MODUL PRAKTIKUM
STRUKTUR DATA

REVISI KEDUA

Disusun Oleh:

Shandi Noris, M.Kom

TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS PAMULANG

Jl. Surya Kencana No. 1 Pamulang Telp (021)7412566, Fax. (021)7412566
Tangerang Selatan – Banten

TATA TERTIB PRAKTIKUM

1) Peserta praktikum sudah terdaftar sebagai mahasiswa Teknik Informatika Universitas Pamulang pada semester bersangkutan dengan nama sudah tertera pada lembar kehadiran dan bersedia menjalankan tata tertib yang sudah ditentukan.
2) Praktikum dilaksanakan sesuai dengan jadwal yang telah ditentukan.
3) Peserta praktikum wajib hadir tepat pada waktunya.
4) Peserta praktikum diharuskan menandatangani lembar kehadiran.
5) Semua peserta praktikum harus memiliki modul praktikum.
6) Selama praktikum berlangsung diwajibkan :
   - Memelihara suasana agar nyaman dan tenang
   - Tidak Membawa makanan dan minuman
   - Tidak Merokok dan mengotori ruangan
   - Tidak Bersuara dengan keras
   - Tidak Hilir mudik yang tidak perlu
   - Tidak Bertindak atau berbicara yang tidak ada hubungannya dengan kegiatan praktikum
   - Tidak diperkenankan bermain Game
   - Tidak mencoret-coret sarana yang ada
   - Tidak merubah, merusak, atau mengambil peralatan di Laboratorium
   - Meletakan tas ditempat yang ditentukan oleh Pengajar
7) Pakaian yang dikenakan peserta praktikum sebagai berikut:
   - Pria
     - Celana Panjang, tidak robek
     - Sepatu Tertutup (bukan sepatu sandal)
     - Kemeja (Bukan kaos berkerah atau almamater atau sejenisnya)
   - Wanita
     - Kemeja (Bukan kaos berkerah atau almamater atau sejenisnya)
     - Sepatu tertutup (bukan Sepatu sandal)
     - Celana Panjang/Rok

Sanksi-Sanksi :

1) Jika Keterlambatan lebih dari 20 menit tanpa alasan yang dapat diterima, maka peserta praktikum tidak diijinkan mengikuti praktikum.
2) Jika peserta praktikum merusak peralatan praktikum akibat melakukan perbuatan yang tidak berkaitan dengan praktikum/prosedur praktikum maka yang bersangkutan harus mengganti kerugian akibat perbuatannya.
3) Jika Tata tertib tidak ditaati maka peserta praktikum dapat diberi sanksi tegas serta dapat diberi nilai E (Tidak Lulus).

PERTEMUAN I
PENGANTAR

TUJUAN PRAKTIKUM

a) Peserta dapat melaksanakan praktikum, membuat laporan praktikum dengan benar dan mengikuti tata tertib dalam melaksanakan praktikum.

ATURAN PRAKTIKUM

a) Mengikuti Tata tertib praktikum

b) Penilaian :

1) Kehadiran : 30%
   - Minimal kehadiran adalah 11 kali (ujian praktikum termasuk kehadiran). Jika kurang dari 11 kali maka peserta praktikum dinyatakan tidak lulus.
   - Absen Jalandiperbolehkan jika disertai alasan yang kuat (seperti surat dokter, surat dinas, atau sejenisnya) dengan batas maksimal 3 kali. Lembar Absen Jalan diberikan kepada pengajar pada saat Ujian Praktikum.
2) Tugas/Laporan Praktikum : 35%
   Nilai Tugas/Laporan terdiri dari: tugas/quiz, laporan awal, dan laporan akhir.
3) Ujian Praktikum: 35%
   Ujian praktikum dilaksanakan pada pertemuan ke-14 atau pertemuan akhir.
4) Salah satu atau lebih dari komponen nilai di atas tidak ada, maka peserta praktikum dinyatakan tidak lulus.

c) Membuat Laporan Praktikum.

1) Laporan Praktikum terdiri dari Laporan Awal dan Laporan Akhir.
2) Laporan Awal
   - Isi dari laporan awal adalah menulis ulang setiap Modul sesuai dengan pertemuan berikutnya, dan menjawab soal Tugas Pendahuluan.
3) Laporan Akhir
   - Isi dari laporan akhir adalah Menulis kesimpulan pada Modul yang telah dipraktekkan dan menjawab soal Tugas Akhir.
4) Tata Cara Mengumpulkan Laporan
   - Laporan dikumpulkan mulai pertemuan ke-2.

- Pada pertemuan ke-2, hanya laporan awal yang dikumpulkan.
- Pada pertemuan ke-3 sampai dengan pertemuan ke-13, laporan yang dikumpulkan adalah laporan awal dan laporan akhir.
- Pada pertemuan ke-14 atau pertemuan akhir atau pada saat ujian praktikum, hanya laporan akhir yang dikumpulkan.
- Contoh:
  - Pada Pertemuan ke-2, peserta praktikum harus membawa laporan awal yang isinya adalah penulisan ulang modul pertemuan ke-2 dan menjawab soal tugas pendahuluan pertemuan ke-2.
  - Pada Pertemuan ke-3, peserta praktikum harus membawa laporan awal yang isinya adalah penulisan ulang modul pertemuan ke-3 dan menjawab soal tugas pendahuluan pertemuan ke-3, serta membawa laporan akhir pertemuan ke-2 dan menjawab soal tugas akhir pertemuan ke-2.
  - Pada pertemuan ke-4 sampai dengan pertemuan ke-13, cara mengumpulkan laporannya sama dengan pada pertemuan ke-3.
  - Pada pertemuan ke-14 atau pada saat ujian praktikum, peserta praktikum hanya membawa Laporan akhir pertemuan ke-13 dan menjawab soal tugas akhir pertemuan ke-13.
- Contoh jika peserta praktikum Tidak Hadir:
  - Jika pada pertemuan ke-4 peserta praktikum tidak hadir dikarenakan sakit dan pada pertemuan ke-5 peserta praktikum hadir, maka peserta praktikum harus membawa laporan awal yang isinya adalah penulisan ulang modul pertemuan ke-5 dan menjawab soal tugas pendahuluan pertemuan ke-5, serta membawa laporan akhir pertemuan ke-3 dan menjawab soal tugas akhir pertemuan ke-3.

5) Format laporan praktikum :
   - Isi Laporan ditulis tangan pada kertas A4.
   - Halaman depan Laporan Awal dan Laporan Akhir boleh diprint atau ditulis tangan, contoh seperti gambar di bawah ini.

Contoh halaman Depan (diPrint) :

LAPORAN AWAL
STRUKTUR DATA

LAPORAN KE-1

Disusun Oleh :

Nama : Cecep Gorbacep
NIM : 123456789
Kelas : II-A Malam

TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS PAMULANG

Jl. Surya Kencana No. 1 Pamulang Telp (021)7412566, Fax. (021)7412566
Tangerang Selatan– Banten

Contoh halaman Depan (diPrint) :

LAPORAN AKHIR
STRUKTUR DATA

LAPORAN KE-1

Disusun Oleh :

Nama: Cecep Gorbacep
NIM : 123456789
Kelas : II-A Malam

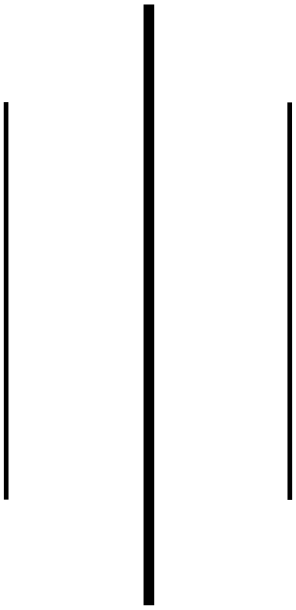
TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS PAMULANG

Jl. Surya Kencana No. 1 Pamulang Telp (021)7412566, Fax. (021)7412566
Tangerang Selatan– Banten

Contoh halaman Depan (diTulis) :

LAPORAN AWAL
STRUKTUR DATA

LAPORAN KE-1



Disusun Oleh :

Nama : Cecep Gorbacep
NIM : 123456789
Kelas : II-A Malam

TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS PAMULANG

Jl. Surya Kencana No. 1 Pamulang Telp (021)7412566, Fax. (021)7412566
Tangerang Selatan– Banten

PERTEMUAN II
ARRAY

TUJUAN PRAKTIKUM

a) Mahasiswa dapat menjelaskan pengertian array dan kegunaan array.
b) Mahasiswa dapat menggunakan array satu dimensi dan dua dimensi dalam penyimpanan data.

TEORI DASAR

a) Pendahuluan

Array merupakan kumpulan elemen yang bertipe sama dalam urutan tertentu yang menggunakan nama yang sama. Letak atau posisi dari elemen array ditunjukan oleh index atau posisi. Dalam beberapa buku array sering juga disebut dengan istilah Larik atau Tabel. Array termasuk dalam struktur data statis, artinya adalah lokasi memoriuntuk suatu array tidak dapat ditambah atau dikurangi selama program dijalankan.

b) Array Satu Dimensi

Dikatakan array satu dimensi karena banyaknya penunjuk indeks hanya satu. Sebelum variabel array digunakan maka variabel array harus dideklarasikan terlebih dahulu.Pendeklarasian variabel array satu dimensi sebenernya hampir sama dengan pendeklarasian variabel yang lain, hanya saja pendeklarasian variabel array diikuti dengan maksimum banyaknya elemen yang dapat disimpan dalam variabel array yang dituliskan dalam pasangan tanda siku penutup. Di dalam bahasa C++, harga awal indeks dimulai dari 0 [nol]. Maka jika dituliskan banyaknya maksimum elemen adalah N, berarti indexs yang akan dugunakan adalah 0,1,2,…,N-1.

Bentuk Umum pendeklarasian array satu dimensi:

```
Tipe_data nama_var_array [ukuran] ;
```

c) Array Dua Dimensi

Array dua dimensi sering digambarkan sebagai sebuah matrik. Merupakan perluasan dari array satu dimensi. Jika array satu dimensi hanya terdiri dari satu baris dan beberapa kolom, maka array dua dimensi terdiri dari beberapa baris dan beberapa kolom. Dengan demikian array dua dimensi tersusun dalam bentuk baris dan kolom, dimana indeks pertama menyatakan baris dan indeks kedua menyatakan kolom.

Sama halnya dengan array satu dimensi sebelum digunakan juga harus dideklarasikan terlebih dahulu. Bentuk umum pendeklarasian array dua dimensi adalah seperti berikut ini:

Tipe_data Nama_var_Array [banyak_baris] [banyak_kolom] ;

TUGAS PRAKTIKUM

a) Buatlah program untuk menginisialisasikan10 bilangan bulat kemudian hitung jumlah dan rata (simpan dengan nama lat2_1.cpp)

```
#include <iostream.h>
#include <conio.h>
using namespace std;
int main()
{
int Nilai[10] = {8, 12, 20, 15, 7, 5, 6, 4, 32, 3} ;
int i, Jumlah = 0;
float Rata_Rata;

//Menghitung jumlah
for (i=0 ; i <10 ; i++)
        Jumlah += Nilai [ i ];
Rata_Rata =( float ) Jumlah / 10;
//Mencetak Elemen Array
cout<<"\n\nDeretan Bilangan      = ";
for ( i=0;i<10;i++)
        cout<<Nilai[i]<<" " ;

//Mencetak Harga Jumlah
cout<<"\nJumlah Bilangan  ="<<Jumlah;
cout<<"\nRata-Rata Bilangan       = "<<Rata_Rata;
getch ();
}
```

b) Buatlah program untuk membaca 10 bilangan bulat kemudian hitung jumlah rata-rata(simpan dengan nama lat2_2.cpp)

```
#include <iostream.h>
#include <conio.h>
using namespace std;
int main()
{
int Nilai[10] ;
int i, Jumlah = 0;
float Rata_Rata;

// membaca dan menghitung jumlah
for (i=0 ; i<10 ; i++)
{
  cout<<"Masukan elemen ke-"<<i<<" = ";
  cin>>Nilai [ i ] ;
  Jumlah+=Nilai [ i ];
}

Rata_Rata= ( float ) Jumlah / 10;

//Mencetak Elemen Array
cout<<"\n\nDeretan Bilangan       = ";
for ( i=0;i<10;i++)
{
  cout<<Nilai[i]<<" " ;
}

//Mencetak Harga Jumlah
cout<<"\nJumlah Bilangan  ="<<Jumlah;
cout<<"\nRata-Rata Bilangan       = "<<Rata_Rata;
getch ();
}
```

c) Buatlah program untuk membaca sederetan bilangan bulat kemudian tentukan bilangan terbesar dari sederetan bilangan tersebut. (simpan dengan nama lat2_3.cpp)

```
#include<iostream.h>
#include<conio.h>
main()
{
int Nilai [20];
int i, N, Terbesar;
cout<<"Masukan Banyaknya Bilangan = ";
cin>>N;
cout<<endl;

//Membaca elemen array
for(i=0; i<N; i++)
{
  cout<<"Masukan elemen ke-"<<i<<"=";
  cin>>Nilai [i];
}
Terbesar = Nilai [1]; //elemen pertama dibuat sebagai Terbesar
for (i=1; i<N; i++)
{
        if (Nilai [i]>Terbesar)
                Terbesar = Nilai [i];
}

//Mencetak Bilangan Terbesar
cout<<"\nBilangan Terbesar = "<< Terbesar;
getch();
```

d) Buatlah program untuk Array Satu dimensi dari media masukan (simpan dengan nama lat2_4.cpp)

```
#include<iostream.h>
#include<conio.h>
main()
{
int Nilai [ 20 ];
int Posisi [ 20 ];
int i, N, Bilangan, Banyak=0;
bool ketemu;
cout<<"Masukan Banyaknya Bilangan = ";
cin>>N;
cout<<endl;
```

```
//Membaca elemen Array
for (i=0; i<N ;i++)
{
  cout<<"Masukan elemen ke-"<<i<<" = ";
  cin>>Nilai [i];
}

//Membaca Elemen Array
cout<<"\n\nDeretan Bilangan = ";
for (i=0; i<N; i++)
  cout<<Nilai [i]<<" ";

cout<<"\n\nMasukan Bilangan yang akan dicabut = ";
cin>>Bilangan ;

//melakukan pencarian
for(i=0; i<N; i++)
{
  if(Nilai [i]==Bilangan)
  {
    ketemu = true;
    Posisi [Banyak]=i;
    Banyak++;
  }
}
if(ketemu)
{
  cout<<"Bilangan "<< Bilangan<<" ditentukan sebanyak "<< Banyak;
  cout<<"\npada posisi ke = ";
  for(i=0; i<Banyak ; i++)
cout<<Posisi [i]<<" ";
}
else
  cout<<"Bilangan "<<Bilangan<<"tidak ditemukan";
getch ();
}
```

e) Buatlah program untuk mencetak elemen-elemen Matriks A berukuran 3x4 yang telah diinisialisasikan. Contoh output sebagai berikut (simpan dengan nama lat2_5.cpp)

```
1  34 5
2  4  6  8
3  5  7  9
```

TUGAS PENDAHULUAN

1. Apa yang dimaksud dengan Array!
2. Jelaskan perbedaan Array dengan Variabel biasa!
3. Jelaskan perbedaan Array Satu dimensi, Dua Dimensi, dan Tiga Dimensi!
4. Buatlah contoh program sederhana menggunakan Array Tiga Dimensi!

