Nama : Yanuar Nurul Hilal
NIM : 222112418
Kelas : 2KS4

Penugasan Struktur Data Praktikum 12

1. Modifikasilah file modul12a.c, modul12b.c, modul12c.c, dan modul12d.c supaya dapat juga mengakomodasi pengurutan data menurun (descending).

```c
// modul12a
#include <stdio.h>
#define MAX 100
// ukuran maksimum array
void fill_data(int data[], int *size)
{ // mengisi data
    printf("Input ukuran array (max 100): ");
    scanf("%d", size);
    printf("Input data: ");
    for (int i = 0; i < *size; i++)
    {
        scanf("%d", &data[i]);
    }
}
void tampil_data(int data[], int size)
{
    for (int i = 0; i < size; i++)
        printf("%d ", data[i]);
    printf("\n");
}
void insertionSort(int arr[], int n)
{
    int i, key, j;
    for (i = 1; i < n; i++)
    {
        key = arr[i];
        j = i - 1;
        while (j >= 0 && arr[j] < key)
        {
            arr[j + 1] = arr[j];
            j = j - 1;
        }
        arr[j + 1] = key;
    }
}
```

```c
void main()
{
    int data[MAX];
    int size; // ukuran array yang dipakai
    fill_data(data, &size);
    insertionSort(data, size);
    printf("data setelah diurutkan:\n");
    tampil_data(data, size);
}
```

```c
// modul12b
#include <stdio.h>
#define MAX 100 // ukuran maksimum array

void fill_data(int data[], int *size)
{ // mengisi data
    printf("Input ukuran array (max 100): ");
    scanf("%d", size);
    printf("Input data: ");
    for (int i = 0; i < *size; i++)
    {
        scanf("%d", &data[i]);
    }
}

void tampil_data(int data[], int size)
{
    for (int i = 0; i < size; i++)
        printf("%d ", data[i]);
    printf("\n");
}

void swap(int *a, int *b)
{
    int temp = *a;
    *a = *b;
    *b = temp;
}
```

```c
void selection_sort(int data[], int size)
{
    for (int step = 0; step < size - 1; step++)
    {
        int min_idx = step;
        for (int i = step + 1; i < size; i++)
            if (data[i] > data[min_idx])
                min_idx = i;
        swap(&data[min_idx], &data[step]);
    }
}

void main()
{
    int data[MAX];
    int size; // ukuran array yang dipakai
    fill_data(data, &size);
    selection_sort(data, size);
    printf("data setelah diurutkan:\n");
    tampil_data(data, size);
}
```

```c
// modul12c
#include <stdio.h>
#define MAX 100 //ukuran maksimum array

void fill_data(int data[], int *size){ //mengisi data
    printf("Input ukuran array (max 100): ");
    scanf("%d", size);
    printf("Input data: ");
    for(int i=0;i<*size;i++){
        scanf("%d",&data[i]);
}
}

void tampil_data(int data[], int size){
    for(int i=0;i<size;i++) printf("%d ",data[i]);
    printf("\n");
}
```

```
void merge(int arr[], int l, int m, int r){
    int i, j, k;
    int n1 = m - l + 1;
    int n2 = r - m;
    int L[n1], R[n2];
    for (i = 0; i < n1; i++)
        L[i] = arr[l + i];
    for (j = 0; j < n2; j++)
        R[j] = arr[m + 1 + j];
    i = 0;
    j = 0;
    k = l;
    while (i < n1 && j < n2){
        if (L[i] >= R[j]) {
            arr[k] = L[i];
            i++;
        }
        else{
            arr[k] = R[j];
            j++;
        }
        k++;
    }
    while (i < n1){
        arr[k] = L[i];
        i++;
        k++;
        }
    while (j < n2){
        arr[k] = R[j];
        j++;
        k++;
    }
}

void mergeSort(int arr[], int l, int r){
    if (l < r) {
        int m = l + (r - l) / 2;
        mergeSort(arr, l, m);
        mergeSort(arr, m + 1, r);
        merge(arr, l, m, r);
    }
}
```

```
void main(){
    int data[MAX];
    int size; //ukuran array yang dipakai
    fill_data(data,&size);
    mergeSort(data,0,size-1);
    printf("data setelah diurutkan:\n");
    tampil_data(data,size);
}
```

```
// modul12d
#include <stdio.h>
#define MAX 100 // ukuran maksimum array

void fill_data(int data[], int *size)
{ // mengisi data
    printf("Input ukuran array (max 100): ");
    scanf("%d", size);
    printf("Input data: ");
    for (int i = 0; i < *size; i++)
    {
        scanf("%d", &data[i]);
    }
}

void tampil_data(int data[], int size)
{
    for (int i = 0; i < size; i++)
        printf("%d ", data[i]);
    printf("\n");
}

void swap(int *a, int *b)
{
    int temp = *a;
    *a = *b;
    *b = temp;
}
```

```c
void bubbleSort(int arr[], int n)
{
    int i, j;
    for (i = 0; i < n - 1; i++)
        for (j = 0; j < n - i - 1; j++)
            if (arr[j] < arr[j + 1])
                swap(&arr[j], &arr[j + 1]);
}

void main()
{
    int data[MAX];
    int size; // ukuran array yang dipakai
    fill_data(data, &size);
    bubbleSort(data, size);
    printf("data setelah diurutkan:\n");
    tampil_data(data, size);
}
```

