PP/018/V/R2

LABORATORIUM
TEKNIK INFORMATIKA
FAKULTAS TEKNOLOGI INDUSTRI
UNIVERSITAS AHMAD DAHLAN

PETUNJUK PRAKTIKUM

EDISI KURIKULUM OBE

ALGORITMA DAN PEMROGRAMAN

Penyusun:
Drs. Wahyu Pujiyono, M.Kom.

2021

HAK CIPTA

PETUNJUK PRAKTIKUM NAMA PRAKTIKUM

Diterbitkan oleh:
Program Studi Teknik Informatika
Fakultas Teknologi Industri
Universitas Ahmad Dahlan
Jalan Ring Road Selatan, Tamanan, Banguntapan, Bantul Yogyakarta 55166

Penulis	: Nama Penyusun 1 Nama Penyusun 2
Editor	: Laboratorium Teknik Informatika, Universitas Ahmad Dahlan
Desain sampul	: Laboratorium Teknik Informatika, Universitas Ahmad Dahlan
Tata letak	: Laboratorium Teknik Informatika, Universitas Ahmad Dahlan
Ukuran/Halaman	: 21 x 29,7 cm / 70 halaman

Didistribusikan oleh:

Laboratorium Teknik Informatika
Universitas Ahmad Dahlan
Jalan Ring Road Selatan, Tamanan, Banguntapan, Bantul Yogyakarta 55166
Indonesia

KATA PENGANTAR

Alhamdulillah, modul praktikum Algoritma dan Pemrograman telah direvisi. Praktikum akan di titik beratkan pada konsep pengembangan algoritma dan diwujudkan dalam program. Untuk menuangkan algoritma, para praktikan dapat menggunakan perangkat lunak Raptor. Perangkat lunak ini dapat memvisualisasikan algoritma. Lebih dari itu, perangkat lunak Raptor dapat juga dieksekusi langkah per langkah, sehingga dapat digunakan untuk mengukur pemahaman praktikan.

Perubahan yang mendasar ini dilandaskan atas usulan dari para alumni dan fakta masih lemahnya mahasiswa informatika untuk mengembangkan solusi secara benar. Dalam berbagai makalah juga telah terbukti bahwa penggunaan Raptor dapat meningkatkan pemahaman mahasiswa dibanding langsung mengimplementasikannya dalam sebuah bahasa pemrograman. Hal ini mengingat bahwa fokus mahasiswa terpecah : mencari solusi masalah (menyusun algoritma) dan mencermati sintaks dari bahasa pemrogramannya sendiri.

Modul ini tak mungkin bisa penulis selesaikan dalam waktu singkat tanpa bantuan para asisten yang memiliki dedikasi tinggi dan kemampuan tinggi. Untuk itu, kami sampaikan penghargaan setinggi-tingginya atas peran sertanya merevisi modul praktikum Algoritma dan Pemrograman ini.

Bisa terjadi, modul ini masih ada kekurangan. Untuk itu, sudilah kiranya para pembaca memberikan masukan untuk perbaikan di masa mendatang.

Yogyakarta, 25 Februari 2022

Penyusun

Wahyu Pujiyono

Kontributor :

1. Afriq Yasin Ramadhan
2. Wisnu Arisandy
3. Indri Fitri Yadini
4. Merlinda Wibowo
5. Arif Budiarti
6. Rifky Maulana
7. Heru Widyastono
8. Nuri Priyanto
9. Ade Budi Prakoso
10. Ahsan Anwar Sandiah
11. Dimas Ragil Triatmaja
12. Astri Yatnasari
13. Rino Abdurrozaq
14. Mohammad Fajar

DAFTAR PENYUSUN

Nama Penyusun 1

Foto Penyusun 1	Tuliskan biografi penyusun 1 disini

Nama Penyusun 2

Foto Penyusun 2	Tuliskan biografi penyusun 2 disini

HALAMAN REVISI

Yang bertanda tangan di bawah ini:

Nama : Drs. Wahyu Pujiyono, M.Kom.

NIP/NIY 60910095

Jabatan : Dosen Pengampu Mata Kuliah **Algoritma dan Pemrograman**

Dengan ini menyatakan pelaksanaan Revisi Petunjuk Praktikum Algoritma dan Pemrograman untuk Program Studi Teknik Informatika telah dilaksanakan dengan penjelasan sebagai berikut:

No	Keterangan Revisi	Tanggal Revisi	Nomor Modul
1	a. Ada tambahan pre test dan post test b. Ada tambahan langkah praktikum c. Pergantian lembar kerja praktikum	4 Maret 2020	PP/018/V/R1
2	a. Pergantian template OBE	25 Februari 2022	PP/018/V/R2

Yogyakarta, 25 Februari 2022
Penyusun

Drs. Wahyu Pujiyono, M.Kom.
NIK/NIY. 60910095

HALAMAN PERNYATAAN

Yang bertanda tangan di bawah ini:

Nama : Drs. Wahyu Pujiyono, M.Kom.

NIK/NIY 60910095

Jabatan : Dosen Pengampu Mata Kuliah **Algoritma dan Pemrograman**

Menerangkan dengan sesungguhnya bahwa Petunjuk Praktikum ini telah direview dan akan digunakan untuk pelaksanaan praktikum di Semester Gasal Tahun Akademik 2020/2021 di Laboratorium Praktikum Teknik Informatika, Program Studi Teknik Informatika, Fakultas Teknologi Industri, Universitas Ahmad Dahlan.

Yogyakarta, 25 Februari 2022

Mengetahui, Ketua Kelompok Keilmuan	Kepala Laboratorium Teknik Informatika
Guntur Maulana Zamroni B.Sc., M.Kom.	**Lisna Zahrotun, S.T., M.Cs.**
NIP/NIY. 0509038402	NIY. 60150773

VISI DAN MISI PRODI TEKNIK INFORMATIKA

VISI

Menjadi Program Studi Informatika yang diakui secara internasional dan unggul dalam bidang Informatika serta berbasis nilai-nilai Islam.

MISI

1. Menjalankan pendidikan sesuai dengan kompetensi bidang Informatika yang diakui nasional dan internasional
2. Meningkatkan penelitian dosen dan mahasiswa dalam bidang Informatika yang kreatif, inovatif dan tepat guna.
3. Meningkatkan kuantitas dan kualitas publikasi ilmiah tingkat nasional dan internasional
4. Melaksanakan dan meningkatkan kegiatan pengabdian masyarakat oleh dosen dan mahasiswa dalam bidang Informatika.
5. Menyelenggarakan aktivitas yang mendukung pengembangan program studi dengan melibatkan dosen dan mahasiswa.
6. Menyelenggarakan kerja sama dengan lembaga tingkat nasional dan internasional.
7. Menciptakan kehidupan Islami di lingkungan program studi.

TATA TERTIB LABORATORIUM TEKNIK INFORMATIKA

DOSEN/KOORDINATOR PRAKTIKUM

1. Dosen harus hadir saat praktikum minimal 15 menit di awal kegiatan praktikum untuk mengisi materi dan menandatangani presensi kehadiran praktikum.
2. Dosen membuat modul praktikum, soal seleksi asisten, pre-test, post-test, dan responsi dengan berkoordinasi dengan asisten dan pengampu mata praktikum.
3. Dosen berkoordinasi dengan koordinator asisten praktikum untuk evaluasi praktikum setiap minggu.
4. Dosen menandatangani surat kontrak asisten praktikum dan koordinator asisten praktikum.
5. Dosen yang tidak hadir pada slot praktikum tertentu tanpa pemberitahuan selama 2 minggu berturut-turut mendapat teguran dari Kepala Laboratorium, apabila masih berlanjut 2 minggu berikutnya maka Kepala Laboratorium berhak mengganti koordinator praktikum pada slot tersebut.