TUGAS AKHIR

1. Buatlah program untuk menjumlahkan 2 buah matriks, masing masing matriks mempunyai ordo yang sama!

PERTEMUAN III
STRUCTURE

TUJUAN PRAKTIKUM

a) Mahasiswa dapat menjelaskan pengertian struct.
b) Mahasiswa dapat menggunakan struct.
c) Mahasiswa dapat menggunakan struct dalam array [array of struct].

TEORI DASAR

a) Pendahuluan

Structure [struktur] adalah kumpulan elemen data yang di gabungkan menjadi satu kesatuan. Dengan katalain, structure merupakan bentuk struktur data yang dapat menyimpan variabel dengan satu nama. Masing-masing elemen data dikenal dengan sebutan *field.*

b) Deklarasi Structure

Pendeklarasian structure selalu diawali kata baku **struct** diikuti nama structure dan deklarasi field-field yang membangun structure di antara pasngan tanda titik koma [;].jika terdapat field dengan tipe data yang sama.

Bentuk umum:

```
Struct nama_struct
{
    <tipe_data>   nama_field_1;
    <tipe_data>   nama_field_2;
    ……..
    <tipe_data>   nama_field_n;
};
```

c) Pemakaian Structure

Untuk menggunakan structure dapat dilakukan dengan menuliskan nama structure yang diikuti dengan nama fieldnya yang dipisahkan dengan titik [.] atau tanda panah [- >].

d) Structure dalam Structure

Suatu struktur juga dapat mengandung suatu struktur yang lain Artinya field-field dalam structure merupakan suatu structure juga. Misalkan biodata Mahasiswa yang terdiri dari NIM, Nama, Alamat dan Tanggal Lahir. Alamat terdiri dari Nama jalan, Kota dan Kode Pos.

e) Array dalam Structure

Suatu structure dideklarasikan menjadi sebuah array apabila hendak menggunakan satu struct untuk beberapa kali. Sebenarnya sama dengan struktur tunggal di atas, perbedaannya adalah hanya pada saat pendeklarasian variabel saja.

Contoh:

```
Struct Mahasiswa {
    Char NIM [ 9 ];
    Char Nama [ 25 ];
    Char Alamat [ 30 ];
    Float Ipk;
};
```

Kemudian deklarasikan variabel bertipe struct di atas.

```
Mahasiswa Mhs [ 5 ] ;
```

TUGAS PRAKTIKUM

a) Buatlah program untuk membaca biodata yang diinput Nim, Nama, Alamat, dan Umur kemudian cetak. (simpan dengan nama lat3_1.cpp)

```
#include<stdio.h>
#include<conio.h>
#include<iostream.h>
struct Mahasiswa
{
char Nim [9] ;
char Nama [25] ;
char Alamat[40] ;
short Umur ;
};
```

```
main ()
{
Mahasiswa Mhs;
cout<<"Nim            : " ;
cin.getline (Mhs.Nim,9);
cout<<"Nama                  :";
cin.getline (Mhs.Nama,25);
cout<<"Alamat        :";
cin.getline (Mhs.Alamat,40);
cout<<"Umur          : ";
cin>>Mhs.Umur ;
cout<<"\n\n\nNim   : "<< Mhs.Nim;
cout<<"\nNama              : "<< Mhs.Nama;
cout<<"\nAlamat        : "<< Mhs.Alamat;
cout<<"\nUmur              : "<< Mhs.Umur;
getch ();
}
```

b) Buatlah program menggunakan ketiga structure di dalam strucure (simpan dengan nama lat3_2.cpp

```
#include<stdio.h>
#include<iostream.h>
#include<conio.h>
struct Tinggal {
  char Jaln [40] ;
  char Kota [15] ;
  char Pos [5] ;
};
struct Tgl_Lahir {
  int Tanggal ;
  int Bulan ;
  int Tahun ;
};
struct Mahasiswa {
  char Nim [9];
  char Nama [25];
  Tinggal Alamat;
  Tgl_Lahir Lahir;
};
```

```
int main ( )
{
  Mahasiswa Mhs ;
  cout<<"NIM                  : "; cin.getline (Mhs.Nim, 9 ) ;
  cout<<"Nama                 : "; cin.getline (Mhs.Nama, 25 ) ;
  cout<<"Alamat      : \n" ;
  cout<<"\tJalan      : "; cin.getline ( Mhs.Alamat.Jaln, 40 ) ;
  cout<<"\tKota       : "; cin.getline ( Mhs.Alamat.Kota, 15 ) ;
  cout<<"\tKode pos          : "; cin.getline ( Mhs.Alamat.Pos, 5 ) ;
  cout<<"Tanggal Lahir : \n";
  cout<<"\tTanggal  : "; cin>> Mhs.Lahir.Tanggal ;
  cout<<"\tBulan     : "; cin>>Mhs.Lahir.Bulan ;
  cout<<"\tTahun     : "; cin>>Mhs.Lahir.Tahun ;
  cout<<"\n\nMencetak Kembali Nilai Anggota\n\n";
  cout<<"NIM                  : "<<Mhs.Nim ;
  cout<<"\nNama    : "<<Mhs.Nama ;
  cout<<"\nAlamat  : \n";
  cout<<"\n\tJalan   : "<<Mhs.Alamat.Jaln;
  cout<<"\n\tKota    : "<<Mhs.Alamat.Kota;
  cout<<"\n\tKode Pos       : "<<Mhs.Alamat.Pos;
  cout<<"\nTanggal Lahir    : "<<Mhs.Lahir.Tanggal<<"-";
  cout<<Mhs.Lahir.Bulan<<"-"<<Mhs.Lahir.Tahun;
  getch ();
}
```

c) Buatlah program array dalam struktur (simpan dengan nama lat3_3.cpp)

```
#include<stdio.h>
#include<conio.h>
#include<iostream>

struct Mahasiswa
{
  char Nim [9] ;
  char Nama [25] ;
  char Alamat [40] ;
  int Umur ;
} ;

main ()
{
```

```
Mahasiswa Mhs [5] ;
  int i ;
  for (i =0; i<5; i++ )
  {
    cout<<"Nim                : ";
    cin>>Mhs[i].Nim ;
    cout <<"Nama              :";
    cin>>Mhs[i].Nama ;
    cout<<"Alamat             : ";
    cin>>Mhs[i].Alamat ;
    cout<<"Umur               : ";
    cin>>Mhs[i].Umur ;
  }
  for( i =0; i<5; i++ )
  {
    cout<<"\n\nNim                    : "<<Mhs [i].Nim;
    cout<<"\n\nNama                   : "<<Mhs [i].Nama;
    cout<<"\n\nAlamat                 : "<<Mhs [i].Alamat;
cout<<"\n\nUmur              : "<<Mhs [i].Umur;
  }
  getch ();
}
```

d) Buatlah program array dalam struktur (simpan dengan nama lat3_4.cpp)

```
#include<stdio.h>
#include<iostream>
#include<conio.h>
struct Tinggal {
  char Jalan [ 40 ];
  char Kota [ 15 ];
  char Pos [ 5 ] ;
        };
struct Tgl_Lahir {
  int Tanggal;
  int Bulan ;
  int Tahun ;
  };
```

```
struct Mahasiswa {
  char Nim [ 9 ];
  char Nama [ 25 ];
Tinggal Alamat ;
  Tgl_Lahir Lahir ;
  };
main ()
{
  Mahasiswa Mhs [5] ;
  int i ;
  for (i=0; i<3; i++)
  {
    cout<<"NIM              : "; cin>>Mhs[i].Nim ;
    cout<<"Nama             : "; cin>>Mhs[i].Nama;
    cout<<"Alamat   :\n";
    cout<<"\tJalan          : "; cin>>Mhs[i].Alamat. Jalan;
cout<<"\tKota               : "; cin>>Mhs[i ]. Alamat. Kota ;
    cout<<"\tKode Pos       : "; cin>>Mhs[i ]. Alamat. Pos;
    cout<<"Tanggal Lahir    : \n";
    cout<<"\tTanggal        : "; cin>>Mhs [i ]. Lahir. Tanggal;
    cout<<"\tBulan          : "; cin>>Mhs [i ]. Lahir. Bulan;
cout<<"\tJalan              : "; cin>>Mhs[i ]. Lahir. Tahun ;
  }
  cout<<"\n\nMencetak Kembali Nilai Anggota\n\n";
  for (i=0; i<3; i++)
  {
    cout<<"\nNIM             : "<<Mhs [ i ]. Nim ;
    cout<<"\nNama            : "<<Mhs [ i ]. Nama ;
  cout<<"\nAlamat   : \n" ;
    cout<<"\tJalan          : "<<Mhs[i ].Alamat. Jalan;
    cout<<"\tKota           : "<<Mhs[i ]. Alamat. Kota ;
    cout<<"\tKode Pos       : "<<Mhs[i ]. Alamat. Pos;
    cout<<"\nTanggal Lahir : "<<Mhs[i ]. Lahir. Tanggal<<"-";
    cout<<Mhs [ i ]. Lahir.Bulan <<"-"<<Mhs[i ]. Lahir. Tahun ;
  }
  getch ();
}
```

e) Buatlah program untuk struktur NILAI yang terdiri dari NIM, Nama, Nilai_Tugas, Nilai_UTS, Nilai_UAS, Nilai_Akhir, Nilai_Huruf.

Ketentuan:

1. Nilai_Akhir : 20% x Nilai_Tugas + 35% x Nilai_UTS + 45% x Nilai_UAS
2. Nilai Huruf : $85 < Nilai\_Akhir \leq 100 \rightarrow A$
   $70 < Nilai\_Akhir \leq 85 \rightarrow B$
   $55 < Nilai\_Akhir \leq 70 \rightarrow C$
   $40 < Nilai\_Akhir \leq 55 \rightarrow D$
   $0 < Nilai\_Akhir \leq 40 \rightarrow E$

(simpan dengan nama lat3_5.cpp)

TUGAS PENDAHULUAN

1. Apa yang dimaksud dengan Structure!
2. Jelaskan kelebihan sebuah program menggunakan Structure!
3. Buatlah contoh program sederhana menggunakan Structure!
4. Buatlah contoh program sederhana kombinasi Array dan Structure!

TUGAS AKHIR

1. Buatlah program untuk membaca dan mencetak biodata pegawai, dengan menggunakan 1) struktur BIODATA yang terdiri dari NIP, Nama, Alamat, Jabatan, Agama, Tanggal_Lahir, Tanggal_Mulai_Kerja, Unit_Kerja. 2) struktur TINGGAL yang terdiri dari Jalan, Kode_Pos, Kota. dan 3) struktur TANGGAL yang terdiri dari Tanggal, Bulan, Tahun. GUNAKAN ARRAY OF STRUCT!!

PERTEMUAN IV
POINTER

TUJUAN PRAKTIKUM

a) Mahasiswa dapat mendeklarasikan Pointer dalam C++.
b) Mahasiswa dapat mengetahui alamat suatu variabel dalam memory dengan C++.
c) Mahasiswa dapat menggunakan pointer dengan Array dalam C++.

TEORI DASAR

a) Pendahuluan

Pointer (variable penunjuk) adalah suatu variable yang berisi alamat memori dari suatu variable lain.Alamat ini merupakan lokasi dari variable lain di dalam memori. Dengan kata lain, pointer berisi alamat dari variabel yang mempunyai nilai tertentu.

b) Operator Pointer

Suatu pointer dapat berisi alamat dari suatu variable lain dan untuk dapat mengakses nilai yang ada di dalam variable berpointer secara langsung dapat dilakukan dengan menggunakan operator. Ada dua oprator pointer yang disediakan oleh Borland C++ yaitu :

1) Operator Deference (&)

Pada umumnya kita tidak dapat menentukan di mana variabel akan di tempatkan dalam memori karena penempatan suatu variabel ditentukan oleh system operasi Untuk suatu keperluan tertentu terkadang kita harus mengetahui alamat suatu variabel di dalam memori, untuk memperoleh alamat dari suatu variabel dapat kita lakukan dengan bentuk

```
Nama_var_pointer = &variabel;
```

2) Operator Reference (*)

Digunakan untuk mengakses secara langsung nilai yang terdapat di dalam alamat yang merupakan nilai dari variabel pointer. Dilakukan dengan

menambahkan Oprator reference [*] didepan nama variabel, agar dapat menerjemahkan nilai sebenarnya dari suatu variabel.

