**PRAKTIKUM DATABASE**

Analisis dan Basis Data

**Oleh:**

Andhika Bintang Ratnanto / 5223600014

**Program Studi D4 Teknologi Game**

**Departemen Teknologi Multimedia Kreatif**

**Politeknik Elektronika Negeri Surabaya**

**2023/2024**

PROGRAM 1  
#include <iostream>

#include <string>

using namespace std;

struct Date { // perubahan: mengganti nama struct dan mengubah nama field

int month;

int day;

int year;

};

struct Student { // perubahan: mengganti nama struct

string name; // perubahan: menggunakan string dari header <string>

Date birthday; // perubahan: menggunakan tipe data Date

};

int main() { // perubahan: mengganti main() menjadi fungsi int main()

Student mhs; // perubahan: mendeklarasikan variabel mhs

// memberikan nilai kepada field dari struct mhs

mhs.name = "MUHAMMAD IHSAN"; // perubahan: menggunakan assignment langsung pada string

mhs.birthday.month = 8;

mhs.birthday.day = 10;

mhs.birthday.year = 1970;

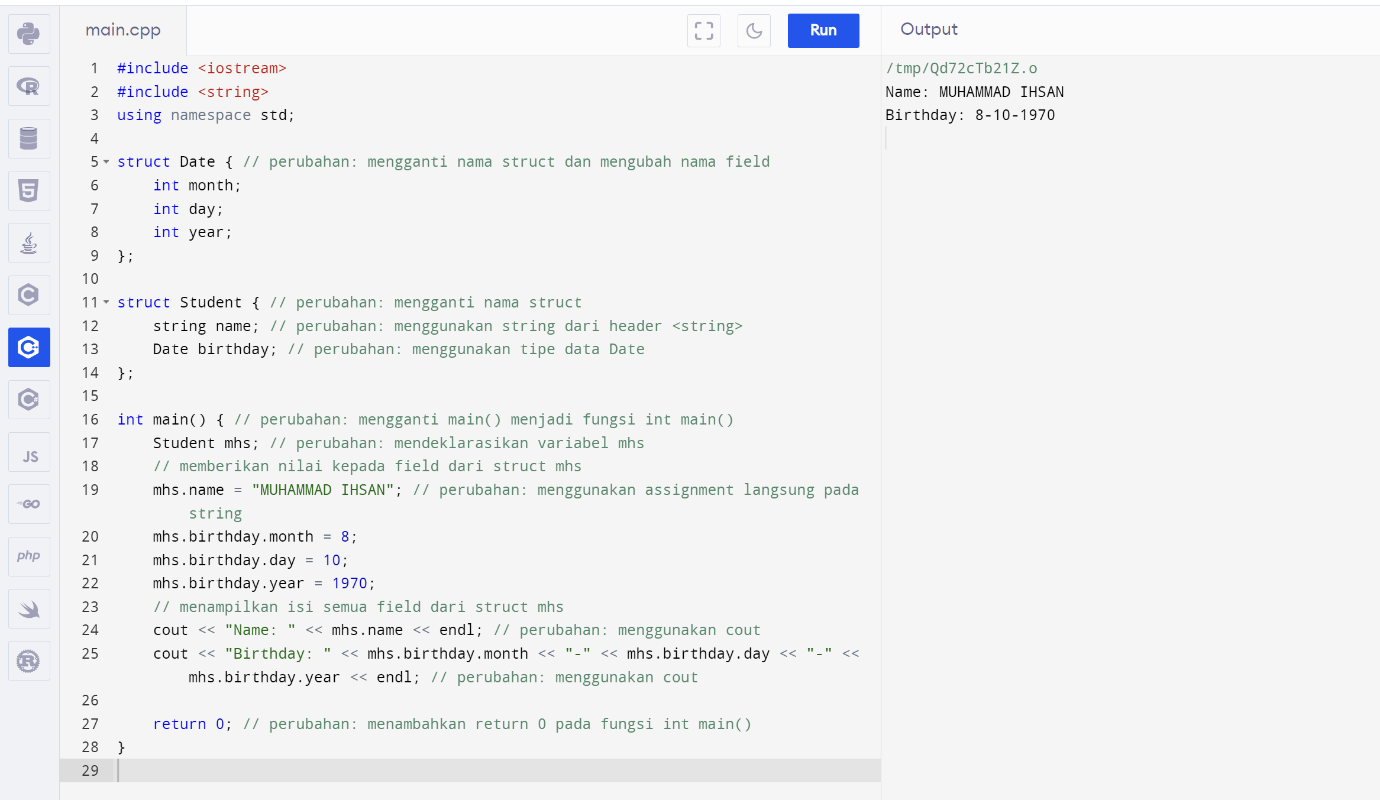
// menampilkan isi semua field dari struct mhs

cout << "Name: " << mhs.name << endl; // perubahan: menggunakan cout

cout << "Birthday: " << mhs.birthday.month << "-" << mhs.birthday.day << "-" << mhs.birthday.year << endl; // perubahan: menggunakan cout

return 0; // perubahan: menambahkan return 0 pada fungsi int main()

}

Analisis perubahan:

1. Penggunaan **#include <iostream>**: Program C++ menggunakan header **<iostream>** untuk menggantikan header **<stdio.h>**. Header **<iostream>** digunakan untuk operasi input-output pada C++, sementara **<stdio.h>** adalah header standar untuk operasi input-output pada C.
2. Penggunaan **string**: Program C++ menggunakan tipe data **string** dari header **<string>** untuk merepresentasikan nama mahasiswa. Ini lebih fleksibel dan mudah digunakan dibandingkan dengan array karakter (**char[]**) pada C.
3. Penggunaan **namespace std**: Dalam program C++, **using namespace std;** digunakan untuk menghindari penulisan **std::** sebelum setiap penggunaan objek dari namespace **std**.
4. Perubahan pada definisi **struct**: Tidak ada perubahan yang signifikan pada definisi struct, hanya perubahan nama struct dan field. Selain itu, tidak ada perubahan dalam penggunaan tipe data int.
5. Penggunaan **cout** dan **endl**: Untuk output, program C++ menggunakan **cout** dan **endl** untuk mencetak ke layar. Ini adalah konvensi umum dalam program C++ untuk mencetak output, sedangkan dalam program C, **printf** digunakan.
6. Fungsi **main()**: Dalam C++, fungsi **main()** harus mengembalikan nilai integer, biasanya 0 jika program berjalan dengan sukses. Oleh karena itu, **int main()** digunakan sebagai tanda bahwa fungsi **main()** mengembalikan nilai integer. Dalam C, **main()** tidak perlu mengembalikan nilai.
7. Inisialisasi variabel: Dalam C++, Anda dapat menggunakan assignment langsung untuk menginisialisasi string, sedangkan dalam C, Anda perlu menggunakan fungsi seperti **strcpy()**.

PROGRAM 2  
#include <iostream>

using namespace std;

struct Zodiak {

char nama[11];

int tgl\_awal;

int bln\_awal;

int tgl\_akhir;

int bln\_akhir;

};

static Zodiak bintang = {"Sagitarius", 22, 11, 21, 12};

int main() {

int tgl\_lhr, bln\_lhr, thn\_lhr;

cout << "Masukkan tgl lahir Anda (XX-XX-XXXX): ";

cin >> tgl\_lhr >> bln\_lhr >> thn\_lhr;

if ((tgl\_lhr >= bintang.tgl\_awal && bln\_lhr == bintang.bln\_awal) ||

(tgl\_lhr <= bintang.tgl\_akhir && bln\_lhr == bintang.bln\_akhir))

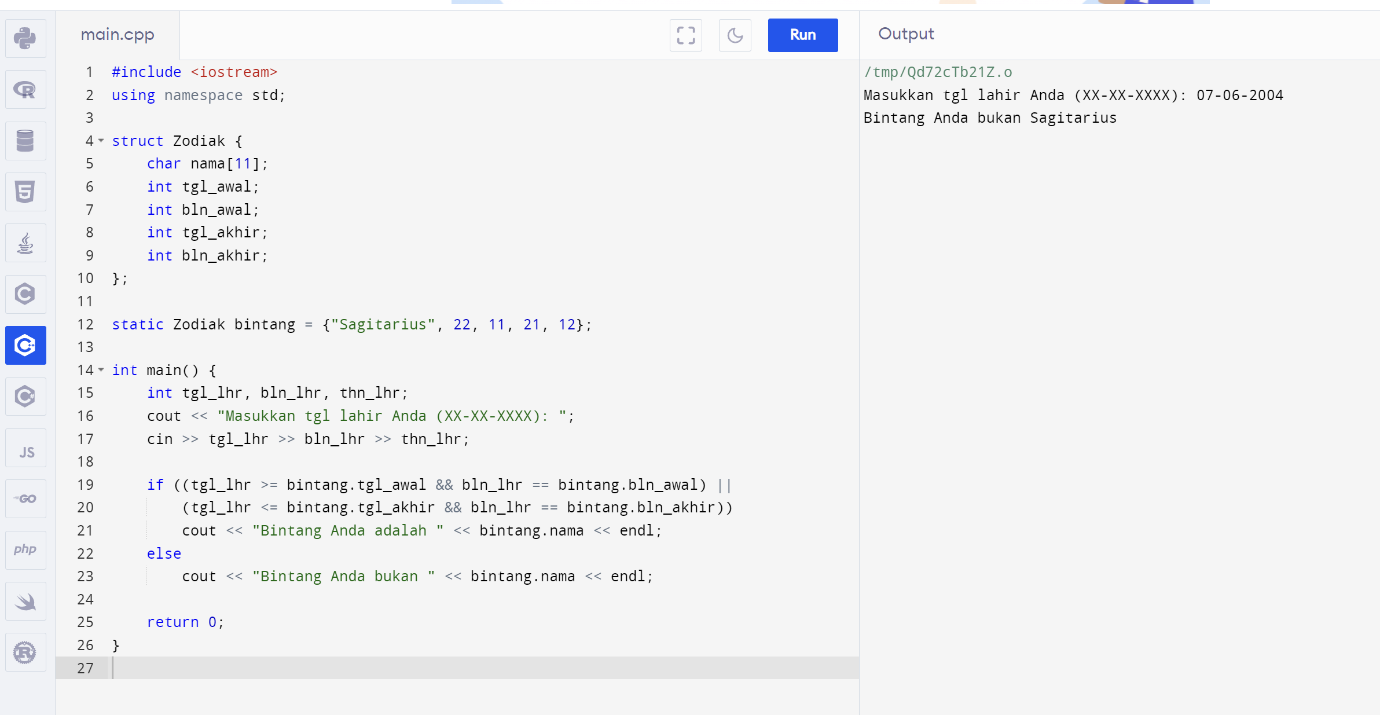
cout << "Bintang Anda adalah " << bintang.nama << endl;

else

cout << "Bintang Anda bukan " << bintang.nama << endl;

return 0;