PRAKTIKAN

1. Praktikan harus hadir 15 menit sebelum kegiatan praktikum dimulai, dan dispensasi terlambat 15 menit dengan alasan yang jelas (kecuali asisten menentukan lain dan patokan jam adalah jam yang ada di Laboratorium, terlambat lebih dari 15 menit tidak boleh masuk praktikum & dianggap Inhal).
2. Praktikan yang tidak mengikuti praktikum dengan alasan apapun, wajib mengikuti INHAL, maksimal 4 kali praktikum dan jika lebih dari 4 kali maka praktikum dianggap GAGAL.
3. Praktikan harus berpakaian rapi sesuai dengan ketentuan Universitas, sebagai berikut:
   a. Tidak boleh memakai Kaos Oblong, termasuk bila ditutupi Jaket/Jas Almamater (Laki-laki / Perempuan) dan Topi harus Dilepas.
   b. Tidak Boleh memakai Baju ketat, Jilbab Minim dan rambut harus tertutup jilbab secara sempurna, tidak boleh kelihatan di jidat maupun di punggung (khusus Perempuan).
   c. Tidak boleh memakai baju minim, saat duduk pun pinggang harus tertutup rapat (Laki-laki / Perempuan).
   d. Laki-laki tidak boleh memakai gelang, anting-anting ataupun aksesoris Perempuan.
4. Praktikan tidak boleh makan dan minum selama kegiatan praktikum berlangsung, harus menjaga kebersihan, keamanan dan ketertiban selama mengikuti kegiatan praktikum atau selama berada di dalam laboratorium (tidak boleh membuang sampah sembarangan baik kertas, potongan kertas, bungkus permen baik di lantai karpet maupun di dalam ruang CPU).
5. Praktikan dilarang meninggalkan kegiatan praktikum tanpa seizin Asisten atau Laboran.
6. Praktikan harus meletakkan sepatu dan tas pada rak/loker yang telah disediakan.
7. Selama praktikum dilarang NGENET/NGE-GAME, kecuali mata praktikum yang membutuhkan atau menggunakan fasilitas Internet.
8. Praktikan dilarang melepas kabel jaringan atau kabel power praktikum tanpa sepengetahuan laboran
9. Praktikan harus memiliki FILE Petunjuk praktikum dan digunakan pada saat praktikum dan harus siap sebelum praktikum berlangsung.
10. Praktikan dilarang melakukan kecurangan seperti mencontek atau menyalin pekerjaan praktikan yang lain saat praktikum berlangsung atau post-test yang menjadi tugas praktikum.
11. Praktikan dilarang mengubah setting software/hardware komputer baik menambah atau mengurangi tanpa permintaan asisten atau laboran dan melakukan sesuatu yang dapat merugikan laboratorium atau praktikum lain.

12. Asisten, Koordinator Praktikum, Kepala laboratorium dan Laboran mempunyai hak untuk menegur, memperingatkan bahkan meminta praktikan keluar ruang praktikum apabila dirasa anda mengganggu praktikan lain atau tidak melaksanakan kegiatan praktikum sebagaimana mestinya dan atau tidak mematuhi aturan lab yang berlaku.
13. Pelanggaran terhadap salah satu atau lebih dari aturan diatas maka Nilai praktikum pada pertemuan tersebut dianggap 0 (NOL) dengan status INHAL.

ASISTEN PRAKTIKUM

1. Asisten harus hadir 15 Menit sebelum praktikum dimulai (konfirmasi ke koordinator bila mengalami keterlambatan atau berhalangan hadir).
2. Asisten yang tidak bisa hadir WAJIB mencari pengganti, dan melaporkan kepada Koordinator Asisten.
3. Asisten harus berpakaian rapi sesuai dengan ketentuan Universitas, sebagai berikut:
   a. Tidak boleh memakai Kaos Oblong, termasuk bila ditutupi Jaket/Jas Almamater (Laki-laki / Perempuan) dan Topi harus Dilepas.
   b. Tidak Boleh memakai Baju ketat, Jilbab Minim dan rambut harus tertutup jilbab secara sempurna, tidak boleh kelihatan di jidat maupun di punggung (khusus Perempuan).
   c. Tidak boleh memakai baju minim, saat duduk pun pinggang harus tertutup rapat (Laki-laki / Perempuan).
   d. Laki-laki tidak boleh memakai gelang, anting-anting ataupun aksesoris Perempuan.
4. Asisten harus menjaga kebersihan, keamanan dan ketertiban selama mengikuti kegiatan praktikum atau selama berada di laboratorium, menegur atau mengingatkan jika ada praktikan yang tidak dapat menjaga kebersihan, ketertiban atau kesopanan.
5. Asisten harus dapat merapikan dan mengamankan presensi praktikum, Kartu Nilai serta tertib dalam memasukan/Input nilai secara Online/Offline.
6. Asisten harus dapat bertindak secara profesional sebagai seorang asisten praktikum dan dapat menjadi teladan bagi praktikan.
7. Asisten harus dapat memberikan penjelasan/pemahaman yang dibutuhkan oleh praktikan berkenaan dengan materi praktikum yang diasisteni sehingga praktikan dapat melaksanakan dan mengerjakan tugas praktikum dengan baik dan jelas.
8. Asisten tidak diperkenankan mengobrol sendiri apalagi sampai membuat gaduh.
9. Asisten dimohon mengkoordinasikan untuk meminta praktikan agar mematikan komputer untuk jadwal terakhir dan sudah dilakukan penilaian terhadap hasil kerja praktikan.
10. Asisten wajib untuk mematikan LCD Projector dan komputer asisten/praktikan apabila tidak digunakan.
11. Asisten tidak diperkenankan menggunakan akses internet selain untuk kegiatan praktikum, seperti Youtube/Game/Medsos/Streaming Film di komputer praktikan.

LAIN-LAIN

1. Pada Saat Responsi Harus menggunakan Baju Kemeja untuk Laki-laki dan Perempuan untuk Praktikan dan Asisten.
2. Ketidakhadiran praktikum dengan alasan apapun dianggap INHAL.
3. Izin praktikum mengikuti aturan izin SIMERU/KULIAH.
4. Yang tidak berkepentingan dengan praktikum dilarang mengganggu praktikan atau membuat keributan/kegaduhan.
5. Penggunaan lab diluar jam praktikum maksimal sampai pukul 21.00 dengan menunjukkan surat ijin dari Kepala Laboratorium Prodi Teknik Informatika.

Yogyakarta, 25 Februari 2022

Kepala Laboratorium Teknik Informatika

Lisna Zahrotun, S.T., M.Cs.
NIY. 60150773

DAFTAR ISI

KATA PENGANTAR 1
DAFTAR PENYUSUN 1
HALAMAN REVISI 2
HALAMAN PERNYATAAN 3
VISI DAN MISI PRODI TEKNIK INFORMATIKA 4
TATA TERTIB LABORATORIUM TEKNIK INFORMATIKA 5
DAFTAR ISI 8
DAFTAR GAMBAR 9
DAFTAR TABEL 10
PRAKTIKUM 1: SEKUEN ..... **Error! Bookmark not defined.**
PRAKTIKUM 2: KONDISIONAL 18
PRAKTIKUM 3: PERULANGAN 26
PRAKTIKUM 4: PERULANGAN (ITERATIF-REKURSIF) 30
PRAKTIKUM 5: REKURSI 33
PRAKTIKUM 6: ARRAY SATU DIMENSI 38
PRAKTIKUM 7: SORTING DAN SEARCHING 44
PRAKTIKUM 8: ARRAY DUA DIMENSI 49
PRAKTIKUM 9: ARRAY 1-2 DIMENSI 56
PRAKTIKUM 10: POINTER 59
DAFTAR PUSTAKA 66

DAFTAR GAMBAR

Gambar 1.1 Ilustrasi perkalian dua matriks ..... **Error! Bookmark not defined.**
Gambar 5.1. Contoh ilustrasi pengurutan decrease and conquer dengan selection sort ..... **Error! Bookmark not defined.**
Gambar 6.1. Contoh graf berarah ..... **Error! Bookmark not defined.**
Gambar 6.2. Algoritma BFS ..... **Error! Bookmark not defined.**
Gambar 6.3 Algoritma DFS ..... **Error! Bookmark not defined.**
Gambar 6.4. Contoh isian folder project anda ..... **Error! Bookmark not defined.**
Gambar 6.5. Setting OpenGL pada DevC++ ..... **Error! Bookmark not defined.**
Gambar 6.6. Hasil algoritma BFS pada graf. ..... **Error! Bookmark not defined.**
Gambar 6.7. Hasil algoritma DFS pada graf ..... **Error! Bookmark not defined.**
Gambar 7.1. Algoritma Bactracking ..... **Error! Bookmark not defined.**
Gambar 7.2 Algoritma Runut-balik Untuk Pewarnaan Graf-1 ..... **Error! Bookmark not defined.**
Gambar 7.3. Algoritma Runut-balik Untuk Pewarnaan Graf-2 ..... **Error! Bookmark not defined.**
Gambar 7.4 Contoh kasus pewarnaan graf ..... **Error! Bookmark not defined.**
Gambar 7.5. Contoh isian folder project ..... **Error! Bookmark not defined.**
Gambar 7.6. Setting OpenGL pada DevC++ ..... **Error! Bookmark not defined.**
Gambar 7.7 Hasil penerapan algoritma Bactracking untuk pewarnaan simpul graf ..... **Error! Bookmark not defined.**
Gambar 8.1 Contoh kasus TSP ..... **Error! Bookmark not defined.**
Gambar 8.2. Solusi algoritma B&B pada kasus TSP ..... **Error! Bookmark not defined.**
Gambar 8.3. Contoh isian folder project ..... **Error! Bookmark not defined.**
Gambar 8.4. Setting OpenGL pada DevC++ ..... **Error! Bookmark not defined.**
Gambar 8.5. Hasil penerapan algoritma B&B pada kasus TSP ..... **Error! Bookmark not defined.**
Gambar 9.1 Contoh kasus pencarian rute terdekat dari simpul A ke simpul F..... **Error! Bookmark not defined.**
Gambar 9.2. Contoh isian folder project ..... **Error! Bookmark not defined.**
Gambar 9.3. Setting OpenGL pada DevC++ ..... **Error! Bookmark not defined.**
Gambar 9.4. Hasil penerapan algoritma A* untuk mencari rute terdekat **Error! Bookmark not defined.**
Gambar 10.1 Ilustrasi algoritma KMP ..... **Error! Bookmark not defined.**
Gambar 10.2. Ilustrasi fungsi pinggiran KMP ..... **Error! Bookmark not defined.**
Gambar 10.3. Contoh penerapan algoritma KMP ..... **Error! Bookmark not defined.**
Gambar 10.4. Contoh isian folder project ..... **Error! Bookmark not defined.**
Gambar 10.5. Setting OpenGL pada DevC++ ..... **Error! Bookmark not defined.**
Gambar 10.6. Hasil penerapan algoritma KMP ..... **Error! Bookmark not defined.**

DAFTAR TABEL

No table of figures entries found.

