FIKOM

MODUL PRAKTIKUM

ALGORITMA & PEMROGRAMAN 2

20
23
20
24

Tim Penyusun

- Lutfi Budi Ilmawan, S.Kom., M.Cs., MTA
- Ramdaniah, S.Kom., M.T., MTA
- Ir. Huzain Azis, S.Kom, M.Cs., MTA
- Siska Anraeni, S.Kom., M.T., MCF
- Ir. Abdul Rachman Manga, S.Kom., M.T., MTA., MCF
- Tim Asisten Laboratorium

Laboratorium Terpadu
Fakultas Ilmu Komputer
Universitas Muslim Indonesia

KATA PENGANTAR

Puji syukur ke hadirat Tuhan Yang Maha Kuasa, yang telah memberikan rahmat-Nya sehingga Modul Praktikum **Algoritma dan Pemrograman II** untuk mahasiswa/i Program Studi Teknik Informatika Fakultas Ilmu Komputer Universitas Muslim Indonesia ini dapat diselesaikan dengan sebaik-baiknya.

Modul praktikum ini dibuat sebagai pedoman dalam melakukan kegiatan praktikum **Algoritma dan Pemrograman II** yang merupakan kegiatan penunjang mata kuliah pada Program Studi Teknik Informatika. Modul praktikum ini diharapkan dapat membantu mahasiswa/i dalam mempersiapkan dan melaksanakan praktikum dengan lebih baik, terarah, dan terencana. Pada setiap topik telah ditetapkan capaian pembelajaran mata kuliah pelaksanaan praktikum dan semua kegiatan yang harus dilakukan oleh mahasiswa/i serta teori singkat untuk memperdalam pemahaman mahasiswa/i mengenai materi yang dibahas.

Penyusun menyakini bahwa dalam pembuatan Modul Praktikum **Algoritma dan Pemrograman II** ini masih jauh dari sempurna. Oleh karena itu penyusun mengharapkan kritik dan saran yang membangun guna penyempurnaan modul praktikum ini dimasa yang akan datang.

Akhir kata, penyusun mengucapkan banyak terima kasih kepada semua pihak yang telah membantu baik secara langsung maupun tidak langsung.

Makassar, Maret 2024

Tim Penyusun

TATA TERTIB PELAKSANAAN PRAKTIKUM

Tata Tertib Pelaksanaan Praktikum pada Laboratorium Terpadu Fakultas Ilmu Komputer UMI adalah sebagai berikut:

1. Seluruh Pengguna laboratorium harus dalam keadaan sehat tidak menunjukkan gejala sakit (batuk, hidung tersumbat, dan suhu badan diatas 37°C).
2. Praktikan hanya diizinkan melaksanakan praktikum apabila :
   a. Pria
      - Berpakaian rapi memakai kemeja putih polos;
      - Menggunakan celana kain berwarna hitam bukan dari bahan jeans/semi jeans;
      - Rambut rapi dan tidak panjang;

   b. Wanita
      - Berpakaian rapi memakai kemeja tunik putih polos (tidak transparan)
      - Memakai Jilbab Segitiga Hitam (bukan pasmina) dan menutupi dada.
      - Menggunakan Rok Panjang berwarna hitam yang tidak terbelah dan tidak span serta bukan dari bahan jeans/semi jeans;
      - Memakai kaos kaki dengan tinggi minimal 10 cm di atas mata kaki;
3. Ketika memasuki dan selama berada dalam ruangan, praktikan diwajibkan :
   - Tenang, tertib, dan sopan;
   - Tidak mengganggu praktikan lain yang sedang melaksanakan praktikum;
   - Tidak diperbolehkan merokok, membawa makanan / minuman senjata tajam dan senjata api ke dalam ruangan praktikum;
   - Tidak diperbolehkan membawa *handphone* ke meja praktikum dan *handphone* dalam mode senyap;
   - Tidak diperbolehkan membawa media penyimpanan eksternal atau *flashdisk* ke meja praktikum tanpa seizin Dosen Pengampu atau Asisten;
4. Dilarang membawa, mengambil, serta memindahkan perangkat yang digunakan pada saat praktikum tanpa instruksi dari Dosen Pengampu atau Asisten.
5. Toleransi keterlambatan praktikan maksimal 5 menit.
6. Praktikan berada diarea laboratorium dengan mengikuti jadwal yang telah ditentukan oleh Kepala Laboratorium.
7. Penggunaan fasilitas Laboratorium menyesuaikan dengan kapasitas ruang Laboratorium.

8. Segala pelanggaran yang dilakukan oleh praktikan akan berakibat pada penutupan dan penghentian penggunaan seluruh fasilitas laboratorium dan ditindak sesuai dengan aturan yang berlaku.

SANKSI-SANKSI

Sanksi terhadap pelanggaran **TATA TERTIB**:

Dosen Pengampu dan Asisten laboratorium berhak menjatuhkan sanksi, sesuai dengan aturan yang berlaku di Laboratorium Terpadu Fakultas Ilmu Komputer UMI apabila :

1. Praktikan merusak peralatan praktikum (*Personal Computer*) secara sengaja, maka praktikan bertanggung jawab untuk mengganti kerusakan tersebut.
2. Praktikan tidak mematuhi dan mentaati aturan praktikum maka tidak diperkenankan mengikuti praktikum.

Pelanggaran point lainnya dikenakan sanksi teguran, dikeluarkan/dicoret namanya dalam kegiatan praktikum (mengulang mata kuliah sesuai dengan semester berjalan) sampai sanksi akademik.

Kepala Laboratorium Terpadu,

Ir. Abdul Rachman Manga', S.Kom., M.T., MTA., MCF

DAFTAR ISI

CAPAIAN PEMBELAJARAN MATA KULIAH (CPMK)

1. CPMK-1 : Mahasiswa dapat menerapkan penggunaan preprocessor pada C++
2. CPMK-2 : Mahasiswa dapat menerapkan konsep OOP pada C++
3. CPMK-3 : Mahasiswa dapat menerapkan Exception pada C++
4. CPMK-4 : Mahasiswa dapat menerapkan dynamic memory pada C++
5. CPMK-5 : Mahasiswa dapat menerapkan penggunaan class vector pada C++
6. CPMK-6 : Mahasiswa dapat menerapkan algoritma rekursif
7. CPMK-7 : Mahasiswa dapat menerapkan algoritma pencarian

MODUL 1 – PREPROCESSOR DIRECTIVES

A. Sub Capaian Pembelajaran Mata Kuliah (CPMK)

Mahasiswa dapat menerapkan penggunaan preprocessor directives pada C++

B. Instrumen dan Prosedur

1. Instrument

a) Perangkat komputer / PC / Laptop / Notebook.
b) Sistem operasi Windows / Linux (optional Mac OS)
c) C++ Compiler (MinGw)
d) Geany / Notepad++ / Dev C++

2. Prosedur

a) Baca dan pahami semua tahapan praktikum dengan cermat.
b) Gunakan fasilitas yang disediakan dengan penuh rasa tanggung jawab.
c) Rapikan kembali setelah menggunakan komputer (mouse, keyboard, kursi, dll)
d) Perhatikan sikap anda untuk tidak mengganggu rekan praktikan lain
e) Pastikan diri anda tidak menyentuh sumber listrik.

C. Teori Dasar

1. Pengenalan Preprocessor Direvtives

Preprocessor directives adalah instruksi khusus pada bahasa pemrograman C++ yang digunakan oleh preprocessor untuk memanipulasi kode sumber sebelum dikompilasi oleh compiler. Preprocessor directives biasanya dimulai dengan karakter "#" (pagar) dan diikuti dengan sebuah kata kunci yang menunjukkan aksi yang harus dilakukan oleh preprocessor.

Preprocessor directives sangat berguna dalam mempermudah pengembangan dan pemeliharaan kode sumber pada program. Dengan menggunakan preprocessor directives, programmer dapat membuat kode sumber yang lebih fleksibel dan mudah untuk diatur sesuai kebutuhan program yang sedang dikembangkan.

2. Penggunaan Preprocessor Directives

Preprocessor directives umumnya digunakan untuk mendefinisikan konstanta, memasukkan file header atau library ke dalam program, menentukan opsi-opsi kompilasi, dan menampilkan pesan error pada saat kompilasi program.

Beberapa contoh preprocessor directives yang sering digunakan dalam bahasa pemrograman seperti C atau C++ antara lain adalah #include, #define, #ifdef, #ifndef, dan #pragma.

a) #define

```
#define PI 3.14159
```

Preprocessor directive #define digunakan untuk mendefinisikan konstanta yang akan digunakan pada program. Konstanta ini dapat berupa angka, string, atau ekspresi matematika sederhana. Konstanta yang telah didefinisikan dengan #define dapat digunakan di seluruh bagian program.

b) #include

```
#include <iostream>
```

Preprocessor directive #include digunakan untuk memasukkan file header atau library ke dalam program. File header biasanya berisi definisi fungsi dan kelas yang akan digunakan pada program, sementara library berisi implementasi fungsi-fungsi yang dapat digunakan pada program.

c) #ifdef, #ifndef, #else, #endif:

```
#ifndef MY_HEADER_H
#define MY_HEADER_H

// Isi dari file header

#endif // MY_HEADER_H
```

Preprocessor directives #ifdef, #ifndef, #else, dan #endif digunakan untuk menentukan kondisi-kondisi tertentu pada program. #ifdef dan #ifndef digunakan untuk menentukan apakah sebuah variabel atau konstanta sudah didefinisikan atau belum. #else digunakan untuk memberikan alternatif dari kondisi yang telah ditentukan sebelumnya, sedangkan #endif digunakan untuk menutup kondisi.

d) #error

```
#if RADIUS <= 0
    #error "Nilai RADIUS harus lebih besar dari 0"
#endif
```

Preprocessor directive #error digunakan untuk menampilkan pesan error pada saat kompilasi program. #error dapat digunakan untuk mengecek nilai dari sebuah variabel atau konstanta pada saat kompilasi program, dan menampilkan pesan error jika nilainya tidak sesuai dengan yang diharapkan.

e) #pragma

```
#pragma once
```

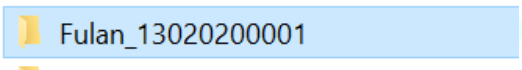
Preprocessor directive #pragma digunakan untuk memberikan instruksi khusus kepada kompiler, seperti optimasi kode atau pengaturan alignment. #pragma biasanya digunakan pada file header atau library yang harus dikompilasi dengan pengaturan khusus.

Dengan menggunakan preprocessor directives, kita bisa melakukan beberapa hal seperti mengimpor file header, mengubah nilai konstanta, melakukan kondisional compilation, dan mengkonfigurasi compiler. Dalam hal ini, preprocessor directives sangat berguna untuk meningkatkan efisiensi dan kehandalan program yang dibuat.

D. Kegiatan Praktikum

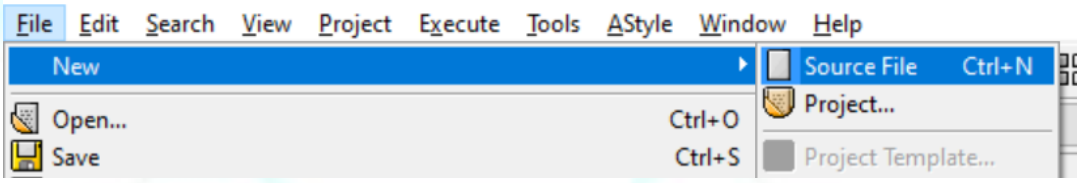
1. Penggunaan preprocessor directives #include #define

a) Buatlah folder dengan nama anda_stambuk (Fulan_13020200001) pada *Windows Explorer* :



b) Bukalah C++ compiler yang ada dikomputer depan anda (Geany/DevC++/Notepad C++).

c) Klik *File* pada header, kemudian *New* lalu *Source File*. Atau anda menekan shortcode pada keyboard dengan menekan *ctrl+N* :



d) Buatlah sebuah file C++ dengan eksistensi *.cpp* dengan format :
M1.1_001.cpp
Dimana :
- **M1 :** Modul yang sedang dikerjakan
- **1 :** nomor kegiatan praktikum yang dikerjakan
- **001 :** 3 stambuk terakhir

e) Ketikkan program dibawah ini :

f) Bukalah C++ compiler yang ada dikomputer depan anda (Geany/DevC++/Notepad C++).

```
#include <iostream>
#define PI 3.14

using namespace std;

int main() {
   double radius, area;

   cout << "Masukkan jari-jari lingkaran: ";
   cin >> radius;

   area = PI * radius * radius;

   cout << "Luas lingkaran adalah: " << area << endl;

   return 0;
}
```

Program tersebut menghitung luas lingkaran dengan menggunakan nilai phi (π) yang telah didefinisikan sebelumnya. Kita dapat menggunakan #define untuk mendefinisikan nilai phi sebagai konstanta sebelum kode program dijalankan.