Sebagai contoh, Misalkan deklarasi sebagai berikut.

```
int x=8,y ;
int *xPtr ;
```

c) Deklarasi Pointer

Seperti halnya variabel yang lain, variabel pointer juga harus dideklarasikan terlebih dahulu sebelum digumakan

Bentuk umum:

```
tipe_data *nama_var_pointer;
```

d) Pointer dan Array

Pointer dan array mempunyai hubungan yang dekat. Secara internal array juga menyatakan alamat, dimana pengenalan array sama dengan alamat pada elemen pertama pada array

d) Pointer Pada Pointer

Variabel pointer menunjukan suatu variabel. Di samping itu suatu variabel pointer juga dapat menunjukan ke variabel pointer lainnya. Dalam hal ini dilakukan dengan cara menambahkan operator reference [*] pada variabel yang akan di tunjuk.

Contoh: misalkan kita mempunyai deklarasi sebagai berikut ini.

```
int Var_x;
int *xPtr1;
int **xPtr2;
```

Artinya:

a. Var_x adalah variabel bertipe int
b. *xPtr1 adalah variabel pointer yang menunjukan ke data bertipe int
c. **xPtr2 adalah variabel pointer yang menunjuk ke pointer int

TUGAS PRAKTIKUM

a) Buatlah program untuk menyimpan nilai pada suatu alamatmenggunakan operator Deference. (simpan dengan nama lat4_1.cpp)

```
#include<iostream.h>
#include<conio.h>
main ()
{
  int x=8, y;
  int *xPtr;
  xPtr = &x;
  cout<<"Nilai                x       = "<< x <<endl ;
  cout<<"Alamat     x       = "<<&x<<endl ;
  cout<<"Alamat     x       = "<<xPtr<<endl ;
  cout<<"Nilai yang disimpan pada alamat " ;
  cout<<xPtr<<" adalah "<< *xPtr;
  getch ();
}
```

b) Buatlah program untuk menyimpan nilai pada suatu alamat menggunakan operator reference. (simpan dengan nama lat4_2.cpp)

```
#include<iostream.h>
#include<conio.h>
main ()
{
  int x=8, y;
  int *xPtr;
  xPtr = &x;
  y = *xPtr;
  cout<<"Nilai                x       = "<< x <<endl ;
  cout<<"Alamat     x       = "<<&x<<endl ;
  cout<<"Alamat     x       = "<<xPtr<<endl ;
  cout<<"Nilai yang disimpan pada alamat ";
  cout<<xPtr<<" adalah "<<y;
  getch();
}
```

c) Buatlah program Pointer dan Array (simpan dengan nama lat4_3.cpp)

```
#include<stdio.h>
#include<conio.h>
#include<iostream.h>
main ()
{
  int Nilai [ ] = {45, 23, 50, 8, 12, 10, 15} ;
  int *Ptr_Nilai ;
  int i;
  Ptr_Nilai = Nilai ;
  cout<<"\nNilai Ptr_Nilai"<<Ptr_Nilai;
  cout<<"\nAlamat array Nilai"<<Ptr_Nilai;
  cout<<"\nNilai yang ada pada alamat "<<Ptr_Nilai<<"adalah
"<<*Ptr_Nilai;
  cout<<"\nElemen Array indeks pertama          : "<<Nilai [0];
  cout<<"\n\nElemen Array (dgn Array )  : ";
  for ( i=0;i<7;i++)
    cout<<Nilai [i]<<" "; //mencetak elemen array
  cout<<"\n\nElemen Array (dgn Pointer)          : ";
  for(i=0;i<7;i++)
    cout<<*( Nilai+i )<<" "; //mencetak elemen array getch ( ) ;
}
```

d) Buatlah program Pointer pada Pointer (simpan dengan nama lat4_4.cpp)

```
#include<iostream>
#include<conio.h>
main ()
{
  int x= 8;
  int *xPtr1;
  int **xPtr2;
  xPtr1          = &x;
  xPtr2          = &xPtr1;
  cout<<"Nilai x = "<<x<<endl;
  cout<<"Nilai x = "<<*xPtr1<<endl;
  cout<<"Nilai x = "<<*xPtr2<<endl;
  getch ();
}
```

e) Buatlah program sederhana dengan menggunakan kombinasi antara Pointer dan perintah While (simpan dengan nama lat5_4.cpp)

TUGAS PENDAHULUAN

1. Apa yang dimaksud dengan Pointer!
2. Apa yang dimaksud dengan Variabel Pointer!
3. Sebutkan dan jelaskan jenis-jenis Operator Pointer!
4. Buatlah contoh program sederhana menggunakan Pointer!

TUGAS AKHIR

1. Buatlah program dengan menggunakan kombinasi antara Pointer dan Array Dua Dimensi!

PERTEMUAN V
FUNCTION

TUJUAN PRAKTIKUM

a) Mahasiswa dapat menjelaskan pengertian dan pembuatan fungsi dengan C++.
b) Mahasiswa dapat menjelaskan keuntungan penggunaan fungsi dengan C++.
c) Mahasiswa dapat mengirimkan nilai parameter secara Nulai dan Alamat.
d) Mahasiswa dapat mengirimkan array, struct, dan pointer sebagai parameter fungsi.
e) Mahasiswa dapat mengimplementasikan fungsi dengan C++.

TEORI DASAR

a) Pendahuluan

Fungsi adalah sekumpulan perintah yang dapt menerima argument input dan dapat memberikan hasil output yang dapat berupa nilai ataupun sebuah hasil operasi. Fungsi merupakan suatu bagian dari program yang dimaksudkan untuk mengerjakan suatu tugas tertentu dan letaknya terpisah dari program yang memanggilnya. Fungsi merupakan elemen utama dalam bahasa C karena bahasa C sendiri terbentuk dari kumpulan fungsi. Fungsi terdiri dari dua jenis yaitu fungsi *build-in* dang fungsi *user defined.*

b) Struktur Fungsi

Pengertian deklarasi fungsi berbeda dengan definisi fungsi. Suatu deklarasi fungsi adalah judul fungsi yang sederhana yang diakhiri dengan tanda semicolon (;) atau sering disebut dengan *Prototipe fungsi.*Sedangkan definisi fungsi adalah fungsi yang lengkap terdiri dari judul dan isinya. Suatu deklarasi fungsi disebut juga sebagai prototype fungsi.

Bentuk umum pendeklarasian fungsi adalah:

```
tipe_fungsi nama_fungsi ( parameter_fungsi ) ;
```

Contoh:

```
Float Jumlah ( float a, float b );
```

Sedangkan bentuk umum pendefinisian fungsi adalah :

```
Penemu_tipe_fungsi Nama_fungsi ( parameter_fungsi )
{
        Statment ;
        Statment ;
        ……
        Statement :
}
```

c) Parameter fungsi

Pada saat pemanggilan dan pendefinisian suatu fungsi, terdapat parameter fungsi. Terdapat 2 (dua) jenis parameter,yaitu Parameter Formal dan Parameter Aktual. Cara melewatkan suatu parameter dari parameter aktual ke dalam parameter formal dapat dilakukan dengan dua cara, yaitu: pemanggilan Secara Nilai (Call by Value) dan Pemanggilan Secara Referensi (Call by Reference)

d) Pernyatan Return ()

Pernyataan return () digunakan untuk mengirimkan nilai dari suatu fungsi kepada fungsi lain yang memanggilnya. Pernyataan return [] diikuti oleh argument yang berupa nilai yang akan dikirimkan.

e) Variabel pada Fungsi

Penggolongan Variabel berdasarkan Kelas Penyimpanan (Storage Class) dibagi atas 3 yaitu:Variabel lokal**,** Variabel Eksternal atau Global**,** dan Variabel Statis

f) Pointer sebagai Argumen Fungsi

Pointer biasa digunakan sebagai argumen fungsi jika nilai argumen yang dimaksudkan untuk diubah di dalam fungsi. Hal ini dilakukan dengan cara menambahkan operator & di depan argumen pada parameter aktual dan operator * di depan argumen pada parameter formal

TUGAS PRAKTIKUM

a) Buatlah program fungsi dengan call by value. (simpan dengan namalat5_1.cpp)

```
#include<iostream.h>
#include<conio.h>
void Tukar ( int a,  int  b);
main()
{
  int  a=8, b=-5;
  cout<<"Nilai a dan b sebelumnya : "<<a<<" & "<<b;
  Tukar (a, b);
  cout<<"\nNilai a dan b Setelah ditukar : "<<a<<" & "<<b;
  getch();
}

void Tukar (int x, int y)
{
  int z ;
  z  =  x ;
  x  =  y ;
  y  =  z ;
}
```

b) Buatlah program fungsi dengan Call by Reference (simpan dengan nama lat5_2.cpp)

```
#include<iostream.h>
#include<conio.h>

void Tukar (int &a, int &b) ;
main ( )
{
  int a=8, b=-5 ;
  cout<<"Nilai a dan b sebelum ditukar : "<<a<<" & "<<b;
  Tukar (a, b);
  cout<<"\nNilai a dan b Setelah ditukar : "<<a<<" & "<<b;
```

```
  getch ();
}

void Tukar ( int &x, int &y)
{
        int z ;
        z = x ;
        x = y ;
        y = z ;
}
```

c) Buatlah program fungsi dengan pernyataan Return (simpan dengan nama lat5_3.cpp)

```
#include<iostream.h>
#include<conio.h>

int Maksimum ( int a, int b, int c );
main ()
{
  int a =8, b=12, c=-5;
  cout<<"Nilai a = "<<a;
  cout<<"\nNilai b = "<<b;
  cout<<"\nNilai c = "<<c;
  cout<<"\nNilai Terbesar :  "<<Maksimum ( a, b, c ) ;
  getch ();
}

int Maksimum ( int x, int y, int z )
{
  int Besar = x;
  if ( y > Besar )
    Besar = y ;
  if ( z > Besar )
    Besar = z;
  return (Besar) ;
}
```

d) Buatlah program Fungsi dengan Pointer (simpan dengan nama lat5_4.cpp)

```
#include<iostream.h>
#include<conio.h>

void Tukar (int *a, int *b);
main ()
{
  int a=8, b=-5;
  cout<<"Nilai a dan b sebelum ditukar    : "<<a<<" & "<<b;
  Tukar (&a, &b);
  cout<<"\nNilai a dan b setelah ditukar  : "<<a<<" & "<<b;
  getch ();
}

void Tukar (int *x, int*y)
{
  int z;
  z = *x;
  *x = *y;
  *y = z ;
}
```

e) Buatlah program Array satu dimensi sebagai Argumen fungsi (simpan dengan nama lat5_5.cpp)

```
#include<stdio.h>
#include<iostream.h>
#include<iomanip.h>
#include<conio.h>
const int N=100 ;
void Baca (int Nilai [ ], int &M) ;
void Cetak (int Nilai [ N ], int &M);
void Jumlah (int Nilai [ N ], int &M, int &Jlh, float &Rata) ;
main ()
{
  int M;
  int Nilai [N];
  int Jlh =0;
  float Rata;
  cout<<"Banyak Elemen : ";
```

```
Baca (Nilai, M); //memanggil fungsi Baca
  cout<<"\nElemen Elemen :";
  Cetak(Nilai,M); //memanggil fungsi cetak
  Jumlah (Nilai, M, Jlh, Rata) ; //memanggil fungsi Jumlah
  cout<<"\Jumlah Bilangan          : "<<Jlh;
  cout<<"\nRata-Rata Bilangan :"<<Rata;
  getch ( ) ;
}

void Cetak (int Nilai [], int &M)
{
  int i;
  for(i=0;i<M;i++)
  {
    cout<<setw (3) <<Nilai [ i ] ;
  }
}

void Baca (int Nilai [], int & M)
{
  int i ;
  for (i=0;i<M;i++)
  {
    cout<<"Elemen ke-"<<i<<" : ";
    cin>>Nilai [i];
  }
}

void Jumlah (int Nilai [], int &M, int &Jlh, float &Rata)
{
  int i;
  for (i=0;i<M;i++)
    Jlh+=Nilai [i] ;
  Rata = (float) Jlh /M;
}
```

f) Buatlah program structure sebagai Argumen Fungsi (simpan dengan nama Lat5_6.cpp)

```
#include<stdio.h>
#include<conio.h>
#include<iostream.h>
struct Mahasiswa
{
  char Nim [ 9 ] ;
  char Nama [ 25 ] ;
  char Alamat [ 40 ] ;
  short Umur ;
};
void Baca (Mahasiswa &Mhs);
void Cetak (Mahasiswa &Mhs);
main ( )
{
  Mahasiswa Mhs;
  cout<<"Membaca Nilai Anggota Struktur \n";
  Baca (Mhs);
  cout<<"\nMencetak Nilai Anggota Struktur ";
  Cetak (Mhs);
  getch ( );
}

void Baca(Mahasiswa &Mhs)
{
  cout<<"NIM          : ";
  cin.getline(Mhs.Nim, 9) ;
  cout<<"Nama  :";
  cin.getline(Mhs.Nama,25) ;
  cout<<"Alamat  :";
  cin.getline(Mhs.Alamat, 40);
  cout<<"Umur        :";
  cin>>Mhs.Umur;
}
void Cetak (Mahasiswa &Mhs)
{
  cout<<"\nNim       : "<< Mhs.Nim;
  cout<<"\nNama     :"<< Mhs.Nama;
  cout<<"\nAlamat   :"<< Mhs.Alamat;
  cout<<"\nUmur     :"<< Mhs.Umur;
}
```

g) Berdasarkan program Lat5 _6.cpp, ubah program tersebut menggunakan Pointer (simpan dengan nama Lat6_6.cpp)

TUGAS PENDAHULUAN

1. Apa yang dimaksud dengan Fungsi!
2. Jelaskan yang dimaksud dengan Pemanggilan secara nilai (Call by Value) dan Pemanggilan secara referensi (Call by Reference)!
3. Jelaskan yang dimaksud dengan Variabel lokal**,** Variabel Eksternal atau Global**,** dan Variabel Statis!
4. Buatlah contoh program sederhana menggunakan Fungsi!