PRAKTIKUM 1: SEKUEN

Pertemuan ke 1

Total Alokasi Waktu : 180 menit (Alokasi waktu disesuaikan dengan RPS)
- Pre-Test : 15 menit
- Praktikum : 60 menit
- Post-Test : 45 menit
- Uji Kompetensi : 60 menit

Total Skor Penilaian : 100%
- Pre-Test : 20 %
- Praktikum : 30 %
- Post-Test : 50 %
- Uji Kompetensi : Kompeten/Belum Kompeten

1.1. TUJUAN DAN INDIKATOR CAPAIAN

Setelah mengikuti praktikum ini mahasiswa diharapkan:

1. Mahasiswa dapat membuat program menyangkut input/proses/output
2. Mahasiswa dapat membedakan untuk menentukan bagian-bagian dari sekuen (IPO)
3. Mahasiswa dapat membuat program dengan header file

1.2. TEORI PENDUKUNG

Seringkali dalam matematika atau fisika kita diminta untuk menyelesaikan suatu masalah yang dapat diselesaikan dengan rumus tertentu. Misalkan, kita diminta untuk menentukan besarnya gaya (F) bila diketahui besarnya massa (m) dan percepatan benda (a) dengan rumus F = m.a.

Masalah di atas dapat diselesaikan dengan menggunakan bahasa pemrograman. Untuk itu kita harus mengetahui komponen dari rumus untuk dikaitkan dengan variabel yang memilah komponen dari rumus di atas menjadi pernyataan yang dikenal oleh bahasa pemrograman.

Pada sisi kiri, yaitu F bisa kita gunakan untuk nama variabel **output**. Kemudian, pada sisi kanan merupakan proses, yaitu ekspresi yang mewakili proses perkalian antara m dan a. Oleh karena m dan a (selanjutnya dapat kita buat sebagai variabel) **harus** diketahui, maka variabel ini merupakan variabel input. Variabel input dapat dimasukkan lewat keyboard atau dapat diinisialisasi saat variabel dibuat. Setelah proses perkalian dilakukan, nilai yang diperoleh di-*assign* (dengan menggunakan operator =) atau diserahkan ke variabel F. Oleh karenanya, tanda = merupakan operator *assignment*.

Berikut ini diberikan versi input lewat keyboard dan input lewat inisialisasi.

	Versi input lewat keyboard
1.	#include <iostream.h>
2.	#include <conio.h>
3.	
4.	void main() {
5.	float m, a;
6.	float F;
7.	cout << "Masukkan besarnya massa = ";
8.	cin >> m;
9.	cout << "Masukkan besarnya percepatan = ";
10.	cin >> a;
11.	F = m * a;
12.	cout << "Besarnya gaya = " << F << endl;
13.	}

	Versi input lewat inisialisasi
1.	#include <iostream.h>
2.	#include <conio.h>
3.	
4.	void main() {
5.	float m = 5, a = 9.8; // inisialisasi nilai
6.	float F;
7.	cout << "Besarnya massa = " << m << endl;
8.	cout << "Besarnya percepatan = " << a << endl;
9.	F = m * a;
10.	cout << "Besarnya gaya = " << F << endl;
11.	}

1.3. ALAT DAN BAHAN

Alat dan bahan yang digunakan dalam praktikum ini yaitu:

1. Komputer.

1.4. PRE TEST

1. Buat lah algoritma menghitung jarak antara dua titik A(x1,y1) dan B(x2,y2). Perhatikan standar penyusunan algoritma yang di jelaskan di kelas.

1.5. LANGKAH PRAKTIKUM

Selain program di atas, ketik dan jalankan program Penjumlahan.cpp di bawah ini.

```
1.  #include <iostream.h>
2.  #include <conio.h>
3.  class Hitung {
4.      friend ostream& operator<<(ostream&, const Hitung&);
5.      friend istream& operator>>(istream&, Hitung&);
6.  public:
7.      Hitung();
8.      void hitung_jumlahnya(){ jumlah = (a + b + c); }
9.  private:
10.     int a,b,c;
11.     int jumlah;
12. };
13.
14. Hitung::Hitung() {
15.     cout << "Program menghitung jumlah 3 integer\n";
16.     cout << "Selamat berkarya\n";
17. }
18. istream& operator>>(istream& in, Hitung& masukan) {
19.     cout << "Masukkan nilai a : ";
20.     in >> masukan.a;
21.     cout << "Masukkan nilai b : ";
22.     in >> masukan.b;
23.     cout << "Masukkan nilai c : ";
24.     in >> masukan.c;
```

```
25.     return in;
26. }
27.
28. ostream& operator<<(ostream& out, const Hitung& keluaran) {
29.     out << "Nilai a : " << keluaran.a << endl;
30.     out << "Nilai b : " << keluaran.b << endl;
31.     out << "Nilai c : " << keluaran.c << endl;
32.     out << "Jumlah 3 integer di atas : " << keluaran.jumlah << endl;
33.     return out;
34. }
35.
36. main() {
37.      Hitung X;
38.      cin >> X;
39.      X.hitung_jumlahnya();
40.      cout << X;
41.      getch();
42.      return 0;
43. }
```

Bila masalah yang diselesaikan tidak terlalu kompleks, cara menulis program dengan menyatukan semua komponen seperti di atas. Namun, bila masalah yang dihadapi sangat kompleks, kita dapat memecah bagian di atas ke dalam komponen lain. Komponen C++ yang lazim adalah header file (dengan ekstensi .h). Ikuti langkah berikut untuk membuat program yang sama dengan cara memecah file menjadi komponen header file dan file utama yang berisi fungsi main.

1. buat workspace baru → namakan jumlah2.
2. kemudian buat header file dengan memilih menu File | New → pilih Files | C/C++ header file. Beri nama _hitung.h yang nantinya berisi definisi class.
3. Ketik program berikut :

```
1.  class Hitung {
2.      friend ostream& operator<<(ostream&, const Hitung&);
3.      friend istream& operator>>(istream&, Hitung&);
4.  public:
5.      Hitung();
6.      void hitung_jumlahnya(){ jumlah = (a + b + c); }
7.  private:
8.      int a,b,c;
9.      int jumlah;
10. };
```

Kemudian, dengan cara yang sama buat file header yang berisi implementasi class, beri nama _hitung2.h.

```
1.  #include "_hitung.h"
2.
3.  Hitung::Hitung() {
4.      cout << "Program menghitung jumlah 3 integer\n";
5.      cout << "Selamat berkarya\n";
6.  }
7.
8.  istream& operator>>(istream& in, Hitung& masukan) {
9.      cout << "Masukkan nilai a : ";
10.     in >> masukan.a;
11.     cout << "Masukkan nilai b : ";
12.     in >> masukan.b;
13.     cout << "Masukkan nilai c : ";
14.     in >> masukan.c;
15.     return in;
16. }
17.
18. ostream& operator<<(ostream& out, const Hitung& keluaran) {
```

```
19.   out << "Nilai a : " << keluaran.a << endl;
20.   out << "Nilai b : " << keluaran.b << endl;
21.   out << "Nilai c : " << keluaran.c << endl;
22.   out << "Jumlah 3 integer di atas : " << keluaran.jumlah << endl;
23.   return out;
24. }
```

Perhatikan dalam jendela Workspace, anda telah mempunyai 2 file dalam tab Header Files.

4. Langkah terakhir, membuat file utama yang berisi fungsi main. Pilih File | New | Files → C++ Source File, lalu ketik program berikut dengan nama Jumlah2.

```
1.  #include <iostream.h>
2.  #include <conio.h>
3.  #include "_hitung2.h"
4.
5.  main() {
6.      Hitung X;
7.      cin >> X;
8.      X.hitung_jumlahnya();
9.      cout << X;
10.     getch();
11.     return 0;
12. }
```

Setelah selesai, kompilasilah menggunakan Ctrl-F5.

