NAMA : MOCH. JOHAN BINTANG PRATAMA

NIM : 1203230063

LAPORAN PRAKTIKUM 4

KOMPONEN PENILAIAN	YA	TIDAK
Soal 1 sesuai dengan output yang diinginkan	YA	
Soal 2 sesuai dengan output yang diinginkan	YA	
Bonus soal 1 dikerjakan	YA	

Soal 1

SS Source Code

```c
#include <stdio.h>

void selectionSort(int arr[], int n, int *numSwaps) {
    int i, j, minIndex, temp;
    for (i = 0; i < n - 1; i++) {
        minIndex = i;
        for (j = i + 1; j < n; j++) {
            if (arr[j] < arr[minIndex]) {
                minIndex = j;
            }
        }
        if (minIndex != i) {
            temp = arr[minIndex];
            arr[minIndex] = arr[i];
            arr[i] = temp;
            (*numSwaps)++;
            printf("Pertukaran %d: ", *numSwaps);
            for (int k = 0; k < n; k++) {
                if (arr[k] >= 2 && arr[k] <= 10) {
                    printf("%d ", arr[k]);
                } else {
                    switch (arr[k]) {
                        case 11:
                            printf("J ");
                            break;
                        case 12:
                            printf("Q ");
                            break;
                        case 13:
                            printf("K ");
                            break;
                    }
                }
            }
```

```
            printf("\n");
        }
    }
}

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

    int kartu[n];
    for (int i = 0; i < n; i++) {
        char c1, c2;
        scanf(" %c", &c1);
        if (c1 == '1') {
            scanf("%c", &c2);
            kartu[i] = 10;
        } else if (c1 >= '2' && c1 <= '9') {
            kartu[i] = c1 - '0';
        } else {
            switch (c1) {
                case 'J':
                    kartu[i] = 11;
                    break;
                case 'Q':
                    kartu[i] = 12;
                    break;
                case 'K':
                    kartu[i] = 13;
                    break;
            }
        }
    }

    printf("\n");

    int numSwaps = 0;
    selectionSort(kartu, n, &numSwaps);

    printf("%d\n", numSwaps);
    for (int i = 0; i < n; i++) {
        if (kartu[i] >= 2 && kartu[i] <= 10) {
        }
    }
    printf("\n");

    return 0;
}
```

Penjelasan

```
#include <stdio.h>
```
Ini berfungsi untuk menampilkan printf, supaya bisa dibaca

```
void selectionSort(int arr[], int n, int *numSwaps) {
```
ini merupakan prosedur yang berfungsi untuk membuat selection sort, didalamnya terdapat beberapa variabel arr,n, dan pointer nuwSwaps yang semua bertipe data integer

```
    int i, j, minIndex, temp;
    for (i = 0; i < n - 1; i++) {
        minIndex = i;
        for (j = i + 1; j < n; j++) {
            if (arr[j] < arr[minIndex]) {
                minIndex = j;
            }
        }
```
Deklarasi variabel i, j, minIndex, dan temp, yang semua bertipe data integer.
for (i = 0; i < n - 1; i++) Loop luar yang bertugas untuk mengiterasi melalui setiap elemen dalam array arr. Dimulai dari indeks 0 dan berakhir pada indeks n - 2. Hal ini dilakukan karena pada iterasi terakhir, elemen terakhir sudah pasti terurut karena telah dibandingkan dengan semua elemen lain sebelumnya.
minIndex = i; Inisialisasi minIndex dengan indeks saat ini (i). Pada awal iterasi, kita anggap elemen pada indeks i adalah elemen terkecil dalam array yang belum diurutkan.
for (j = i + 1; j < n; j++) {: Loop dalam yang bertugas untuk mencari elemen terkecil dalam bagian array yang belum diurutkan. Dimulai dari indeks i + 1 karena elemen pada indeks i sudah dianggap sebagai elemen terkecil.
if (arr[j] < arr[minIndex]) Pemeriksaan kondisi apakah nilai elemen pada indeks j lebih kecil dari nilai elemen pada minIndex. Jika ya, maka minIndex diupdate dengan nilai j, yang berarti elemen pada indeks j adalah elemen terkecil yang ditemukan saat ini.
Setelah loop dalam selesai, minIndex akan menyimpan indeks dari elemen terkecil dalam bagian array yang belum diurutkan.
Iterasi loop luar akan berlanjut hingga semua elemen dalam array diurutkan. Pada setiap iterasi, elemen terkecil ditemukan dan diletakkan pada posisi yang sesuai dalam array yang diurutkan.

```
        if (minIndex != i) {
            temp = arr[minIndex];
            arr[minIndex] = arr[i];
            arr[i] = temp;
            (*numSwaps)++;
            printf("Pertukaran %d: ", *numSwaps);
            for (int k = 0; k < n; k++) {
                if (arr[k] >= 2 && arr[k] <= 10) {
                    printf("%d ", arr[k]);
                } else {
                    switch (arr[k]) {
                        case 11:
                            printf("J ");
                            break;
                        case 12:
```

```
                            printf("Q ");
                            break;
                        case 13:
                            printf("K ");
                            break;
                    }
                }
            }
            printf("\n");
        }
    }
}
```

