Task 1. Ketik dan jalankan program di bawah ini secara mandiri, pahami bagaimana masing-masing komponen pada program ini bekerja :

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

struct Mahasiswa {
    char nama[50];
    int nim;
    float ipk;
}

int main() {
    int angka = 10;
    int *ptrAngka = &angka;

    printf("Nilai angka: %d\n", *ptrAngka);

    int arrayAngka[5] = {1,2,3,4,5};

    printf("Elemen-elemen array");

    for (int i = 0; i < 5; ++i) {
        printf("%d ", arrayAngka[i]);
    }
    printf("\n");

    struct Mahasiswa mhs1;
    strcpy(mhs1.nama, "John Doe");
    mhs1.nim = 12345;
    mhs1.ipk = 3.75;

    printf("Data Mahasiswa:\n");
    printf("Nama: %s\n", mhs1.nama);
    printf("NIM: %d\n", mhs1.nim);
    printf("IPK: %.2f\n", mhs1.ipk);

    std::vector<int> vectorAngka;
    for (int i = 0; i < 5; ++i) {
        vectorAngka.push_back(i * 2);
    }
    printf("Elemen-elemen vector: ");

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

```
    // 1. Modifikasi nilai variabel angka menggunakan pointer
    *ptrAngka = 20;

    // 2. Modifikasi IPK mahasiswa mhs1 menggunakan pointer
    float *ptrIpk = &mhs1.ipk;
    *ptrIpk = 3.90;

    // 3. Tambahkan dua elemen baru ke dalam vectorAngka
    vectorAngka.push_back(12);
    vectorAngka.push_back(15);

    // Tampilkan hasil setelah modifikasi
    printf("Setelah dimodifikasi:\n");
    printf("Nilai angka setelah dimodifikasi: %d\n", *ptrAngka);
    printf("IPK mahasiswa setelah dimodifikasi: %.2f\n", mhs1.ipk);
    printf("Elemen-elemen vector setelah dimodifikasi: ");
    for (int i = 0; i < vectorAngka.size(); ++i) {
        printf("%d ", vectorAngka[i]);
    }

    printf("\n");

    return 0;
}
```

```
Nilai angka: 10
Elemen-elemen array1 2 3 4 5
Data Mahasiswa:
Nama: John Doe
NIM: 12345
IPK: 3.75
Elemen-elemen vector: 0 2 4 6 8
Setelah dimodifikasi:
Nilai angka setelah dimodifikasi: 20
IPK mahasiswa setelah dimodifikasi: 3.90
Elemen-elemen vector setelah dimodifikasi: 0 2 4 6 8 12 15
```

Task 2. Setelah Anda menjalankan program tersebut, diskusikan tugas di bawah ini dengan kelompok Anda, lalu kumpulkan hasilnya secara mandiri (berupa file nim.cpp dan penjelasannya dalam nim.docx menggunakan kalimat Anda masing-masing):

1. Modifikasilah program agar dapat menyimpan informasi lebih dari satu mahasiswa menggunakan array dari struct data Mahasiswa. Berikan penjelasan mengenai modifikasi yg Anda lakukan.

```
void addmhs(std::vector<Mahasiswa>& mhs, const char* nama, int nim, float
ipk) {
    Mahasiswa temp;
    strcpy(temp.nama, nama);
    temp.nim = nim;
    temp.ipk = ipk;
    mhs.push_back(temp);
}

int main() {

    std::vector<Mahasiswa> mhs;

    addmhs(mhs, "A", 64011, 4.00);
    addmhs(mhs, "B", 64012, 3.90);
    addmhs(mhs, "C", 64013, 3.90);
}
```

Penjelasan

Saya membuat fungsi baru yang untuk menambahkan mahasiswa ke dalam vektor dari mahasiswa. Ini membuat vector data mahasiswa sementara dengan sebagaimana pada fungsi struct yang telah dibuat, lalu mengcopy nya ke dalam vektor mahasiswa