Dalam contoh di atas, kita menggunakan #define untuk mendefinisikan konstanta PI sebagai 3.14. Dengan cara ini, nilai PI akan diketahui oleh seluruh bagian program yang membutuhkan nilai tersebut, sehingga program menjadi lebih efisien dan mudah dimengerti.

g) Kemuidan *save* program yang telah diketik pada folder yang telah anda buat sebelumnya pada modul 1, dengan menekan *ctrl+s*

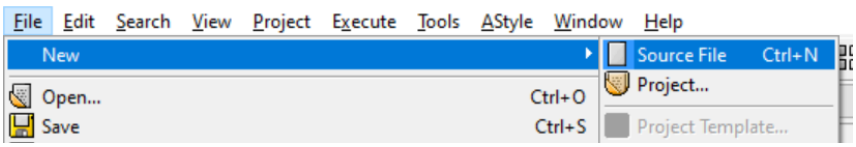
h) Kemudian, apabila telah selesai menuliskan kode diatas maka anda dapat menekan tombol *compile* atau shortcode pada keyboard : *F9*

i) Jika program anda berhasil tanpa error, maka tampilan akan seperti berikut :

```
Masukkan jari-jari lingkaran: 5
Luas lingkaran adalah: 78.5
```

2. Penggunaan preprocessor directives #ifdef, #ifndef, #else, #endif

a) Klik *File* pada header, kemudian *New* lalu *Source File*. Atau anda menekan shortcode pada keyboard dengan menekan *ctrl+N* :



b) Buatlah sebuah file C++ dengan eksistensi *.cpp* dengan format :

M1.2_001.cpp

Dimana :

- **M1 :** Modul yang sedang dikerjakan
- **2 :** nomor kegiatan praktikum yang sedang dikerjakan
- **001 :** 3 stambuk terakhir

c) Ketikkan program dibawah ini :

```
#include <iostream>

#define DEBUG_MODE 1
#define FILE_MODE 1

int main() {
    #ifdef DEBUG_MODE
        std::cout << "Program dijalankan dalam mode debug." << std::endl;
    #else
        std::cout << "Program dijalankan dalam mode release." << std::endl;
    #endif

    #ifndef FILE_MODE
        std::cout << "File output tidak diaktifkan." << std::endl;
    #else
        std::cout << "File output diaktifkan." << std::endl;
    #endif

    return 0;
}
```

d) Kemuidan *save* program yang telah diketik pada folder yang telah dibuat sebelumnya pada modul 1, dengan menekan *ctrl+s*

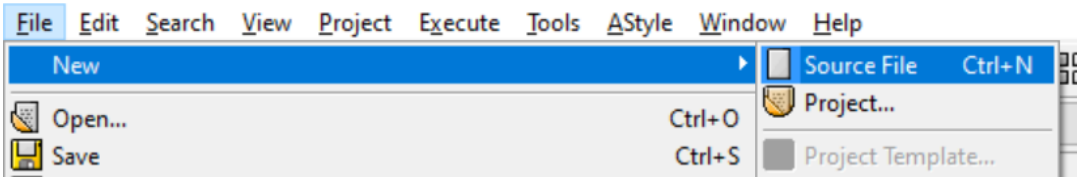
e) Kemudian, apabila telah selesai menuliskan kode diatas maka anda dapat menekan tombol *compile* atau shortcode pada keyboard : *F9*

f) Jika program anda berhasil tanpa error, maka tampilan akan seperti berikut :

```
Program dijalankan dalam mode debug.
File output diaktifkan.

--------------------------------
Process exited after 0.07754 seconds with return value 0
Press any key to continue . . .
```

3. Penggunaan preprocessor directives #error

a) Klik *File* pada header, kemudian *New* lalu *Source File.* Atau anda menekan shortcode pada keyboard dengan menekan *ctrl+N* :



b) Buatlah sebuah file C++ dengan eksistensi *.cpp* dengan format :

M1.3_001.cpp

Dimana :

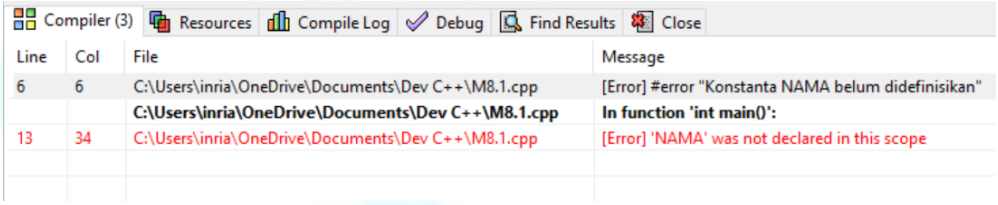
- **M1 :** Modul yang sedang dikerjakan
- **3 :** nomor kegiatan praktikum yang sedang dikerjakan
- **001 :** 3 stambuk terakhir

c) Ikuti instruksi dibawah ini :

I. Ketikkan program berikut

```
#include <iostream>

#define VERSI "1.0"

#ifndef NAMA
    #error "Konstanta NAMA belum didefinisikan"
#endif

using namespace std;

int main() {
    cout << "Program versi " << VERSI << endl;
    cout << "Nama pengguna: " << NAMA << endl;
    return 0;
}
```

II. Apabila program tersebut anda jalankan, maka outputntya akan seperti berikut



III. Compiler memberitahukan bahwa konstanta ‘NAMA’ belum didefinisikan, hal ini berarti preprocessor directives #error berfungsi dan memberitahu pesan error setelahnya.

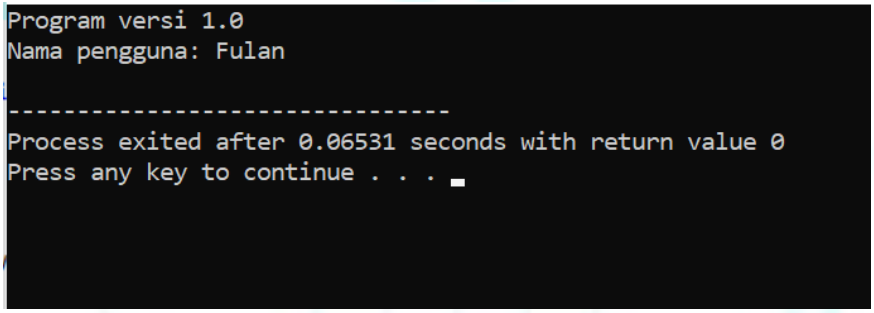
IV. Tambahkan konstanta NAMA dengan mengunakan preprocessor directives #define

```
#define NAMA "Fulan"
```

d) Kemuidan *save* program yang telah diketik pada folder yang telah dibuat sebelumnya pada modul 1, dengan menekan *ctrl+s*

e) Kemudian, apabila telah selesai menuliskan kode diatas maka anda dapat menekan tombol *compile* atau shortcode pada keyboard : *F9*

f) Jika program anda berhasil tanpa error, maka tampilan akan seperti berikut :



LEMBAR EVALUASI PRAKTIKUM

1. Apa itu Preprocessor directives? Apa yang membedakan Preprocessor directives dengan kode C++ biasa?
2. Bagaimana cara menggunakan Preprocessor directives #pragma dalam C++?
3. Apa yang dimaksud algoritma rekursif?

Evaluasi Praktikum 1:

No	Indikator	Skor Penilaian				
		Sangat Kurang (E) <=40	Kurang (D) 41-55	Cukup (C) 55-65	Baik (B) 66-85	Sangat Baik (A) >=86
1.	Pemahaman mengenai preprocessor directive #define					
2.	Pemahamah mengenai preprocessor directive #ifdef, #ifndef, #else, #endif					
3.	Pemahamah mengenai preprocessor directive #error					
4.	Pemahamah mengenai preprocessor directive #pragma					

Catatan Asisten:

Asisten 1 : _____

Asisten 2 : _____

MODUL 2 – ALGORITMA REKURSIF

A. Sub Capaian Pembelajaran Mata Kuliah (CPMK)

Mahasiswa dapat menerapkan algoritma rekursif pada C++

B. Instrumen dan Prosedur

1. Instrument

a) Perangkat komputer / PC / Laptop / Notebook.
b) Sistem operasi Windows / Linux (optional Mac OS)
c) MinGW/C++ Compiler
d) Geany/Notepad++/Dev C++

2. Prosedur

a) Baca dan pahami semua tahapan praktikum dengan cermat.
b) Gunakan fasilitas yang disediakan dengan penuh rasa tanggung jawab.
c) Rapikan kembali setelah menggunakan komputer (mouse, keyboard, kursi, dll)
d) Perhatikan sikap anda untuk tidak mengganggu rekan praktikan lain
e) Pastikan diri anda tidak menyentuh sumber listrik.

C. Teori Dasar

1. Algoritma rekursif

Algoritma rekursif adalah teknik pemrograman dimana sebuah fungsi memanggil dirinya sendiri secara berulang-ulang dengan parameter yang berbeda-beda. Algoritma rekursif dapat digunakan untuk menyelesaikan banyak masalah dalam pemrograman. Namun, karena sifatnya yang berulang-ulang, algoritma rekursif dapat memakan waktu dan memori yang cukup besar jika tidak diimplementasikan dengan benar. Oleh karena itu, perlu untuk mempertimbangkan dengan baik penggunaan algoritma rekursif dalam sebuah program.

2. Deklarasi algoritma rekursif

Deklarasi algoritma rekursif dalam bahasa C++ sama dengan deklarasi fungsi biasa, namun dengan tambahan pemanggilan dirinya sendiri di dalam tubuh fungsi tersebut. Berikut adalah sintaks untuk mendeklarasikan fungsi rekursif dalam bahasa C++:

```
tipe_data nama_fungsi(argumen) {
    // Base Case
    if (kondisi_base_case) {
        return nilai_base_case;
    }

    // Recursive Case
    return operasi_rekursif(nama_fungsi(argumen_rekursif));
}
```

Penjelasan :

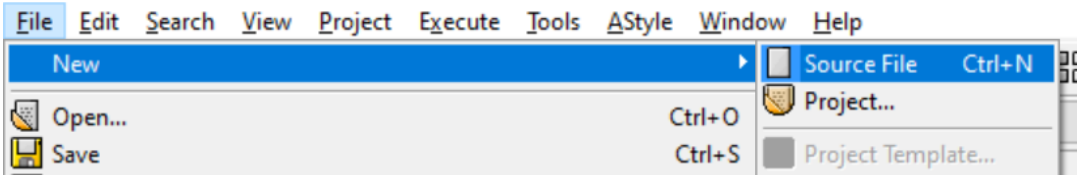
- **tipe_data** adalah jenis data yang dihasilkan oleh fungsi tersebut, seperti int, double, float, void, dll.
- **nama_fungsi** adalah nama dari fungsi tersebut, harus unik dalam satu program.
- **argumen** adalah parameter yang diterima oleh fungsi.
- **kondisi_base_case** adalah kondisi yang menentukan apakah fungsi harus mengembalikan nilai base case atau melakukan operasi rekursif.
- **nilai_base_case** adalah nilai yang akan dikembalikan ketika kondisi base case terpenuhi.
- **operasi_rekursif** adalah operasi yang dilakukan pada setiap iterasi rekursif.
- argumen_rekursif adalah argumen yang dikirim ke fungsi itu sendiri dalam setiap iterasi rekursif.

D. Kegiatan Praktikum

1. Program untuk menghitung nilai factorial dengan algortima rekursif

a) Bukalah C++ compiler yang ada dikomputer depan anda (Geany/DevC++/Notepad C++).

b) Klik *File* pada header, kemudian *New* lalu *Source File.* Atau anda menekan shortcode pada keyboard dengan menekan *ctrl+N* :



c) Buatlah sebuah file C++ pada folder yang telah anda buat pada Langkah *b* dengan eksistensi *.cpp* dengan format :
M2.1_001.cpp
Dimana :

- **M2 :** Modul yang sedang dikerjakan

- **1 :** nomor kegiatan praktikum yang dikerjakan
- **001 :** 3 stambuk terakhir

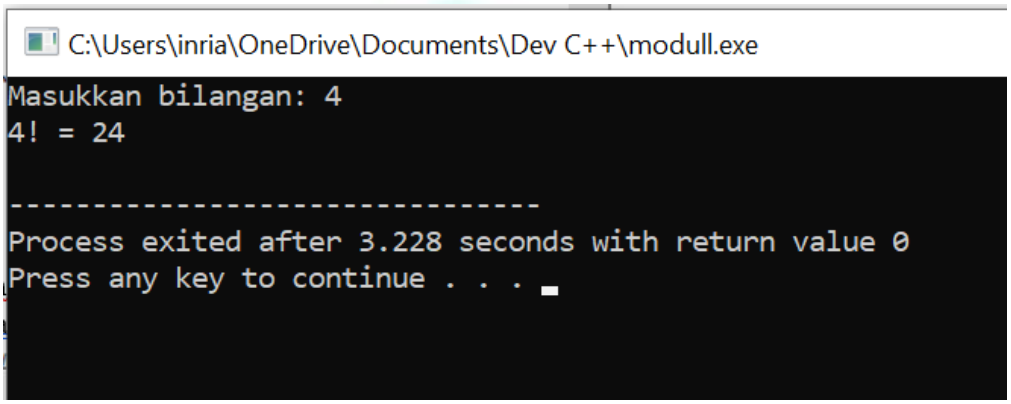
d) Ketikkan program dibawah ini :

```
#include <iostream>
using namespace std;

int faktorial(int n) {
    // Base Case
    if (n <= 1) {
        return 1;
    }

    // Recursive Case
    return n * faktorial(n-1);
}

int main() {
    int n;
    cout << "Masukkan bilangan: ";
    cin >> n;

    cout << n << "! = " << faktorial(n) << endl;

    return 0;
}
```

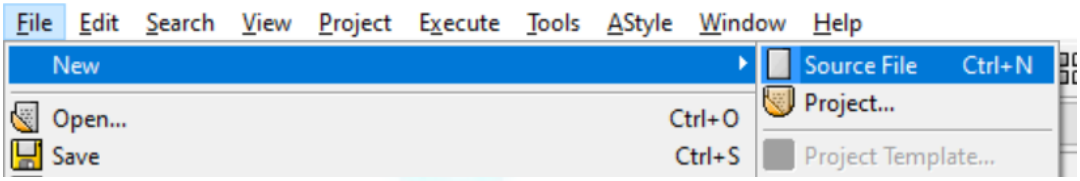
e) Kemuidan *save* program yang telah diketik pada folder yang telah dibuat pada modul 1, dengan menekan *ctrl+s*

f) Kemudian, apabila telah selesai menuliskan kode diatas maka anda dapat menekan tombol *compile* atau shortcode pada keyboard : *F9*

g) Jika program anda berhasil tanpa error, maka tampilan akan seperti berikut :



2. **Program untuk mencari nilai bilangan permutasi P(n,r) dengan algoritma rekursif:**
   a) Klik *File* pada header, kemudian *New* lalu *Source File.* Atau anda menekan shortcode pada keyboard dengan menekan *ctrl+N* :



   b) Buatlah sebuah file C++ pada folder yang telah anda buat pada Langkah *b* dengan eksistensi *.cpp* dengan format :

   M2.2_001.cpp

   Dimana :
   - **M2 :** Modul yang sedang dikerjakan
   - **2 :** nomor kegiatan praktikum yang sedang dikerjakan
   - **001 :** 3 stambuk terakhir

   c) Ketikkan program dibawah ini :

```
#include <iostream>
using namespace std;

int permutasi(int n, int r) {
    // Base Case
    if (r == 0) {
        return 1;
    } else {
        return n * permutasi(n-1, r-1);
    }
}
```

```
int main() {
    int n, r;
    cout << "Masukkan nilai n: ";
    cin >> n;
    cout << "Masukkan nilai r: ";
    cin >> r;

    cout << "P(" << n << "," << r << ") = " << permutasi(n, r) << endl;

    return 0;
}
```

   d) Kemuidan *save* program yang telah diketik pada folder yang telah dibuat pada modul 1, dengan menekan *ctrl+s*

   e) Kemudian, apabila telah selesai menuliskan kode diatas maka anda dapat menekan tombol *compile* atau shortcode pada keyboard : *F9*

   f) Jika program anda berhasil tanpa error, maka tampilan akan seperti berikut :

```
C:\Users\inria\OneDrive\Documents\Dev C++\M1.1_001.exe
Masukkan nilai n: 9
Masukkan nilai r: 2
P(9,2) = 72

--------------------------------
Process exited after 3.065 seconds with return value 0
Press any key to continue . . .
```

LEMBAR EVALUASI PRAKTIKUM

1. Buatlah program C++ untuk mencetak deret bilangan Fibonacci hingga suatu nilai tertentu yang dimasukkan oleh pengguna menggunakan algoritma rekursif.

```
Masukkan nilai batas: 13
1 1 2 3 5 8 13
```

2. Buatlah program C++ untuk mencari nilai bilangan kombinasi C(n,r) dengan menggunakan algoritma rekursif. Nilai n dan r dimasukkan oleh pengguna.

```
Masukkan nilai n: 6
Masukkan nilai r: 3
C(6,3) = 20
```

3. Jelaskan menurut pendapat anda apa itu algoritma pengurutan!
4. Tuliskan bagaimana proses algoritma selection sort!
5. Tuliskan bagaimana proses algoritma bubble sort!