2. Gabungkan keempat metode sorting tersebut, kemudian buatlah sebuah menu sehingga pengguna dapat memilih metode pengurutan yang diinginkan dan juga memilih urutan menaik atau menurun.

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

#define MAX 100 // ukuran maksimum array

void fill_data(int data[], int size)
{ // mengisi data
    printf("Input data: ");
    for (int i = 0; i < size; i++)
    {
        scanf("%d", &data[i]);
    }
}
void tampil_data(int data[], int size)
{
    for (int i = 0; i < size; i++)
        printf("%d ", data[i]);
    printf("\n");
}
```

```c
void swap(int *a, int *b)
{
    int temp = *a;
    *a = *b;
    *b = temp;
}

void insertionSortasc(int arr[], int n)
{
    int i, key, j;
    for (i = 1; i < n; i++)
    {
        key = arr[i];
        j = i - 1;
        while (j >= 0 && arr[j] > key)
        {
            arr[j + 1] = arr[j];
            j = j - 1;
        }
        arr[j + 1] = key;
    }
}

void insertionSortdesc(int arr[], int n)
{
    int i, key, j;
    for (i = 1; i < n; i++)
    {
        key = arr[i];
        j = i - 1;
        while (j >= 0 && arr[j] < key)
        {
            arr[j + 1] = arr[j];
            j = j - 1;
        }
        arr[j + 1] = key;
    }
}


void selection_sortasc(int data[], int size)
{
    for (int step = 0; step < size - 1; step++)
    {
        int min_idx = step;
        for (int i = step + 1; i < size; i++)
            if (data[i] < data[min_idx])
                min_idx = i;
        swap(&data[min_idx], &data[step]);
    }
}

void selection_sortdesc(int data[], int size)
{
    for (int step = 0; step < size - 1; step++)
    {
        int min_idx = step;
        for (int i = step + 1; i < size; i++)
            if (data[i] > data[min_idx])
                min_idx = i;
        swap(&data[min_idx], &data[step]);
    }
}


void mergeasc(int arr[], int l, int m, int r)
{
    int i, j, k;
    int n1 = m - l + 1;
    int n2 = r - m;
    int L[n1], R[n2];
    for (i = 0; i < n1; i++)
        L[i] = arr[l + i];
    for (j = 0; j < n2; j++)
        R[j] = arr[m + 1 + j];
    i = 0;
    j = 0;
    k = l;
    while (i < n1 && j < n2)
    {
        if (L[i] <= R[j])
        {
            arr[k] = L[i];
            i++;
        }
        else
        {
            arr[k] = R[j];
            j++;
        }
        k++;
    }
    while (i < n1)
    {
        arr[k] = L[i];
        i++;
        k++;
    }
    while (j < n2)
    {
        arr[k] = R[j];
        j++;
        k++;
    }
}