2. Tambahkan sebuah fungsi untuk mencetak semua mahasiswa beserta informasi mereka menggunakan pointer. Apa kelebihan dan kekurangan cara ini dibandingkan penggunaan index array?

```
void printmhs(const std::vector<Mahasiswa>& mhs) {
    printf("Data Semua Mahasiswa:\n");
    for (const Mahasiswa* ptr = mhs.data(); ptr < mhs.data() + mhs.size();
++ptr) {
        printf("Nama: %s\n", ptr->nama);
        printf("NIM: %d\n", ptr->nim);
        printf("IPK: %.2f\n", ptr->ipk);
        printf("\n");
    }
}
```

Penjelasan

mhs.data() memberikan pointer ke elemen pertama dalam std::vector. Pointer ptr digunakan untuk mengakses elemen-elemen dalam vektor, dan iterasi dilakukan hingga akhir vektor (mhs.data() + mhs.size()). Operator panah digunakan untuk mengakses anggota dari elemen Mahasiswa yang ditunjuk oleh pointer.

Kelebihan Arrays: Lebih mudah, resiko kesalahan dalam memprogramnya lebih kecil

Kekurangan Arrays: Kurang fleksibel, peforma lebih lambat.

Kelebihan Pointer: Flesibelitas tinggi, performa lebih cepat.

Kekurangan Pointer: Mudah melakukan error pada pembuatan fungsi yang kompleks.

3. Gantilah array yang digunakan untuk menyimpan mahasiswa dengan vektor dari struct data Mahasiswa. Apa kelebihan dan kekurangan penggunaan array dan vector?

```
void addmhs(std::vector<Mahasiswa>& mhs, const char* nama, int nim, float
ipk) {
    Mahasiswa temp;
    strcpy(temp.nama, nama);
    temp.nim = nim;
    temp.ipk = ipk;
    mhs.push_back(temp);
}

int main() {

       std::vector<Mahasiswa> mhs;

    addmhs(mhs, "A", 64011, 4.00);
    addmhs(mhs, "B", 64012, 3.90);
    addmhs(mhs, "C", 64013, 3.90);
}
```

Kelebihan & Kekurangan Array:

Respon time lebih cepat karena ukuran array sudah ditetapkan dan tidak perlu memanajemen memori saat program berjalan, akan tetapi ini membuatnya tidak fleksibel dan jika terlalu besar akan memakan banyak memori

Kelebihan & Kekurangan Vector:

Ukuran data fleksibel karena ukuran vektor dapat menyesuaikan sebagaimana banyak data yang dibutuhkan, akan tetapi proses ini memakan waktu respon saat program berjalan.

3. Tambahkan dua mahasiswa baru ke dalam vector tersebut menggunakan fungsi dari STL. Bagaimana cara kerja fungsi dari STL ini?

```
void addmhs(std::vector<Mahasiswa>& mhs, const char* nama, int nim, float
ipk) {
    Mahasiswa temp;
    strcpy(temp.nama, nama);
    temp.nim = nim;
    temp.ipk = ipk;
    mhs.push_back(temp);
}

int main() {

    std::vector<Mahasiswa> mhs;

    addmhs(mhs, "A", 64011, 4.00);
    addmhs(mhs, "B", 64012, 3.90);
    addmhs(mhs, "C", 64013, 3.90);
    addmhs(mhs, "D", 64014, 3.80);
    addmhs(mhs, "E", 64015, 3.70);
}

    printf("Data Mahasiswa:\n");
    for (size_t i = 0; i < mhs.size(); ++i) {
        printf("Mahasiswa %zu:\n", i + 1);
        printf("Nama: %s\n", mhs[i].nama);
        printf("NIM: %d\n", mhs[i].nim);
        printf("IPK: %.2f\n", mhs[i].ipk);
        printf("\n");
    }
```

Penjelasan

Saya membuat fungsi untuk menambahkan mahasiswa yang terdapat push_back yang dimana akan menambahakan data ke dalam vektor mahasiswa, sehingga saya cukup menggunakan fungsi untuk menambahkan mahasiswa yang telah saya buat, dan untuk menampilkannya semua saya cukup men-set iterasinya sebanyak size dari banyak siswanya.