TUGAS AKHIR

1. Buatlah program dengan menggunakan kombinasi antara Fungsi dan Pointer!

PERTEMUAN VI
SEARCHING

TUJUAN PRAKTIKUM

a) Mahasiswa dapat menjelaskan pengertian Serching dengan C++
b) Mahasiswa dapat Serching untuk beberapa metode yang ada dengan C++
c) Mahasiswa dapat mengimplementasikan Queue dengan C++

TEORI DASAR

a) Pendahuluan

Pencarian merupakan proses yang mendasar di dalam pemrograman. Pencarian (*Searching)* merupakan tindakan untuk mendapatkan suatu data dalam kumpulan data berdasarkan suatu kunci (*key*) atau acuan data.Dalam kehidupan sehari-hari seringkali kita berurusan dengan pencarian; misalnya menemukan nomor telepon seseorang pada buku telepon atau mencari istilah dalam kamus, dan masih banyak lagi. Pada aplikasi komputer pencarian kerapkali dilakukan. Misalnya untuk peroses penghapusan data/record atau mengubah data/record tertentu didalam suatu tabel atau file.

b) Sequensial Search

Sequensial Search (pencarian beruntun) adalah metode pencarian yang paling mudah. Pencarian berurutan adalah proses membandingkan setiap elemen array satu per satu secara berurutan yang dimulai dari elemen pertama hingga elemen yang dicari ditemukan atau hingga elemen terakhir dari array. Pencarian beruntun dapat dilakukan terhadap elemen array yang belum terurut atau terhadap elemen array yang terurut.

Dengan kata lain sequential search akan mencari data dengan cara membandingkannya satu-persatu dengan data yang ada. Metode ini di sarankan untuk digunakan pada data yang sedikit saja.

c) Binary Search

Binary search adalah metode pencarian suatu data atau elemen di dalam suatu array dengan kondisi data dalam keadaan terurut. Proses pencarian binary search hanya dapat dilakukan pada kumpulan data yang sudah diurutkan terlebih dahulu (menaik atau menurun). Prinsip dari binary search terhadap N elemen dapat dijelaskan seperti berikut :

1) Tentukan posisi awal = 0 dan posisi akhir = N-1.
2) Hitung posisi tengah = [posisi awal + posisi akhir] /2
3) Bandingkan data yang dicari dengan elemen posisi tengah.
   - Jika sama maka catat posisi dan cetak kemudian berhenti.
   - Jika lebih besar maka akan dilakukan pencarian kembali ke bagian kanan dengan posisi awal = posisi tengah + 1 dan posisi akhir tetap kemudian ulangi mulai poin 2.
   - Jika lebih kecil maka akan dilakukan pencarian kembali ke bagian kiri dengan nilai posisi awal tetap dan nilai posisi akhir = posisi tengah -1 kemudian ulangi mulai poin 2.

TUGAS PRAKTIKUM

a) Buatlah program Sequential Search (simpan dengan nama lat6_1.cpp)

```
#include<iostream.h>
#include<conio.h>
main ()
{
  int Nilai [ 20 ] ;
  int i, N, angka, Bilangan ;
  cout<<"Masukan Banyaknya Bilangan =";
  cin>>N;

  //Membaca elemen array
  for(i=0; i<N; i++)
  {
    cout<<"Masukan elemen ke-"<<i<<" = ";
    cin>>Nilai [ i ] ;
  }
```

```
//Mencetak elemen array
  cout<<"\n\nDeretan Bilangan =";
  for(i=0; i<N; i++)
    cout<<Nilai [ i ]<<" ";
  cout<<"\n\nMasukan Bilangan yang akan dicari  = ";
  cin>>Bilangan ;

  //melakukan pencarian
  i=0;
  do
  {
    if(Nilai [ i ]==Bilangan)
      angka = Nilai [i];
    i++;
  }
  while (i<N);
  if (angka==Bilangan)
    cout<<"Bilangan "<<Bilangan <<" ditemukan ";
  else
    cout<<"Bilangan "<<Bilangan<<" tidak ditemukan ";
  getch ();
}
```

b) Buatlah program Sequential Search (simpan dengan nama lat6_2.cpp)

```
#include<iostream.h>
#include<conio.h>
main()
{
  int Nilai [20];
  int Posisi [ 20 ];
  int i, N, Bilangan, Banyak= 0;
  bool Ketemu;
  cout<<"Masukan Banyaknya Bilangan = ";
  cin>>N;

  //Membaca elemen Array
  for(i=0; i<N; i ++)
  {
    cout<<"Masukan elemen ke-"<<i<<" = ";
    cin>>Nilai [ i ];
```

```
}

  //Mencetak Elemen Array
  cout<<"\n\nDeretan Bilangan =";
  for (i=0; i<N; i++)
    cout<<Nilai [ i ] <<" ";
    cout<<"\n\nMasukan Bilangan yang akan dicari = ";
    cin>> Bilangan;

  //Melakukan Pencarian
  for (i=0; i<N; i++)
  {
    if (Nilai [ i ] ==Bilangan)
    {
            Ketemu = true;
            Posisi [Banyak] = i;
            Banyak++;
    }
  }
  if (Ketemu)
  {
    cout<<"Bilangan "<<Bilangan<<" ditemukan Sebanyak "<<Banyak;
    cout<<"\npada posisi ke =";
    for(i=0; i<Banyak; i ++)
      cout<<Posisi [ i ]<<" ";
  }
  else
  {
    cout<<"Bilangan "<<Bilangan<<" tidak ditemukan";
  }
  getch ();
}
```

c) Buatlah program untuk pencarian dengan metode binary search (simpan dengan nama lat6_3.cpp)

```
#include<stdio.h>
#include<iostream.h>
#include<conio.h>
#include<iomanip.h>
main ( )
{
```

```
//deklarasi variabel
int Nilai [ 20 ];
int i, j, N;
int temp, awal, akhir, tengah, Bilangan ;
//proses penginputan data
cout<<"Banyak bilangan : ";
cin>>N;
for (i=0; i<N; i++)
{
  cout<<"Elemen ke-"<<i<<"= ";
  cin>>Nilai [ i ] ;
}
cout<<"\nElemen Sebelumnya diurut = ";
for (i=0; i<N; i++ )
  cout<<setw ( 3 )<<Nilai[i] ;

//proses pengurutan data
for (i=0; i<N; i++ )
{
  for (j=i+1; j<N;   j++ )
  {
    if (Nilai [ i ] > Nilai [ j ] )
    {
      temp   = Nilai [ i ];
                Nilai [ i ]     = Nilai [ j ];
          Nilai [ j ]  = temp;
    }
  }
}
cout<<"\nElemen Setelah diurut = " ;
for ( i=0; i<N; i++ )
  cout<<setw ( 3 )<< Nilai [ i ];
  cout<<"\nindeks Elemen                    = " ;
for ( i=0; i<N; i++ ) cout<<setw ( 3 )<<i;
  cout<<"\nMasukan data yang akan anda cari : " ;
  cin>>Bilangan;

//proses pencarian data
awal = 0 ;
akhir = N-1;
do
{
```

```
    tengah =( akhir + awal ) / 2;
    if ( Bilangan < Nilai [ tengah ] )
      akhir = tengah - 1;
    else
      awal = tengah + 1 ;
  }
  while (( akhir >= awal) && ( Nilai [ tengah ] != Bilangan )) ;
  if ( Nilai [ tengah ] == Bilangan )
  {
    cout<<"\nData "<<Bilangan<<" ada dalam array ";
    cout<<" pada posisi "<<tengah;
  }
  else
    cout<<"\nData "<<Bilangan<<" tidak ada dalam array\n";
getch () ;
}
```

d) Buatlah program dengan Array of Struct seperti pada Pertemuan III : program biodata pegawai yang terdiri dari field-field seperti NIP, Nama, Alamat, Agama dan Jabatan. **Kemudian tambahkan program menggunakan metode sequential search untuk mencari seorang pegawai lalu tampilkan biodatanya. (simpan dengan nama lat6_4.cpp)**

TUGAS PENDAHULUAN

1. Apa yang dimaksud dengan Searching!
2. Jelaskan perbedaan dari Sequential Search, Binary Search, dan Interpolation Search sebagai metode-metode searching !
3. Jelaskan apa saja yang mempengaruhi kecepatan proses pencarian data di dalam penyimpanan data !
4. Buatlah contoh program sederhana menggunakan Searching!

TUGAS AKHIR

1. Buatlah program untuk melakukan pencarian terhadap data 12,15 dan 37 dari sederetan data 34, 8, 50, 74, 87, 90, 12, 25, 20, 30, 35, 45, 40, 22, 29, 72, 60, 55, 53, 12, 32, 33, 12, 41, 12 ! Jika data yang dicari terdapat lebih dari satu, tentukan banyaknya dan sebutkan berada pada posisi berapa saja data yang dicari berada pada sedereatan data!

PERTEMUAN VII
SORTING

TUJUAN PRAKTIKUM

a) Mahasiswa dapat menjelaskan pengertian Sort dengan C++.
b) Mahasiswa dapat melakukan pengurutan data dengan beberapa metode pengurutan dengan C++.

TEORI DASAR

a) Pendahuluan

Salah satu bagian terpenting dari struktur data adalah proses pengurutan data. Data terkadang akan berada dalam bentuk yang tidak berpola ataupun dengan pola tertentu yang tidak kita inginkan. Namun dalam penggunaannya, kita akan selalu ingin menggunakan data tersebut dalam bentuk yang rapi atau berpola sesuai dengan yang kita inginkan. Maka dari itu proses sorting adalah proses yang sangat penting dalam struktur data. Proses pengurutan banyak ditemukan dalam pemrosesan komputer.

b) Definisi Sorting

Pengurutan (sorting) adalah proses mengatur sekumpulan objek menurut urutan atau susunan tertentu. Urutan objek tersebut dapat menaik (ascending), yaitu urutan objek yang disusun mulai dari Nilai terkecil hingga terbesar atau menurun (descending), yaitu urutan objek yang disusun mulai dari Nilai terbesar hingga terkecil. Jika N buah objek atau data disimpan di dalam array Nilai, maka pengurutan menaik berarti menyusun elemen array sedemikian sehingga:

$$NILAI[0] \leq NILAI[1] \leq NILAI[2] < \ldots < NILAI[N-1]$$

sedangkan pengurutan menurun berarti menyusun elemen array, sedemikian sehingga:

$$NILAI[0] \geq NILAI[1] \geq NILAI[2] \geq \ldots \geq NILAI[N-1]$$

Pengurutan (Sorting) dibedakan menjadi dua kelompok, yaitu: Pengurutan Internal dan Pengurutan Eksternal.

c) Bubble Sort

Bubble Sort adalah metode pengurutan yang membandingkan elemen yang sekarang dengan elemen-elemen berikutnya. Pembanding elemen dapat dimulai dari awal atau mulai dari paling akhir. Apabila elemen yang sekarang lebih besar (untuk urut menaik) atau lebih kecil (untuk urut menaik) dari elemen berikutnya, maka posisinya di tukar, tetapi jika tidak maka posissnya tetap.

Contoh : Misalkan kita mempunyai array sebanyak 8 elemen diurutkan secara menaik dengan metode bubble Sort: 25,72,30,45,20,15,6,50. Urutan langkah pengurutannya yang dimulai dari belakang.

d) Quick Sort

*Quick Sort*merupakan metode terdapat dalam peruses pengurutan data dengan menggunakan perinsip rekursif. Metode ini menggunakan strategi "pecah belah" dengan mekanisme berikut ini.

Misalkan kita mempunyai array Naik [k..1]. Array dipartisi menjadi 2 bagian array kiri niali[k..m] dan array kanan Nilai [m+1..l] Dasar mempartisi menjadi dua adalah dengan mengambil elemen yang pertama sebagai elemen pivot.

TUGAS PRAKTIKUM

a) Buatlah program Pengurutan dengan Metode Bubble Sort "Pengurutan secara menaik" (simpan dengan nama lat7_1.cpp)

```
#include<iostream.h>
#include<conio.h>
#include<iomanip.h>
main ( )
{
  int Nilai [ 20 ];
  int i, j, k ,N ;
  int temp ;
  bool tukar ;
  cout<<"Masukan Banyak Bilangan : ";
  cin>>N;
```

```
  for (i=0;  i<N;  i++)
  {
    cout<<"Elemen ke-"<<i<<"  :  ";
    cin>>Nilai [ i ];
  }

  //Proses Cetak Sebelum diurutkan
  cout<<"\nData sebelumnya diurut :";
  for (i=0;  i<N ; i++)
    cout<<setw ( 3 )<<Nilai [ i ];

  //Proses Pengurutan
  i=0;
  tukar = true;
while ((i<=N-2) && (tukar))
{
    tukar=false;
    for (j=N-1;  j>=i+1;  j--)
    {
      if( Nilai [ j ]  < Nilai [ j-1 ] )
      {
        temp = Nilai [ j ];
        Nilai [ j ] = Nilai [ j-1];
        Nilai [ j-1 ] = temp;
        tukar = true;
        cout<<"\nUntuk j = "<<j<<" : ";
        for (k=0;  k<N;  k++)
                cout<<setw(3)<<Nilai [ k ] ;
      }
    }
    i++ ;
  }
  //Proses Cetak setelah diurutkan
  cout<<"\nData setelah di urut :  ";
  for (i=0;  i<N; i++ )
    cout<<setw ( 3 )<<Nilai [ i ] ;
  getch () ;
}
```