Evaluasi Praktikum 2:

No	Indikator	Skor Penilaian				
		Sangat Kurang (E) <=40	Kurang (D) 41-55	Cukup (C) 55-65	Baik (B) 66-85	Sangat Baik (A) >=86
1.	Dapat memhamai algoritma rekursif					
2.	Pemahaman bagaimana menghitung nilai factorial dengan menggunakan algoritma rekursif					
3.	Pemahaman bagaimana menghitung nilai permutasi dengan menggunakan algoritma rekursif					

Catatan Asisten:

Asisten 1 : _____

Asisten 2 : _____

MODUL 3 – ALGORITMA PENGURUTAN (Selection Sort & Bubble Sort)

A. Sub Capaian Pembelajaran Mata Kuliah (CPMK)

Mahasiswa dapat menerapkan algoritma pengurutan (selection sort dan bubble sort) pada C++

B. Instrument dan Prosedur

1. Instrument

a) Perangkat komputer / PC / Laptop / Notebook.
b) Sistem operasi Windows / Linux (optional Mac OS)
c) C++ Compiler (MinGw)
d) Geany / Notepad++ / Dev C++

2. Prosedur

a) Baca dan pahami semua tahapan praktikum dengan cermat.
b) Gunakan fasilitas yang disediakan dengan penuh rasa tanggung jawab.
c) Rapikan kembali setelah menggunakan komputer (mouse, keyboard, kursi, dll)
d) Perhatikan sikap anda untuk tidak mengganggu rekan praktikan lain
e) Pastikan diri anda tidak menyentuh sumber listrik.

C. Teori Dasar

Algoritma pengurutan merupakan algoritma yang digunakan untuk mengurutkan sebuah kumpulan data atau elemen, seperti array atau list, dari yang terkecil hingga yang terbesar atau sebaliknya. Algoritma pengurutan sangat penting dalam pemrograman karena seringkali kita perlu mengurutkan data untuk memudahkan analisis atau pengolahan data.

Dalam praktikum ini, kita akan membahas dua algoritma pengurutan sederhana yang sering digunakan, yaitu selection sort dan bubble sort.

1. Selection Sort

Algoritma selection sort bekerja dengan memilih elemen terkecil dari kumpulan data dan menukarnya dengan elemen pertama. Kemudian, elemen terkecil kedua dipilih dari sisa kumpulan data dan ditukar dengan elemen kedua, dan seterusnya hingga seluruh kumpulan data terurut.

Berikut adalah langkah-langkah dalam algoritma selection sort:

a) Cari elemen terkecil dalam kumpulan data.
b) Tukar elemen terkecil dengan elemen pertama dalam kumpulan data.
c) Cari elemen terkecil kedua dalam sisa kumpulan data.
d) Tukar elemen terkecil kedua dengan elemen kedua dalam kumpulan data, dan seterusnya hingga seluruh kumpulan data terurut.

Berikut adalah implementasi selection sort dalam bahasa C++:

```
void selectionSort(int arr[], int n) {
    for (int i = 0; i < n-1; i++) {
        int min_idx = i;
        for (int j = i+1; j < n; j++) {
            if (arr[j] < arr[min_idx]) {
                min_idx = j;
            }
        }
        int temp = arr[min_idx];
        arr[min_idx] = arr[i];
        arr[i] = temp;
    }
}
```

2. Bubble Sort

Algoritma bubble sort bekerja dengan membandingkan setiap pasang elemen yang bersebelahan dalam kumpulan data dan menukarkan elemen tersebut jika urutannya salah. Proses ini diulangi hingga seluruh kumpulan data terurut.
Berikut adalah langkah-langkah dalam algoritma bubble sort:

a) Bandingkan elemen pertama dan kedua dalam kumpulan data. Jika urutannya salah, tukar posisi keduanya.
b) Bandingkan elemen kedua dan ketiga dalam kumpulan data. Jika urutannya salah, tukar posisi keduanya.
c) Seterusnya hingga seluruh pasangan elemen dalam kumpulan data dibandingkan dan posisinya diperbaiki.

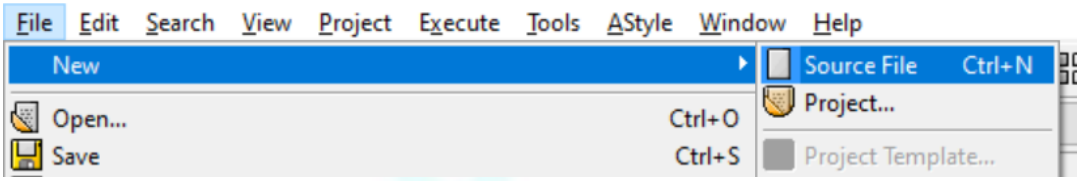
Berikut adalah implementasi bubble sort dalam bahasa C++:

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

D. Kegiatan Praktikum

1. Studi kasus 1

a) Klik *File* pada header, kemudian *New* lalu *Source File*. Atau anda menekan shortcode pada keyboard dengan menekan *ctrl+N* :



b) Buatlah sebuah file C++ pada folder yang telah anda buat pada Langkah *b* dengan eksistensi *.cpp* dengan format :

M3.1_001.cpp

Dimana :

- **M3 :** Modul yang sedang dikerjakan
- **1 :** nomor kegiatan praktikum yang sedang dikerjakan
- **001 :** 3 stambuk terakhir

c) Ketikkan program dibawah ini :

```cpp
#include <iostream>
using namespace std;

void selectionSort(int arr[], int n) {
    for (int i = 0; i < n-1; i++) {
        int min_index = i;
        for (int j = i+1; j < n; j++) {
            if (arr[j] < arr[min_index]) {
                min_index = j;
            }
        }
        int temp = arr[i];
        arr[i] = arr[min_index];
        arr[min_index] = temp;
    }
}

int main() {
    int arr[] = {64, 25, 12, 22, 11};
    int n = sizeof(arr)/sizeof(arr[0]);
    selectionSort(arr, n);
    cout << "Sorted array: ";
    for (int i = 0; i < n; i++) {
        cout << arr[i] << " ";
    }
    cout << endl;
    return 0;
}
```

d) Kemuidan *save* program yang telah diketik pada folder yang telah dibuat pada modul 1, dengan menekan *ctrl+s*

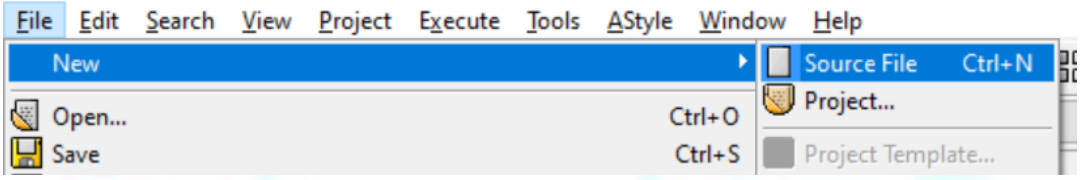
e) Kemudian, apabila telah selesai menuliskan kode diatas maka anda dapat menekan tombol *compile* atau shortcode pada keyboard : *F9*

f) Jika program anda berhasil tanpa error, maka tampilan akan seperti berikut :

```
/tmp/eKZOIkxVz1.o
Sorted array: 11 12 22 25 64
```

2. Studi kasus 2

a) Klik *File* pada header, kemudian *New* lalu *Source File*. Atau anda menekan shortcode pada keyboard dengan menekan *ctrl+N* :



b) Buatlah sebuah file C++ pada folder yang telah anda buat pada Langkah *b* dengan eksistensi *.cpp* dengan format :

M3.2_001.cpp

Dimana :

- **M3 :** Modul yang sedang dikerjakan
- **2 :** nomor kegiatan praktikum yang sedang dikerjakan
- **001 :** 3 stambuk terakhir

c) Ketikkan program dibawah ini :

```cpp
#include <iostream>
using namespace std;

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

Kemudian pada function main ketikan program di bawah ini :

```
int main() {
    int n;
    cout << "Enter the size of the array: ";
    cin >> n;

    int arr[n];
    cout << "Enter " << n << " integers: ";
    for (int i = 0; i < n; i++) {
        cin >> arr[i];
    }

    cout << "Original array: ";
    for (int i = 0; i < n; i++) {
        cout << arr[i] << " ";
    }
    cout << endl;

    bubbleSort(arr, n);

    cout << "Sorted array: ";
    for (int i = 0; i < n; i++) {
        cout << arr[i] << " ";
    }
    cout << endl;

    return 0;
}
```

d) Kemuidan *save* program yang telah diketik pada folder yang telah dibuat pada modul 1, dengan menekan *ctrl+s*

e) Kemudian, apabila telah selesai menuliskan kode diatas maka anda dapat menekan tombol *compile* atau shortcode pada keyboard : *F9*

f) Jika program anda berhasil tanpa error, maka tampilan akan seperti berikut :

```
/tmp/eKZOIkxVz1.o
Masukkan panjang array: 5
Masukkan Data Ke - 1: 2
Masukkan Data Ke - 2: 1
Masukkan Data Ke - 3: 5
Masukkan Data Ke - 4: 2
Masukkan Data Ke - 5: 6
Sebelum Sorting: 2 1 5 2 6
Sesudah Sorting: 1 2 2 5 6
```

LEMBAR EVALUASI PRAKTIKUM

1. Lakukan pengurutan array dibawah ini, dengan menggunakan Selection Sort dan hitung jumlah perbandingan dan jumlah pertukaran yang dilakukan :

```
[10, 5, 8, 3, 2]
```

Output :

```
Sorted array: 2 3 5 8 10
Number of comparisons: 10
Number of swaps: 4
```

2. Buatlah program algoritma pengurutan bubble sort yang menampilkan proses pengurutan dengan data nilai yang diinputkan melalui keyboard, lalu menampilkan proses bubble sort serta juga menampilkan jumlah perandingan dan jumlah pertukaran. Output seperti dibawah ini :

```
Masukkan jumlah data = 5

Masukkan data ke-1 = 2
Masukkan data ke-2 = 4
Masukkan data ke-3 = 1
Masukkan data ke-4 = 6
Masukkan data ke-5 = 3

Proses Bubble Sort
Iterasi 1: [1] [2] [4] [3] [6]
Iterasi 2: [1] [2] [3] [4] [6]
Iterasi 3: [1] [2] [3] [4] [6]
Iterasi 4: [1] [2] [3] [4] [6]

Data yang telah diurutkan: 1 2 3 4 6
Jumlah perbandingan: 10
Jumlah pertukaran: 4
```

3. Jelaskan menurut anda algortima pengurutan insertion sort, marge sort, dan quick sort!

Evaluasi Praktikum 3:

No	Indikator	Skor Penilaian				
		Sangat Kurang (E) <=40	Kurang (D) 41-55	Cukup (C) 55-65	Baik (B) 66-85	Sangat Baik (A) >=86
1.	Memhamai algoritma pengurutan					
2.	Dapat membuat algoritma pengurutan dengan selection sort					
3.	Dapat membuat algoritma pengurutan dengan bubble sort					

Catatan Asisten:

Asisten 1 : _____

Asisten 2 : _____

MODUL 4 – ALGORITMA PENGURUTAN (Insertion Sort, Merge Sort, dan Quick Sort)

A. Sub Capaian Pembelajaran Mata Kuliah (CPMK)

Mahasiswa dapat menerapkan algoritma pengurutan (insertion sort, marge sort, dan quick sort) pada C++

B. Instrument dan Prosedur

1. Instrument

a) Perangkat komputer / PC / Laptop / Notebook.
b) Sistem operasi Windows / Linux (optional Mac OS)
c) C++ Compiler (MinGw)
d) Geany / Notepad++ / Dev C++

2. Prosedur

a) Baca dan pahami semua tahapan praktikum dengan cermat.
b) Gunakan fasilitas yang disediakan dengan penuh rasa tanggung jawab.
c) Rapikan kembali setelah menggunakan komputer (mouse, keyboard, kursi, dll)
d) Perhatikan sikap anda untuk tidak mengganggu rekan praktikan lain
e) Pastikan diri anda tidak menyentuh sumber listrik.

C. Teori Dasar

1. Insertion sort

Insertion Sort adalah algoritma pengurutan sederhana yang membandingkan dan menggeser elemen-elemen sebuah array satu per satu untuk mengurutkannya secara berurutan. Algoritma ini cocok untuk mengurutkan data yang relatif kecil atau data yang sudah hampir terurut.