void mergedesc(int arr[], int l, int m, int r){
    int i, j, k;
    int n1 = m - l + 1;
    int n2 = r - m;
    int L[n1], R[n2];
    for (i = 0; i < n1; i++)
        L[i] = arr[l + i];
    for (j = 0; j < n2; j++)
        R[j] = arr[m + 1 + j];
    i = 0;
    j = 0;
    k = l;
    while (i < n1 && j < n2){
        if (L[i] >= R[j]) {
            arr[k] = L[i];
            i++;
        }
        else{
            arr[k] = R[j];
            j++;
        }
        k++;
    }
    while (i < n1){
        arr[k] = L[i];
        i++;
        k++;
        }
    while (j < n2){
        arr[k] = R[j];
        j++;
        k++;
    }
}


void mergeSortasc(int arr[], int l, int r)
{
    if (l < r)
    {
        int m = l + (r - l) / 2;
        mergeSortasc(arr, l, m);
        mergeSortasc(arr, m + 1, r);
        mergeasc(arr, l, m, r);
    }
}

void mergeSortdesc(int arr[], int l, int r)
{
    if (l < r)
    {
        int m = l + (r - l) / 2;
        mergeSortdesc(arr, l, m);
        mergeSortdesc(arr, m + 1, r);
        mergedesc(arr, l, m, r);
    }
}



void bubbleSortasc(int arr[], int n)
{
    int i, j;
    for (i = 0; i < n - 1; i++)
        for (j = 0; j < n - i - 1; j++)
            if (arr[j] > arr[j + 1])
```

```c
                swap(&arr[j], &arr[j + 1]);
}

void bubbleSortdesc(int arr[], int n)
{
    int i, j;
    for (i = 0; i < n - 1; i++)
        for (j = 0; j < n - i - 1; j++)
            if (arr[j] < arr[j + 1])
                swap(&arr[j], &arr[j + 1]);
}


void main()
{
    int data[MAX];
    int size; // ukuran array yang dipakai
    int metode;
    char NT;
    char NT_string[5];

    printf("Input ukuran array (max 100): ");
    scanf("%d", &size);

    printf("Metode Sorting yang tersedia\n1. Insertion Sort\n2. Selection
Sort\n3. Merge Sort\n4. Bubble Sort\nPilih Metode Sorting (1/2/3/4): ");
    scanf("%d",&metode);

    printf("Pilih pengurutan Naik/Turun(N/T): ");
    getchar();
    scanf("%c",&NT);

    if (NT=='N')
    {
         strcpy(NT_string,"Naik");
    }else
    {
         strcpy(NT_string,"Turun");
    }

    switch (metode)
    {
    case 1:
        fill_data(data, size);
        printf("Menjalankan sorting dengan metode Insertion Sort");

        if (NT=='N')
            insertionSortasc(data,size);
        else
            insertionSortdesc(data,size);

        break;

    case 2:
        fill_data(data, size);
        printf("Menjalankan sorting dengan metode Selection Sort");

        if(NT=='N')
            selection_sortasc(data,size);
        else
            selection_sortdesc(data,size);
        break;

    case 3:
        fill_data(data, size);
        printf("Menjalankan sorting dengan metode Merge Sort");
        if(NT=='N')
            mergeSortasc(data,0,size-1);
        else
            mergeSortdesc(data,0,size-1);
        break;


    case 4:
        fill_data(data, size);
        printf("Menjalankan sorting dengan metode Bubble Sort");
        if(NT=='N')
            bubbleSortasc(data, size);
        else
            bubbleSortdesc(data, size);
        break;