if (minIndex != i) {: Ini adalah kondisi yang memeriksa apakah elemen terkecil yang ditemukan (minIndex) tidak sama dengan elemen saat ini (i). Jika mereka tidak sama, itu berarti ada elemen yang lebih kecil dari elemen saat ini yang ditemukan dalam bagian array yang belum diurutkan.
Di dalam blok if, pertukaran dilakukan antara elemen saat ini (arr[i]) dan elemen terkecil yang ditemukan (arr[minIndex]). Ini adalah langkah yang diperlukan untuk memindahkan elemen terkecil ke posisi yang benar dalam array yang diurutkan.
temp = arr[minIndex]; arr[minIndex] = arr[i]; arr[i] = temp;: Ini adalah langkah pertukaran nilai menggunakan variabel temp sebagai variabel sementara. Nilai dari arr[minIndex] disimpan di temp, kemudian nilai dari arr[i] disalin ke arr[minIndex], dan akhirnya nilai dari temp (yang sebelumnya adalah nilai dari arr[minIndex]) disalin ke arr[i].
(*numSwaps)++ Jumlah pertukaran (numSwaps) yang dilakukan ditambahkan satu setiap kali pertukaran dilakukan. Ini membantu dalam menghitung berapa kali proses pertukaran dilakukan.
printf("Pertukaran %d: ", *numSwaps);: Ini mencetak pesan yang menunjukkan nomor pertukaran yang sedang dilakukan.
Loop for digunakan untuk mencetak urutan kartu setelah pertukaran. Setiap kartu dicetak sesuai dengan format yang ditentukan: angka jika kartunya berada dalam rentang 2-10, atau huruf J, Q, atau K jika kartunya adalah 11, 12, atau 13.
printf("\n") Ini mencetak newline setelah semua kartu dalam urutan baru dicetak.

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

    int kartu[n];
    for (int i = 0; i < n; i++) {
        char c1, c2;
        scanf(" %c", &c1);
        if (c1 == '1') {
            scanf("%c", &c2);
            kartu[i] = 10;
        } else if (c1 >= '2' && c1 <= '9') {
            kartu[i] = c1 - '0';
        } else {
            switch (c1) {
                case 'J':
```

```
                kartu[i] = 11;
                break;
            case 'Q':
                kartu[i] = 12;
                break;
            case 'K':
                kartu[i] = 13;
                break;
        }
    }
}

