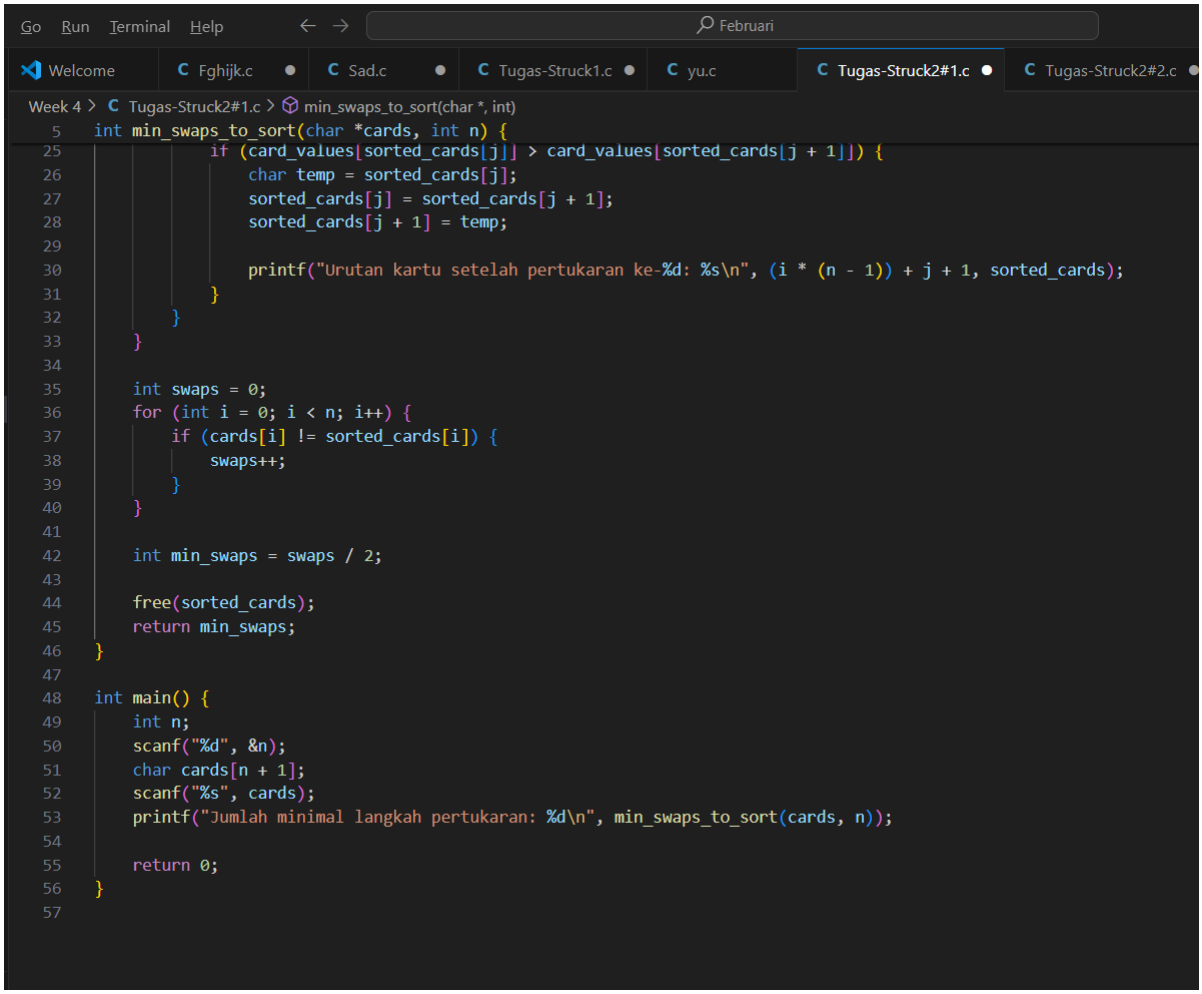