```
void insertionSort(int arr[], int n) {
    int i, key, j;
    for (i = 1; i < n; i++) {
        key = arr[i];
        j = i - 1;

        /* Memindahkan elemen yang lebih besar dari key ke posisi depan */
        while (j >= 0 && arr[j] > key) {
            arr[j + 1] = arr[j];
            j = j - 1;
        }
        arr[j + 1] = key;
    }
}
```

Penjelasan:

- Fungsi insertionSort menerima dua argumen: array integer arr yang akan diurutkan, dan integer n yang menunjukkan panjang array.
- Variabel i, key, dan j adalah variabel yang digunakan dalam iterasi dan pengurutan.

- Iterasi dimulai dari indeks ke-1 (i=1) hingga akhir array (i<n).
- Nilai pada indeks ke-i disimpan di dalam variabel key.
- Variabel j diinisialisasi dengan nilai i-1.
- Dalam loop while, jika nilai pada indeks ke-j lebih besar dari key, maka nilai pada indeks ke-j dipindahkan ke indeks ke-j+1.
- Variabel j dikurangi satu, dan loop while dilanjutkan hingga nilai pada indeks ke-j tidak lagi lebih besar dari key.
- Nilai key disimpan pada indeks ke-j+1.
- Proses pengurutan diulangi untuk setiap elemen pada array.

2. Marge sort

Merge Sort adalah yaitu metode pengurutan dengan memecah kemudian menyelesaikan setiap bagian kemudian menggabungkannya kembali. Pertama data dipecah menjadi 2 bagian dimana bagian pertama merupakan setengah (jika data genap) atau setengah minus satu (jika data ganjil) dari seluruh data, kemudian dilakukan pemecahan kembali untuk masing-masing blok sampai hanya terdiri dari satu data tiap blok.

```
#include <iostream>
using  namespace std;

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 printArray(int arr[], int size) {
    for (int i = 0; i < size; i++)
        cout << arr[i] << "  ";
}
int main() {
    int arr[] = {12, 11, 13, 5, 6, 7};
    int arr_size = sizeof(arr) / sizeof(arr[0]);

    cout << "Data Sebelum Pengurutan : " << endl;

    printArray(arr, arr_size);
    mergeSort(arr, 0, arr_size - 1);

    cout << "\nData Setelah Pengurutan : " << endl;

    printArray(arr, arr_size);

    return 0;
}
```

3. Quick sort

a) **Algortima quicksort dengan cara iterative :**

```
partition(a[],start, end){
    pivot ← a[end]
    partitionIndex ← (start-1)
    for (j←start; j<end; j++)
        if (a[j] <= pivot) {
            partitionIndex++
            swap(a[partitionIndex], a[j])
        }
    swap(a[partitionIndex+1], a[end])
    return (partitionIndex+1)
}

quickSort(a[], start, end){
    stack[end - start + 1]
    top ← -1
    stack[++top] ← start
    stack[++top] ← end
    while (top >= 0) {
        end ← stack[top--]
        start ← stack[top--]
        p ← partition(a, start, end)
        if (p-1 > start) {
            stack[++top] ← start
            stack[++top] ← p-1
        }
        if (p+1 < end) {
            stack[++top] ← p+1
            stack[++top] ← end
        }
    }
}
```

b) **Algoritma quicksort dengan rekursif**

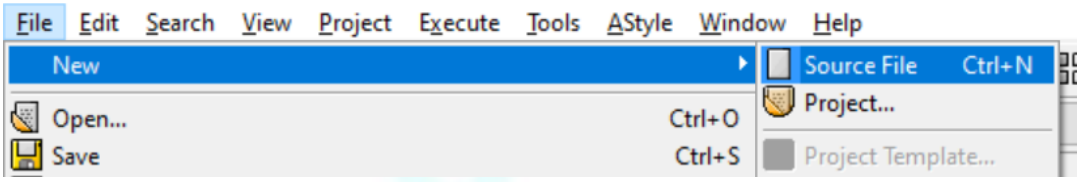
```
partition(a[], start, end){
      pivot ← a[end]
      partitionIndex ← start
      for(i←start; i<end; i++)
            if(a[i] < pivot){
                  swap(a[i], a[partitionIndex])
                  partitionIndex++
            }
      swap(a[partitionIndex], a[end])
      return partitionIndex
}

quickSort(a[], start, end) {
      if (start<end){
            partitionIndex ← partition(a, start, end)
            quickSort(a, start, partitionIndex-1)
            quickSort(a, partitionIndex+1, end)
      }
}
```

Parameter ***start*** merupkan indeks awal dari array, dan parameter ***end*** adalah indeks akhir dari array

D. Kegiatan Praktikum

1. Insertion Sort

a) Klik *File* pada header, kemudian *New* lalu *Source File*. Atau anda menekan shortcode pada keyboard dengan menekan *ctrl+N* :



b) Buatlah sebuah file C++ dengan eksistensi *.cpp* dengan format :

M4.1_001.cpp

Dimana :

- **M4 :** Modul yang sedang dikerjakan
- **1 :** nomor kegiatan praktikum yang sedang dikerjakan
- **001 :** 3 stambuk terakhir

c) Ketikkan program dibawah ini :

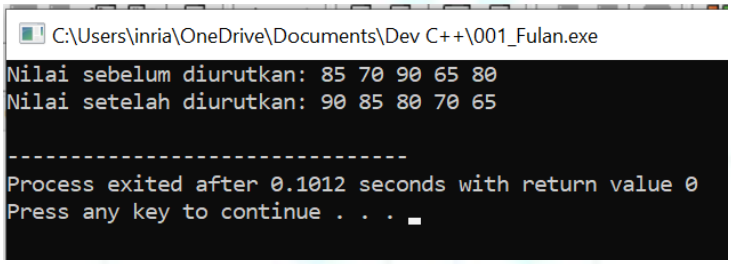
```
#include <iostream>
using namespace std;

void insertionSort(int arr[], int n) {
    int i, key, j;
    for (i = 1; i < n; i++) {
        key = arr[i];
        j = i - 1;

        /* Memindahkan elemen yang lebih besar dari key ke posisi depan */
        while (j >= 0 && arr[j] < key) {
            arr[j + 1] = arr[j];
            j = j - 1;
        }
        arr[j + 1] = key;
    }
}
```

```
int main() {
    int nilai[] = {85, 70, 90, 65, 80};
    int n = sizeof(nilai) / sizeof(nilai[0]);

    cout << "Nilai sebelum diurutkan: ";
    for (int i = 0; i < n; i++) {
        cout << nilai[i] << " ";
    }
    cout << endl;

    insertionSort(nilai, n);

    cout << "Nilai setelah diurutkan: ";
    for (int i = 0; i < n; i++) {
        cout << nilai[i] << " ";
    }
    cout << endl;

    return 0;
}
```

d) Kemuidan *save* program yang telah diketik pada folder yang telah dibuat sebelumnya pada modul 1, dengan menekan *ctrl+s*

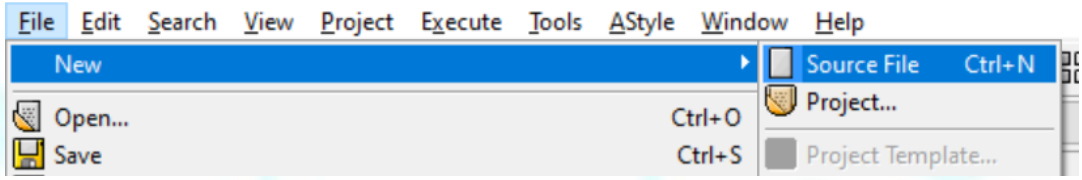
e) Kemudian, apabila telah selesai menuliskan kode diatas maka anda dapat menekan tombol *compile* atau shortcode pada keyboard : *F9*

f) Jika program anda berhasil tanpa error, maka tampilan akan seperti berikut :



2. Marge sort

a) Klik *File* pada header, kemudian *New* lalu *Source File.* Atau anda menekan shortcode pada keyboard dengan menekan *ctrl+N* :



b) Buatlah sebuah file C++ dengan eksistensi *.cpp* dengan format :

M4.2_001.cpp

Dimana :

- **M4 :** Modul yang sedang dikerjakan
- **2 :** nomor kegiatan praktikum yang sedang dikerjakan
- **001 :** 3 stambuk terakhir

c) Ikuti instruksi dibawah ini :

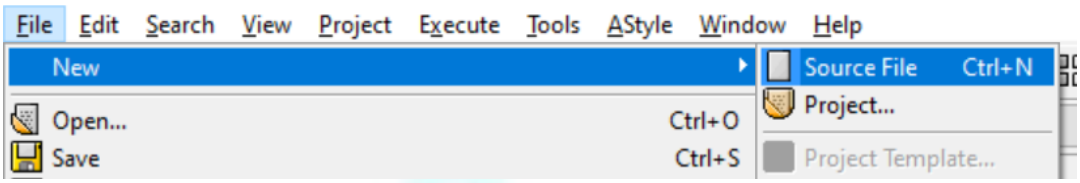
I. Buka text editor yang akan digunakan
II. Kerjakan contoh program insertion sort, dan merge sort yang ada pada teori dasar.
III. Setelah percobaan selesai, tutup semua perangkat lunak yang telah digunakan.
IV. Matikan PC dan rapihkan meja praktikum.

d) Kemuidan *save* program yang telah diketik pada folder yang telah dibuat sebelumnya pada modul 1, dengan menekan *ctrl+s*

e) Kemudian, apabila telah selesai menuliskan kode diatas maka anda dapat menekan tombol *compile* atau shortcode pada keyboard : *F9*

3. Quick Sort

a) Klik *File* pada header, kemudian *New* lalu *Source File*. Atau anda menekan shortcode pada keyboard dengan menekan *ctrl+N* :



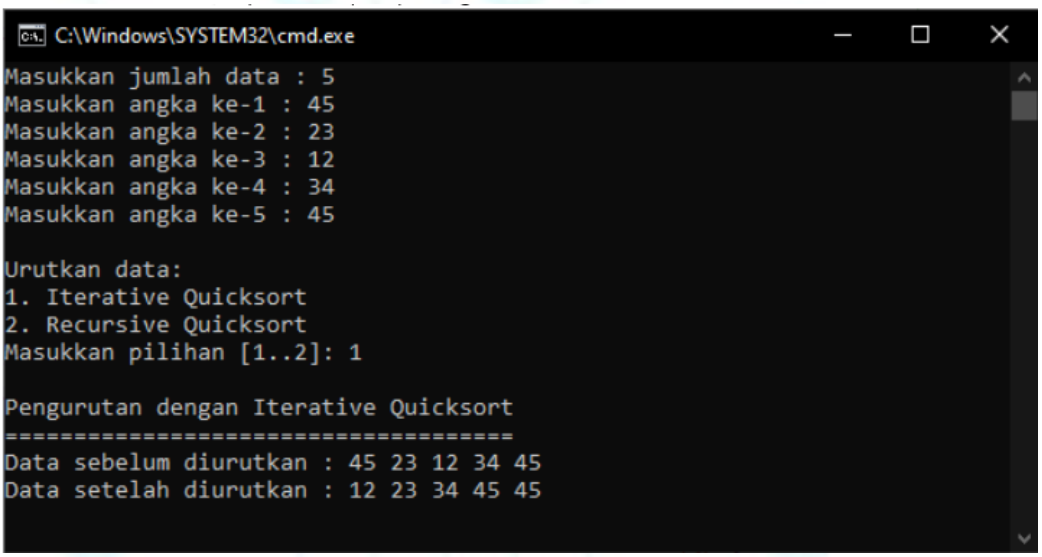
b) Buatlah sebuah file C++ dengan eksistensi *.cpp* dengan format :

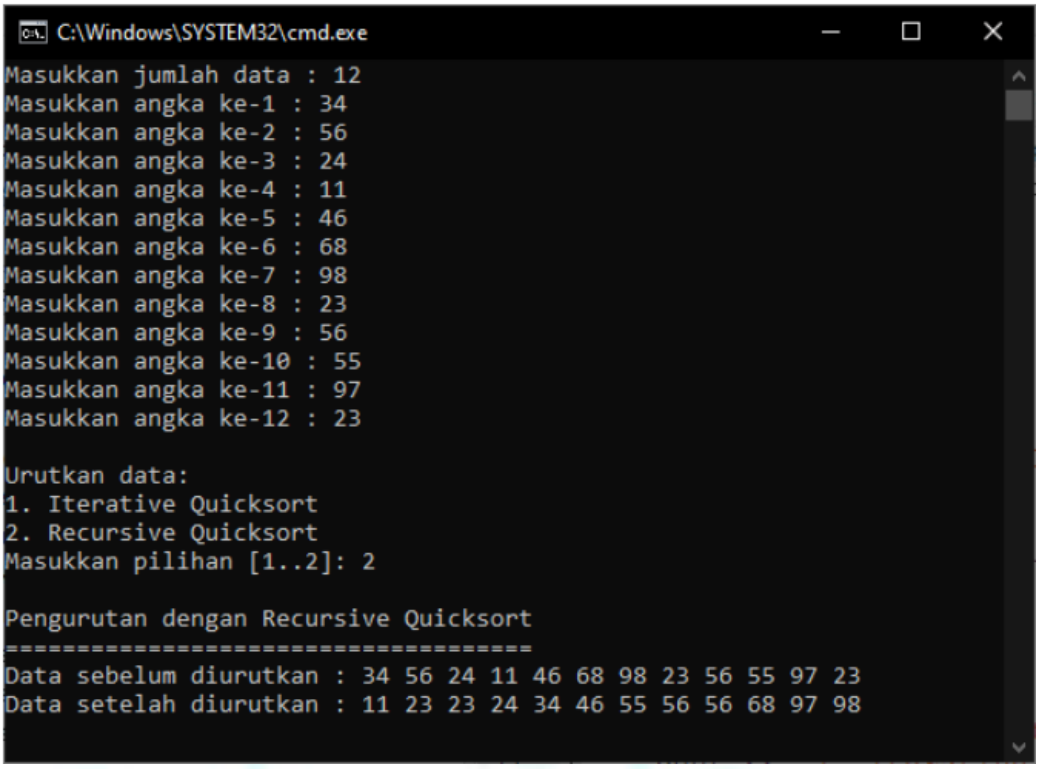
M4.3_001.cpp

Dimana :

- **M4 :** Modul yang sedang dikerjakan
- **3 :** nomor kegiatan praktikum yang sedang dikerjakan
- **001 :** 3 stambuk terakhir

c) Ketikkan program dibawah ini :





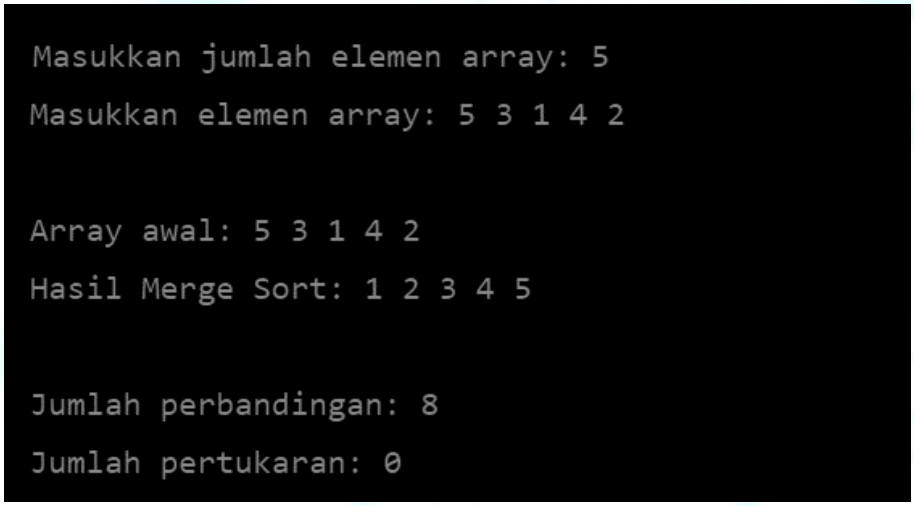
d) Kemuidan *save* program yang telah diketik pada folder yang telah dibuat sebelumnya pada modul 1, dengan menekan *ctrl+s*

e) Kemudian, apabila telah selesai menuliskan kode diatas maka anda dapat menekan tombol *compile* atau shortcode pada keyboard : *F9*

f) Periksa jika ada kesalahan.

LEMBAR EVALUASI PRAKTIKUM

1. Buatlah program untuk mengurutkan data menggunakan algoritma pengurutan insertion sort pada bahasa pemrograman C++! Program menerima input jumlah data yang akan diurutkan, kemudian meminta pengguna untuk memasukkan nilai-nilai data tersebut. Setelah itu, program melakukan pengurutan data menggunakan algoritma insertion sort dan menampilkan hasil pengurutan beserta jumlah perbandingan dan jumlah pergeseran data yang dilakukan selama proses pengurutan.
2. Buatlah sebuah program untuk mengurutkan array dengan menggunakan algoritma Merge Sort. Program tersebut meminta user untuk memasukkan jumlah elemen array yang diinginkan dan elemen-elemen tersebut. Tampilkan array awal dan hasil pengurutan array menggunakan Merge Sort. Hitung jumlah perbandingan dan pertukaran yang terjadi selama proses pengurutan, dan tampilkan jumlah tersebut.



Evaluasi Praktikum 4:

No	Indikator	Skor Penilaian				
		Sangat Kurang (E) <=40	Kurang (D) 41-55	Cukup (C) 55-65	Baik (B) 66-85	Sangat Baik (A) >=86
1.	Dapat membuat algoritma pengurutan dengan insertion sort					
2.	Dapat membuat algoritma pengurutan dengan marge sort					
3	Dapat membuat algoritma pengurutan dengan quick sort					

Catatan Asisten:

Asisten 1 : _____

Asisten 2 : _____

MODUL 5 – ALGORITMA PENCARIAN

A. Sub Capaian Pembelajaran Mata Kuliah (CPMK)

Mahasiswa dapat menerapkan algoritma pencarian pada C++

B. Instrument dan Prosedur

1. Instrument

a) Perangkat komputer / PC / Laptop / Notebook.
b) Sistem operasi Windows / Linux (optional Mac OS)
c) C++ Compiler (MinGw)
d) Geany / Notepad++ / Dev C++

2. Prosedur

a) Baca dan pahami semua tahapan praktikum dengan cermat.
b) Gunakan fasilitas yang disediakan dengan penuh rasa tanggung jawab.
c) Rapikan kembali setelah menggunakan komputer (mouse, keyboard, kursi, dll)
d) Perhatikan sikap anda untuk tidak mengganggu rekan praktikan lain
e) Pastikan diri anda tidak menyentuh sumber listrik.

C. Teori Dasar

1. Algoritma quicksort dengan rekursif