printf("\n");
```

int n; scanf("%d", &n);: Deklarasi variabel n yang akan menyimpan jumlah kartu yang akan dimasukkan oleh pengguna. Input jumlah kartu diminta dari pengguna menggunakan fungsi scanf().

int kartu[n];: Deklarasi array kartu dengan ukuran yang sama dengan nilai yang dimasukkan oleh pengguna untuk menyimpan kartu-kartu tersebut.

for (int i = 0; i < n; i++) {: Loop for yang akan mengiterasi melalui setiap elemen array kartu untuk memasukkan kartu-kartu yang dimasukkan oleh pengguna.

char c1, c2;: Deklarasi variabel karakter c1 dan c2 yang akan digunakan untuk menyimpan input karakter kartu yang dimasukkan oleh pengguna.

scanf(" %c", &c1);: Input karakter kartu pertama dari pengguna. Diperhatikan bahwa ada spasi sebelum %c dalam fungsi scanf(), hal ini untuk menangani karakter newline atau spasi yang mungkin tersisa di buffer setelah input sebelumnya. Ini memastikan bahwa karakter yang dimasukkan oleh pengguna benar-benar karakter kartu.

Seleksi kondisional if-else digunakan untuk mengecek jenis kartu yang dimasukkan oleh pengguna:

Jika kartu dimasukkan adalah karakter '1', maka akan dilakukan pemrosesan khusus karena '1' diikuti oleh karakter lain ('0') yang menunjukkan kartu 10.

Jika kartu adalah angka antara '2' dan '9', nilai numerik dari karakter tersebut akan disimpan ke dalam array kartu. Pengurangan '0' dilakukan untuk mendapatkan nilai angka sebenarnya dari karakter ASCII.

Jika kartu adalah karakter 'J', 'Q', atau 'K', nilai numerik yang sesuai dengan kartu tersebut akan disimpan dalam array kartu.

Setelah semua kartu dimasukkan, pernyataan printf("\n"); digunakan untuk mencetak baris kosong untuk memisahkan input kartu dengan output hasil urutan kartu setelah pengurutan.

```
int numSwaps = 0;
```

```
    selectionSort(kartu, n, &numSwaps);

    printf("%d\n", numSwaps);
    for (int i = 0; i < n; i++) {
        if (kartu[i] >= 2 && kartu[i] <= 10) {
        }
    }
    printf("\n");

    return 0;
```

int numSwaps = 0;: Deklarasi variabel numSwaps yang akan digunakan untuk menyimpan jumlah pertukaran yang dilakukan selama proses pengurutan menggunakan algoritma Selection Sort.

selectionSort(kartu, n, &numSwaps);: Pemanggilan fungsi selectionSort() untuk mengurutkan array kartu yang telah dimasukkan oleh pengguna. Argumen yang diberikan adalah array kartu, ukuran array n, dan alamat variabel numSwaps yang akan diubah nilainya selama proses pengurutan untuk mencatat jumlah pertukaran yang terjadi.

printf("%d\n", numSwaps);: Setelah proses pengurutan selesai, nilai dari variabel numSwaps dicetak. Ini menunjukkan jumlah total pertukaran yang dilakukan selama proses pengurutan.

Loop for digunakan untuk mencetak urutan kartu setelah pengurutan. Namun, perhatikan bahwa dalam blok if, tidak ada tindakan yang dilakukan. Ini bisa menjadi sebuah kekurangan, karena seharusnya ada pernyataan cetak atau tindakan lainnya yang menunjukkan urutan kartu yang diurutkan. Hal ini mungkin diperlukan untuk menambahkan kode yang mencetak nilai kartu untuk menyelesaikan fungsi ini.

printf("\n") mencetak baris kosong untuk memisahkan hasil cetak dari urutan kartu setelah pengurutan.

return 0;: Mengakhiri fungsi main() dan mengembalikan nilai 0 kepada sistem operasi, menandakan bahwa program berakhir secara normal.

Output

```
PS D:\IT TELKOM\Tugas\Semester 2\Algoritma Pemrograman\Praktikum 4> cd "d:\IT TELKOM\Tugas\Semester 2\Algoritma Pemrograman\Praktikum 4\" ; if ($?) { gcc praktikum4_
1.c -o praktikum4_1 } ; if ($?) { .\praktikum4_1 }
4
6 6 9 7

Pertukaran 1: 6 6 7 9
1
```

```
PS D:\IT TELKOM\Tugas\Semester 2\Algoritma Pemrograman\Praktikum 4> cd "d:\IT TELKOM\Tugas\Semester 2\Algoritma Pemrograman\Praktikum 4\" ; if ($?) { gcc praktikum4_
1.c -o praktikum4_1 } ; if ($?) { .\praktikum4_1 }
5
3 2 8 7 4

Pertukaran 1: 2 3 8 7 4
Pertukaran 2: 2 3 4 7 8
2
```

```
PS D:\IT TELKOM\Tugas\Semester 2\Algoritma Pemrograman\Praktikum 4> cd "d:\IT TELKOM\Tugas\Semester 2\Algoritma Pemrograman\Praktikum 4\" ; if ($?) { gcc praktikum4_
1.c -o praktikum4_1 } ; if ($?) { .\praktikum4_1 }
6
10 J K Q 3 2