1.6. POST TEST

1. Buat lah flowchart untuk menghitung jarak antara dua titik A(x1,y1) dan B(x2,y2)!
2. Seperti nomor 1, gunakan subprogam dalam flowchart untuk menghitung jarak antara dua titik A(x1,y1) dan B(x2,y2)!
3. Konversikan hasil dari flowchart nomor 1 dan 2 menjadi progam C++.

LEMBAR JAWABAN PRE-TEST / POST-TEST / EVALUASI PRAKTIKUM 1: SEKUEN

Nama : **NIM :**	**Asisten:** **Paraf Asisten:**	**Tanggal:** **Nilai:**

PRAKTIKUM 2: KONDISIONAL

Pertemuan ke 2

Total Alokasi Waktu : 180 menit (Alokasi waktu disesuaikan dengan RPS)
- Pre-Test : 15 menit
- Praktikum : 60 menit
- Post-Test : 45 menit
- Uji Kompetensi : 60 menit

Total Skor Penilaian : 100%
- Pre-Test : 20 %
- Praktikum : 30 %
- Post-Test : 50 %
- Uji Kompetensi : Kompeten/Belum Kompeten

2.1. TUJUAN DAN INDIKATOR CAPAIAN

Setelah mengikuti praktikum ini mahasiswa diharapkan:

1. Mahasiswa dapat membuat program kondisional 1 alternatif
2. Mahasiswa dapat membuat program kondisional 2 alternatif
3. Mahasiswa dapat membuat program kondisional banyak alternatif

2.2. TEORI PENDUKUNG

Dalam kehidupan sehari-hari, sering kita menjumpai suatu tindakan baru dilakukan apabila telah terpenuhi syaratnya. Sebagai contoh, terdapat sepatu dengan harga 75.000 rupiah. Bila kita ingin membelinya, maka kita harus mempunyai sekurang-kurangnya uang sejumlah 75.000 rupiah. Bila uang kita kurang dari 75.000 rupiah maka kita tidak dapat membelinya. Artinya tindakan membeli sepatu seharga 75.00 rupai tidak dilakukan.

Persoalan di atas dapat diselesaiakan menggunakan bahasa pemrograman. Pernyataan yang digunakan adalah pernyataan if. Dalam struktur syarat selalu ada variabel (mewakili keadaan keuangan kita) yang dibandingkan dengan syarat (yang mewakili harga sepatu). Kita dapat membuat pernyataan :

Jika uangku lebih atau sama dengan 75.000 maka aku akan membeli sepatu

Atau

If (uang >= 75000) { // pernyataan membeli sepatu }.

Namun kadang pilihan yang ada tidak hanya satu. Artinya ada alternatif pilihan yang bisa ditawarkan toko sepatu. Misalkan terdapat kelompok sepatu dengan model I seharga di antara 75 ribu sampai 200 ribu, sedang model II harganya lebih dari 200 ribu. Maka kita dapat membuat pernyataan :

Jika uangku di antara 75 ribu sampai 200 ribu maka aku akan membeli sepatu model I

Namun jika uangku lebih dari 200 ribu maka aku akan membeli sepatu model II

Atau

If (75000 <= uangku <= 200000) { // pernyataan aku akan membeli sepatu model I }

Else if (uangku > 200000) { // pernyataan aku akan membeli sepatu model II }

Namun apabila terdapat sejumlah alternatif, kita tentu menggunakan perny if – else berulang akan menyita banyak waktu. Kita dapat menggunakan switch untuk menggantikannya. (Silahkan refer ke pelajaran Pemrograman C++).

Versi pernyataan kondisional dengan 1 alternatif.

```
1.  #include <iostream.h>
2.  void main() {
3.          float uangku;
4.          cout << "Berapa uangku ?";
5.          cin >> uangku;
6.          if (uangku >= 75000)
7.                  cout << "Sepatuku baru.\n";
8.  }
```

Versi pernyataan kondisional dengan 2 alternatif.

```
1.  #include <iostream.h>
2.  void main() {
3.          float uangku;
4.          cout << "Berapa uangku ?";
5.          cin >> uangku;
```

```
6.      if ((uangku >= 75000) && (uangku <= 200000))
7.          cout << "Sepatu model I yang kubeli.\n";
8.      else if (uangku > 200000)
9.          cout << "Sepatu model II yang kubeli.\n";
10.     }
```

2.3 PRE TEST

1. Buatlah algoritma untuk menentukan input tahun tersebut kabisat atau bukan.

2.4 ALAT DAN BAHAN

Alat dan bahan yang digunakan dalam praktikum ini yaitu:

1. Komputer.
2. Software Dev C++.

2.5 LANGKAH PRAKTIKUM

Selain program di atas, ketik dan jalankan program membilang.cpp di bawah ini. Program ini mengkonversi bilangan 1 sampai dengan 11 menjadi kalimat. Amati cara kerjanya.

```
1.  #include <iostream.h>
2.  #include <conio.h>
3.  class konversi {
4.     friend istream& operator>>(istream&, konversi&);
5.  public :
6.     konversi(unsigned int b=0) { bilangan = b; }
7.     void membilang();
8.  private:
9.     unsigned int bilangan;
10. };
11. istream& operator>>(istream& in, konversi& x) {
12.    cout << "Masukkan bilangan : ";
13.    in >> x.bilangan;
14.    return in;
15. }
16.
17. void konversi::membilang() {
```

```
18.     switch (bilangan) {
19.     case 0 : cout << "nol"; break;
20.     case 1 : cout << "satu"; break;
21.     case 2 : cout << "dua"; break;
22.     case 3 : cout << "tiga"; break;
23.     case 4 : cout << "empat"; break;
24.     case 5 : cout << "lima"; break;
25.     case 6 : cout << "enam"; break;
26.     case 7 : cout << "tujuh"; break;
27.     case 8 : cout << "delapan"; break;
28.     case 9 : cout << "sembilan"; break;
29.     case 10 : cout << "sepuluh"; break;
30.     case 11 : cout << "sebelas"; break;
31.     default : cout << "di luar range\n";
32.     }
33. }
34.
35. void main() {
36.     konversi a;
37.     cin >> a;
38.     a.membilang();
39.     getch();
40. }
```

Namun adakalanya kita juga ingin membilang dengan bilangan yang melebihi 11. Programnya ditunjukkan pada program berikut. Program ini dapat mengkoversi hingga angka 99. Namakan program ini dengan membilang2.cpp.

```
1.  #include <iostream.h>
2.  #include <conio.h>
3.
4.  class konversi {
5.      friend istream& operator>>(istream&, konversi&);
6.  public :
7.      konversi(unsigned int b=0) { bilangan = b; }
8.      void membilang1();
9.      void membilang2();
10.     void membilang3();
11.     void konversikan();
12. private:
13.     unsigned int bilangan;
14. };
15.
16. istream& operator>>(istream& in, konversi& x) {
17.     cout << "Masukkan bilangan : ";
18.     in >> x.bilangan;
19.     return in;
20. }
21.
22. void konversi::konversikan() {
23.     if (bilangan <= 11) membilang1();
24.     else if (bilangan > 19) membilang3();
25.     else membilang2();
26. }
27.
28. void konversi::membilang3() {
29.     int satuan;
30.     if (bilangan > 19) {
31.         satuan = bilangan%10;
```

```
32.         bilangan = bilangan/10;
33.         konversikan();
34.         cout << " puluh ";
35.         bilangan = satuan;
36.         konversikan();
37.       }
38.   }
39.
40.   void konversi::membilang1() {
41.       switch (bilangan) {
42.       case 0 : cout << "nol"; break;
43.       case 1 : cout << "satu"; break;
44.       case 2 : cout << "dua"; break;
45.       case 3 : cout << "tiga"; break;
46.       case 4 : cout << "empat"; break;
47.       case 5 : cout << "lima"; break;
48.       case 6 : cout << "enam"; break;
49.       case 7 : cout << "tujuh"; break;
50.       case 8 : cout << "delapan"; break;
51.       case 9 : cout << "sembilan"; break;
52.       case 10 : cout << "sepuluh"; break;
53.       case 11 : cout << "sebelas"; break;
54.       default : cout << "di luar range\n";
55.       }
56.   }
57.
58.   void konversi::membilang2() {
59.       int temp;
60.       if (bilangan > 11) {
61.           bilangan %= 10;
62.           membilang1();
```

63.	cout << " belas";
64.	}
65.	}
66.	
67.	void main() {
68.	konversi a;
69.	cin >> a;
70.	a.konversikan();
71.	getch();
72.	}

2.6 POST TEST

1. Buat lah flowchart untuk menentukan tahun tersebut kabisat atau bukan.
2. Seperti nomor 1, gunakan subprogam dalam flowchart untuk menentukan tahun tersebut kabisat atau bukan.
3. Konversikan hasil dari flowchart nomor 1 dan 2 menjadi progam C++.