b) Buatlah program Pengurutan dengan Metode Bubble Sort“Pengurutan secaramenurun” (simpan dengan nama lat7_2.cpp)

```
#include<iostream.h>
#include<conio.h>
#include<iomanip.h>
main ( )
{
  int Nilai [ 20 ];
  int i, j, k ,N ;
  int temp ;
  bool tukar ;
  cout<<"Masukan Banyak Bilangan : ";
  cin>>N;
  for (i=0;  i<N;  i++)
{
cout<<"Elemen ke-"<<i<<"  :  ";
 cin>>Nilai [ i ];
  }

//Proses Cetak Sebelum diurutkan
  cout<<"\nData sebelumnya diurut :";
  for (i=0;  i<N ; i++)
    cout<<setw ( 3 )<<Nilai [ i ];

  //Proses Pengurutan
  i=0;
  tukar = true;
  while ((i<=N-2) && (tukar))
  {
    tukar=false;
    for (j=N-1;  j>=i+1;  j--)
    {
      if( Nilai [ j ]  < Nilai [ j-1 ] )
      {
        temp = Nilai [ j ];
        Nilai [ j ] = Nilai [ j-1];
        Nilai [ j-1 ] = temp;
        tukar = true;
        cout<<endl;
        for(k=0; k<N;  k++ )
```

```
        cout<<setw (3)<<Nilai [ k ] ;
      }
    }
    i++ ;
  }

  //Proses Cetak setelah diurutkan
  cout<<"\nData setelah di urut :  ";
  for (i=0;  i<N; i++ )
    cout<<setw ( 3 )<<Nilai [ i ] ;
  getch () ;
}
```

c) Buatlah program untuk Pengurutan dengan Metode Quick Sort "Pengurutan Secara menaik" (simpan dengan nama lat7_3.cpp)

```
#include<iostream.h>
#include<conio.h>
#include<iomanip.h>
void Cetak (int data [], int n)
{
  int i ;
  for ( i=0 ; i<n ; i++)
  cout<<setw ( 3 )<<data [ i ] ;
  cout<<"\n" ;
}

int Partisi ( int data [], int p, int r )
{
  int x, i, j, temp;
  x = data [ p ];
  i=p;
  j=r;
  while (1)
  {
    while( data[j]>x)
      j--;
    while( data[i]<x)
      i++;
```

```
    if(i<j)
    {
      temp = data [ i ];
      data [ i ] = data [ j ];
      data [ j ] = temp;
    }
    else
      return j;
  }
}

void Quick_Sort (int data [], int p, int r )
{
  int q ;
  if (p<r)
  {
    q=Partisi (data, p,r+1) ;
Quick_Sort (data, p, q ) ;
Quick_Sort (data, q+1, r) ;
  }
}

main ()
{
  int Nilai [ 20 ];
  int i, N ;
  cout<<"Masukan Banyak Bilangan : ";
  cin>>N;
  for(i=0;  i<N;  i++ )
  {
    cout<<"Elemen ke-"<<i<<"  :  ";
    cin>>Nilai [ i ] ;
  }
  cout<<"\nData Sebelum di urut  :  ";
  Cetak ( Nilai, N ) ;
  cout<<endl;
  Quick_Sort (Nilai, 0, N-1 );
  cout<<"\nData Setelah di urut  :  ";
  Cetak (Nilai, N ) ;
getch ( ) ;
}
```

d) Buatlah program menu untuk menampilkan 3 program di atas, menggunakan perintah switch (simpan dengan nama lat7_4.cpp)

TUGAS PENDAHULUAN

1. Apa yang dimaksud dengan Sorting!
2. Jelaskan Perbedaan dari Pengurutan Internal dan Pengurutan Eksternal!
3. Jelaskan perbedaan metode-metode Sorting seperti: Bubble Sort, Quick Sort, Selection Sort, Merge Sort, Tree Sort, Maximum Sort, dan Insertion Sort!
4. Buatlah contoh program sederhana menggunakan Sorting!

TUGAS AKHIR

1. Buatlah program untuk mengurutkan sederetan data: 34, 12, 56, 78, 6, 43, 32, 20, 90, 50, 55, 75, 85, 95, 25!

PERTEMUAN VIII
SORTING (Lanjut1)

TUJUAN PRAKTIKUM

a) Mahasiswa dapat menjelaskan pengertian Sort dengan C++.
b) Mahasiswa dapat melakukan pengurutan data dengan beberapa metode pengurutan dengan C++.

TEORI DASAR

a) Metode Maximum / Minimum Sort

Metode Maximum / Minimum Sort dilakukan berdasarkan pemilihan elemen maksimum / minimum, maka metode ini disebut juga dengan metode pemilih/seleksi (*Selection Sort*).

b) Metode Maximum Sort

Metode ini disebut juga dengan metode Maksimum karena didasarkan pada pemilihan elemen maksimum sebagai dasar pengurutan. Konsepnya adalah memilih elemen maksimum kemudian mempertukarkan elemen maksimum tersebut dengan elemen paling akhir untuk urut menaik dan elemen pertama untuk urut menurun. Demikian seterusnya hingga semua elemen terurut.

c) Metode Minimum Sort

Metode ini disebut juga dengan metode minimum karena didasarkan pada pemilihan elemen minimum sebagai dasar pengurutan. Konsepnya adalah memilih elemen minimum kemudian mempertukarkan elemen minimum tersebut dengan elemen paling akhir untuk urut menaik dan elemen pertama untuk urut menurun. Demikian seterusnya hingga semua elemen terurut.

TUGAS PRAKTIKUM

a) Buatlah program untuk Pengurutan dengan Metode Maximum Sort "Pengurutan Secara Menaik" (simpan dengan nama lat8_1.cpp)

```
#include<iostream.h>
#include<conio.h>
#include<iomanip.h>
main ()
{
  int Nilai [ 20 ];
  int i, j , N, l ;
  int temp, U, Imaks;
  cout<<"Masukan Banyaknya Bilangan :";
  cin>>N;
  for(i=0; i<N; i++)
  {
    cout<<"Elemen ke-"<<i<<" : ";
    cin>>Nilai [ i ];
  }
  //Proses Cetak sebelum diurutkan
  cout<<"\nData sebelum diurut :";
  for(i=0; i<N; i++)
    cout<<setw ( 3 )<<Nilai [ i ];
  //Peroses Pengurutan
  U=N-1 ;
  for (i=0 ; i<=N-2; i++)
  {
    Imaks =0;
    for(j=1; j<=U; j++)
    {
      if( Nilai [ j ] > Nilai [ Imaks] )
      Imaks = j;
    }
    temp =Nilai [ U ];
    Nilai [ U ] = Nilai [ Imaks];
    Nilai [ Imaks ]= temp;
    U--;
    cout<<endl;
    for(l=0; l<N; l++)
      cout<<setw ( 3 )<<Nilai [ 1 ];
  }
```

```
  cout<<"\nData Setelah di urut : ";
  for(i=0; i<N; i++)
    cout<<setw (3 )<<Nilai [ i ] ;
  getch ( ) ;
}
```

b) Buatlah program untuk Pengurutan dengan Metode Maximum Sort "Pengurutan Secara Menurun" (simpan dengan nama lat8_2.cpp)

```
#include<iostream.h>
#include<conio.h>
#include<iomanip.h>
main( )
{
  int Nilai [ 20 ];
  int i, j, N, l;
  int temp, U, Imaks;
  cout<<"Masukan Banyaknya Bilangan : ";
  cin>>N;
  for(i=0; i<N; i++)
  {
    cout<<"Elemen ke -"<<i<<" : ";
    cin>>Nilai [ i ];
  }
  //Peroses Cetak Sebelum diut=rutkan
  cout<<"\nData Sebelum diurut :";
  for (i=0; i<N; i++)
    cout<<setw (3)<< Nilai [ i ] ;

  //Peroses pengurutan
  U=N-1;
  for(i=0; i<=N-2; i++)
  {
    Imaks = i;
    for(j=i+1; j<=U; j++)
    {
      if(Nilai[ j ] > Nilai [Imaks])
        Imaks = j;
    }
```

```
    temp = Nilai [ i ];
    Nilai [ i ] = Nilai [ Imaks ];
    Nilai [Imaks] = temp;
    cout<<endl;
    for(l=0; l<N; l++)
      cout<<setw(3)<<Nilai [l];
  }
  cout<<"\nData Setelah di urut : " ;
  for(i=0; i<N; i++)
    cout<<setw(3)<<Nilai [i];
  getch ( );
}
```

c) Buatlah program untuk Pengurutan dengan Metode Minimum Sort "Pengurutan Secara Menaik" (simpan dengan nama lat8_3.cpp)

```
#include<iostream.h>
#include<conio.h>
#include<iomanip.h>
main ( )
{
  int Nilai [ 20 ];
  int i, j, N, l;
  int temp, Imin;
  cout<<"Masukan Banyak bilangan        : ";
  cin>>N;
  for (i=0; i<N; i++ )
  {
    cout<<"Elemen ke-"<<i<<" : ";
    cin>>Nilai [ i ] ;
  }
  //Proses Cetak Sebelum Diurutkan
  cout<<"\nData sebelum diurut :";
  for(i=0; i<N; i++)
    cout<<setw (3)<<Nilai [ i ];
  //Proses pengurutan
  for (i=0;i<=N-2; i++)
  {
    Imin = i;
    for(j=i+1; j<N; j++)
    {
```

```
      if(Nilai [j] < Nilai [Imin])
        Imin = j;
    }
    temp = Nilai [i];
    Nilai [i] = Nilai [Imin];
    Nilai [Imin] = temp;
    cout<<endl;
    for(l=0; l<N; l++)
      cout<<setw(3)<<Nilai [l];
  }
  cout<<"\nData Setelah di urut ; ";
  for(i=0; i<N; i++)
    cout<<setw(3)<<Nilai [i];
  getch ( );
}
```

d) Buatlah program untuk Pengurutan dengan Metode Minimum Sort "Pengurutan Secara Menurun" (simpan dengan nama lat8_4.cpp)

```
#include<iostream.h>
#include<conio.h>
#include<iomanip.h>
main ( )
{
  int Nilai [20];
  int i, j, N, l;
  int temp,U, Imin;
  cout<<"Masukan Banyak Bilangan : ";
  cin>>N;
  for (i=0; i<N; i++)
  {
    cout<<"Elemen ke-"<<i<<" : ";
    cin>>Nilai [ i ];
  }
  //Proses Cetak Sebelum diurutkan
  cout<<"\nData sebelum diurut : ";
  for (i=0; i<N; i++)
    cout<<setw ( 3 )<<Nilai [ i ];
```

```
  //Proses Pengurutan
  U = N - 1;
  for(i=0; i<=N-2; i++)
  {
    Imin = 0;
    for (j=1; j<=U; j++)
    {
      if(Nilai [ j ] <Nilai [Imin])
      Imin = j;
    }
    temp = Nilai [ U ];
    Nilai [ U ]=  Nilai [ Imin ];
    Nilai [ Imin ] = temp;
    cout<<endl;
    U--;
    for(l=0; l<N; l++)
      cout<<setw ( 3 )<<Nilai [ l ];
  }
  cout<<"\nData Setelah di urut : ";
  for(i=0; i<N; i++)
  cout<<setw ( 3 )<<Nilai [ i ];
  getch ( );
}
```

e) Buatlah program menu untuk menampilkan 4 program di atas, menggunakan perintah IF (simpan dengan nama lat8_5.cpp)

TUGAS PENDAHULUAN

1. Jelaskan kekurangan menggunakan metode Maximum/Minimum Sort dengan metode-metode Sorting lainnya!
2. Jelaskan perbedaan program Sorting dengan menggunakan antara metode Maximum Sort dan Minimum Sort!
3. Jelaskan tahapan-tahapan Sorting menggunakan metode Maximum Sort!
4. Jelaskan tahapan-tahapan Sorting menggunakan metode Minimum Sort!

TUGAS AKHIR

1. Buatlah tambahan program menggunakan sistem menu pada program yang telah dipraktekkan!

PERTEMUAN IX
SORTING (Lanjut2)

TUJUAN PRAKTIKUM

a) Mahasiswa dapat menjelaskan pengertian Sort dengan C++.
b) Mahasiswa dapat melakukan pengurutan data dengan beberapa metode pengurutan dengan C++.

TEORI DASAR

a) Metode Shell Sort

Metode ini mirip dengan Bubble Sort, hanya saja perbandingan dilakukan bukan antara dua bilangan yang berurutan, akan tetapi antara dua bilangan dengan jarak tertentu. Jarak ditentukan dengan N Div 2, dimana N adalah banyaknya elemen array. Lakukanpertukaran tempat jika setiap kali perbandingan dipenuhi (lebih besar untuk urut menaik dan lebih kecil untuk urut menurun).Setiap kali perbandingan terhadap keseluruhan elemen selesai dilakukan, maka perbandingan yang baru dilakukan kembali dimana jarak diperoleh dengan jarakDiv 2 (Jarak diperoleh dari Nilai jarak sebelumnya). Perbandingan keseluruhan dilakukan sampai Nilai jarak sama dengan 1 (satu). Pada saat jarak bernilai 1, maka metode Shell Sort sama dengan metode Bubble Sort.

b) Metode Insertion Sort

Metode ini merupakan metode pengurutan dengan cara menyisipkan elemen array pada posisi yang tepat.Pencarian yang tepat dilakukan dengan melakukan pencarian beruntun di dalam array. Selama pencarian posisi yang tepat dilakukan pergeseran elemen array. Algoritma pengurutan ini tepat untuk persoalan menyisipkan elemen baru ke dalam array yang sudah terurut.