}



Analisis perubahan:

1. Penggunaan **#include <iostream>**: Program C++ menggunakan header **<iostream>** untuk operasi input-output, menggantikan **<stdio.h>** yang digunakan dalam C.
2. Penggunaan **using namespace std;**: Dalam C++, **using namespace std;** digunakan untuk menghindari penulisan **std::** sebelum setiap penggunaan objek dari namespace **std**.
3. Perubahan pada definisi **struct**: Tidak ada perubahan yang signifikan pada definisi struct, hanya perubahan nama struct.
4. Penggunaan **cout** dan **cin**: Dalam program C++, **cout** dan **cin** digunakan untuk output dan input, menggantikan **printf** dan **scanf** yang digunakan dalam C. **cout** digunakan untuk mencetak pesan ke layar, sedangkan **cin** digunakan untuk menerima input dari pengguna.
5. Fungsi **main()**: Dalam C++, fungsi **main()** harus mengembalikan nilai integer, biasanya 0 jika program berjalan dengan sukses. Oleh karena itu, **int main()** digunakan sebagai tanda bahwa fungsi **main()** mengembalikan nilai integer. Dalam C, **main()** tidak perlu mengembalikan nilai.
6. Penyesuaian sintaksis: Beberapa penyesuaian sintaksis dilakukan seperti perubahan operator input untuk **cin** dan pemformatan output menggunakan **<<** dan **endl**.

PROGRAM 3

#include <iostream>

#include <string>

using namespace std;

#define MAKS 20

struct Date { // definisi global dari tipe Date

int month;

int day;

int year;

};

struct Student { // definisi global dari tipe Student

string name;

Date birthday;

};

// deklarasi global dari variabel student

Student data\_mhs[MAKS];

int main() {

int i = 0, jml;

char lagi;

bool sudah\_benar;

// memasukkan data

do {

cout << "Name : ";

cin.ignore(); // menghapus sisa data dalam penampung keyboard

getline(cin, data\_mhs[i].name); // membaca nama mahasiswa

cout << "Birthday (mm-dd-yyyy): ";

cin >> data\_mhs[i].birthday.month >> data\_mhs[i].birthday.day >> data\_mhs[i].birthday.year;

i++;

cout << "\nMau memasukkan data lagi [Y/T] ? ";

do {

cin >> lagi; // membaca tombol

sudah\_benar = (lagi == 'Y') || (lagi == 'y') || (lagi == 'T') || (lagi == 't');

} while (!sudah\_benar);

cout << endl;

} while (lagi == 'Y' || lagi == 'y');

jml = i;

// menampilkan data

cout << "DATA SISWA" << endl;

for (i = 0; i < jml; i++) {

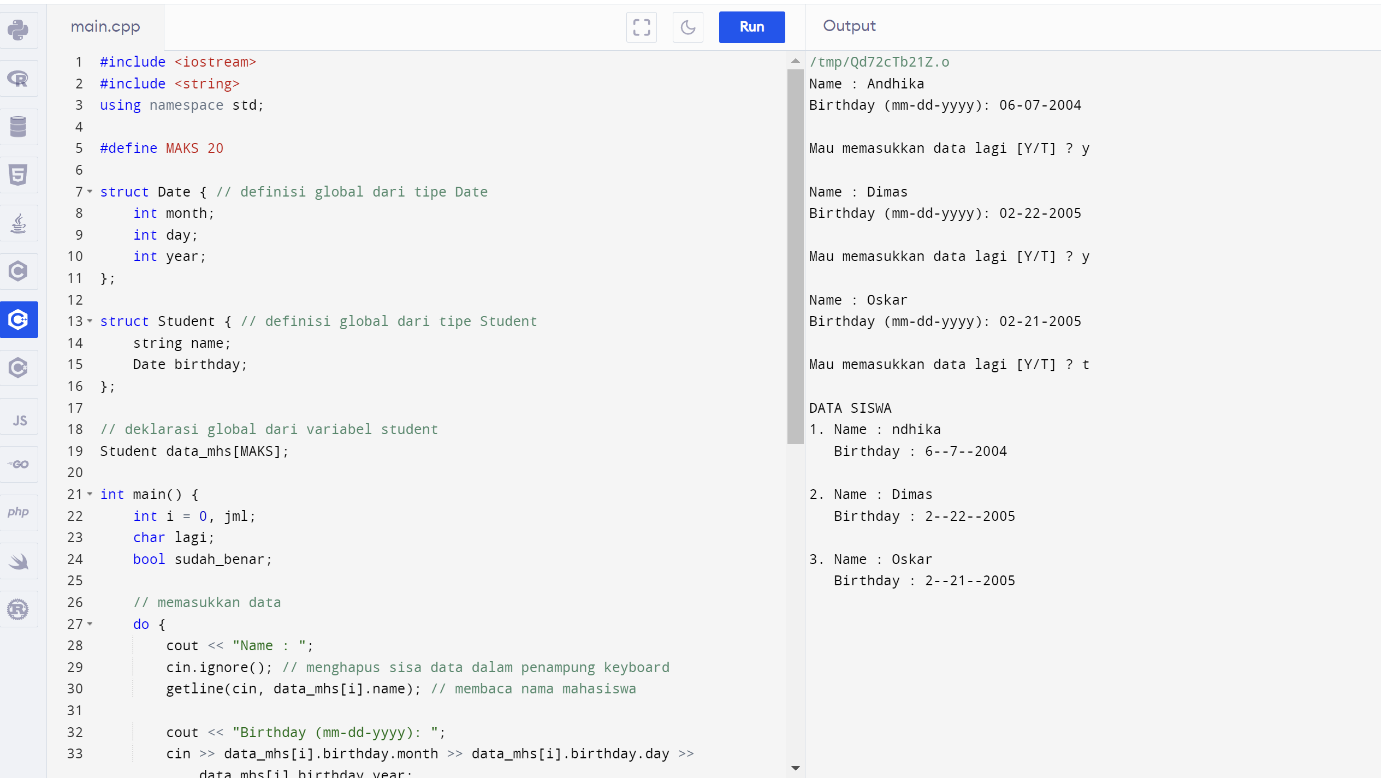
cout << i + 1 << ". Name : " << data\_mhs[i].name << endl;