LEMBAR JAWABAN PRE-TEST / POST-TEST / EVALUASI PRAKTIKUM 2: KONDISIONAL

Nama : **NIM :**	**Asisten:** **Paraf Asisten:**	**Tanggal:** **Nilai:**

PRAKTIKUM 3: PERULANGAN

Pertemuan ke 3

Total Alokasi Waktu : 180 menit (Alokasi waktu disesuaikan dengan RPS)
- Pre-Test : 15 menit
- Praktikum : 60 menit
- Post-Test : 45 menit
- Uji Kompetensi : 60 menit

Total Skor Penilaian : 100%
- Pre-Test : 20 %
- Praktikum : 30 %
- Post-Test : 50 %
- Uji Kompetensi : Kompeten/Belum Kompeten

3.1. TUJUAN DAN INDIKATOR CAPAIAN

Setelah mengikuti praktikum ini mahasiswa diharapkan:

1. Mahasiswa dapat menerapkan konsep perulangan dengan berbagai macam pernyataan perulangan
2. Mahasiswa dapat mengubah dari satu bentuk perulangan ke bentuk perulangan yang lain

3.2. TEORI PENDUKUNG

Ada 3 pernyataan perulangan dalam C++, yaitu menggunakan struktur for, while dan do – while. Bentuk umum dari ketiga pernyataan tersebut adalah sebagai berikut :

Tabel 3.1 Bentuk umum perulangan

For	While	do – while

for (**i = awal; i <= akhir; i++**)
aksi;
	i = awal;
	while (**i <= akhir**) {
	aksi;
	i++;
	}
		i = awal;
		do {
		aksi;
		i++;
		} while (**i <= akhir**);

Pola di atas adalah pola perulangan naik. Ini ditunjukkan dengan i++. Tentu saja syarat aksi dilaksanakan bila akhir >= awal. Namun bila dikehendaki pola turun bagian i++ diganti dengan i--, dan syarat menjadi awal >= akhir. Perhatikan contoh berikut ini.

Perulangan pola naik : awal = 1, akhir = 4, syarat akhir >= awal terpenuhi. Program mencetak bilangan 1 sampai 4.

Tabel 3.2 Contoh perulangan naik

for loop	while loop	do while loop

#include <iostream.h>
main() {
int i;
for (i=1; i<=4; i++)
cout << " " << i;
return 0;
}
	#include <iostream.h>
	main() {
	int i=1;
	while (i <= 4) {
	cout << " " << i;
	i++;
	}
	return 0;
	}
		#include <iostream.h>
		main() {
		int i=1;
		do { cout << " " << i;
		i++;
		} while (i <= 4);
		return 0;
		}

Perulangan pola turun : awal = 4, akhir = 1, syarat awal >= aakhir terpenuhi. Program mencetak bilangan 4 sampai 1.

Tabel 3.3 Contoh perulangan turun

for loop	while loop	do while loop

#include <iostream.h>
main() {
int i;
for (i=4; i>=1; i--)
cout << " " << i;
return 0;
}
	#include <iostream.h>
	main() {
	int i=4;
	while (i >= 4) {
	cout << " " << i;
	i--;
	}
	return 0;
	}
		#include <iostream.h>
		main() {
		int i=1;
		do {
		cout << " " << i;
		i--;
		} while (i >= 1);
		return 0;
		}

3.3. ALAT DAN BAHAN

Alat dan bahan yang digunakan dalam praktikum ini yaitu:

1. Komputer.
2. Software Dev C++.

3.4. PRE TEST

1. Buatlah algoritma menghitung nilai x(pangkat)y dengan x bilangan real dan y bilangan bulat baik negatif maupun positif.

3.5. LANGKAH PRAKTIKUM

Langkah 1

Dari tabel 3.1 di atas buatlah versi raptor nya (ingat versi raptor hanya memiliki dua versi.) Amati proses yang di lakukan pada variabel i.

Langkah 2

Dengan cara yang sama lakukan untuk tabel 3.2 perhatikan perbedaan dari masing masing kondisional dari raptor yang anda buat dari tabel 1 dan tabel 2.

3.6. POST TEST

1. Buat lah flowchart untuk menghitung nilai x(pangkat)y dengan x bilangan real dan y bilangan bulat baik negatif maupun positif.
2. Seperti nomor 1, gunakan subprogam dalam flowchart untuk menghitung nilai x(pangkat)y dengan x bilangan real dan y bilangan bulat baik negatif maupun positif.
3. Konversikan hasil dari flowchart nomor 1 dan 2 menjadi progam C++.

LEMBAR JAWABAN PRE-TEST / POST-TEST / EVALUASI PRAKTIKUM 3: PERULANGAN

Nama : **NIM :**	**Asisten:** **Paraf Asisten:**	**Tanggal:** **Nilai:**

PRAKTIKUM 4: PERULANGAN (ITERATIF-REKURSIF)

Pertemuan ke 4

Total Alokasi Waktu : 180 menit (Alokasi waktu disesuaikan dengan RPS)
- Pre-Test : 15 menit
- Praktikum : 60 menit
- Post-Test : 45 menit
- Uji Kompetensi : 60 menit

Total Skor Penilaian : 100%
- Pre-Test : 20 %
- Praktikum : 30 %
- Post-Test : 50 %
- Uji Kompetensi : Kompeten/Belum Kompeten

4.1. TUJUAN DAN INDIKATOR CAPAIAN

1. Mahasiswa dapat menerapkan konsep perulangan dari iteratif menjadi bentuk rekursif dan sebaliknya.
2. Mahasiswa dapat mengkonversi pernyataan perulangan iteratif ke rekursif dan sebaliknya

4.2. TEORI PENDUKUNG

Fungsi rekursif dapat dipandang sebagai sebuah “operator”. Misalkan kita lihat kasus 5.4. di mana secara iteratif n faktorial didefinisikan sebagai :

$$n! = 1 \times 2 \times 3 \times \ldots \times n = \prod_{i=1}^{n} i$$

n faktorial didefinisikan secara rekrusif sebagai berikut :

n! = 1 , untuk n = 0 atau n = 1

= n*(n-1)! , n > 2.

Dapat dilihat ternyata pada sisi kanan masih terdapat operator yang sama dengan sisi kiri. Dengan demikian, nilai n faktorial baru bisa diperoleh bila (n-1) faktorial telah diperoleh. Tentu saja n faktorial “lebih kompleks” dibanding dengan (n-1) faktorial. Pencarian ini berakhir bila sudah sampai pada nilai konstan, yakni telah dicapai harga n=0 atau n=1 yaitu 0!=1 atau 1!=1.

Untuk itu dapat diambil kesimpulan bahwa untuk fungsi rekursif ada 2 syarat, yaitu :

1. kasus penyetop. Dalam kasus ini terdapat nilai konstan

2. kasus pemanggilan rekursif. Dalam kasus ini terdapat pemanggilan fungsi itu sendiri, tetapi harus mengarah kepada kasus penyetop.

4.3. PRE TEST

1. Buatlah algoritma mencetak bilangan yang habis dibagi 5 dan 7 antara 1 sampai dengan 100 secara iteratif kemudian dirubah ke rekursif.

4.4. ALAT DAN BAHAN

Alat dan bahan yang digunakan dalam praktikum ini yaitu:

1. Komputer.
2. Software Dev C++.

4.5. LANGKAH PRAKTIKUM

1. Ketiklah progam di bawah ini.

```
1  class Operator {
2      friend ostream& operator<<(ostream&, Operator&);
3      friend istream& operator>>(istream&, Operator&);
4  public:
5      long faktorial();
6      long faktorial(int);
7  private:
8      int n;
9      long hasil;
10 };
11
12 long Operator::faktorial() {
13     long fak = 1;
14     for (int i = 1; i<=n; i++)
15         fak = fak * i;
16     return fak;
17 }
18
19 long Operator::faktorial(int n)
20 { if ((n==0) || (n==1)) return(1);
21     else return (n*faktorial(n-1));
22 }
```

2. Untuk menelusuri progam di atas buatlah raptor dari baris 12 sampai baris 17 amati nilai variabel fak dan i.

3. Buatlah raptor dari baris 19 sampai baris 22 jalankan raptor dan amati perubahan dari nilai variabel n.

4.6. POST TEST

1. Buat lah flowchart untuk mencetak bilangan yang habis dibagi 5 dan 7 antara 1 sampai dengan 100 secara iteratif kemudian dirubah ke rekursif.
2. Seperti nomor 1, gunakan subprogam dalam flowchart untuk mencetak bilangan yang habis dibagi 5 dan 7 antara 1 sampai dengan 100 secara iteratif kemudian dirubah ke rekursif.
3. Konversikan hasil dari flowchart nomor 1 dan 2 menjadi progam C++.