```
partition(a[], start, end){
      pivot ← a[end]
      partitionIndex ← start
      for(i←start; i<end; i++)
            if(a[i] < pivot){
                  swap(a[i], a[partitionIndex])
                  partitionIndex++
            }
      swap(a[partitionIndex], a[end])
      return partitionIndex
}

quickSort(a[], start, end) {
      if (start<end){
            partitionIndex ← partition(a, start, end)
            quickSort(a, start, partitionIndex-1)
            quickSort(a, partitionIndex+1, end)
      }
}
```

2. Algoritma linear search mengembalikan index dari nilai yang dicari

```
linearSearch(a[], n, x){
      for(int i←0; i<n; i++){
            if (x == a[i]) return i
      }
      return -1
}
```

Parameter **a[ ]** adalah array dari nilai-nilai yang diinputkan, parameter **n** adalah kapasitas arraynya, parameter **x** adalah nilai yang dicari dari array **a[ ].**

3. **Algoritma iterative binay search, mengembalikan index dari nilai yang dicari :**

```
binarySearch(a[], n, x){
      start ← 0;
      end   ← n-1;
      while(start <=end){
            mid ← start + (end-start)/2;;
            if (x == a[mid]) return mid;
            else if (x < a[mid]) end = mid-1;
            else if (x > a[mid]) start = mid+1;
      }
      return -1;
}
```

Parameter **a[ ]** adalah array dari nilai-nilai yang diinputkan, parameter **n** adalah kapasitas arraynya, parameter **x** adalah nilai yang dicari dari array **a[ ].**

4. **Algoritma recursive binary search, mengembalikan index dari nilai yang dicari :**

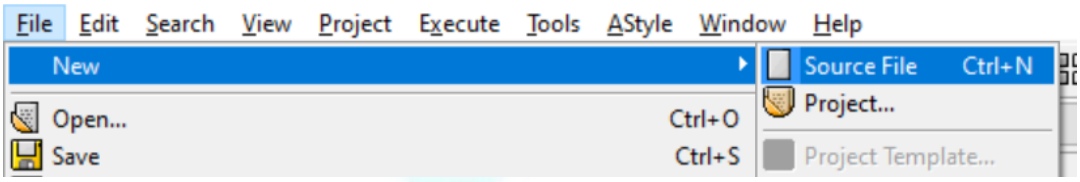
```
binarySearch(a[], start, end, x){
      if (start > end) return -1;
      int mid = start + (end-start)/2;
      if (x == a[mid]) return mid;
      else if (x < a[mid]) return binarySearch(a, start, mid-1, x);
      else if (x > a[mid]) return binarySearch(a, mid+1, end, x);
}
```

Parameter **a[ ]** adalah array dari nilai-nilai yang diinputkan, parameter **start** adalah index awal dari arry, parameter **end** adalah index terakhir dari arry, parameter **x** adalah nilai yang dicari dari array **a[ ].**

D. Kegiatan Praktikum

1. Studi Kasus

a) Klik *File* pada header, kemudian *New* lalu *Source File.* Atau anda menekan shortcode pada keyboard dengan menekan *ctrl+N* :



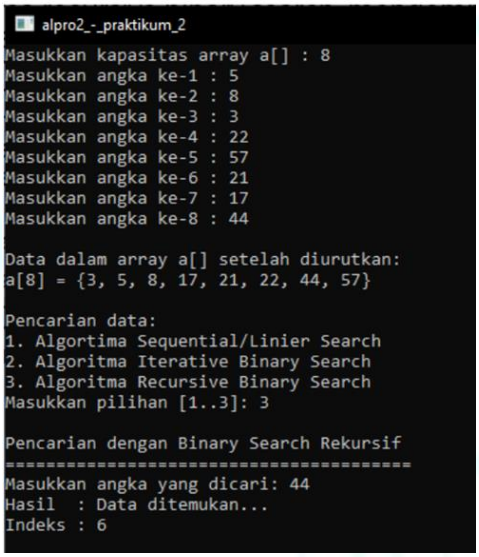
b) Buatlah sebuah file C++ dengan eksistensi *.cpp* dengan format :

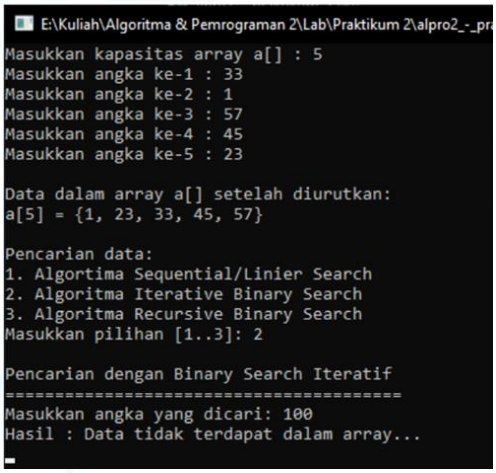
M5.1_001.cpp

Dimana :

- **M5 :** Modul yang sedang dikerjakan
- **1 :** nomor kegiatan praktikum yang sedang dikerjakan
- **001 :** 3 stambuk terakhir

c) Buatlah program agar tampilan seperti berikut :





d) Kemuidan *save* program yang telah diketik pada folder yang telah dibuat sebelumnya pada modul 1, dengan menekan *ctrl+s*

e) Kemudian, apabila telah selesai menuliskan kode diatas maka anda dapat menekan tombol *compile* atau shortcode pada keyboard : *F9*

LEMBAR EVALUASI PRAKTIKUM

1. Buatlah sebuah program untuk mencari nilai minimum dari sebuah array dengan menggunakan algoritma pencarian biner. Program tersebut meminta user untuk memasukkan jumlah elemen array yang diinginkan dan elemen-elemen tersebut. Pastikan elemen-elemen array sudah diurutkan terlebih dahulu sebelum dilakukan pencarian nilai minimum. Tampilkan array yang diinput, nilai minimum yang ditemukan, dan jumlah perbandingan yang dilakukan pada algoritma pencarian biner. Output dari program :

```
Masukkan jumlah elemen array: 5
Masukkan elemen-elemen array: 5 8 3 5 9

Array yang diinput: 3 5 5 8 9
Nilai minimum yang ditemukan: 3

--------------------------------
Process exited after 9.937 seconds with return value 0
Press any key to continue . . .
```

2. Jelaskan apa yang dimaksud dengan pointer!
3. Apa saja jenis dari pointer? Jelaskan!

Evaluasi Praktikum 5:

No	Indikator	Skor Penilaian				
		Sangat Kurang (E) <=40	Kurang (D) 41-55	Cukup (C) 55-65	Baik (B) 66-85	Sangat Baik (A) >=86
1.	Dapat memhami algoritma pencarian					
2.	Dapat membuat program algoritma linear search					
3.	Dapat membuat program algoritma binary search					

Catatan Asisten:

Asisten 1 : _____

Asisten 2 : _____

MODUL 6 – POINTER PADA C++

A. Sub Capaian Pembelajaran Mata Kuliah (CPMK)

Mahasiswa dapat menerapkan penggunaan pointer pada C++

B. Instrument dan Prosedur

1. Instrument

a) Perangkat komputer / PC / Laptop / Notebook.
b) Sistem operasi Windows / Linux (optional Mac OS)
c) MinGW/C++ Compiler
d) Geany/Notepad++/Dev C++

2. Prosedur

a) Baca dan pahami semua tahapan praktikum dengan cermat.
b) Gunakan fasilitas yang disediakan dengan penuh rasa tanggung jawab.
c) Rapikan kembali setelah menggunakan komputer (mouse, keyboard, kursi, dll)
d) Perhatikan sikap anda untuk tidak mengganggu rekan praktikan lain
e) Pastikan diri anda tidak menyentuh sumber listrik.

C. Teori Dasar

Pointer pada C++ adalah sebuah variabel yang berisi alamat memori dari variabel lain. Dengan menggunakan pointer, kita dapat mengakses dan memodifikasi nilai dari variabel yang diacu melalui alamat memori yang tersimpan pada pointer.

1. Operator dereference (&)

Untuk deklarasi pointer dereference pada C++, kita dapat menggunakan operator dereferencing * untuk mengambil nilai yang disimpan pada alamat memori yang diacu oleh pointer. Berikut adalah contoh deklarasi pointer dereference pada C++:

```
int num = 5;
int * numPtr = &num;
int value = *numPtr;
```

2. Operator reference (*)

Pointer reference di C++ adalah cara untuk membuat sebuah referensi (alias) dari sebuah variabel, sehingga kita dapat mengakses dan memodifikasi variabel tersebut melalui referensi. Pointer reference mirip dengan pointer, namun pointer reference memiliki sintaks yang lebih sederhana dan lebih mudah dibaca.

Deklarasi syntax pointer reference :