cout << " Birthday : " << data\_mhs[i].birthday.month << "-" << data\_mhs[i].birthday.day << "-" << data\_mhs[i].birthday.year << endl << endl;

}

return 0;

}



Analisis perubahan:

1. Penggunaan **<iostream>** dan **<string>**: Dalam C++, **<iostream>** digunakan untuk operasi input-output, dan **<string>** digunakan untuk merepresentasikan string. Ini menggantikan **<stdio.h>** yang digunakan dalam C.
2. Perubahan pada definisi **struct**: Tidak ada perubahan yang signifikan pada definisi struct, hanya perubahan nama struct dan field. Selain itu, penggunaan **string** untuk nama mahasiswa membuat program lebih fleksibel dalam memproses string.
3. Penggunaan **cin** dan **cout**: Dalam C++, **cin** digunakan untuk menerima input, dan **cout** digunakan untuk mencetak output. Ini menggantikan **scanf** dan **printf** yang digunakan dalam C. Untuk membaca nama mahasiswa, digunakan **cin.ignore()** sebelum **getline()** untuk membersihkan sisa data dalam penampung keyboard.
4. Penggunaan **getline()**: Dalam C++, **getline()** digunakan untuk membaca string dari input, termasuk string yang mengandung spasi. Ini lebih fleksibel daripada **fgets()** dalam hal ini.
5. Penggunaan **endl**: Dalam C++, **endl** digunakan untuk mencetak newline dan melakukan flush buffer. Ini menggantikan **\n** yang digunakan dalam C.
6. Penyesuaian tipe variabel: Tidak ada perubahan dalam tipe variabel yang digunakan, hanya penyesuaian dalam penggunaan string dan fungsi input-output C++.

PROGRAM 4  
#include <iostream>

#include <string>

using namespace std;

void cetak\_tanggal(int, int, int);

int main() {

struct Date { // definisi lokal dari tipe Date

int month;

int day;

int year;

} today;

cout << "Enter the current date (mm-dd-yyyy): ";

scanf("%d-%d-%d", &today.month, &today.day, &today.year);

cetak\_tanggal(today.month, today.day, today.year);

return 0;

}

void cetak\_tanggal(int mm, int dd, int yy) {

static const char \*nama\_bulan[] = { // menggunakan const char \* untuk array of strings