LEMBAR JAWABAN PRE-TEST / POST-TEST / EVALUASI PRAKTIKUM 4: PERULANGAN (ITERATIF-REKURSIF)

Nama : **NIM :**	**Asisten:** **Paraf Asisten:**	**Tanggal:** **Nilai:**

PRAKTIKUM 5: REKURSI

Pertemuan ke 5

Total Alokasi Waktu : 180 menit (Alokasi waktu disesuaikan dengan RPS)
- Pre-Test : 15 menit
- Praktikum : 60 menit
- Post-Test : 45 menit
- Uji Kompetensi : 60 menit

Total Skor Penilaian : 100%
- Pre-Test : 20 %
- Praktikum : 30 %
- Post-Test : 50 %
- Uji Kompetensi : Kompeten/Belum Kompeten

5.1. TUJUAN DAN INDIKATOR CAPAIAN

1. Mahasiswa dapat menerapkan konsep perulangan menggunakan teknik rekursi
2. Mahasiswa dapat mengkonversi pernyataan perulangan iteratif ke rekursif dan sebaliknya

5.2. TEORI PENDUKUNG

Fungsi atau prosedur yang memanggil dirinya sendiri dinamakan fungsi atau prosedur rekursi. Prinsip dari proses rekursi adalah sebagai berikut :

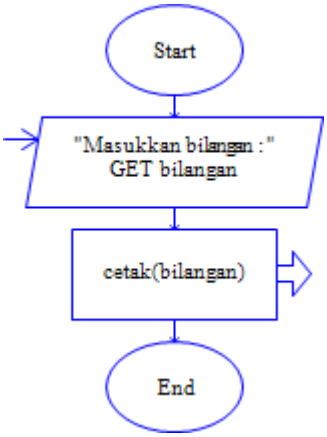
1. Memiliki kasus non rekursi (sederhana)
2. Kasus awal diarahkan menuju ke kasus sederhana
3. Mendefinisikan proses rekursi.

Dalam bentuk pernyataan, biasanya menggunakan pernyataan if (atau if … else) sebagai berikut :

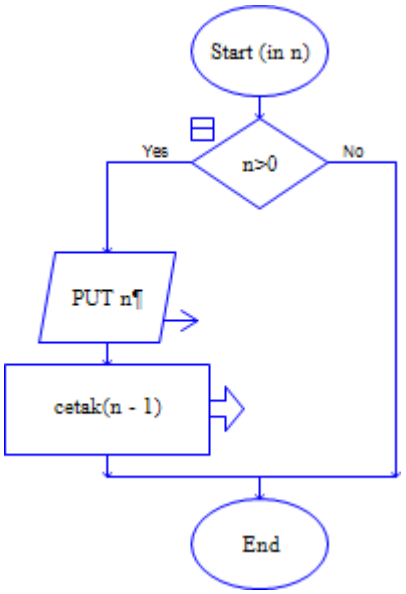
if (kasus sederhana) then eksekusi solusinya

else buat definisi ulang dari kasus menuju kasus sederhana

Contoh untuk rekursif dapat kita lihat pada raptor berikut ini:



Pada raptor diatas ada pemanggilan fungsi cetak, dan berikut fungsi cetaknya:



Perbedaan (atau cara konversi) dapat dilihat pada fungsi berikut ini :

Perulangan for	Perulangan rekursif
void perulangan::ulang_for(int n) {	void perulangan::ulang_rekursif(int n) {
for (int i = 1; i <=n; i++)	if (n >= 1) {
cout << i << " ";	ulang_rekursif(n-1);
}	cout << n << " ";
	}
	}

Perbedaan utama antara fungsi rekursi dan prosedur rekursi adalah pada bagian aksi setelah pernyataan if. Fungsi rekursi memerlukan else karena mengembalikan nilai (untuk kasus sederhana yang harus memiliki nilai) kemudian baru pemanggilan rekursi. Dalam prosedur rekursi, karena tidak

memerlukan pengembalian nilai maka setelah pernyataan if hanya ada pemanggilan rekursi. Kapan prosedur rekursi selesai? Ini terleltak pada kondisi pernyataan if-nya. Kebalikan (atau nilai pernyataan if menjadi FALSE) akan menghentikan proses rekursinya.

Perulangan for	Perulangan rekursif
Mulai i = 1	Nilai parameter n berakhir bila n < 1, dengan kata lain aksi akan diulang bila nilai n >= 1
Menaik i++	Parameter ditambah dengan 1, yaitu n+1
Implisit perulangan dalam for	Eksplisit perulangan dengan memanggil fungsinya sendiri.

5.3. PRE TEST

1. Buatlah algoritma fungsi rekursif untuk menyelesaikan deret dibawah ini:

$$1-\frac{1}{2}+\frac{1}{3}-\frac{1}{4}+\ldots+\frac{1}{n}$$

Gunakan prinsip dari proses rekursi untuk menyelesaikan algoritma di atas.

5.4. ALAT DAN BAHAN

Alat dan bahan yang digunakan dalam praktikum ini yaitu:

1. Komputer.
2. Software Dev C++.

5.5. LANGKAH PRAKTIKUM

Program selengkapnya dalam bentuk class diberikan di bawah ini. Amati setiap perulangan dengan menambahkan fungsi getch() setiap kali perulangan dilakukan.

```
1.  #include <iostream.h>
2.  class perulangan {
3.  public:
4.          perulangan() { n = 4; }
5.          void ulang_for(int);
6.          void ulang_rekursif(int);
7.  private:
8.          int n; };
9.  void perulangan::ulang_for(int n) {
10.         for (int i = 1; i <=n; i++)
11.                 cout << i << " ";
```

```
12.  }
13.  void perulangan::ulang_while(int n) {
14.          int i = 4;
15.          while (i >= n) {
16.                  cout << i << " ";
17.          i--; }
18.  }
19.  void main() {
20.          perulangan X;
21.          cout << "Menggunakan cara iteratif\n";
22.          X.ulang_for(4);
23.          cout << "\nMenggunakan cara rekursif\n";
24.          X.ulang_rekursif(4);
25.  }
```

5.6. POST TEST

1. Buat lah flowchart untuk membuat fungsi rekursif untuk menyelesaikan deret dibawah ini:

$$1-\frac{1}{2}+\frac{1}{3}-\frac{1}{4}+\ldots+\frac{1}{n}$$

2. Seperti nomor 1, gunakan subprogam dalam flowchart untuk membuat fungsi rekursif untuk menyelesaikan deret dibawah ini:

$$1-\frac{1}{2}+\frac{1}{3}-\frac{1}{4}+\ldots+\frac{1}{n}$$

3. Konversikan hasil dari flowchart nomor 1 dan 2 menjadi progam C++.

LEMBAR JAWABAN PRE-TEST / POST-TEST / EVALUASI PRAKTIKUM 5: REKURSI

Nama : **NIM :**	**Asisten:** **Paraf Asisten:**	**Tanggal:** **Nilai:**

PRAKTIKUM 6: ARRAY SATU DIMENSI

Pertemuan ke 6

Total Alokasi Waktu : 180 menit (Alokasi waktu disesuaikan dengan RPS)
- Pre-Test : 15 menit
- Praktikum : 60 menit
- Post-Test : 45 menit
- Uji Kompetensi : 60 menit

Total Skor Penilaian : 100%
- Pre-Test : 20 %
- Praktikum : 30 %
- Post-Test : 50 %
- Uji Kompetensi : Kompeten/Belum Kompeten

6.1. TUJUAN DAN INDIKATOR CAPAIAN

1. Mahasiswa dapat membuat dan memanipulasi array satu dimensi
2. Mahasiswa dapat membuat aplikasi menggunakan array satu dimensi

6.2. TEORI PENDUKUNG

Dalam kehidupan sehari-hari, kita dapat menemukan sekelompok benda pada tempat tertentu. Sebagai contoh : sederetan buku di rak, VCD yang ditempatkan pada lemari, kelas yang berisi murid-murid dan sebagainya. Kita dapat memberikan nomor sebagai tempat pada sebuah objek. Misalkan buku berjudul Harry Potter : Chamber of Secret ada pada deretan ke-5 dari kiri. Kita dapat menganggap posisi kiri sebagai awal (sering dinamakan offset) penomoran.

Andaikan kita mempunyai 5 buah data berupa integer, yaitu 8, 3, 9, 2, dan 5. Data ini akan kita tempatkan pada tempat seperti di bawah ini.

Data	8	3	9	2	5
Posisi ke-	**[0]**	**[1]**	**[2]**	**[3]**	**[4]**

Kita bisa mengidentifikasi setiap elemen dari setiap tempat (selanjutnya kita namakan array atau larik) di atas. Misalkan saja nama array di atas adalah **data.** Kita bisa menunjuk elemen ke-3 dengan data[3]. Namun kita juga bisa menunjuk elemen ke-0 sampai dengan elemen ke-4. Untuk melakukan

hal itu tentu saja kita tidak akan mengatakan data[0], data[1], ..., data[4]. Oleh karena tempatnya urut naik kita dapat menggunakan struktur for untuk menggantikannya :

```
for (int i=0; i<=4, i++)
        cout << data[i];
```

Artinya kita menjalani setiap elemen dari elemen ke-0 sampai dengan elemen ke-4. Variabel yg mewakili “perjalanan” ini adalah variabel i.