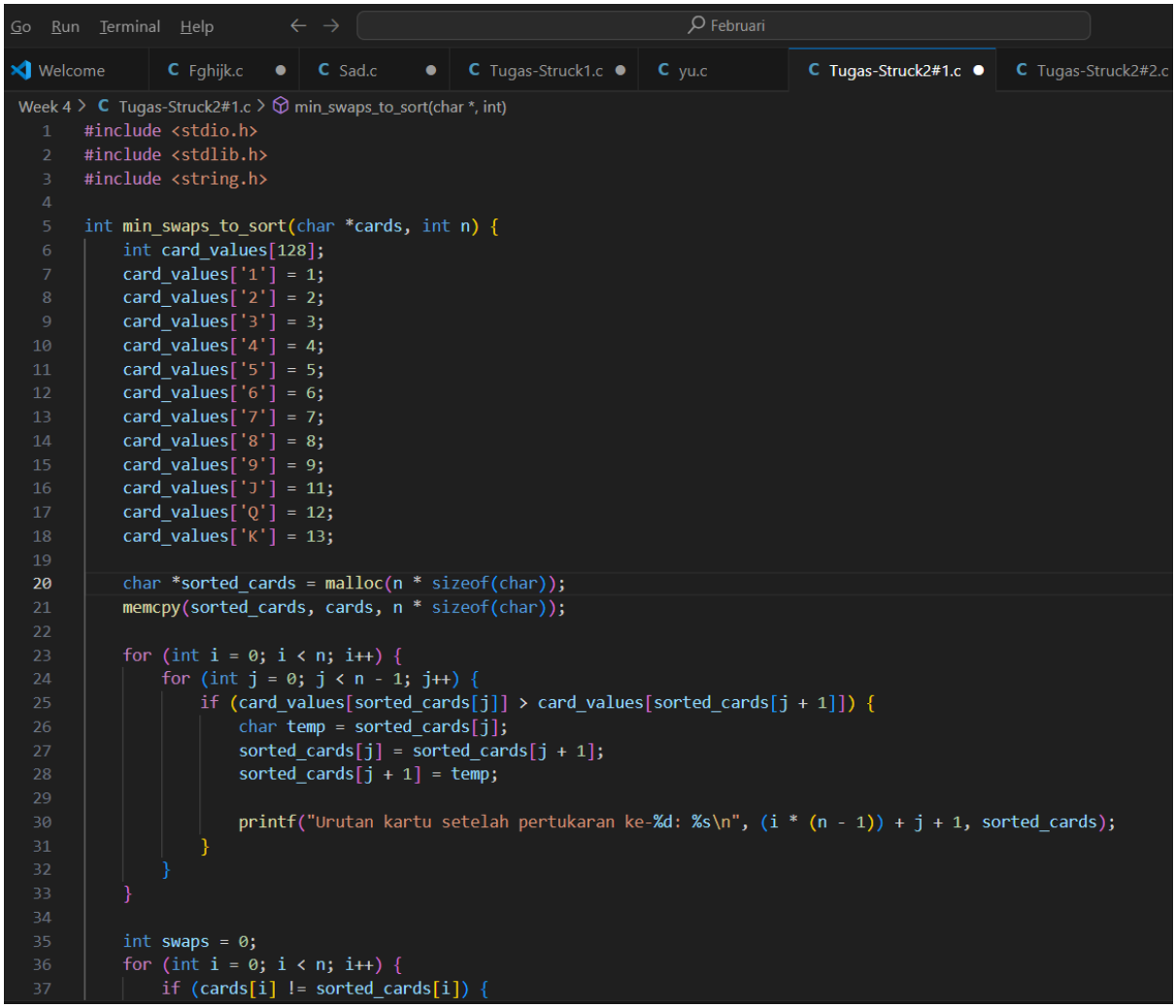
Nama:M.AGUNG BALYA