```
tipe_data& nama_pointer_reference = variabel;
```

Dimana :

- **tipe_data** adalah tipe data variabel yang ingin dijadikan pointer reference.
- **nama_pointer_reference** adalah nama dari pointer reference yang ingin dideklarasikan.
- **variabel** adalah variabel yang akan diacu oleh pointer reference.

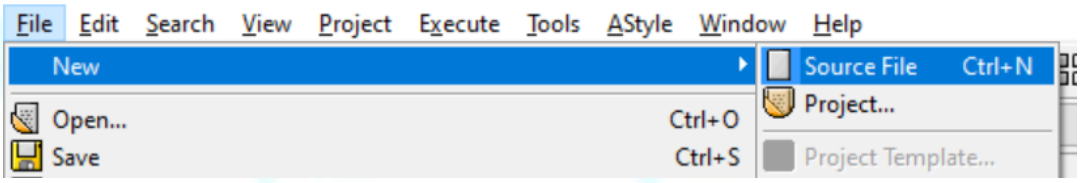
```
int num = 5;
int& numRef = num;
```

D. Kegiatan Praktikum

1. Program pointer yang menggunakan operator dereference :

a) Bukalah C++ compiler yang ada dikomputer depan anda (Geany/DevC++/Notepad C++).

b) Klik *File* pada header, kemudian *New* lalu *Source File*. Atau anda menekan shortcode pada keyboard dengan menekan *ctrl+N* :



c) Buatlah sebuah file C++ pada folder yang telah anda buat pada Langkah *b* dengan eksistensi *.cpp* dengan format :
M6.1_001.cpp
Dimana :
- **M6 :** Modul yang sedang dikerjakan
- **1 :** nomor kegiatan praktikum yang dikerjakan
- **001 :** 3 stambuk terakhir

d) Ketikkan program dibawah ini :

```
#include <iostream>
using namespace std;

int main() {
  int x, y;
  int* ptr;

  // meminta pengguna memasukkan dua nilai
  cout << "Masukkan dua nilai bilangan bulat: ";
  cin >> x >> y;

  // menampilkan nilai awal
  cout << "Nilai awal x: " << x << endl;
  cout << "Nilai awal y: " << y << endl;
```

```
    // mengubah nilai x dan y melalui pointer
    ptr = &x;
    *ptr = 10;

    ptr = &y;
    *ptr = 20;

    // menampilkan nilai setelah diubah melalui pointer
    cout << "Nilai setelah diubah melalui pointer:" << endl;
    cout << "Nilai x: " << x << endl;
    cout << "Nilai y: " << y << endl;

    return 0;
}
```

e) Kemuidan *save* program yang telah diketik pada folder yang telah dibuat pada modul 1, dengan menekan *ctrl+s*

f) Kemudian, apabila telah selesai menuliskan kode diatas maka anda dapat menekan tombol *compile* atau shortcode pada keyboard : *F9*

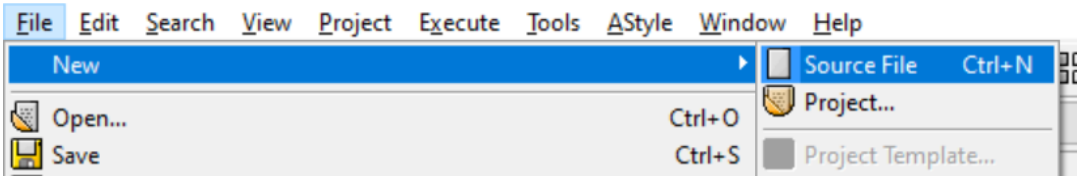
g) Jika program anda berhasil tanpa error, maka tampilan akan seperti berikut :

```
Masukkan dua nilai bilangan bulat: 10 20
Nilai awal x: 10
Nilai awal y: 20
Nilai setelah diubah melalui pointer:
Nilai x: 10
Nilai y: 20

--------------------------------
Process exited after 2.222 seconds with return value 0
Press any key to continue . . .
```

2. Program pointer yang menggunakan operator reference :

a) Bukalah C++ compiler yang ada dikomputer depan anda (Geany/DevC++/Notepad C++).

b) Klik *File* pada header, kemudian *New* lalu *Source File.* Atau anda menekan shortcode pada keyboard dengan menekan *ctrl+N* :



c) Buatlah sebuah file C++ pada folder yang telah anda buat pada Langkah *b* dengan eksistensi *.cpp* dengan format :
M6.2_001.cpp
Dimana :

- **M6 :** Modul yang sedang dikerjakan
- **2 :** nomor kegiatan praktikum yang dikerjakan
- **001 :** 3 stambuk terakhir

d) Ketikkan program dibawah ini :

```
#include <iostream>
using namespace std;

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

```
int main() {
    int a = 5;
    int &ref = a; // membuat reference 'ref' yang merujuk pada variabel 'a'

    cout << "Nilai a: " << a << endl;
    cout << "Nilai ref: " << ref << endl;

    ref = 10; // mengubah nilai variabel 'a' melalui reference 'ref'

    cout << "Nilai a setelah diubah: " << a << endl;

    int x = 3, y = 5;

    cout << "Sebelum swap: x = " << x << ", y = " << y << endl;
    swap(x, y); // mengubah nilai variabel 'x' dan 'y' melalui reference '&a' dan '&b' pada fungsi 'swap'
    cout << "Setelah swap: x = " << x << ", y = " << y << endl;

    return 0;
}
```

e) Kemuidan *save* program yang telah diketik pada folder yang telah dibuat pada modul 1, dengan menekan *ctrl+s*

f) Kemudian, apabila telah selesai menuliskan kode diatas maka anda dapat menekan tombol *compile* atau shortcode pada keyboard : *F9*

g) Jika program anda berhasil tanpa error, maka tampilan akan seperti berikut :

```
Nilai a: 5
Nilai ref: 5
Nilai a setelah diubah: 10
Sebelum swap: x = 3, y = 5
Setelah swap: x = 5, y = 3

--------------------------------
Process exited after 0.05439 seconds with return value 0
Press any key to continue . . .
```

LEMBAR EVALUASI PRAKTIKUM

1. Deklarasikan sebuah pointer integer yang menunjuk ke variabel integer 'a', lalu berikan nilai 10 ke variabel 'a' dan tampilkan nilai variabel 'a' melalui pointer.
2. Deklarasikan sebuah array integer dengan 5 elemen, lalu berikan nilai 1 sampai 5 untuk masing-masing elemen array. Deklarasikan sebuah pointer integer yang menunjuk ke alamat memori elemen pertama dari array, lalu tampilkan nilai masing-masing elemen array melalui pointer.
3. Tuliskan program untuk menukar nilai dari dua variabel integer menggunakan pointer.
4. Tuliskan program untuk menampilkan alamat memori dan isi dari setiap elemen array menggunakan pointer.
5. Tuliskan program untuk mengalikan dua buah array integer berukuran sama menggunakan pointer, lalu tampilkan hasilnya.
6. Jelaskan mengenai class/struct pada pemrograman C++
7. Jelaskan mengenai objek pada algoritma pemrograman C++.

Evaluasi Praktikum 6:

No	Indikator	Skor Penilaian				
		Sangat Kurang (E) <=40	Kurang (D) 41-55	Cukup (C) 55-65	Baik (B) 66-85	Sangat Baik (A) >=86
1.	Dapat memahami pointer					
2.	Dapat membedakan operator dereference dan reference					
3.	Dapa memahami program C++ pointer dengan operator dereference					
4.	Dapa memahami program C++ pointer dengan operator reference					

Catatan Asisten:

Asisten 1 : _____

Asisten 2 : _____

MODUL 7 – CLASS/STRUCT & OBJEK PADA C++

A. Sub Capaian Pembelajaran Mata Kuliah (CPMK)

Mahasiswa dapat menerapkan konsep OOP pada C++

B. Instrument dan Prosedur

1. Instrument

a) Perangkat komputer / PC / Laptop / Notebook.
b) Sistem operasi Windows / Linux (optional Mac OS)
c) MinGW/C++ Compiler
d) Geany/Notepad++/Dev C++/NetBeans IDE

2. Prosedur

a) Baca dan pahami semua tahapan praktikum dengan cermat.
b) Gunakan fasilitas yang disediakan dengan penuh rasa tanggung jawab.
c) Rapikan kembali setelah menggunakan komputer (mouse, keyboard, kursi, dll)
d) Perhatikan sikap anda untuk tidak mengganggu rekan praktikan lain
e) Pastikan diri anda tidak menyentuh sumber listrik.

C. Teori Dasar

1. Objek dan Class

- Class merupakan blueprint untuk menghasilkan objek
- Objek memiliki: State, Behaviour, dan Identity
- Class dapat memiliki : Field, Method, Constructor, Class dan Interface bercabang

2. Deklarasi Class/Struct :

```
class NamaClass{
      private field;
      private function;
};

struct NamaClass{
      public field;
      public function;
};
```

3. Pembuatan Object dan Class :

```
NamaClass namaObjek;
```

4. Pembuatan konstruktor :

```
public:
      NamaClass(){
            //isi konstruktor
      }
```

5. **Inisialisasi nilai pada instance variable secara langsung:**

```
class A{
      public:
            int x;
            int y;
};
```

```
int main(){
      A a;
      a.x = 10;
      a.y = 20;

      cout << "x = " << a.x << endl;
      cout << "y = " << a.y << endl;
}
```

Pengaksesan secara langsung terhadap member *class* diperbolehkan jika *access modifier* dari variable atau method bersifat public.

6. **Inisialisasi nilai pada instance variable yang bersifat private melalui public function:**

```
class A{
    int x;
    int y;

    public:
          void setX(int x) {  //public method untuk pemberian nilai variabel x
              this->x = x;
          }

          void setY(int y) {  //public method untuk pemberian nilai variabel y
              this->y = y;
          }

          int getX() {        //public method untuk mengambil nilai variabel x
              return x;
          }

          int getY() {        //public method untuk mengambil nilai variabel y
              return y;
          }
};

int main(){
      A a;
      a.setX(10);  //inisialisasi x melalui public method setX()
      a.setY(20);  //inisialisasi y melalui public method setY()

      cout << "x = " + a.getX() << endl; //pengaksesan x melalui method getX()
      cout << "y = " + a.getY() << endl; //pengaksesan y melalui method getY()
}
```

Pengaksesan dari public method dilakukan jika access modifier dari variable atau method yang akan diakses bersifat private.

7. Inisialisasi nilai pada instance variable melalui konstruktor:

```
class A{
    int x;
    int y;

    public:
      A(int x, int y) { //konstruktor untuk inisialisasi nilai ke variabel x dan y
        this->x = x;
        this->y = y;
      }

      int getX() {        //public method untuk mengambil nilai dari variabel x
        return x;
      }
```

```
       int getY() {        //public method untuk mengambil nilai dari variabel y
         return y;
       }
};

int main(){
       A a(10, 20);   //inisialisasi x dan y melalui konstruktor

       cout << "x = " + a.getX() << endl; //pengaksesan x melalui method getX()
       cout << "y = " + a.getY() << endl; //pengaksesan y melalui method getY()
}
```

Pengaksesan dari public method dilakukan jika access modifier dari variable atau method yang akan diakses bersifat **private.** Dilakukan untuk mempersingkat penulisan kode jika banyak *field* yang akan diinisialisasi.

8. Pengaksesan melalui pointer

Untuk melakukan pengaksesan data field sebuah objek melaui pointer, digunakan operator -> (arrow). Contoh :

```
struct A{
       int x;
       int y;
};

int main(){
       A a;
       a.x = 10;
       a.y = 20;

       A *pA;    //deklarasi pointer untuk menyimpan alamt memori objek dari kelas A
       pA = &a;  //pointer pA menyimpan alamat memori a

       cout << "x = " << pA->x << endl; //nilai x diakses melalui pointer pA
       cout << "y = " << pA->y << endl; //nilai y diakses melalui pointer pA

}
```

D. Kegiatan Praktikum

1. Studi Kasus

a) Buatlah sebuah file C++ dengan eksistensi *.cpp* dengan format :

M7.1_001.cpp

Dimana :

- **M7 :** Modul yang sedang dikerjakan
- **1 :** nomor kegiatan praktikum yang sedang dikerjakan
- **001 :** 3 stambuk terakhir

b) Ikuti instruksi dibawah ini :

I. Buat sebuah class dengan nama **Mahasiswa** yang memiliki beberapa field yang memiliki *access modifier* **private:**

- **Stambuk:** tipe data **string**
- **Nama:** tipe data **string**
- **Angkatan:** tipe data **int**
- **Kelas:** tipe data **string**

II. Buat objek dari class **Mahasiswa** dengan nama **mahasiswa.**

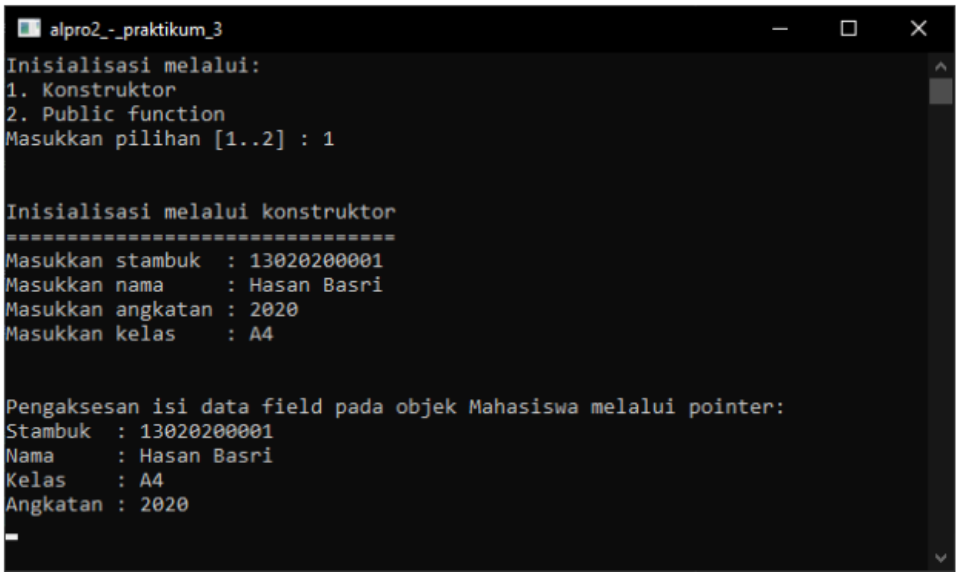
III. Lakukan penginputan untuk semua field dari objek **mahasiswa.** Inisialisasi (pemberian nilai) data field dapat dilakukan melalui konstruktor atau public function tergantung pilihan yang dipilih.