Pertukaran 1: 2 J K Q 3 10
Pertukaran 2: 2 3 K Q J 10
Pertukaran 3: 2 3 10 Q J K
Pertukaran 4: 2 3 10 J Q K
4
```

Output Bonus

```
PS D:\IT TELKOM\Tugas\Semester 2\Algoritma Pemrograman\Praktikum 4> cd "d:\IT TELKOM\Tugas\Semester 2\Algoritma Pemrograman\Praktikum 4\" ; if ($?) { gcc praktikum4_
1.c -o praktikum4_1 } ; if ($?) { .\praktikum4_1 }
8
9 4 2 J K 8 4 Q

Pertukaran 1: 2 4 9 J K 8 4 Q
Pertukaran 2: 2 4 4 J K 8 9 Q
Pertukaran 3: 2 4 4 8 K J 9 Q
Pertukaran 4: 2 4 4 8 9 J K Q
Pertukaran 5: 2 4 4 8 9 J Q K
5
```

Soal 2

Source Code

```c
#include <stdio.h>

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

  int dx[] = {2, 1, -1, -2, 2, 1, -1, -2};
  int dy[] = {1, 2, 2, 1, -1, -2, -2, -1};

  for (int k = 0; k < 8; k++) {
    int new_i = i + dx[k];
    int new_j = j + dy[k];

    if (0 <= new_i && new_i < size && 0 <= new_j && new_j < size) {
      chessboard[new_i * size + new_j] = 1;
    }
  }
}

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

  for (int i = 0; i < size; i++) {
    for (int j = 0; j < size; j++) {
      chessboard[i * size + j] = 0;
    }
  }

  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;
}
```

Penjelasan

```c
#include <stdio.h>
```
Ini berfungsi untuk menampilkan printf, supaya bisa dibaca

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

  int dx[] = {2, 1, -1, -2, 2, 1, -1, -2};
  int dy[] = {1, 2, 2, 1, -1, -2, -2, -1};

  for (int k = 0; k < 8; k++) {
    int new_i = i + dx[k];
    int new_j = j + dy[k];

    if (0 <= new_i && new_i < size && 0 <= new_j && new_j < size) {
      chessboard[new_i * size + new_j] = 1;
    }
  }
}
```
chessboard[i * size + j] = 1;: Pada baris ini, posisi awal kuda (ditentukan oleh koordinat (i, j)) ditandai di papan catur dengan menetapkan nilai 1 pada indeks yang sesuai dalam array chessboard. Ini menunjukkan bahwa kuda berada pada posisi awalnya.

int dx[] = {2, 1, -1, -2, 2, 1, -1, -2}; dan int dy[] = {1, 2, 2, 1, -1, -2, -2, -1};: Deklarasi array dx dan dy yang berisi perubahan posisi (delta) yang mungkin terjadi dalam koordinat x dan y saat kuda bergerak. Terdapat delapan kemungkinan gerakan yang dapat dilakukan oleh kuda.

for (int k = 0; k < 8; k++) {: Loop for yang akan melakukan iterasi melalui delapan kemungkinan gerakan yang mungkin dilakukan oleh kuda.

Di dalam loop, dilakukan perhitungan koordinat baru (new_i dan new_j) untuk setiap kemungkinan gerakan kuda dengan menambahkan perubahan posisi (dx[k] dan dy[k]) ke koordinat awal kuda (i, j).

if (0 <= new_i && new_i < size && 0 <= new_j && new_j < size) {: Dilakukan pemeriksaan apakah koordinat baru berada dalam batas-batas papan catur (antara 0 dan size - 1). Jika ya, maka posisi tersebut valid dan akan ditandai pada papan catur dengan menetapkan nilai 1 pada indeks yang sesuai dalam array chessboard. Ini menunjukkan bahwa kuda dapat mencapai posisi tersebut dalam satu langkah.