5. Implementasikan fungsi atau metode untuk mencari dan menampilkan mahasiswa dengan NIM tertentu.

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

struct Mahasiswa {
```

```
    char nama[50];
    int nim;
    float ipk;
};

void addmhs(std::vector<Mahasiswa>& mhs, const char* nama, int nim, float
ipk) {
    Mahasiswa temp;
    strcpy(temp.nama, nama);
    temp.nim = nim;
    temp.ipk = ipk;
    mhs.push_back(temp);
}

void srcmhs(const std::vector<Mahasiswa>& mhs, int nim) {
    bool found = false;
    for (const auto& mahasiswa : mhs) {
        if (mahasiswa.nim == nim) {
            printf("Mahasiswa ditemukan:\n");
            printf("Nama: %s\n", mahasiswa.nama);
            printf("NIM: %d\n", mahasiswa.nim);
            printf("IPK: %.2f\n", mahasiswa.ipk);
            found = true;
            break;
        }
    }
    if (!found) {
        printf("Mahasiswa dengan NIM %d tidak ditemukan.\n", nim);
    }
}

int main() {
    int angka = 10;
    int *ptrAngka = &angka;
    printf("Nilai angka: %d\n", *ptrAngka);

    int arrayAngka[5] = {1, 2, 3, 4, 5};
    printf("Elemen-elemen array: ");
    for (int i = 0; i < 5; ++i) {
        printf("%d ", arrayAngka[i]);
    }
    printf("\n");

    std::vector<Mahasiswa> mhs;

    addmhs(mhs, "A", 64011, 4.00);
    addmhs(mhs, "B", 64012, 3.90);
    addmhs(mhs, "C", 64013, 3.90);
```

```
    addmhs(mhs, "D", 64014, 3.80);
    addmhs(mhs, "E", 64015, 3.70);

    printf("Data Mahasiswa:\n");
    for (size_t i = 0; i < mhs.size(); ++i) {
        printf("Mahasiswa %zu:\n", i + 1);
        printf("Nama: %s\n", mhs[i].nama);
        printf("NIM: %d\n", mhs[i].nim);
        printf("IPK: %.2f\n", mhs[i].ipk);
        printf("\n");
    }

    // Mencari mahasiswa dengan NIM tertentu
    int nimDicari = 64012; // Ganti dengan NIM yang ingin dicari
    srcmhs(mhs, nimDicari);

    std::vector<int> vectorAngka;
    for (int i = 0; i < 5; ++i) {
        vectorAngka.push_back(i * 2);
    }
    printf("Elemen-elemen vector: ");
    for (int i = 0; i < vectorAngka.size(); ++i) {
        printf("%d ", vectorAngka[i]);
    }
    printf("\n");

    *ptrAngka = 20;

    float *ptrIpk = &mhs[0].ipk;
    *ptrIpk = 3.90;

    vectorAngka.push_back(12);
    vectorAngka.push_back(15);

    printf("Setelah dimodifikasi:\n");
    printf("Nilai angka setelah dimodifikasi: %d\n", *ptrAngka);
    printf("IPK mahasiswa pertama setelah dimodifikasi: %.2f\n",
mhs[0].ipk);
    printf("Elemen-elemen vector setelah dimodifikasi: ");
    for (int i = 0; i < vectorAngka.size(); ++i) {
        printf("%d ", vectorAngka[i]);
    }
    printf("\n");

    return 0;
}
```

```
Nilai angka: 10
Elemen-elemen array: 1 2 3 4 5
Data Mahasiswa:
Mahasiswa 1:
Nama: A
NIM: 64011
IPK: 4.00

Mahasiswa 2:
Nama: B
NIM: 64012
IPK: 3.90

Mahasiswa 3:
Nama: C
NIM: 64013
IPK: 3.90

Mahasiswa 4:
Nama: D
NIM: 64014
IPK: 3.80

Mahasiswa 5:
Nama: E
NIM: 64015
IPK: 3.70

Mahasiswa ditemukan:
Nama: B
NIM: 64012
IPK: 3.90

Elemen-elemen vector: 0 2 4 6 8
Setelah dimodifikasi:
Nilai angka setelah dimodifikasi: 20
IPK mahasiswa pertama setelah dimodifikasi: 3.90
Elemen-elemen vector setelah dimodifikasi: 0 2 4 6 8 12 15
```