Manipulasi array tidaklah hanya menampilkan saja. Kita bisa, misalnya, menambah setiap elemen data di atas dengan angka 3. Pernyataan yg dapat dipakai adalah :

```
for (int i=0; i<=4, i++)
        data[i] = data[i] + 3;
```

Pernyataan for di atas dapat berarti juga “untuk semua nilai i berlaku ... “. Sehingga pernyataan for di atas dapat diartikan sebagai :

Untuk setiap data dari data ke-0 sampai data ke-4 tambahkan nilanya dengan 3.

Kitapun juga dapat menghitung berapa jumlah semua nilai dalam array data di atas menggunakan pernyataan :

```
total = 0;
for (int i=0; i<=4, i++)
        total = total + data[i];     // gantikan dengan total += data[i];
```

6.3. PRE TEST

1. NIM ganjil buatlah algoritma dan flowchart untuk menggeser ke kiri elemen array baik secara iteratif maupun rekursi.

2. NIM genap buatlah algoritma dan flowchart untuk menggeser ke kanan elemen array baik secara iteratif maupun rekursi.

6.4. ALAT DAN BAHAN

Alat dan bahan yang digunakan dalam praktikum ini yaitu:

1. Komputer.
2. Dev C++.
3. OpenGL Library.

6.5. LANGKAH PRAKTIKUM

Class Vektor di bawah ini dapat digunakan untuk tipe data abstrak sebuah array beserta dengan manipulasi yg dilakukan pada array. Ada 2 method yang digunakan untuk memanipulasi array yaitu :

1. penjumlahan_vektor, digunakan untuk menjumlah dua array
2. perkalian_vektor, digunakan untuk mengalikan setiap elemen array dengan bilangan integer

Class Vektor selengkapnya disajikan di bawah ini, namakan vektor.h.

```
1.  class Vektor {
2.      friend ostream& operator<<(ostream&, Vektor&);
3.      friend istream& operator>>(istream&, Vektor&);
4.  public:
5.      Vektor();
6.      void penjumlahan_vektor(const Vektor&, const Vektor&);
7.      void perkalian_vektor(float, const Vektor&);
8.      void beri_nilaiBanyak(int);
9.  private:
10.     int elemen[100];
11.     int banyak;
12. };
13. // konstruktor class digunakan untuk memberi nilai ke bagian private data
14. Vektor::Vektor() {
15.     banyak = 3;
16.     for (int i=0; i<banyak; i++)
17.         elemen[i] = 0;
18. }
19.
20. void Vektor::beri_nilaiBanyak(int i) {
21.     banyak = i;
22. }
23. istream& operator>>(istream& in, Vektor& A) {
24.     cout << "\nBanyak elemen : ";
25.     in >> A.banyak;
26.     cout << "Masukkan data vektor\n";
27.     for (int i=0; i<A.banyak; i++) {
```

```
28.         cout << "Data [" << i+1 << "] : ";
29.         cin >> A.elemen[i];
30.     }
31.     return in;
32. }
33. ostream& operator<<(ostream& out, Vektor& A) {
34.     cout << endl;
35.     for (int i=0; i<A.banyak; i++)
36.         cout << "s[" << i+1 << "] = " << setw(5) << A.elemen[i] << "\n";
37.     return out;
38. }
39. void Vektor::penjumlahan_vektor(const Vektor& A, const Vektor& B) {
40.     if (A.banyak > B.banyak) banyak = A.banyak;
41.         else banyak = B.banyak;
42.     for (int i=0; i<banyak; i++)
43.         elemen[i] = A.elemen[i] + B.elemen[i];
44. }
45. void Vektor::perkalian_vektor(float k, const Vektor& A) {
46.     banyak = A.banyak;
47.     for (int i=0; i<banyak; i++)
48.         elemen[i] = k*A.elemen[i];
49. }
```

Implementasi untuk menggunakan class Vektor di atas diberikan berikut ini, namakan Vektor.cpp.

```
1. #include <iostream.h>
2. #include <iomanip.h>
3. #include "vektor.h"
4. void main() {
5.     Vektor X, Y, Z;
6.     cin >> X;
7.     cout << X;
```

8.	cin >> Y;
9.	cout << Y;
10.	Z.penjumlahan_vektor(X,Y);
11.	cout << "\nHasil penjumlahan 2 vektor\n" << Z;
12.	Z.perkalian_vektor(3,X);
13.	cout << "\nHasil perkalian skalar dengan vektor\n" << Z;
14.	cout << Z;
15.	getch();
16.	}

6.6. POST TEST

1. Buat lah flowchart untuk menggeser ke kiri atau ke kanan elemen array baik secara iteratif maupun rekursi.

2. Seperti nomor 1, gunakan subprogam dalam flowchart untuk menggeser ke kiri atau ke kanan elemen array baik secara iteratif maupun rekursi.

3. Konversikan hasil dari flowchart nomor 1 dan 2 menjadi progam C++.

LEMBAR JAWABAN PRE-TEST / POST-TEST / EVALUASI PRAKTIKUM 6: ARRAY SATU DIMENSI

Nama : **NIM :**	**Asisten:** **Paraf Asisten:**	**Tanggal:** **Nilai:**

PRAKTIKUM 7: SORTING DAN SEARCHING

Pertemuan ke 7

Total Alokasi Waktu : 180 menit (Alokasi waktu disesuaikan dengan RPS)
- Pre-Test : 15 menit
- Praktikum : 60 menit
- Post-Test : 45 menit
- Uji Kompetensi : 60 menit

Total Skor Penilaian : 100%
- Pre-Test : 20 %
- Praktikum : 30 %
- Post-Test : 50 %
- Uji Kompetensi : Kompeten/Belum Kompeten

7.1 TUJUAN DAN INDIKATOR CAPAIAN

1. Mahasiswa dapat membuat program pencarian data
2. Mahasiswa dapat membuat program pengurutan data
3. Mahasiswa dapat membuat program aplikasi

7.2 TEORI PENDUKUNG

Bila kita ke perpustakaan, kita mudah sekali menemukan buku yang kita kehendaki. Kemudahan ini disebabkan buku-buku di perpustakaan sudah ditata dengan urut menurut aturan tertentu.

Apabila kita ingin mengetahui nomor telepon teman, kita dapat mencarinya dengan mudah pada buku telepon. Ini bisa terjadi karena nomor telepon seseorang dihubungkan dengan namanya yang telah terurut pada buku telepon.

Persoalan mencari data menjadi sangat penting dalam bidang informatika. Secara umum, data yang telah terurut lebih mudah dicari apabila kita membutuhan data tertentu. Usaha mengurutkan data menjadi usaha yang tidak pernah berhenti dilakukan untuk mencari cari yang efisien (cepat). Kita mengenal pengurutan dengan metode : gelembung (bubble), seleksi (selection), penyisipan (insertion), quick, dan merge (penggabungan). Apabila data telah terurut, maka metode pencarian yang paling tepat adalah metode binary search.

7.3 PRE TEST

1. Mengurutkan deret bilangan **9 2 1 4 11 10 18 6** dengan buble sort dan tampilkan tiap langkah sortingnya secara manual.

2. Buatlah algoritma mencari huruf **K** pada kalimat "Algoritma Pemrograman Menyenangkan" dan terdapat pada index berapa.

7.4 ALAT DAN BAHAN

Alat dan bahan yang digunakan dalam praktikum ini yaitu:

1. Komputer.
2. DevC++.
3. OpenGL Library.

7.5 LANGKAH PRAKTIKUM

Untuk praktikum, kita menggunakan pengurutan menggunakan cara seleksi. Class Sorting yang menerapkan metode pengurutan seleksi adalah sebagai berikut :

```
1.  class Sorting {
2.      friend istream& operator>>(istream& in, Sorting& a);
3.      friend ostream& operator<<(ostream& out, Sorting& a);
4.  public:
5.      void selection_sort();
6.      int pencarianBiner(int low, int high);
7.  private:
8.      void minimum(int , int, int&);
9.      void tukar (int&, int&);
10.     int data[10], n;
11. };
12. void Sorting::minimum(int dari, int n, int &tempat)
13. {
14.      int min = data[dari];
15.     tempat = dari;
16.     for (int i = dari+1; i<n; i++)
17.         if (data[i] < min)
18.         {   min = data[i];
19.             tempat = i;
20.         }
```

```
21.  }
22.  void Sorting::tukar (int &a, int &b)
23.  {  int temp;
24.      temp = a;
25.      a = b;
26.      b = temp;
27.  }
28.  void Sorting::selection_sort()
29.  {  int t;
30.     for (int i = 0; i<n; i++) {
31.         minimum(i, n, t);
32.         tukar(data[i], data[t]);
33.     }
34.  }
35.
36.  int Sorting::pencarianBiner(int low, int high)
37.  {    int middle;
38.      while (low <= high) {
39.          middle = (low+high) / 2;
40.          cetakBaris(low, middle, high);
41.          if (kunciPencarian == data[middle])
42.                return middle;
43.          else if (kunciPencarian < data[middle])
44.                high = middle - 1;
45.          else low = middle + 1;
46.      }
47.      return -1;
48.  }
```

Untuk implementasi operator overloading input dan output, silahkan lihat pada diktat.