TUGAS PRAKTIKUM

a) Buatlah program untuk Pengurutan dengan Metode Shell Sort "Pengurutan secara menaik" (simpan dengan nama lat9_1.cpp)

```
#include<iostream.h>
#include<conio.h>
#include<iomanip.h>
main ( )
{
  int Nilai [ 20 ];
  int i, k, N, l;
  int temp, jarak, s;
  cout<<"Masukan Banyak Bilangan : ";
  cin>>N;
  for (i=0; i<N; i++)
  {
    cout<<"Elemen ke-"<<i<<" : ";
    cin>>Nilai [ i ] ;
  }
  //Proses Cetak Sebelum diurutkan
  cout<<"\nData sebelum diurut : ";
  for (i=0; i<N; i++)
    cout<<setw (4)<<Nilai [ i ];
  //Proses pengurutan
  jarak = N/2;
  cout<<"\nJarak= "<<jarak;
  while (jarak >= 1)
  {
    do
    {
      s=0;
      for (i =0; i<=(N-jarak)-1; i++)
      {
        k=i+ jarak;
        if(Nilai [i] > Nilai [k])
        {
          temp = Nilai [i];
          Nilai [i] = Nilai [k];
          Nilai [k] = temp;
          s=1;
          for(l=0; l<N; l++)
```

```
 cout<<setw (4)<<Nilai [l];
           cout<<"\n\t";
         getch();
         }
       }
     }
     while(s!=0);
     jarak /=2;
     cout<<"\nJarak= "<<jarak;
   }
   cout<<"\nData Setelah diurut : ";
   for(i=0; i<N ;i++)
     cout<<setw (4)<<Nilai [i];
   getch ();
}
```

b) Buatlah program untuk Pengurutan denganMetode Shell Sort "Pengurutan secara menurun" (simpan dengan nama lat9_2.cpp)

```
#include<iostream.h>
#include<conio.h>
#include<iomanip.h>
main ( )
{
  int Nilai [ 20 ];
  int i, k, N, l;
  int temp, jarak, s;
  cout<<"Masukan Banyak Bilangan : ";
  cin>>N;
  for (i=0; i<N; i++)
  {
    cout<<"Elemen ke-"<<i<<" : ";
    cin>>Nilai [ i ] ;
  }
  //Proses Cetak Sebelum diurutkan
  cout<<"\nData sebelum diurut : ";
  for (i=0; i<N; i++)
    cout<<setw (4)<<Nilai [ i ];
  //Proses pengurutan
  jarak = N/2;
  cout<<"\nJarak= "<<jarak;
```

```
  while (jarak >= 1)
  {
    do
    {
      s=0;
      for (i =0; i<=(N-jarak)-1; i++)
      {
        k=i+ jarak;
        if(Nilai [i] < Nilai [k])
        {
          temp = Nilai [i];
          Nilai [i] = Nilai [k];
          Nilai [k] = temp;
          s=1;
          for(l=0; l<N; l++)
            cout<<setw (4)<<Nilai [l];
          cout<<"\n\t";
        getch();
        }
      }
    }
    while(s!=0);
    jarak /=2;
    cout<<"\nJarak= "<<jarak;
  }
  cout<<"\nData Setelah diurut : ";
  for(i=0; i<N ;i++)
    cout<<setw (4)<<Nilai [i];
  getch ();
}
```

c) Buatlah program untuk Pengurutan dengan Metode Insertion Sort "Pengurutan Secara menaik" (simpan dengan nama lat9_3.cpp)

```
#include<iostream.h>
#include<conio.h>
#include<iomanip.h>
main ( )
{
  int Nilai [ 20 ];
  int i,  j,  k,  N;
```

```
  int temp;
  cout<<"Masukan Banyak Bilangan : ";
  cin>>N;
  for (i=0; i<N; i++)
  {
    cout<<"Elemen ke-"<<i<<" : ";
    cin>>Nilai [ i ] ;
  }
  //Proses Cetak Sebelum diurutkan
  cout<<"\nData sebelum diurut : ";
  for (i=0; i<N; i++)
  cout<<setw ( 3 )<<Nilai [ i ];
  //Proses pengurutan
  for(i=1; i<N; i++)
  {
    temp = Nilai [ i ] ;
    j=i-1 ;
    while ((temp <= Nilai [ j ]) && (j>=1))
    {
            Nilai [j+1] = Nilai [ j ];
            j--;
    }
    if(temp >= Nilai [ j ])
      Nilai [j+1] = temp;
    else
    {
      Nilai [ j + 1] = Nilai [ j ];
      Nilai [ j ] = temp;
    }
    cout<<endl;
    for(k=0; k<N; k++)
      cout<<setw ( 3 )<<Nilai [ k ];
  }
  //Proses Cetak Setelah diurutkan
 cout<<"\nData Setelah diurut : ";
  for (i=0; i<N; i++)
    cout<<setw (3)<<Nilai [ i ] ;
  getch ( ) ;
}
```

d) Buatlah program untuk Pengurutan dengan Metode Insertion Sort "Pengurutan Secara menurun" (simpan dengan nama lat9_4.cpp)

```
#include<iostream.h>
#include<conio.h>
#include<iomanip.h>

main ( )
{
  int Nilai [ 20 ];
  int i,  j,  k,  N;
  int temp;
  cout<<"Masukan Banyak Bilangan : ";
  cin>>N;
  for (i=0; i<N; i++)
  {
    cout<<"Elemen ke-"<<i<<" : ";
    cin>>Nilai [ i ] ;
  }
  //Proses Cetak Sebelum diurutkan
  cout<<"\nData sebelum diurut : ";
  for (i=0; i<N; i++)
  cout<<setw ( 3 )<<Nilai [ i ];
  //Proses pengurutan
  for(i=1; i<N; i++)
  {
    temp = Nilai [ i ] ;
    j=i-1 ;
    while ((temp > Nilai [ j ]) && (j>=1))
    {
            Nilai [j+1] = Nilai [ j ];
            j--;
    }
    if(temp <= Nilai [ j ])
      Nilai [j+1] = temp;
    else
    {
      Nilai [ j + 1] = Nilai [ j ];
      Nilai [ j ] = temp;
    }
```

```
    cout<<endl;
    for(k=0; k<N; k++)
      cout<<setw ( 3 )<<Nilai [ k ];
  }
  //Proses Cetak Setelah diurutkan
  cout<<"\nData Setelah diurut : ";
  for (i=0; i<N; i++)
    cout<<setw (3)<<Nilai [ i ] ;
  getch ( ) ;
}
```

e) Buatlah program menu untuk menampilkan 4 program di atas, menggunakan perintah switch (simpan dengan nama lat9_5.cpp)

TUGAS PENDAHULUAN

1. Jelaskan kekurangan menggunakan metode Shell Sort dan Insertion Sort dengan metode-metode Sorting lainnya!
2. Jelaskan perbedaan program Sorting dengan menggunakan antara metode Shell Sort dan Insertion Sort!
3. Jelaskan tahapan-tahapan Sorting menggunakan metode Shell Sort!
4. Jelaskan tahapan-tahapan Sorting menggunakan metode Insertion Sort!

TUGAS AKHIR

1. Buatlah program untuk mengurutkan sederetan data: suka, aku, sama, kamu, dulu, sampai, dari, sekarang. Dengan menggunakan salah satu metode Shell Sort dan Insertion Sort!

PERTEMUAN X
LINKED LIST

TUJUAN PRAKTIKUM

a) Mahasiswa dapat menjelaskan pengertian dan pembuatan Linked List dengan C++.
b) Mahasiswa dapat melakukan operasi penyisipan maupun penghapusan simpul pada Linked List dengan C++.
c) Mahasiswa dapat mengimplementasikan Linked List dengan C++.

TEORI DASAR

a) Pendahuluan

Linked List adalah struktur berupa rangkaian elemen saling berkait dimana tiap elemen dihubungkan ke elemen yang lain melalui pointer. Pointer adalah alamat elemen. Penggunaan pointer untuk mengacu elemen berakibat elemen-elemen bersebelahan secara logic walaupun tidak bersebelahan secara fisik di memori.

Linked List merupakan kumpulan komponen yang saling berkaitan satu dengan yang lain melalui pointer. Masing-masing komponen sering disebut dengan simpul atau verteks.

b) Singly Linked List

Merupakan Linked List yang paling sederhana. Setiap simpul dibagi menjadi dua bagian yaitu bagian isi dan bagian pointer. Bagian isi merupakan bagian yang berisi data yang disimpan oleh simpul, sedangkan bagian pointer merupakan bagian yang berisi alamat dari simpul berikutnya.

Deklarasi Doubly Linked List :

```
typedef struct node *simpul
struct node
      {
        type_data Isi;
        simpul  Next;
      };
```

c) Doubly Linked List

Doubly Linked Listmerupakan Linked List dimana setiap simpul dibagi menjadi tiga bagian, yaitu bagian isi, bagian pointer kiri, dan bagian pointer kanan. Bagian isi merupakan bagian yang berisi data yang disimpan oleh simpul, bagian pointer

kiri merupakan bagian yang berisi alamat dari simpul sebelumnya dan bagian pointer kanan merupakan bagian yang berisi alamat dari simpul berikutnya.

Deklarasi Doubly Linked List :

```
typedef struct node *simpul
struct node
      {
        char Isi;
        simpul kanan;
        simpul kiri;
      };
```

TUGAS PRAKTIKUM

a) Buatlah program untuk Operasi Singly Linked (simpan dengan nama lat10_1.cpp)

```
#include<iostream>
#include<conio.h>
#include<stdlib.h>
using namespace std;
typedef struct node *simpul;
struct node
  {
    char Isi;
    simpul Next;
  };
//====================
//==Prototype Function=
//====================
void Sisip_Depan (simpul &L, char elemen );
void Sisip_Belakang (simpul &L, char elemen ) ;
void Sisip_Tengah1 (simpul &L, char elemen1, char elemen2 ) ;
void Sisip_Tengah2 (simpul &L, char elemen1, char elemen2 ) ;
void Hapus_Depan (simpul &L);
void Hapus_Belakang (simpul &L);
void Hapus_Tengah (simpul &L, char elemen);
void Cetak (simpul L);

//===================
//==Function Main====
//===================
main ( )
{
```

```
  char huruf, huruf2;
  simpul L = NULL; //Pastikan Bahwa L kosong
  cout<<"==OPERASI PADA SINGLE LINKED LIST=="<<endl<<endl;
  //===============
  //==Sisip Depan==
  //===============
  cout<<"Penyisipan Simpul Di Depan"<<endl<<endl;
  cout<<"Masukan Huruf : "; cin>>huruf;
  Sisip_Depan (L, huruf );
  cout<<"Masukan Huruf :"; cin>>huruf ;
  Sisip_Depan (L, huruf );
  cout<<"Masukan Huruf :"; cin>>huruf ;
  Sisip_Depan (L, huruf );
  cout<<"Masukan Huruf :"; cin>>huruf ;
  Sisip_Depan (L, huruf );
  Cetak (L);
  //=================
  //==Sisip Belakang=
  //=================
  cout<<"\n\nPenyisipan Simpul Di Belakang"<<endl<<endl;
  cout<<"Masukan Huruf : "; cin>>huruf;
  Sisip_Belakang (L, huruf );
  cout<<"Masukan Huruf :"; cin>>huruf ;
  Sisip_Belakang (L, huruf );
  cout<<"Masukan Huruf :"; cin>>huruf ;
  Sisip_Belakang (L, huruf );
  cout<<"Masukan Huruf :"; cin>>huruf ;
  Sisip_Belakang (L, huruf );
  Cetak (L);
  //=====================================
  //==Sisip Simpul Setelah Simpul Tertentu=
  //=====================================
  cout<<endl<<endl<<"Masukan Huruf    : "; cin>>huruf;
  cout<<"Disisip Setelah Huruf      : "; cin>>huruf2;
  cout<<huruf<<" Disisip Setelah "<<huruf2<<endl;
  Sisip_Tengah1 (L, huruf, huruf2);
  Cetak (L) ;
//=====================================
  //==Sisip Simpul Sebelum Simpul Tertentu=
  //=====================================
```

```
 cout<<endl<<endl<<"Masukan Huruf      : "; cin>>huruf;
  cout<<"Disisip Sebelum Huruf     : "; cin>>huruf2;
  cout<<huruf<<" Disisip Sebelum "<<huruf2<<endl;
  Sisip_Tengah2 (L, huruf, huruf2);
  Cetak (L) ;
  //====================
  //==Hapus Simpul Depan=
  //====================
  cout<<endl<<endl<<"Setelah Hapus Simpul Depan "<<endl;
  Hapus_Depan (L);
  Cetak (L);
  //=======================
  //==Hapus Simpul Belakang=
  //=======================
  cout<<endl<<endl<<"Setelah Hapus Simpul Belakang "<<endl;
  Hapus_Belakang (L);
  Cetak (L);
  //=======================
  //==Hapus Simpul TENGAH==
  //=======================
  cout<<"\n\nMasukkan Huruf Tengah Yang akan dihapus :
";cin>>huruf;
  Hapus_Tengah (L,huruf);
  Cetak (L);
  getch ( ) ;
}

//***********************************
//**FUNCTION SISIP SIMPUL DI DEPAN****
//***********************************
void Sisip_Depan (simpul &L, char elemen)
{
  simpul baru; // = new simpul ;
  baru = (simpul) malloc (sizeof(simpul));
  baru->Isi = elemen ;
  baru-> Next = NULL;
  if (L== NULL)
    L=baru;
else
{
    baru->Next = L;
```

```
    L= baru;
  }
}
//***********************************************
//**FUNCTION SISIP SIMPUL SETELAH SIMPUL TERTENTU**
//***********************************************
void Sisip_Tengah1 (simpul &L, char elemen1, char elemen2)
{
  simpul bantu,baru;
  baru = (simpul) malloc (sizeof(simpul));
  baru->Isi = elemen1 ;
  baru-> Next = NULL;
  if (L== NULL)
    cout << "List Kosong ....."<<endl;
  else
  {
    bantu = L;
    while (bantu ->Isi != elemen2) bantu = bantu -> Next;
    baru->Next = bantu ->Next ;
    bantu->Next = baru ;
  }
}
//*****************************************************
//**FUNCTION SISIP SIMPUL SEBELUM SIMPUL TERTENTU********
//*****************************************************
void Sisip_Tengah2 (simpul &L, char elemen1, char elemen2)
{
  simpul bantu, baru;
  baru = (simpul) malloc (sizeof(simpul));
  baru->Isi = elemen1 ;
  baru-> Next = NULL;
  if (L== NULL)
    cout<<"List Kosong....."<<endl;
  else
  {
    bantu = L;
    while (bantu->Next->Isi != elemen2) bantu = bantu -> Next;
    baru->Next = bantu ->Next ;
    bantu->Next = baru ;
  }
}
```

```
//*********************************************
//**FUNCTION SISIP SIMPUL DI BELAKANG***********
//*********************************************
void Sisip_Belakang (simpul &L, char elemen)
{
  simpul bantu, baru;
  baru = (simpul) malloc (sizeof(simpul));
  baru->Isi = elemen ;
  baru-> Next = NULL;
  if (L== NULL)
    L=baru;
  else
  {
    bantu = L;
    while (bantu->Next != NULL)
    bantu= bantu -> Next ;
    bantu->Next = baru ;
  }
}
//*********************************************
//**FUNCTION MENCETAK ISI LINKED LIST************
//*********************************************
void Cetak (simpul L)
{
  simpul bantu ;
  if (L==NULL)
    cout<<"Linked List Kosong ........"<<endl;
  else
  {
    bantu =L;
    cout<<"Isi Linked List : ";
    while (bantu ->Next != NULL)
    {
      cout<<bantu->Isi<<"-->";
            bantu=bantu->Next;
          }
    cout<<bantu->Isi;
        }
}
```

```
//************************************
//**FUNCTION HAPUS SIMPUL DEPAN*******
//************************************
void Hapus_Depan (simpul &L)
{
  simpul Hapus ;
  if (L==NULL)
    cout<<"Linked List Kosong.....";
  else
  {
    Hapus = L;
    L = L-> Next ;
    Hapus -> Next = NULL;
    free (Hapus);
  }
}
//************************************
//**FUNCTION HAPUS SIMPUL BELAKANG*****
//************************************
void Hapus_Belakang (simpul &L)
{
  simpul bantu, hapus;
  if (L==NULL)
    cout<<"Linked List Kosong.....";
  else
  {
    bantu = L;
    while (bantu ->Next->Next != NULL) bantu=bantu->Next;
    hapus = bantu -> Next;
          bantu -> Next = NULL;
    free (hapus);
  }
}
//************************************
//**FUNCTION HAPUS SIMPUL DI TENGAH****
//************************************
void Hapus_Tengah(simpul &L, char elemen)
{
simpul bantu,hapus;
```

```
  if (L==NULL)
    cout<<"Linked List Kosong.....";
  else
  {
    bantu = L;
    while (bantu ->Next->Isi != elemen) bantu=bantu->Next;
    hapus = bantu -> Next;
    bantu ->Next = bantu -> Next ->Next;
hapus -> Next = NULL;
    free (hapus);
  }
}
 //======================eof=========================
```