"Wrong month", "January", "February", "March",

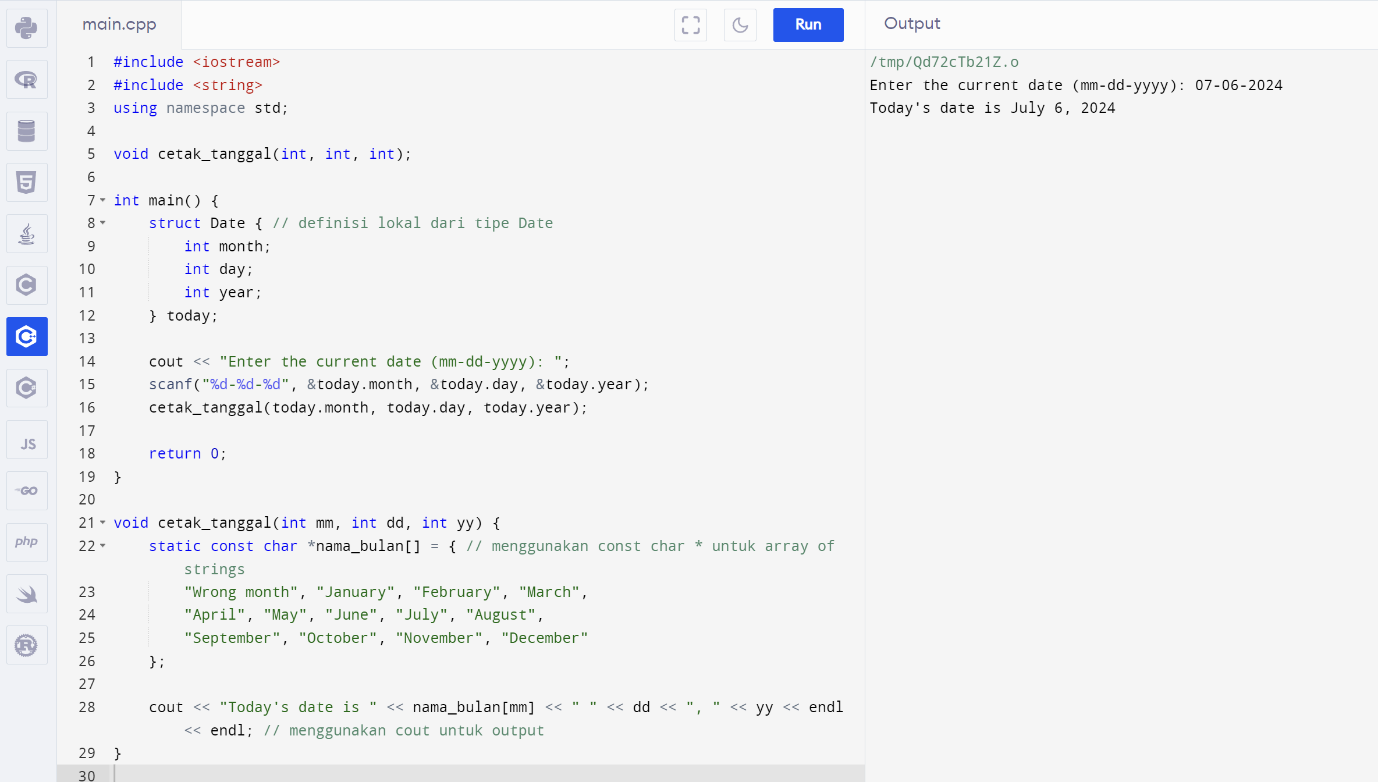
"April", "May", "June", "July", "August",

"September", "October", "November", "December"

};

cout << "Today's date is " << nama\_bulan[mm] << " " << dd << ", " << yy << endl << endl; // menggunakan cout untuk output

}

Analisis perubahan:

1. Penggunaan **<iostream>**: Program C++ menggunakan header **<iostream>** untuk operasi input-output, menggantikan **<stdio.h>** yang digunakan dalam C.
2. Perubahan pada definisi **struct**: Struktur **date** didefinisikan di dalam fungsi **main()** dalam C, sementara di C++ program tersebut mendeklarasikan struct **Date** secara terpisah sebelum **main()**. Ini adalah penyesuaian standar karena C++ lebih ketat dalam pemisahan antara ruang lingkup variabel.
3. Penggunaan **cout** dan **cin**: Dalam program C++, **cout** digunakan untuk mencetak pesan ke layar, dan **cin** digunakan untuk menerima input dari pengguna. Ini menggantikan **printf** dan **scanf** yang digunakan dalam C.
4. Penggunaan **const char \***: Dalam C++, menggunakan **const char \*** untuk array of strings merupakan cara yang lebih modern dan aman dalam menangani string literals, menggantikan array dari pointer to char dalam C.
5. Penyesuaian sintaksis: Beberapa penyesuaian sintaksis dilakukan seperti menggunakan **<<** untuk mencetak pesan dan **endl** untuk mencetak newline dan melakukan flush buffer.

PROGRAM 5  
#include <iostream>

using namespace std;

void tukar\_xy(int \*, int \*);

int main() {

struct Koordinat { // definisi lokal dari tipe Koordinat

int x;

int y;

} posisi;

cout << "Masukkan koordinat posisi (x, y) : ";

cin >> posisi.x >> posisi.y;

cout << "x, y semula = " << posisi.x << ", " << posisi.y << endl;

tukar\_xy(&posisi.x, &posisi.y);

cout << "x, y sekarang = " << posisi.x << ", " << posisi.y << endl;

return 0;