    default:
        break;
    }
    printf("\nPilihan pengurutan %s", NT_string);
    printf("\nData setelah diurutkan: ");
    tampil_data(data, size);
}
```

3. Buat program untuk input dan pengurutan data students berisi int nim, char nama[50], int nilai dengan struktur array. Pengurutan bisa berdasarkan nim, nama, atau nilai terserah pada pilihan pengguna.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#define MAX 100 // ukuran maksimum array

void fill_nim(int nim[],int nilai[], char nama[][50], int *size)
{ // mengisi nim
    printf("Masukkan jumlah mahasiswa (max 100): ");
    scanf("%d", size);
    for (int i = 0; i < *size; i++)
    {
        printf("NIM Mahasiswa ke-[%d] : ", i+1);
        scanf("%d", &nim[i]);
        getchar();
        printf("Nama Mahasiswa-[%d] : ", i+1);
        gets(nama[i]);
        printf("Nilai Mahasiswa ke-[%d] : ",i+1);
        scanf("%d",&nilai[i]);
        printf("\n");
    }
}

void tampil_data(int nim[],int nilai[], char nama[][50], int size)
{
    printf("NIM\tNama\tNilai");
    for (int i = 0; i < size; i++)
        printf("\n%d\t%s\t%d", nim[i], nama[i], nilai[i]);
    printf("\n");
}

void swap(int *a, int *b)
{
    int temp = *a;
    *a = *b;
```

```c
    *b = temp;
}
void swapNama(char a[][50], char b[][50])
{
    char temp[100];
    strcpy(temp, a[0]);
    strcpy(a[0], b[0]);
    strcpy(b[0], temp);
}

void bubbleSort(int arr[], char name[][50], int n)
{
    int i, j;
    for (i = 0; i < n - 1; i++)
        for (j = 0; j < n - i - 1; j++)
            if (arr[j] > arr[j + 1])
            {
                swap(&arr[j], &arr[j + 1]);
                swapNama(&name[j], &name[j + 1]);
            }
}

void bubbleSortNama(int arr[],int nilai[], char name[][50], int n)
{
    int i, j;
    for (i = 0; i < n - 1; i++)
        for (j = 0; j < n - i - 1; j++)
            if (strcmp(name[j], name[j + 1]) > 0)
            {
                swap(&arr[j], &arr[j + 1]);
                swapNama(&name[j], &name[j + 1]);
            }
}

void bubbleSortNIM(int arr[],int nilai[], char name[][50], int n)
{
    int i, j;
    for (i = 0; i < n - 1; i++)
        for (j = 0; j < n - i - 1; j++)

            if (arr[j] > arr[j + 1])
                swap(&arr[j], &arr[j + 1]);
}
```

```c
void bubbleSortNilai(int arr[],int nilai[], char name[][50], int n)
{
    int i, j;
    for (i = 0; i < n - 1; i++)
        for (j = 0; j < n - i - 1; j++)

            if (nilai[j] < nilai[j + 1])
                swap(&nilai[j], &nilai[j + 1]);
}

void main()
{
    int nim[MAX];
    int nilai [MAX];
    char nama[MAX][50];
    int size; // ukuran array
    int choice;
    fill_nim(nim,nilai, nama, &size);

    printf("Pilih mekanisme pengurutan : \n");
    printf("1.Berdasarkan Nama\n2.Berdasarkan NIM\n3.Berdasarkan Nilai\n");
    printf("Pilihan Anda : ");scanf("%d",&choice);
    switch(choice)
    {
        case 1:
            bubbleSortNama(nim,nilai, nama, size);
            printf("\nData setelah diurutkan berdasarkan Nama:\n");
            tampil_data(nim,nilai, nama, size);
            break;
        case 2:
            bubbleSortNIM(nim,nilai,nama,size);
            printf("\nData setelah diurutkan berdasarkan NIM:\n");
            tampil_data(nim,nilai, nama, size);
            break;
        case 3:
            bubbleSortNilai(nim,nilai,nama,size);
            printf("\nData setelah diurutkan berdasarkan Nilai:\n");
            tampil_data(nim,nilai, nama, size);
            break;
        default:
            break;
    }
}
```