b) Buatlah program untuk Operasi Doubly Linked (simpan dengan nama lat10_2.cpp)

```
#include<iostream>
#include<conio.h>
#include<stdlib.h>
#define true 1
#define false 0
using namespace std;
typedef struct node *simpul;
struct node
  {
    char Isi;
    simpul kanan;
    simpul kiri;
  };
//=====================
//==Prototype Function=
//=====================
void Sisip_Depan (simpul &DL, char elemen );
void Sisip_Belakang (simpul &DL, char elemen ) ;
void Sisip_Tengah1 (simpul &DL, char elemen1, char elemen2 ) ;
void Sisip_Tengah2 (simpul &DL, char elemen1, char elemen2 ) ;
void Hapus_Depan (simpul &DL);
void Hapus_Belakang (simpul &DL);
void Hapus_Tengah (simpul &DL, char elemen);
void Cetak (simpul DL);
```

```
//==================
//==Function Main====
//==================
main ( )
{
  char huruf, huruf2;
  simpul DL = NULL; //Pastikan Bahwa DL kosong
  int i;
  cout<<"\t\t==OPERASI PADA DOUBLY LINKED LIST==\n\n";
  //===============
  //==Sisip Depan==
  //===============
  cout<<"Penyisipan Simpul Di Depan"<<endl<<endl;
  for (i=1;i<=4;i++)
  {
    cout<<"Masukan Huruf : "; cin>>huruf;
    Sisip_Depan (DL, huruf );
  }
  Cetak (DL);
  //=================
  //==Sisip Belakang=
  //=================
  cout<<"\n\nPenyisipan Simpul Di Belakang"<<endl<<endl;
  for (i=1;i<=4;i++)
  {
    cout<<"Masukan Huruf : "; cin>>huruf;
    Sisip_Belakang (DL, huruf );
  }
  Cetak (DL);
  //=====================================
  //==Sisip Simpul Setelah Simpul Tertentu=
  //=====================================
  cout<<endl<<endl<<"Masukan Huruf    : "; cin>>huruf;
  cout<<"Disisip Setelah Huruf      : "; cin>>huruf2;
  cout<<huruf<<" Disisip Setelah "<<huruf2<<endl;
  Sisip_Tengah1 (DL, huruf, huruf2);
  Cetak (DL) ;
  //=====================================
  //==Sisip Simpul Sebelum Simpul Tertentu=
  //=====================================
  cout<<endl<<endl<<"Masukan Huruf    : "; cin>>huruf;
  cout<<"Disisip Sebelum Huruf    : "; cin>>huruf2;
  cout<<huruf<<" Disisip Sebelum "<<huruf2<<endl;
  Sisip_Tengah2 (DL, huruf, huruf2);
```

```
  Cetak (DL) ;
  //====================
  //==Hapus Simpul Depan=
  //====================
  cout<<endl<<endl<<"Setelah Hapus Simpul Depan "<<endl;
  Hapus_Depan (DL);
  Cetak (DL);
  //======================
  //==Hapus Simpul Belakang=
  //======================
  cout<<endl<<endl<<"Setelah Hapus Simpul Belakang "<<endl;
  Hapus_Belakang (DL);
  Cetak (DL);
  //======================
  //==Hapus Simpul TENGAH==
  //======================
  cout<<"\n\nMasukkan Huruf Tengah Yang akan dihapus :
";cin>>huruf;
  Hapus_Tengah (DL,huruf);
  Cetak (DL);
  getch ( ) ;
}

//***********************************
//**FUNCTION SISIP SIMPUL DI DEPAN****
//***********************************
void Sisip_Depan (simpul &DL, char elemen)
{
  simpul baru;
  baru = (simpul) malloc (sizeof(simpul));
  baru->Isi = elemen ;
  baru-> kanan = NULL;
  baru-> kiri = NULL;
  if (DL== NULL)
    DL=baru;
  else
  {
    baru->kanan = DL;
    DL->kiri = baru;
          DL= baru;
  }
}
//************************************************
//**FUNCTION SISIP SIMPUL SETELAH SIMPUL TERTENTU **
//************************************************
```

```
void Sisip_Tengah1 (simpul &DL, char elemen1, char elemen2)
{
  simpul bantu,baru;
  baru = (simpul) malloc (sizeof(simpul));
  baru->Isi = elemen1 ;
  baru-> kanan = NULL;
  baru-> kiri = NULL;
  if (DL== NULL)
    cout << "List Kosong ....."<<endl;
  else
  {
    bantu = DL;
    while (bantu ->Isi != elemen2)
            bantu = bantu -> kanan;
      baru->kanan = bantu ->kanan;
      baru->kiri = bantu;
      bantu->kanan->kiri = baru;
      bantu->kanan = baru;
  }
}
//*******************************************************
//**FUNCTION SISIP SIMPUL SEBELUM SIMPUL TERTENTU*******
//*******************************************************
void Sisip_Tengah2 (simpul &DL, char elemen1, char elemen2)
{
  simpul bantu, baru;
  baru = (simpul) malloc (sizeof(simpul));
  baru->Isi = elemen1 ;
  baru-> kanan = NULL;
  baru-> kiri = NULL;
  if (DL== NULL)
    cout<<"List Kosong....."<<endl;
  else
  {
    bantu = DL;
    while (bantu->kanan->Isi != elemen2)
            bantu = bantu -> kanan;
      baru->kanan = bantu ->kanan;
      baru->kiri = bantu;
      bantu->kanan->kiri = baru;
      bantu->kanan = baru;
  }
}
```

```
//*********************************************
//**FUNCTION SISIP SIMPUL DI BELAKANG**********
//*********************************************
void Sisip_Belakang (simpul &DL, char elemen)
{
  simpul bantu, baru;
  baru = (simpul) malloc (sizeof(simpul));
  baru->Isi = elemen ;
  baru-> kanan = NULL;
  baru-> kiri = NULL;
  if (DL== NULL)
    DL=baru;
  else
  {
    bantu = DL;
    while (bantu->kanan != NULL)
      bantu = bantu -> kanan;
      bantu->kanan = baru;
      baru->kiri = bantu;
  }
}
//*********************************************
//**FUNCTION MENCETAK ISI LINKED LIST**********
//*********************************************
void Cetak (simpul DL)
{
  simpul bantu ;
  if (DL==NULL)
    cout<<"Linked List Kosong ........"<<endl;
  else
  {
    bantu =DL;
    cout<<"Isi Linked List : ";
    while (bantu ->kanan != NULL)
    {
      cout<<bantu->Isi<<"<= =>";
            bantu=bantu->kanan;
         }
    cout<<bantu->Isi;
        }
}
```

```
//************************************
//**FUNCTION HAPUS SIMPUL DEPAN********
//************************************
void Hapus_Depan (simpul &DL)
{
  simpul Hapus ;
  if (DL==NULL)
    cout<<"Linked List Kosong.....";
  else
  {
    Hapus = DL;
    DL = DL-> kanan ;
    DL ->kiri = NULL;
    Hapus -> kanan = NULL;
    free (Hapus);
  }
}
//************************************
//**FUNCTION HAPUS SIMPUL BELAKANG*****
//************************************
void Hapus_Belakang (simpul &DL)
{
  simpul bantu, hapus;
  if (DL==NULL)
    cout<<"Linked List Kosong.....";
  else
  {
    bantu = DL;
    while (bantu ->kanan->kanan != NULL) bantu=bantu->kanan;
      hapus = bantu -> kanan;
            bantu -> kanan = NULL;
            hapus -> kiri = NULL;
      free (hapus);
  }
}
//************************************
//**FUNCTION HAPUS SIMPUL DI TENGAH****
//************************************
void Hapus_Tengah(simpul &DL, char elemen)
{
```

```
  simpul bantu,hapus;
  if (DL==NULL)
    cout<<"Linked List Kosong.....";
  else
  {
    bantu = DL;
    while (bantu ->kanan->Isi != elemen)
          bantu=bantu->kanan;
      hapus = bantu -> kanan;
      bantu ->kanan->kanan->kiri = bantu;
          bantu ->kanan = bantu->kanan->kanan;
          hapus -> kanan = NULL;
          hapus -> kiri = NULL;
      free (hapus);
  }
}
 //======================eof==========================
```

TUGAS PENDAHULUAN

1. Apa yang dimaksud dengan Linked List!
2. Jelaskan perbedaan antara Singly Linked List, Doubly Linked List, dan Circular Linked List !
3. Jelaskan Operasi-Operasi pada Singly Linked List!
4. Jelaskan Operasi-Operasi pada Doubly Linked List!

TUGAS AKHIR

1. Buatlah program menu untuk menampilkan 2 program di atas !

PERTEMUAN XI

STACK

TUJUAN PRAKTIKUM

a) Mahasiswa dapat menjelaskan pengertian dan pembuatan Stack dengan C++.
b) Mahasiswa dapat melakukan operasi penyisipan dan penghapusan elemen dalam Stack dengan C++.
c) Mahasiswa dapat mengimplementasikan Stack dengan C++.

TEORI DASAR

a) Pendahuluan

Stack atau tumpukan adalah kumpulan elemen yang hanya dapat di tambahatau dihapus dari satu ujung (gerbang) yang sama. Hal ini menunjukan bahwa seolah-olah suatu elemen diletakan di atas elemen yang lain. Yang memberi gambaran bahwa Stack mempunyai sifat LIFO(*Last In First Out*) yang berarti bahwa elemen yang terakhir masuk akan pertama keluar.

Secara sederhana stack dimisalkan kita mempunyai 4 buah kotak (A,B,C, dan D) yang ditumpukkan. Kotak A diletakkan paling bawah, lalu diikuti kotak B, C, dan yang teratas atau terakhir adalah D. Maka untuk mengambil tiap kotak harus dilakukan berurutan dari kotak D, C, B kemudian A. Karena jika kita mengambil kotak B tanpa terlebih dahulu mengambil kotak di atasnya maka tumpukan akan roboh.

b) Deklarasi Stack

Bentuk deklarasi pertama:

```
#define MaxSn
TypeData      Isi[MaxS]
TypeData      Top;
```

Bentuk deklarasi kedua :

```
#define MaxS n
struct Stack
        {
        TypeData Isi [MaxS] ;
        TypeData      Top ;
        };
```

c) Operasi Pada Stack

Ada dua operasi dasar yang dapat dilakukan terhadap sebuah Stack, yaitu operasi insert atau penyisipkan elemen yang sering disebut istilah PUSH dan operasi Delete atau Penghapusan elemen yang sering disebut istilah POP.

d) Inisialisasi Stack

Sebelum stack dapat dioperasikan, terlebih dahulu diinisialisasikan dengan memberi harga S.Top =0.

```
void   INITS ( Stack  &S)
{
        S.Top = 0 ;
}
```

e) Mencetak Stack

Isi suatu Stack dapat dicetak dengan menggunakan fungsi berikut.

```
void CETAK ( Stack S)
{
        int i;
        cout<<endl<<"Isi Stack : ";
        if (S.Top  !=0)
        {
           for(i=1;i<=S.Top ; i++)
           cout<<S.Isi [ i ] ;
        }
        else
        cout<<"Stack Kosong….";
        }
```

f) Karakteristik Stack

Karakteristik dari suatu stack meliputi : Elemen Stack, Top, Max, Stack Kosong, dan stack penuh.

g) Aplikasi Stack

1) Simulasi stack dalam dunia nyata
2) Pemanggilan fungsi/procedure
3) Rekursif
4) Penanganan interupsi
5) Evaluasi ekspresi
6) Konversi notasi infiks ke notasi postfiks
7) Konversi bilangan basis 10 (decimal) ke basis 2 (biner)