Setelah loop selesai, fungsi akan mengubah chessboard untuk menandai semua posisi yang dapat dicapai oleh kuda dalam satu langkah.

```c
int main() {
  int size = 8;
  int chessboard[size * size];

  for (int i = 0; i < size; i++) {
    for (int j = 0; j < size; j++) {
      chessboard[i * size + j] = 0;
    }
```
int size = 8;: Variabel size dideklarasikan dan diinisialisasi dengan nilai 8. Ini menunjukkan ukuran papan catur yang akan digunakan, yaitu papan catur dengan ukuran 8x8.

int chessboard[size * size];: Array chessboard dideklarasikan dengan ukuran size * size, yang sesuai dengan ukuran papan catur. Array ini akan digunakan untuk merepresentasikan papan catur, di mana setiap elemen array akan merepresentasikan kotak pada papan catur.

for (int i = 0; i < size; i++) {: Loop for luar yang digunakan untuk mengiterasi baris-baris papan catur.

for (int j = 0; j < size; j++) {: Loop for dalam yang digunakan untuk mengiterasi kolom-kolom papan catur.

chessboard[i * size + j] = 0;: Setiap kotak pada papan catur diinisialisasi dengan nilai 0. Ini menunjukkan bahwa pada awalnya, tidak ada kuda yang menduduki kotak tersebut.

Dengan menggunakan nested loop (for luar dan dalam), setiap elemen pada array chessboard diatur menjadi 0, menandakan bahwa tidak ada kuda yang menduduki posisi tersebut pada awalnya.

```
 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;
}
```

int i, j; scanf("%d %d", &i, &j);: Dua variabel i dan j dideklarasikan untuk menyimpan koordinat awal kuda. Nilai-nilai ini dimasukkan oleh pengguna melalui fungsi scanf(). Pengguna diminta untuk memasukkan dua bilangan bulat yang dipisahkan oleh spasi, yang masing-masing akan menjadi koordinat baris dan kolom awal kuda pada papan catur.

koboImaginaryChess(i, j, size, chessboard);: Fungsi koboImaginaryChess() dipanggil untuk mensimulasikan gerakan kuda pada papan catur imajiner berdasarkan koordinat awal yang dimasukkan oleh pengguna. Fungsi ini akan memodifikasi array chessboard sesuai dengan posisi-posisi yang dapat dicapai oleh kuda dalam satu langkah.

for (int i = 0; i < size; i++) {: Loop for luar yang digunakan untuk mengiterasi baris-baris papan catur.

for (int j = 0; j < size; j++) {: Loop for dalam yang digunakan untuk mengiterasi kolom-kolom papan catur.

printf("%d", chessboard[i * size + j]);: Pada setiap iterasi loop dalam, nilai dari setiap kotak pada papan catur (diwakili oleh nilai di dalam array chessboard) dicetak menggunakan fungsi printf(). Nilai 1 menunjukkan bahwa kotak tersebut telah dikunjungi oleh kuda dalam simulasi gerakan, sedangkan nilai 0 menunjukkan bahwa kotak tersebut belum dikunjungi oleh kuda.

printf("\n");: Setelah setiap baris papan catur selesai dicetak, dilakukan pencetakan baris baru untuk membentuk tata letak papan catur yang sesuai.

return 0;: Mengakhiri fungsi main() dan mengembalikan nilai 0 kepada sistem operasi, menandakan bahwa program berakhir secara normal.

Output

```
PS D:\IT TELKOM\Tugas\Semester 2\Algoritma Pemrograman\Praktikum 4> cd "d:\IT TELKOM\Tugas\Semester 2\Algoritma Pemrograman\Praktikum 4\" ; if ($?) { gcc praktikum4_
2.c -o praktikum4_2 } ; if ($?) { .\praktikum4_2 }
2 2
01010000
10001000
00100000
10001000
01010000
00000000
00000000
00000000
```

```
PS D:\IT TELKOM\Tugas\Semester 2\Algoritma Pemrograman\Praktikum 4> cd "d:\IT TELKOM\Tugas\Semester 2\Algoritma Pemrograman\Praktikum 4\" ; if ($?) { gcc praktikum4_
2.c -o praktikum4_2 } ; if ($?) { .\praktikum4_2 }
3 7
00000000
00000010
00000100
00000001
00000100
00000010
00000000
00000000
```