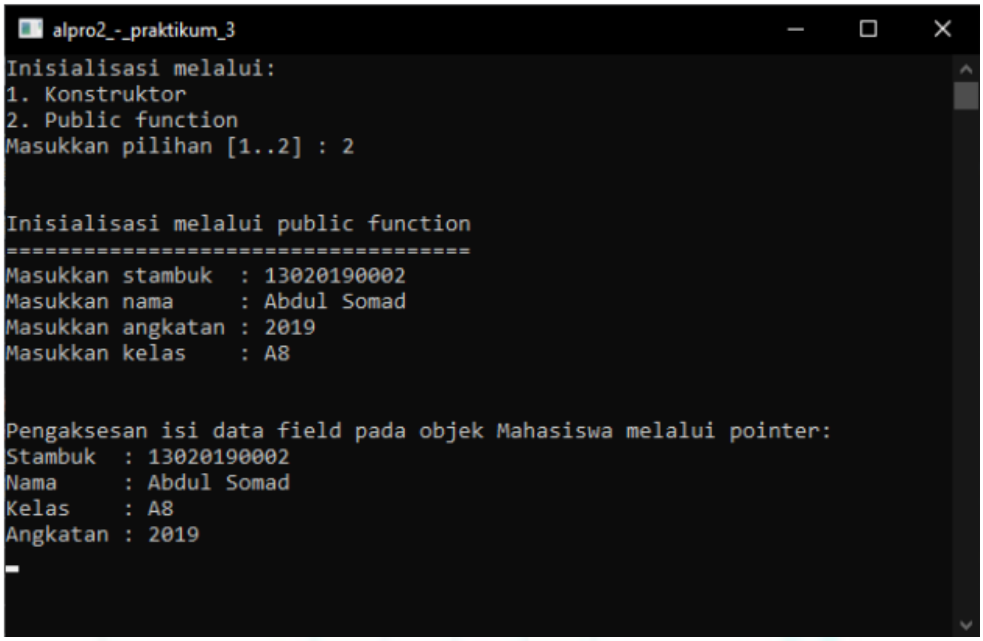
IV. Tampilkan Kembali semua field dari objek **mahasiswa** yang telah diinisialisasi sebelumnya melalui pointer yang menyimpan alamat memori dari objek mahasiswa.

c) Kemuidan *save* program yang telah diketik pada folder yang telah dibuat sebelumnya pada modul 1, dengan menekan *ctrl+s*

d) Kemudian, apabila telah selesai menuliskan kode diatas maka anda dapat menekan tombol *compile* atau shortcode pada keyboard : *F9*

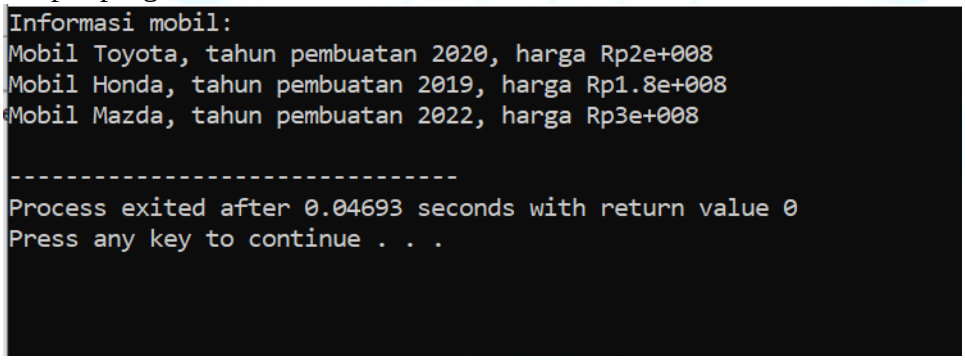
e) Jika program anda berhasil tanpa error, maka tampilan akan seperti berikut :





LEMBAR EVALUASI PRAKTIKUM

1. Buatlah sebuah class bernama "Mobil" dengan atribut "merk", "tahun_pembuatan", dan "harga". Buatlah constructor untuk class Mobil yang dapat menerima nilai untuk ketiga atribut tersebut. Buatlah getter method untuk masing-masing atribut Mobil. Buatlah setter method untuk atribut harga. Buatlah function untuk menampilkan informasi mobil dengan format "Mobil [merk], tahun pembuatan [tahun_pembuatan], harga [harga]". Buatlah program utama yang menggunakan class Mobil untuk membuat beberapa objek Mobil, kemudian menampilkan informasi mobil-mobil tersebut.
   Output program :



2. Buatlah program C++ untuk menghitung luas dan keliling sebuah lingkaran dengan menggunakan konsep class atau struct. Class atau struct yang memiliki variabel jari-jari dan method/function untuk menghitung luas dan keliling lingkaran. Program tersebut meminta user untuk memasukkan nilai jari-jari. Program menampilkan luas dan keliling lingkaran dengan menggunakan class/struct yang telah dibuat. Terakhir, tambahkan opsi untuk mengulang program atau keluar dari program setelah menampilkan output.

Output program :

```
Masukkan nilai jari-jari lingkaran: 8
Luas lingkaran: 200.96
Keliling lingkaran: 50.24

Apakah Anda ingin mengulang program? (Y/N): y
Masukkan nilai jari-jari lingkaran: 4
Luas lingkaran: 50.24
Keliling lingkaran: 25.12

Apakah Anda ingin mengulang program? (Y/N): n

--------------------------------
Process exited after 10.46 seconds with return value 0
Press any key to continue . . .
```

3. Jelaskan mengenai vector class pada C++ yang telah anda pelajari.

Evaluasi Praktikum 7:

No	Indikator	Skor Penilaian				
		Sangat Kurang (E) <=40	Kurang (D) 41-55	Cukup (C) 55-65	Baik (B) 66-85	Sangat Baik (A) >=86
1.	Dapat memahami class/struct dan objek pada C++					
2.	Dapat membuat class/struct C++					
3.	Dapat membuat objek C++					
4.	Dapat membuat konstruktor C++					

Catatan Asisten:

Asisten 1 : _____

Asisten 2 : _____

MODUL 8 – VECTOR CLASS PADA C++

A. Sub Capaian Pembelajaran Mata Kuliah (CPMK)

Mahasiswa dapat menerapkan penggunaan class vector pada C++

B. Instrument dan Prosedur

1. Instrument

a) Perangkat komputer / PC / Laptop / Notebook.
b) Sistem operasi Windows / Linux (optional Mac OS)
c) MinGW/C++ Compiler
d) Geany/Notepad++/Dev C++/NetBeans IDE

2. Prosedur

a) Baca dan pahami semua tahapan praktikum dengan cermat.
b) Gunakan fasilitas yang disediakan dengan penuh rasa tanggung jawab.
c) Rapikan kembali setelah menggunakan komputer (mouse, keyboard, kursi, dll)
d) Perhatikan sikap anda untuk tidak mengganggu rekan praktikan lain
e) Pastikan diri anda tidak menyentuh sumber listrik.

C. Teori Dasar

1. Vector Class

Merupakan struktur data dinamis yang disediakan pada pemrograman c++. Berasal dari <vector> directive, Namespace srd::vector. Contoh deklarasi :

```
Deklarasi:
vector<tipe_data/struct/class> namaObjek;

Function penambahan data:
namaObjek.push_back(data);

Function pengaksesan data berdasarkan index:
namaObjek.at(int index);

Function menghapus data:
namaObjek.erase(namaObjek.begin() + index);
```

Contoh syntax vector :

```
class A{
    int x;
    int y;

    public:
        A(){}
        A(int x, int y){
            this->x = x;
            this->y = y;
```

```
        }

        void setX(int x){
            this->x = x;
        }

        void setY(int y){
            this->y = y;
        }

        int getX(){
            return x;
        }

        int getY(){
            return y;
        }
};

int main(){
    vector<A> vObjectA;

    //contoh memasukkan 4 data dalam objek vObjectA
    vObjectA.push_back(A(10, 20));              //data index 0
    vObjectA.push_back(A(100, 200));            //data index 1
    vObjectA.push_back(A(1000, 2000));          //data index 2
    vObjectA.push_back(A(10000, 20000));        //data index 3

    //menghapus data pada indeks ke-2 (object dgn nilai x=1000, y=2000)
    int index = 2;
    vObjectA.erase(vObjectA.begin() + index);

    //menampilkan semua isi dari vObjectA
    for(int i=0; i<(int)vObjectA.size(); i++){
        cout << "x = " << vObjectA.at(i).getX() << endl;
        cout << "y = " << vObjectA.at(i).getY() << endl;
        cout << "-----------------" << endl;
    }
}
```

D. Kegiatan Praktikum

1. Studi Kasus

a) Buatlah sebuah file C++ dengan eksistensi *.cpp* dengan format :

M8.1_001.cpp

Dimana :

- **M8 :** Modul yang sedang dikerjakan
- **1 :** nomor kegiatan praktikum yang sedang dikerjakan
- **001 :** 3 stambuk terakhir

b) Ikuti instruksi dibawah ini :

I. Buatlah sebuah class dengan nama **Mahasiswa** yang memiliki beberapa field yang memiliki *access modifier* **private :**
   a. **Stambuk** : tipe data **string**
   b. **Nama :** tipe data **string**
   c. **Angkatan :** tipe data **int**
   d. **Kelas :** tipe data string

II. Buat objek dari class vector dengan tipe **Mahasiswa (vector<Mahasiswa>)** dengan nama **vMhs.**

III. Buat menu tampilan sesuai output di bawah yang terdiri dari Input Data, Tamil data, Ciri Data, Hapus Data, dan Keluar dari program :

IV. Pada menu Input Data, dilakukan penginputan semua data field pada object Mahasiswa.

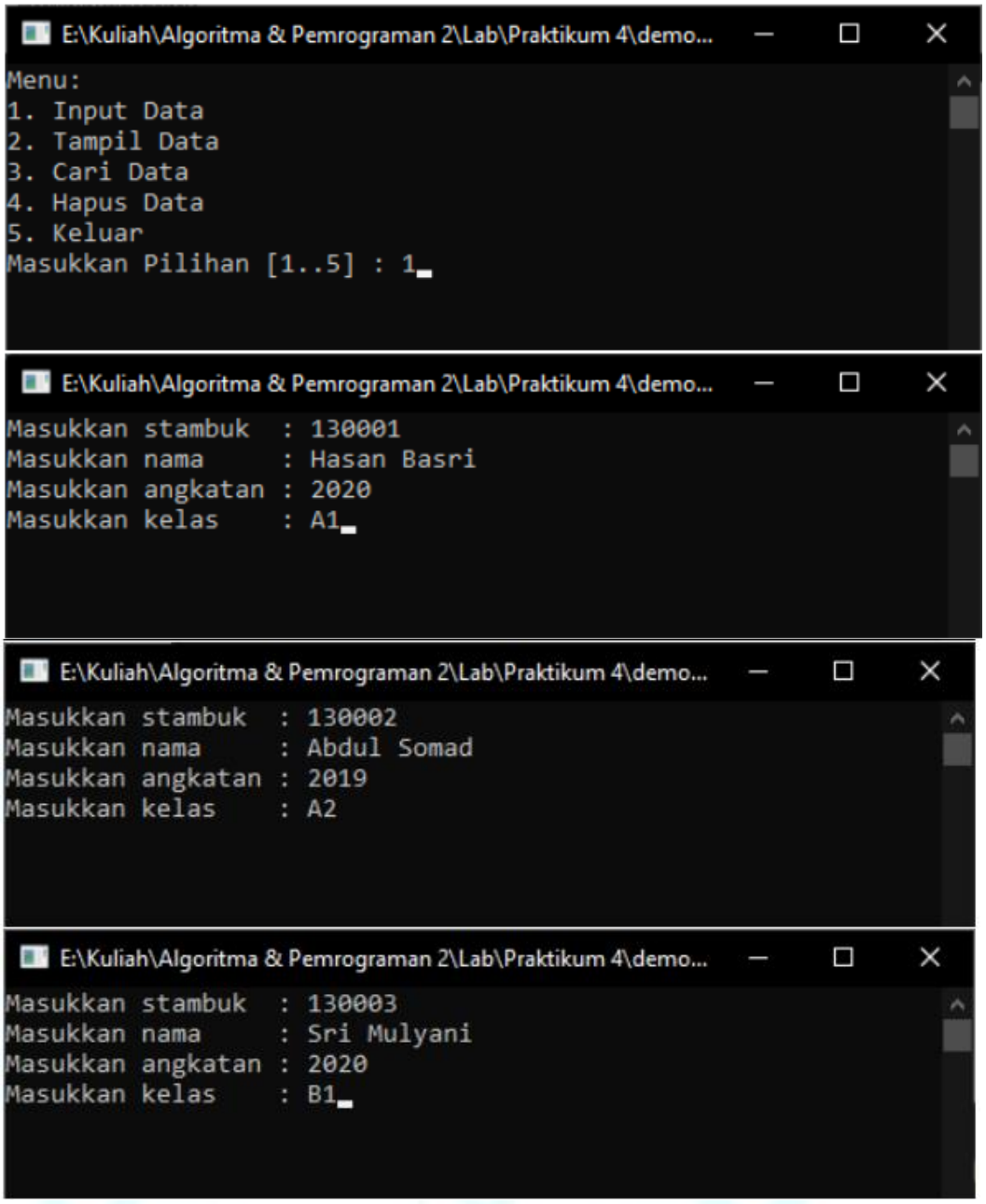
V. Pada menu Tampil Data, menampilkan semua data yang telah diinputkan melalui menu Input Data.

