dan kudaY sebagai letak [kudaY][8] secara vertikal.
- Dan ada sizeBidak dengan membagi sizeof(Catur) bernilai 256, berasal dari 8*8*(**INT** type value) dan sizeof(Catur[0]) bernilai 32 dari size array indeks ke-0 [*][0] dan **INT** type value = 4 adalah 8*(**INT** type value).
- Dan yang terakhir ada fungsi koboImaginaryChess() untuk memanggil fungsi sekaligus Output-nya.

```c
#include <stdio.h>

void koboImaginaryChess(int (*bidakCatur)[8], int kudaX, int kudaY, int size){
    for (int i = 0; i < size; i++){
        for (int j = 0; j < size; j++){
            // Mengatur bidak kuda pada posisi yang di input
            if (i == kudaX && j == kudaY){
                bidakCatur[i][j] = 0;
            }

            // Menandai sel yang dapat dijangkau oleh kuda
            // Misal angka kudaX dan kudaY adalah 2 & 2, dan saat index i dan j adalah 2
            // Maka ia akan meng-inisialisasi algoritma perhitungan kemungkinan perpindahan
            else if ((i == kudaX - 2 || i == kudaX + 2)) {
                if (j == kudaY - 1 || j == kudaY + 1)
                    bidakCatur[i][j] = 1;
            }
            else if ((i == kudaX - 1 || i == kudaX + 1)) {
                if (j == kudaY - 2 || j == kudaY + 2)
                    bidakCatur[i][j] = 1;
            }
            // Jika tidak dijangkau, maka tetap 0
            else bidakCatur[i][j] = 0;

            // Mencetak nilai 0 atau 1
            printf("%d ", bidakCatur[i][j]);
        }
        printf("\n");
    }
}
```

(Fungsi koboImaginaryChess)

- Pada parameter **koboImaginaryChess** terdapat:
  Pointer(*) **INT** (*bidakCatur)[8] sebagai Variabel yang dijadikan sebagai letak titik dimana kuda itu berada, dan letak yang bisa dia langkahi.

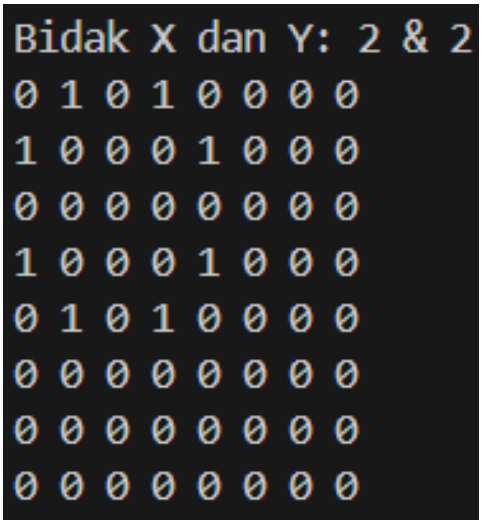
- Pada **IF:**
  (i == kudaX && j == kudaY) sebagai penentu letak dimana kuda itu berada dengan mengubah **bidakCatur[i][j] = 0 atau <u>bisa dengan 1</u>** agar dapat dibaca pengguna.

- Pada **ELSE IF:**
  (i == kudaX – 2 || i == kudaX + 2) dengan **NESTED IF** (j == kudaY – 1 || j == kudaY + 1), dan bidakCatur[i][j] = 1. Itu sebagai penanda dimana kuda bisa menentukan petak kotak mana yang di tuju.

  Contoh: posisi kuda berada di bidakCatur[2][2] dengan kudaX dan kudaY adalah (2,2), Nah mulai dari i = 0 dan j = 1, berarti dapat ditulis (0 == 2 – 2 || 0 == 2 + 2) yang jelas kiri bernilai **TRUE**, lalu (1 == 2 – 1 || 1 == 2 + 1) berarti ini yang kiri lagi.

  Bisa kita simpulkan maksud dari fungsi **ELSE IF** di gambar, merupakan pembacaan per **<u>baris</u>** kayak kita menganggap i adalah Y dan j adalah X yang berarti 0 kurang lebih sama dengan i dan j < 8.

- **ELSE:**
  Terakhir ada pada **ELSE** bidakCatur[i][j] = 0, maksudnya jika bidak kuda tidak dijangkau ke posisi lain. Dan print out hasil dari bidakCatur[i][j].



(Hasil dari fungsi koboImaginaryChess)