NIM:1203230095

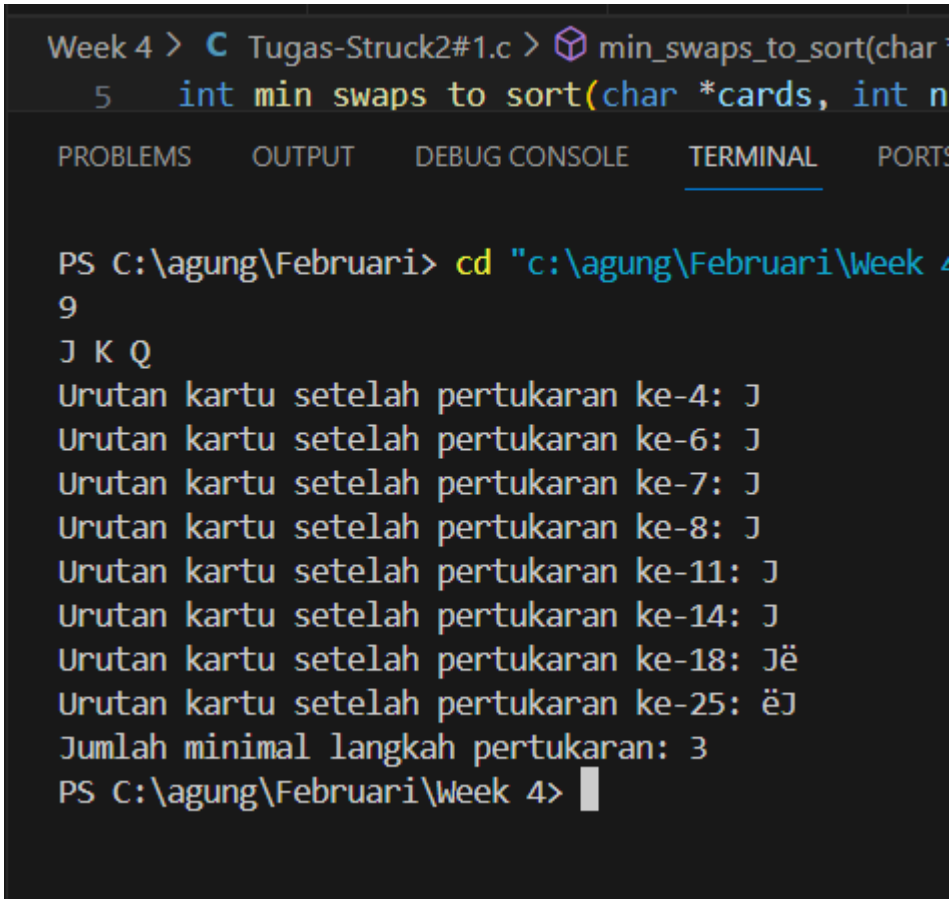
Kelas:IF 03-01

Code




Go Run Terminal Help
Februari
Welcome
Fghijk.c
Sad.c
Tugas-Struck1.c
yu.c
Tugas-Struck2#1.c
Tugas-Struck2#2.c
Week 4 > Tugas-Struck2#1.c > min_swaps_to_sort(char *, int)
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
int min_swaps_to_sort(char *cards, int n) {
int card_values[128];
card_values['1'] = 1;
card_values['2'] = 2;
card_values['3'] = 3;
card_values['4'] = 4;
card_values['5'] = 5;
card_values['6'] = 6;
card_values['7'] = 7;
card_values['8'] = 8;
card_values['9'] = 9;
card_values['J'] = 11;
card_values['Q'] = 12;
card_values['K'] = 13;
char *sorted_cards = malloc(n * sizeof(char));
memcpy(sorted_cards, cards, n * sizeof(char));
for (int i = 0; i < n; i++) {
for (int j = 0; j < n - 1; j++) {
if (card_values[sorted_cards[j]] > card_values[sorted_cards[j + 1]]) {
char temp = sorted_cards[j];
sorted_cards[j] = sorted_cards[j + 1];
sorted_cards[j + 1] = temp;
printf("Urutan kartu setelah pertukaran ke-%d: %s\n", (i * (n - 1)) + j + 1, sorted_cards);
}
}
}
int swaps = 0;
for (int i = 0; i < n; i++) {
if (cards[i] != sorted_cards[i]) {

Output



Penjelasan

```c
int min_swaps_to_sort(char *cards, int n) {
    int card_values[128];
    card_values['1'] = 1;
    card_values['2'] = 2;
    card_values['3'] = 3;
    card_values['4'] = 4;
    card_values['5'] = 5;
    card_values['6'] = 6;
    card_values['7'] = 7;
    card_values['8'] = 8;
    card_values['9'] = 9;
    card_values['J'] = 11;
    card_values['Q'] = 12;
    card_values['K'] = 13;
```

fungsi ‘min_swaps_to_sort’ mengonversi nilai-nilai kartu ke menjadi nilai numerik

int ‘card_values[128];’ Membuat array ‘card_values’ dengan ukuran 128, yang akan digunakan untuk menyimpan nilai numerik dari kartu-kartu.