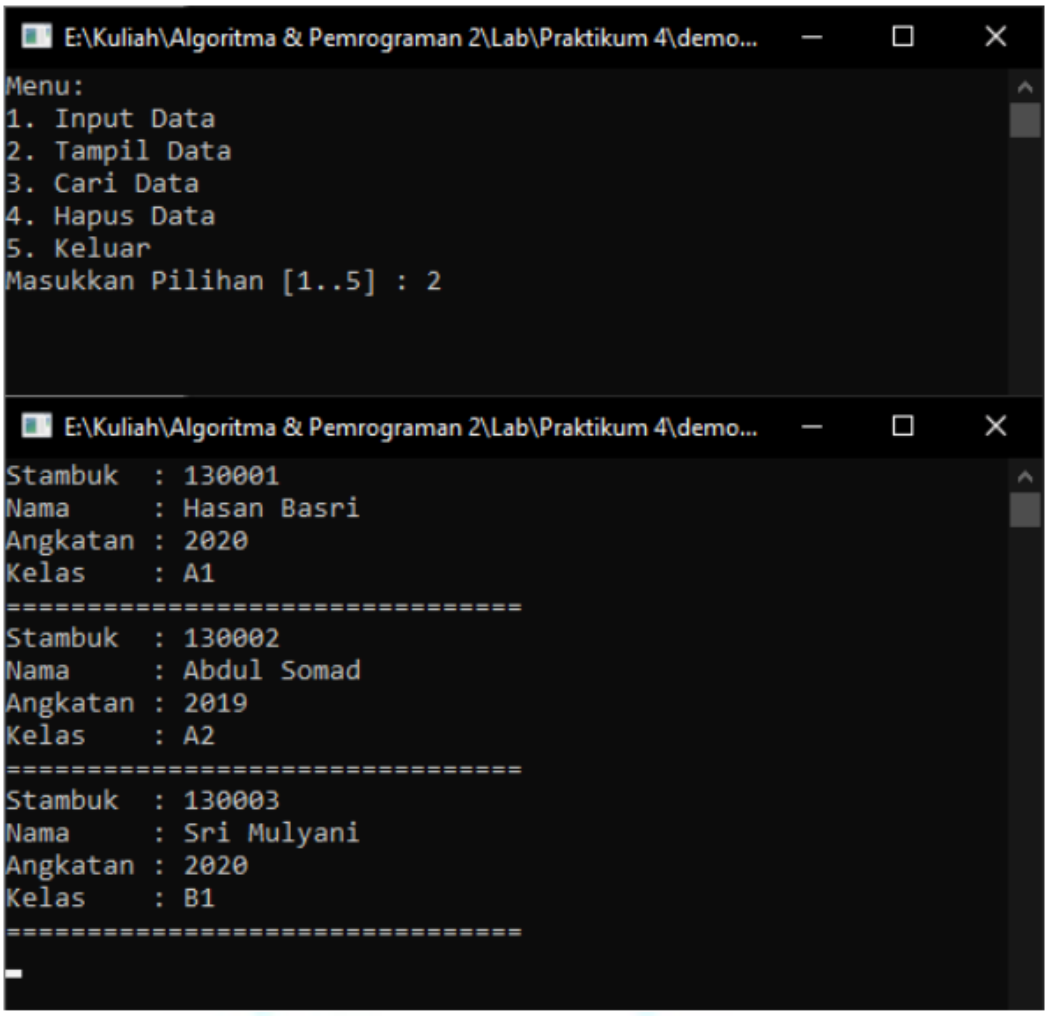
VI. Pada menu Cari data, dilakaukan pencarian data berdasarkan Stambuk. Jika data ditemukan, maka ditampilkan seluruh deatil data mahasiswanya. Jika data tidak ditemukan, maka tampilan pesan “Data tidak ditemukan”.

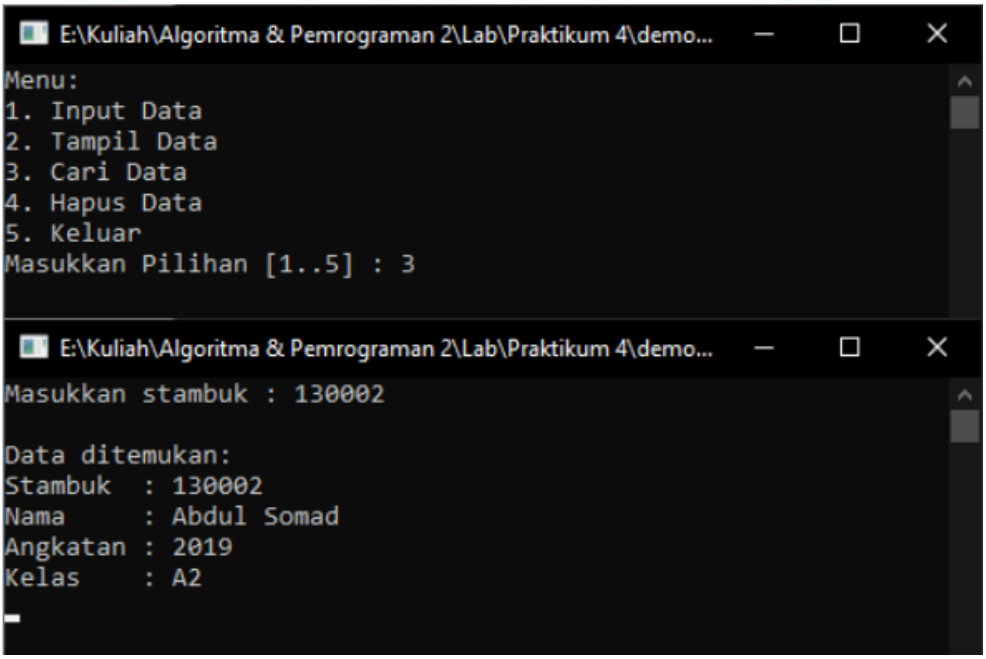
VII. Pada menu Hapus Data, penhapusan data dilakukan berdasarkan stambuk yang dimasukkan.

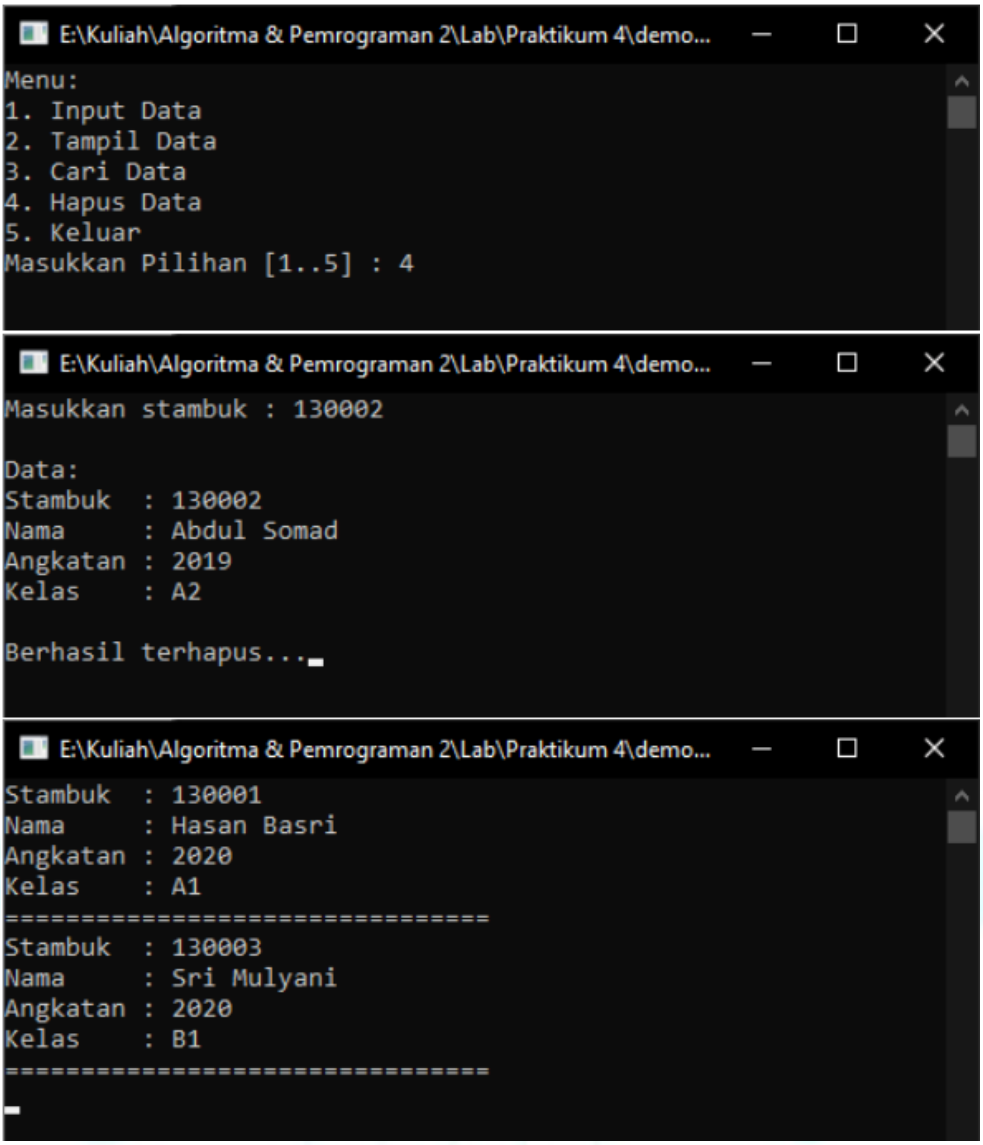
c) Kemuidan *save* program yang telah diketik pada folder yang telah dibuat sebelumnya pada modul 1, dengan menekan *ctrl+s*

d) Kemudian, apabila telah selesai menuliskan kode diatas maka anda dapat menekan tombol *compile* atau shortcode pada keyboard : *F9*


E:\Kuliah\Algoritma & Pemrograman 2\Lab\Praktikum 4\demo...
Menu:
1. Input Data
2. Tampil Data
3. Cari Data
4. Hapus Data
5. Keluar
Masukkan Pilihan [1..5] : 1
E:\Kuliah\Algoritma & Pemrograman 2\Lab\Praktikum 4\demo...
Masukkan stambuk : 130001
Masukkan nama : Hasan Basri
Masukkan angkatan : 2020
Masukkan kelas : A1
E:\Kuliah\Algoritma & Pemrograman 2\Lab\Praktikum 4\demo...
Masukkan stambuk : 130002
Masukkan nama : Abdul Somad
Masukkan angkatan : 2019
Masukkan kelas : A2
E:\Kuliah\Algoritma & Pemrograman 2\Lab\Praktikum 4\demo...
Masukkan stambuk : 130003
Masukkan nama : Sri Mulyani
Masukkan angkatan : 2020
Masukkan kelas : B1

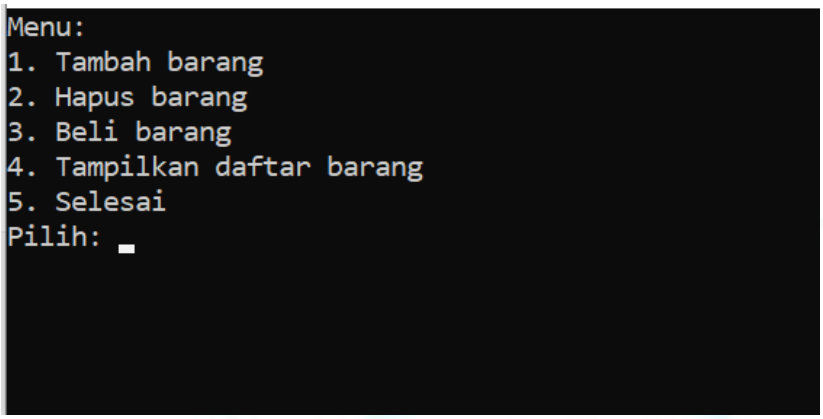

E:\Kuliah\Algoritma & Pemrograman 2\Lab\Praktikum 4\demo...
Menu:
1. Input Data
2. Tampil Data
3. Cari Data
4. Hapus Data
5. Keluar
Masukkan Pilihan [1..5] : 2
E:\Kuliah\Algoritma & Pemrograman 2\Lab\Praktikum 4\demo...
Stambuk : 130001
Nama : Hasan Basri
Angkatan : 2020
Kelas : A1
==========
Stambuk : 130002
Nama : Abdul Somad
Angkatan : 2019
Kelas : A2
==========
Stambuk : 130003
Nama : Sri Mulyani
Angkatan : 2020
Kelas : B1
==========


E:\Kuliah\Algoritma & Pemrograman 2\Lab\Praktikum 4\demo...
Menu:
1. Input Data
2. Tampil Data
3. Cari Data
4. Hapus Data
5. Keluar
Masukkan Pilihan [1..5] : 3
E:\Kuliah\Algoritma & Pemrograman 2\Lab\Praktikum 4\demo...
Masukkan stambuk : 130002
Data ditemukan:
Stambuk : 130002
Nama : Abdul Somad
Angkatan : 2019
Kelas : A2


E:\Kuliah\Algoritma & Pemrograman 2\Lab\Praktikum 4\demo...
Menu:
1. Input Data
2. Tampil Data
3. Cari Data
4. Hapus Data
5. Keluar
Masukkan Pilihan [1..5] : 4
E:\Kuliah\Algoritma & Pemrograman 2\Lab\Praktikum 4\demo...
Masukkan stambuk : 130002
Data:
Stambuk : 130002
Nama : Abdul Somad
Angkatan : 2019
Kelas : A2
Berhasil terhapus...
E:\Kuliah\Algoritma & Pemrograman 2\Lab\Praktikum 4\demo...
Stambuk : 130001
Nama : Hasan Basri
Angkatan : 2020
Kelas : A1
==========
Stambuk : 130003
Nama : Sri Mulyani
Angkatan : 2020
Kelas : B1
==========

LEMBAR EVALUASI PRAKTIKUM

1. Buatlah program yang menggunakan Vector class untuk menyimpan data barang pada sebuah toko. Setiap barang memiliki nama, harga, dan jumlah stok. Program ini harus dapat menambahkan barang baru ke dalam Vector, menghapus barang dari Vector, mengurangi stok barang saat ada pembelian, serta menampilkan daftar barang beserta harga dan jumlah stok mereka.



Evaluasi Praktikum 8:

No	Indikator	Skor Penilaian				
		Sangat Kurang (E) <=40	Kurang (D) 41-55	Cukup (C) 55-65	Baik (B) 66-85	Sangat Baik (A) >=86
1.	Dapat memahami penggunaan vector pada C++					
2.	Dapat membuat program C++ dengan menggunakan vector					

Catatan Asisten:

Asisten 1 : _____

Asisten 2 : _____