TUGAS PRAKTIKUM

a) Buatlah program lengkap dari suatu Stack dengan menggunakan Array (simpan dengan nama lat11_1.cpp)

```
#include<iostream.h>
#include<conio.h>
#define MaxS 10
struct Stack
  {
    char Isi [MaxS] ;
    unsigned int Top;
  };
void INITS (Stack &S) ;
void PUSH (Stack &S, char Data) ;
void CETAK (Stack S) ;
void POP (Stack &S, char &Hsl) ;
main ( )
{
  char huruf ;
  Stack S;
  INITS (S) ;
  cout<<"Masukan Karakter :";
  cin>>huruf ;
  PUSH(S, huruf);
  cout<<"Masukan Karakter :";
  cin>>huruf ;
  PUSH(S, huruf);
  cout<<"Masukan Karakter :";
cin>>huruf ;
  PUSH(S, huruf);
```

```
  CETAK (S);
  POP (S, huruf);
  cout<<endl<<"Yang Dihapus ....."<<huruf;
  CETAK (S) ;
  cout<<endl<<"Masukan Karakter :";
  cin>>huruf ;
  PUSH(S, huruf);
  cout<<"Masukan karakter :";
  cin>>huruf;
  PUSH(S, huruf);
  cout<<"Masukan karakter :";
  cin>>huruf;
  PUSH(S, huruf);
  CETAK (S) ;
  POP (S, huruf);
  cout<<endl<<"Yang Dihapus ....."<<huruf;
  CETAK (S) ;
  getch ( ) ;
}
void INITS (Stack &S)
{
        S.Top = 0;
}
void PUSH (Stack &S, char Data)
{
  if (S.Top <MaxS)
  {
    S.Top++;
    S.Isi [S.Top] = Data;
  }
  else
    cout<<"Stack penuh.....";
}
void CETAK (Stack S)
{
  int i;
  cout<<endl<<"Isi Stack : ";
  if (S.Top != 0)
  {
    for(i=1;i<S.Top;i++)
```

```
    {
      cout<<S.Isi [i];
    }
  }
  else
    cout<<"Stack Kosong .....";
}
void POP (Stack &S, char &Hsl)
{
  if (S.Top !=0)
  {
    Hsl = S.Isi [S.Top];
    S.Top--;
  }
  else
    cout<<"Stack Kosong.....";
}
//=================end of file=================
```

b) Buatlah program untuk membalik karakter-karakter dalam suatu kalimat (karakter depan menjadi belakang dan karakter belakang menjadi karakter depan) dengan menggunakan Stack (simpan dengan nama lat11_2.cpp)

TUGAS PENDAHULUAN

1. Apa yang dimaksud dengan Stack!
2. Bagaimana tahapan-tahapan proses operasi PUSH!
3. Bagaimana tahapan-tahapan proses operasi POP!
4. Jelaskan karakteristik-karakteristik dari Stack!

TUGAS AKHIR

1. Buatlah program untuk mengkonversi bilangan desimal menjadi bilangan biner dengan menggunakan Stack!

PERTEMUAN XII

STACK (Lanjut)

TUJUAN PRAKTIKUM

a) Mahasiswa dapat menjelaskan pengertian dan pembuatan Stack dengan C++.
b) Mahasiswa dapat melakukan operasi penyisipan dan penghapusan elemen dalam Stack dengan C++.
c) Mahasiswa dapat mengimplementasikan Stack dengan C++.

TEORI DASAR

a) Pendahuluan

Stack atau tumpukan adalah kumpulan elemen yang hanya dapat di tambah atau dihapus dari satu ujung (gerbang) yang sama. Hal ini menunjukan bahwa seolah-olah suatu elemen diletakan di atas elemen yang lain. Yang memberi gambaran bahwa Stack mempunyai sifat LIFO (*Last In First Out*) yang berarti bahwa elemen yang terakhir masuk akan pertama keluar.

Representasi Stack dapat dilakukan menggunakan Array atau Linked List. Kedua representasi mempunyai keunggulan dan kelemahan. Dengan Array, stack juga dapat disajikan dengan Single Stack dan Double Stack.

TUGAS PRAKTIKUM

a) Buatlah program Stack dengan menggunakan Singly Linked List (simpan dengan nama lat12_1.cpp)

```
#include<iostream.h>
#include<conio.h>
#include<stdlib.h>
#define true1
#define false 0
typedef struct node *simpul;
struct node
  {
    char Isi;
    simpul next ;
  };
```

```
//=====================
//==Prototype Function==
//=====================
void Sisip_Belakang (simpul &L, char elemen);
void Hapus_Belakang (simpul &L);
void Cetak (simpul L);
//=================
//==Function Main==
//=================
main ( )
{
  char huruf ;
  simpul L = NULL; //Pastikan bahwa L kosong
  cout<<"Operasi Single Linked List Pada Stack==\n\n";
  //==================
  //==Sisip Belakang==
  //==================
  cout<<endl<<endl<<"Penyiapan Stack "<<endl<<endl;
  cout<<"Masukan Elemen : "; cin>>  huruf;
  Sisip_Belakang (L, huruf);
  cout<<"Masukan Elemen : "; cin>>  huruf;
  Sisip_Belakang (L, huruf);
  cout<<"Masukan Elemen : "; cin>>  huruf;
  Sisip_Belakang (L, huruf);
  cout<<"Masukan Elemen : "; cin>>  huruf;
  Sisip_Belakang (L, huruf);
  cout<<"Masukan Elemen : "; cin>>  huruf;
  Sisip_Belakang (L, huruf);
  cout<<"Masukan Elemen : "; cin>>  huruf;
  Sisip_Belakang (L, huruf);
  Cetak (L);
  //========================
  //==Hapus Simpul Belakang==
  //========================
  cout<<endl<<endl<<"Hapus Elemen "<<endl;
  Hapus_Belakang (L);
  Cetak (L);
  cout<<endl<<endl<<"Hapus Elemen "<<endl;
  Hapus_Belakang (L);
  Cetak (L);
  cout<<endl<<endl<<"Hapus Elemen "<<endl;
  Hapus_Belakang (L);
```

```
  Cetak (L);
  cout<<endl<<endl<<"Hapus Elemen "<<endl;
  Hapus_Belakang (L);
  Cetak (L);
  getch () ;
}
//*****************************************
//**FUNCTION SISIP SIMPUL DI BELAKANG******
//*****************************************
void Sisip_Belakang (simpul & L, char elemen)
{
  simpul bantu, baru;
  baru= (simpul) malloc(sizeof(simpul));
  baru->Isi = elemen;
  baru->next = NULL;
  if(L == NULL)
    L=baru;
  else
  {
    bantu=L;
    while(bantu->next != NULL)
      bantu=bantu->next;
    bantu->next=baru;
        }
}
//*****************************************
//**FUNCTION HAPUS SIMPUL BELAKANG*********
//*****************************************
void Hapus_Belakang (simpul & L)
{
  simpul bantu, hapus;
  if(L == NULL)
    cout<<"Linked List Kosong.....";
  else
  {
bantu=L;
  while(bantu->next->next != NULL)
      bantu=bantu->next;
hapus = bantu ->next;
bantu->next = NULL;
free(hapus);
  }
}
```

```
//****************************************
//**FUNCTION MENCETAK ISI LINKED LIST**
//****************************************
void Cetak(simpul L)
{
  simpul bantu;
  if (L==NULL)
    cout<<"Linked List Kosong ……."<<endl;
  else
  {
    bantu=L;
    cout<<endl<<"Isi Linked List : ";
    while (bantu->next != NULL)
    {
      cout<<bantu->Isi <<"->";
      bantu=bantu->next;
    }
  cout<<bantu->Isi;
  }
} //==================== eof================
```

b) Buatlah program menu untuk menampilkan program-program pada pertemuan XI dan XII, menggunakan perintah IF (simpan dengan nama lat12_2.cpp)

TUGAS PENDAHULUAN

1. Jelaskan perbedaan program Stack antara menggunakan Array dan Linked List!
2. Jelaskan Aplikasi-Aplikasi Stack dalam dunia nyata!
3. Tuliskan contoh program pada operasi Full!
4. Tuliskan contoh program pada operasi Empty!

TUGAS AKHIR

1. Buatlah program untuk mengetahui suatu kalimat adalah polindrom atau tidak! Polindrom adalah suatu kata atau kalimat yang jika dibaca dari depan akan sama maknanya dengan jika dibaca dari belakang. Contoh : "KASUR NABABAN RUSAK" maka jika kalimat tersebut dibalik akan mempunyai makna yang sama yaitu: "KASUR NABABAN RUSAK"

PERTEMUAN XIII
QUEUE

TUJUAN PRAKTIKUM

a) Mahasiswa dapat menjelaskan pengerian dan pembuatan Queue dengan C++.
b) Mahasiswa dapat melakukan operasi penyisipan dan penghapusan elemen dalam queue dengan C++.
c) Mahasiswa dapat mengimplementasikan Queue dengan C++.

TEORI DASAR

a) Pendahuluan

Queue atau Antrian merupakan kumpulan elemen dengan penyisipan dan penghapusan elemen yang dilakukan dari sisi/gerbang yang berbeda. Penyisipan dilakukan dari gerbang belakang dan penghapusan dilakukan dari gerbang depan. Hal ini menunjukan bahwa untuk Queue mempunyai dua gerbang yaitu gerbang depan dan gerbang belakang. Dengan demikian dapat dilihat bahwa Queue mempunyai sifat FIFO (*first In Firs Out)*, yaitu elemen yang pertama masuk akan keluar pertama juga. Queue dapat direpresentasikan dengan menggunakan Array atau Linked List.

b) Operasi-operasi pada Queue

Sama halnya dengan stack, operasi yang dapat dilakukan pada suatu Queue pada dasarnya adalah penyisipan elemen dan penghapusan elemen. Disamping itu juga dapat dilakukan operasi untuk mengecek apakah Queue kosong atau penuh, macam-macam operasi untuk mengecek Queue antara lain :

1) Operasi Inisialisasi
2) Operasi Queue Kosong
3) Operasi Queue Penuh
4) Operasi Mengosongkan Queue
5) Operasi Penyisipan Elemen Queue
6) Operasi Penghapusan Elemen Queue
7) Operasi Pencetakan Isi Queue

TUGAS PRAKTIKUM

a) Buatlah program untuk operasi Queue dengan menggunakan Singly Linked List (simpan dengan nama lat13_1.cpp)

```
#include <iostream.h>
#include <conio.h>
#include <stdlib.h>
typedef struct node *simpul ;
struct node
  {
    char Isi ;
    simpul Next ;
  };
//=====================
//==Prototype Function=====
//=====================
void Sisip_Belakang (simpul &L, char elemen ) ;
void Hapus_Depan (simpul &L);
void Cetak (simpul L);
//====================
//==Function Main=======
//====================
main()
{
  char huruf ;
  simpul L = NULL; //Pastikan bahwa L kosong
  int i;
  cout<<"==OPERASI PADA SINGLE LINKED LIST=="<<endl<<endl;
  //=====================
  //==Sisip Belakang========
  //=====================
  cout<<"\nPenyisipan Simpul \n\n";
  for(i=1; i<=3;i++)
  {
    cout<<"Masukan Huruf  :";
    cin>>huruf;
    Sisip_Belakang (L, huruf);
  }
  Cetak (L) ;
  //=====================
```

```
//==Hapus simpul Depan====
  //=====================
  cout<<"\nSetelah Hapus Simpul "<<endl;
  Hapus_Depan (L) ;
  Cetak(L) ;
  cout<<"\nSetelah Hapus Simpul "<<endl ;
  Hapus_Depan ( L ) ;
  Cetak (L) ;
  cout<<"\nSetelah Hapus Simpul "<<endl;
  Hapus_Depan (L) ;
  Cetak(L) ;
  cout<<"\nPenyisipan simpul \n\n";
  for (i=1 ; i<=3;i++)
  {
    cout<<"Masukan Huruf  :";
    cin>>huruf;
    Sisip_Belakang (L, huruf);
  }
  Cetak (L) ;
  cout<<"\nSetelah Hapus Simpul "<<endl;
  Hapus_Depan ( L ) ;
  Cetak (L) ;
  cout<<"\nSetelah Hapus Simpul "<<endl;
  Hapus_Depan (L) ;
  Cetak (L) ;
  getch ( ) ;
}
//************************************
//**FUNCTION SISIP SIMPUL DI BELAKANG****
//************************************
void Sisip_Belakang (simpul &L, char elemen)
{
  simpul bantu, baru;
  baru = (simpul) malloc (sizeof(simpul));
  baru-> Isi = elemen;
  baru -> Next = NULL;
  if (L == NULL)
    L= baru;
  else
  {
    bantu =L;
```

```
    while (bantu->Next != NULL)
      bantu =bantu->Next;
    bantu->Next=baru;
        }
}
//**************************************
//**FUNCTION MENCETAK ISI LINKED LIST*******
//**************************************
void Cetak (simpul L)
{
  simpul bantu;
  if(L==NULL)
    cout<<"Linked List Kosong.....\n";
  else
  {
    bantu=L;
    cout<<"\nIsi Linked List  :  ";
    while (bantu->Next != NULL)
    {
      cout<<bantu->Isi<<"->";
      bantu=bantu->Next;
    }
    cout<<bantu->Isi;
        }
}
//**************************************
//**FUNCTION HAPUS SIMPUL DEPAN********
//**************************************
void Hapus_Depan  (simpul &L)
{
  simpul Hapus;
  if(L==NULL)
    cout<<"Linked List Kosong.....\n";
  else
  {
    Hapus=L;
    L = L->Next;
    Hapus->Next = NULL;
    free(Hapus);
  }
}
//=========================eof=======================
```

TUGAS PENDAHULUAN

1. Apa yang dimaksud dengan Queue!
2. Tuliskan Deklarasi sintaks Queue!
3. Sebutkan dan Jelaskan operasi-operasi pada Queue!
4. Jelaskan Aplikasi-Aplikasi Queue dalam dunia nyata!

TUGAS AKHIR

1. Buatlah program **lat13_1.cpp** di atas, dengan menggunakan sistem menu!