‘card_values['1'] = 1;’ = Nilai kartu angka 1

‘card_values['2'] = 2;’ = Nilai kartu angka 2

‘card_values['3'] = 3;’ = Nilai kartu angka 3

‘card_values['4'] = 4;’ = Nilai kartu angka 4

‘card_values['5'] = 5;’ = Nilai kartu angka 5

‘card_values['6'] = 6;’ = Nilai kartu angka 6

‘card_values['7'] = 7;’ = Nilai kartu angka 7

‘card_values['8'] = 8;’ = Nilai kartu angka 8

‘card_values['9'] = 9;’ = Nilai kartu angka 9

‘card_values['J'] = 11;’ = Nilai kartu angka 11

‘card_values['Q'] = 12;’ = Nilai kartu angka 12

‘card_values['K'] = 13;’ = Nilai kartu angka 13

```
char *sorted_cards = malloc(n * sizeof(char));
    memcpy(sorted_cards, cards, n * sizeof(char));

    for (int i = 0; i < n; i++) {
        for (int j = 0; j < n - 1; j++) {
            if (card_values[sorted_cards[j]] > card_values[sorted_cards[j +
1]]) {
                char temp = sorted_cards[j];
                sorted_cards[j] = sorted_cards[j + 1];
                sorted_cards[j + 1] = temp;

                printf("Urutan kartu setelah pertukaran ke-%d: %s\n", (i * (n
- 1)) + j + 1, sorted_cards);
            }
        }
    }
```

Bubble kode diatas merupakan proses dari pengurutan kartu mengguanakan Bubble Sort

char *sorted_cards = malloc(n * sizeof(char));: Membuat sebuah array dinamis baru bernama ‘sorted_cards’ dengan ukuran sepanjang n (jumlah kartu) dan tipe data char.

memcpy(sorted_cards, cards, n * sizeof(char));: Menggunakan fungsi ‘memcpy’ untuk menyalin isi dari array cards (kartu-kartu yang belum diurutkan) ke dalam array sorted_cards.

for (int i = 0; i < n; i++) {: Memulai loop pertama yang akan melakukan iterasi sebanyak n kali, mewakili jumlah kartu yang belum diurutkan.

for (int j = 0; j < n - 1; j++) {: Memulai loop kedua yang akan melakukan iterasi sebanyak n - 1 kali. Ini adalah jumlah perbandingan antara dua kartu yang berdekatan dalam proses Bubble Sort.

if (card_values[sorted_cards[j]] > card_values[sorted_cards[j + 1]]) {: Memeriksa apakah nilai kartu pada posisi j lebih besar dari nilai kartu pada posisi j + 1. Jika ya, maka kartu tersebut perlu ditukar posisinya agar sesuai dengan urutan yang diinginkan.

char temp = sorted_cards[j]; sorted_cards[j] = sorted_cards[j + 1]; sorted_cards[j + 1] = temp;: Melakukan pertukaran posisi antara kartu pada posisi j dengan kartu pada posisi j + 1, menggunakan sebuah variabel sementara temp untuk menyimpan nilai sementara dari kartu.

printf("Urutan kartu setelah pertukaran ke-%d: %s\n", (i * (n - 1)) + j + 1, sorted_cards);: Menampilkan urutan kartu setelah terjadi pertukaran. Ini membantu untuk melihat bagaimana urutan kartu berubah pada setiap langkah pertukaran.

```
  int swaps = 0;
    for (int i = 0; i < n; i++) {
        if (cards[i] != sorted_cards[i]) {
            swaps++;
        }
    }

    int min_swaps = swaps / 2;

    free(sorted_cards);
    return min_swaps;
}
```

int swaps = 0;, Inisialisasi variabel swaps

for (int i = 0; i < n; i++) , Memulai loop untuk iterasi melalui setiap kartu.

if (cards[i] != sorted_cards[i]) , Memeriksa apakah kartu pada posisi i dalam array cards tidak sama dengan kartu pada posisi i dalam array sorted_cards.

swaps++ ; , Menambahkan nilai counter swaps jika terjadi perbedaan antara kartu

free(sorted_cards); , Melakukan dealokasi memori yang dialokasikan sebelumnya

return min_swaps;: Mengembalikan nilai jumlah minimal langkah pertukaran yang telah dihitung.

```
int main() {
    int n;
    scanf("%d", &n);
    char cards[n + 1];
    scanf("%s", cards);
    printf("Jumlah minimal langkah pertukaran: %d\n", min_swaps_to_sort(cards,
n));

    return 0;
}
```

char cards[n + 1]; , Mendeklarasikan array cards dengan panjang sebanyak n + 1

printf("Jumlah minimal langkah pertukaran: %d\n", min_swaps_to_sort(cards, n)); , Memanggil fungsi ‘min_swaps_to_sort’ dengan argumen cards (array nilai-nilai kartu) dan n (jumlah kartu), dan mencetak hasilnya ke layar. Ini akan mencetak jumlah minimal langkah