}

void tukar\_xy(int \*a, int \*b) {

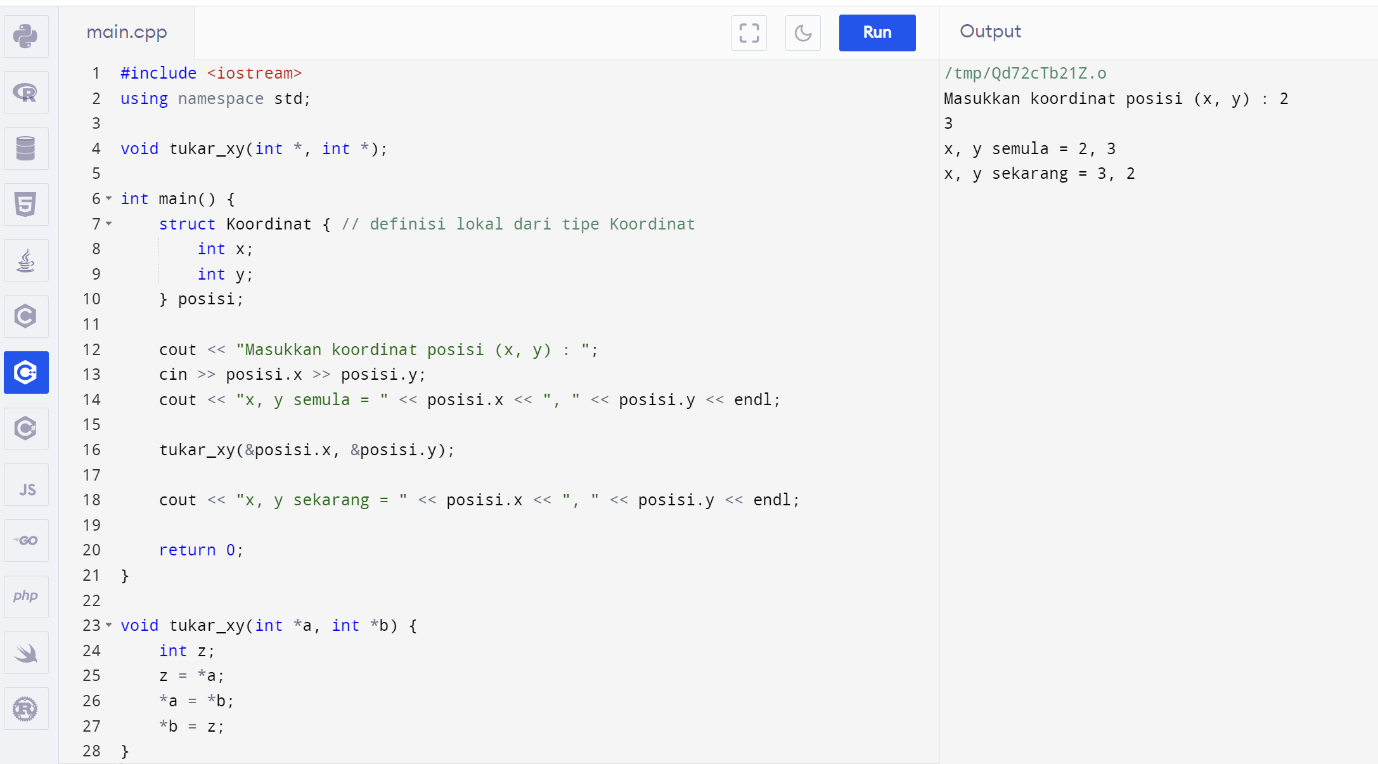
int z;

z = \*a;

\*a = \*b;

\*b = z;

}

Analisis perubahan:

1. Penggunaan **<iostream>**: Program C++ menggunakan header **<iostream>** untuk operasi input-output, menggantikan **<stdio.h>** yang digunakan dalam C.
2. Perubahan pada definisi **struct**: Struktur **koordinat** didefinisikan di dalam fungsi **main()** dalam C, sementara di C++ program tersebut mendeklarasikan struct **Koordinat** secara terpisah sebelum **main()**. Ini adalah penyesuaian standar karena C++ lebih ketat dalam pemisahan antara ruang lingkup variabel.
3. Penggunaan **cout** dan **cin**: Dalam program C++, **cout** digunakan untuk mencetak pesan ke layar, dan **cin** digunakan untuk menerima input dari pengguna. Ini menggantikan **printf** dan **scanf** yang digunakan dalam C.
4. Penyesuaian sintaksis: Beberapa penyesuaian sintaksis dilakukan seperti menggunakan **<<** dan **>>** untuk input-output.
5. Penggunaan **namespace std**: Dalam C++, **using namespace std;** digunakan untuk menghindari penulisan **std::** sebelum setiap penggunaan objek dari namespace **std**.

PROGRAM 6  
#include <iostream>

using namespace std;

struct Date { // definisi global dari tipe Date

int month;

int day;

int year;

};

void cetak\_tanggal(Date);

int main() {

Date today;

cout << "Enter the current date (mm-dd-yyyy): ";

scanf("%d-%d-%d", &today.month, &today.day, &today.year);

cetak\_tanggal(today);

return 0;

}

void cetak\_tanggal(Date now) {

static const char \*nama\_bulan[] = {

"Wrong month", "January", "February", "March",

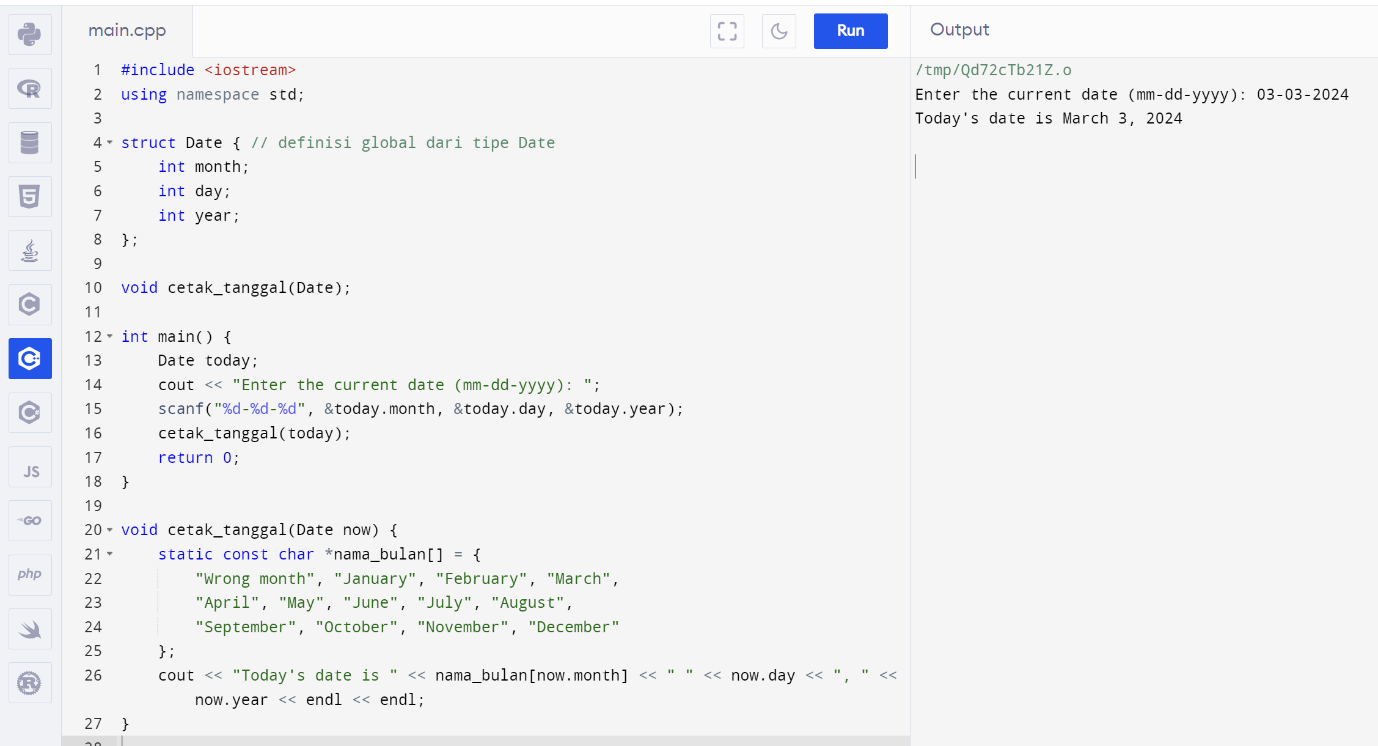
"April", "May", "June", "July", "August",

"September", "October", "November", "December"

};

cout << "Today's date is " << nama\_bulan[now.month] << " " << now.day << ", " << now.year << endl << endl;

}

Analisis perubahan:

1. Penggunaan **<iostream>**: Program C++ menggunakan header **<iostream>** untuk operasi input-output, menggantikan **<stdio.h>** yang digunakan dalam C.
2. Perubahan pada definisi **struct**: Struct **date** diganti menjadi **Date** untuk mengikuti konvensi penamaan C++ yang umum, serta didefinisikan di dalam **namespace** global. Ini adalah standar dalam pemrograman C++ untuk menghindari konflik nama.
3. Penggunaan **cout** dan **cin**: Dalam program C++, **cout** digunakan untuk mencetak pesan ke layar, dan **cin** digunakan untuk menerima input dari pengguna. Ini menggantikan **printf** dan **scanf** yang digunakan dalam C.
4. Penyesuaian fungsi **main()**: Dalam C++, fungsi **main()** harus mengembalikan nilai integer, biasanya 0 jika program berjalan dengan sukses. Oleh karena itu, **int main()** digunakan sebagai tanda bahwa fungsi **main()** mengembalikan nilai integer. Dalam C, **main()** tidak perlu mengembalikan nilai.
5. Penggunaan **namespace std**: Dalam C++, **using namespace std;** digunakan untuk menghindari penulisan **std::** sebelum setiap penggunaan objek dari namespace **std**.

PROGRAM 7  
#include <iostream>

using namespace std;

struct Koordinat {

int x;

int y;

};

void tukar\_xy(Koordinat \*);

int main() {

Koordinat posisi;

cout << "Masukkan koordinat posisi (x, y) : ";

cin >> posisi.x >> posisi.y;

cout << "x, y semula = " << posisi.x << ", " << posisi.y << endl;

tukar\_xy(&posisi);

cout << "x, y sekarang = " << posisi.x << ", " << posisi.y << endl;

return 0;

}

void tukar\_xy(Koordinat \*pos\_xy) {

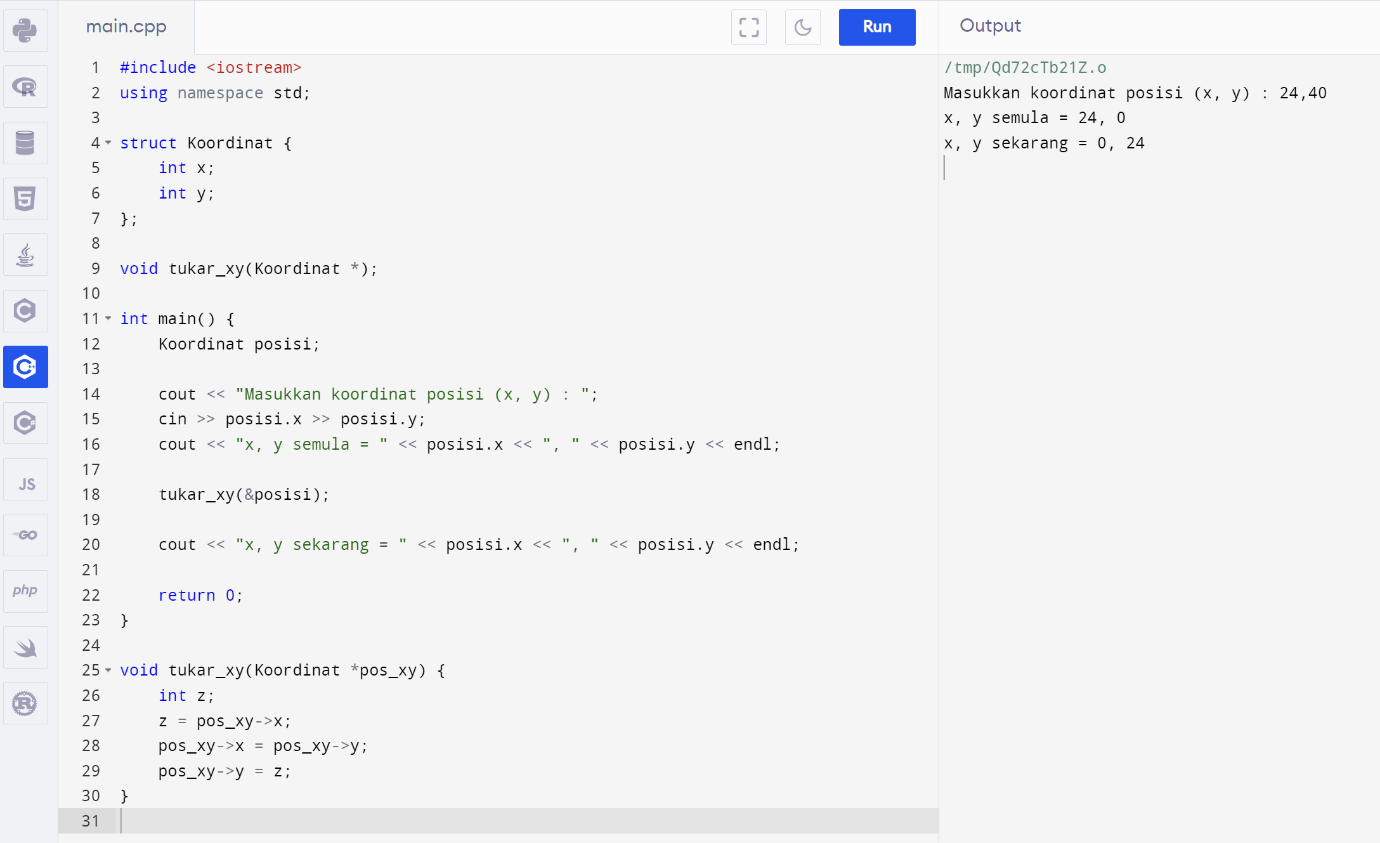
int z;

z = pos\_xy->x;

pos\_xy->x = pos\_xy->y;

pos\_xy->y = z;

}

Analisis perubahan:

1. Penggunaan **<iostream>**: Program C++ menggunakan header **<iostream>** untuk operasi input-output, menggantikan **<stdio.h>** yang digunakan dalam C.
2. Perubahan pada definisi **struct**: Struktur **koordinat** didefinisikan di dalam fungsi **main()** dalam C, sementara di C++ program tersebut mendeklarasikan struct **Koordinat** secara terpisah sebelum **main()**. Ini adalah penyesuaian standar karena C++ lebih ketat dalam pemisahan antara ruang lingkup variabel.
3. Penggunaan **cout** dan **cin**: Dalam program C++, **cout** digunakan untuk mencetak pesan ke layar, dan **cin** digunakan untuk menerima input dari pengguna. Ini menggantikan **printf** dan **scanf** yang digunakan dalam C.
4. Penyesuaian sintaksis: Beberapa penyesuaian sintaksis dilakukan seperti menggunakan **<<** dan **>>** untuk input-output.
5. Penggunaan **->** operator: Dalam C++, **->** digunakan untuk mengakses member struct dari pointer ke struct, menggantikan **.** yang digunakan dalam C.

LATIHAN :

1. Definisikan sebuah struktur (misalkan namanya = record) yang memiliki 3 buah field berupa sebuah integer (misalkan namanya = loop), sebuah array karakter dengan 5 elemen (misalkan namanya = word) dan sebuah float (misalkan Namanya = sum).

**JAWAB:**  
*#include <iostream>*

*#include <cstring>*

*using namespace std;*

*struct Record {*

*int loop;*

*char word[6];*

*float sum;*

*};*

*int main() {*

*Record data;*

*data.loop = 10;*

*strncpy(data.word, "Hello", sizeof(data.word));*

*data.word[sizeof(data.word) - 1] = '\0';*

*data.sum = 3.14;*

*cout << "Loop: " << data.loop << endl;*

*cout << "Word: " << data.word << endl;*

*cout << "Sum: " << data.sum << endl;*

*return 0;*

*}*  
Dalam struktur **Record** di atas:

* **loop** adalah sebuah integer.
* **word** adalah sebuah array karakter dengan panjang 5.
* **sum** adalah sebuah float.

1. Deklarasikan sebuah variabel struktur (misalkan namanya = sample) yang didefinisikan memiliki tipe struktur record.

**JAWAB:***record sample;*Dengan deklarasi ini, variabel sample telah dideklarasikan sebagai sebuah variabel yang memiliki tipe struktur record, sehingga Anda dapat menggunakan variabel ini untuk menyimpan data sesuai dengan struktur yang telah Anda definisikan sebelumnya.