7.6 POST TEST

1. Buat lah flowchart untuk mengurutkan deret bilangan **9 2 1 4 11 10 18 6** dengan buble sort dan tampilkan tiap langkah sortingnya secara manual.

2. Seperti nomor 1, gunakan subprogam dalam flowchart untuk mengurutkan deret bilangan **9 2 1 4 11 10 18 6** dengan buble sort dan tampilkan tiap langkah sortingnya secara manual.

3. Konversikan hasil dari flowchart nomor 1 dan 2 menjadi progam C++.

.

LEMBAR JAWABAN PRE-TEST / POST-TEST / EVALUASI PRAKTIKUM 7: SORTING DAN SEARCHING

Nama : **NIM :**	**Asisten:** **Paraf Asisten:**	**Tanggal:** **Nilai:**

PRAKTIKUM 8: ARRAY DUA DIMENSI

Pertemuan ke 8

Total Alokasi Waktu : 180 menit (Alokasi waktu disesuaikan dengan RPS)
- Pre-Test : 15 menit
- Praktikum : 60 menit
- Post-Test : 45 menit
- Uji Kompetensi : 60 menit

Total Skor Penilaian : 100%
- Pre-Test : 20 %
- Praktikum : 30 %
- Post-Test : 50 %
- Uji Kompetensi : Kompeten/Belum Kompeten

8.1 TUJUAN DAN INDIKATOR CAPAIAN

1. Mahasiswa dapat membuat dan memanipulasi array dua dimensi
2. Mahasiswa dapat membuat aplikasi menggunakan array dua dimensi

8.2 TEORI PENDUKUNG

Kita turunkan pengertian array dua dimensi dari pengertian array satu dimensi pada praktikum sebelumnya. Apabila kita perhatikan, pada rak buku terdapat larik dari atas ke bawah. Andaikan dalam sebuah rak buku terdapat 4 larik, maka buku berjudul Harry Potter : Chamber of Secret dapat dilihat pada larik ke-2 deretan ke-5 dari kiri. Dengan demikian kita mempunyai dua offset, posisi larik sebagai offset pertama (dari atas ke bawah), posisi deret sebagai offset kedua (dari kiri ke kanan).

Larik [1]							
Larik [2]					**Harry.**		
Larik [3]							
Larik [4]							
	Deret [1]	Deret [2]	Deret [3]	Deret [4]	Deret [5]	Deret [6]	Deret [7]

Contoh lain, kita dapat melihat posisi tempat duduk pada sebuah gedung bioskop. Baris tempat duduk biasanya menggunakan huruf sedangkan posisi tempat duduk (atau kolom) menggunakan angka. Jika seseorang ingin memperoleh tempat duduk pada posisi balkon di tengah

(andaikan posisi balkon pada mulai pada baris K dan nomor kursi dari 1 sampai 30) dia dapat memperkirakan nomor tiket yang dikehendaki, misalkan nomor K15. Ini juga berlaku untuk pembelian tiket kereta api, dengan nomor gerbang mulai dari posisi 1, biasanya terdapat 15 baris bernomor 1 sampai 15 dan 4 kolom dari A sampai D. Tempat duduk dekat jendela pada kolom A dan D, sedangkan posisi tengah (dekat jalan) pada kolom B dan C. Sehingga bila kita ingin pesan tiket pada posisi tengah kita dapat memesan tiket nomor 7B atau 7C.

Secara umum, array dua dimensi dapat kita pandang sebagai matriks. Matriks mempunyai dua offset, yaitu baris dan kolom. Pada setiap baris dan kolom tertentu, kita dapat meletakkan data yang akan kita simpan. Misalkan kita punya matriks :

$$\begin{bmatrix} 4 & -1 & 7 & 3 \\ 0 & 2 & -5 & 6 \\ -2 & 8 & -3 & 9 \end{bmatrix}$$

Matriks di atas dapat kita simpan dalam array dua dimensi (dalam bahasa C++, offset array dimulai dari 0) berikut ini :

Baris [0]	**4**	**-1**	**7**	**3**
Baris [1]	**0**	**2**	**-5**	**6**
Baris [2]	**-2**	**8**	**-3**	**9**
	Kolom [0]	Kolom [1]	Kolom [2]	Kolom [3]

Implementasi penyimpanan array dua dimensi di atas dapat kita buat class sebagai berikut :

```
1.  class Matriks {
2.      friend ostream& operator<<(ostream&, Matriks&);
3.      friend istream& operator>>(istream&, Matriks&);
4.  public:
5.      Matriks();
6.      void matriks_jumlah(const Matriks&, const Matriks&);
7.      void perkalian_matriks(const Matriks&, const Matriks&);
8.      void beri_nilaiBaris(int);
9.      void beri_nilaiKolom(int);
10. private:
```

```
11.     int A[10][10];
12.     int baris, kolom;
13. };
```

Setiap objek dari class Matriks di atas akan diidentifikasi sebagai array dua dimensi yang maksimal barisnya adalah 10 dan maksimal kolomnya 10. Pada konstruktor class Matriks pada umumnya kita gunakan untuk memberikan nilai awal 0 pada setiap elemen matriks.

8.3 PRE TEST

1. Diketahui 2 buah array 2 dimensi 3x3. Gabungkan 2 array tersebut menjadi satu dimensi kemudian lakukan penyortingan di mulai dari yang terkecil. Buatlah algoritma hingga program C++nya.

8.4 ALAT DAN BAHAN

Alat dan bahan yang digunakan dalam praktikum ini yaitu:

1. Komputer.
2. DevC++.
3. OpenGL Library.

8.5 LANGKAH PRAKTIKUM

Ketiklah program operasi matriks berikut ini. Amati tata cara setiap operasi matriks berlaku pada setiap elemen matriks. Buat kesimpulan tentang pentingnya syarat setiap operasi matriks dapat dilakukan.

```
1.  #include <iostream.h>
2.  #include <conio.h>
3.  #include <stdio.h>
4.  #include <iomanip.h>
5.
6.  class Matriks {
7.      friend ostream& operator<<(ostream&, Matriks&);
8.      friend istream& operator>>(istream&, Matriks&);
9.  public:
10.     void baca_matriks();
11.     void matriks_jumlah(const Matriks&, const Matriks&);
12.     void perkalian_matriks(const Matriks&, const Matriks&);
13.     void cetak_matriks();
14. private:
```

```
15.   int A[10][10];
16.   int baris, kolom;
17.   };
18.   void Matriks::baca_matriks ()
19.   {   int i,j;
20.       for (i=0; i<baris; i++)
21.           for (j=0; j<kolom; j++) {
22.               cout << "Data [" << i+1 << "," << j+1 << "] : ";
23.               cin >> A[i][j];
24.           }
25.   }
26.   void Matriks::matriks_jumlah(const Matriks& matriks1, const Matriks& matriks2)
27.   {   int i,j;
28.       baris = matriks1.baris;
29.       kolom = matriks1.kolom;
30.       for (i=0; i<baris; i++)
31.           for (j=0; j<kolom; j++)
32.               A[i][j]=matriks1.A[i][j]+matriks2.A[i][j];
33.       cetak_matriks();
34.   }
35.   void Matriks::perkalian_matriks(const Matriks& matriks1, const Matriks& matriks2)
36.   {   int i,j,k;
37.       int barkol;
38.       baris = matriks1.baris;
39.       kolom = matriks1.kolom;
40.       barkol = matriks1.kolom;
41.       for (i=0; i<baris; i++)
42.           for (j=0; j<kolom; j++) {
43.               A[i][j] = 0;
44.               for (k=0; k<barkol; k++)
45.                   A[i][j] = A[i][j] + matriks1.A[i][k] * matriks2.A[k][j];
```

```
46.             }
47.     cetak_matriks();
48. }
49.
50. void Matriks::cetak_matriks ()
51. {   int i,j;
52.     for (i=0; i<baris; i++)
53.     {  for (j=0; j<kolom; j++)
54.             cout << setw(5) << A[i][j] << " ";
55.         cout << endl;
56.     }
57. }
58.
59. istream& operator>>(istream& in, Matriks& A) {
60.     cout << "Banyak baris : ";
61.     in >> A.baris;
62.     cout << "Banyak kolom : ";
63.     in >> A.kolom;
64.     cout << "Masukkan data matriks\n";
65.     A.baca_matriks();
66.     cout << "Matriks yang dibuat adalah : \n";
67.     A.cetak_matriks();
68.     return in;
69. }
70.
71. ostream& operator<<(ostream& out, Matriks& A) {
72.     int i,j;
73.     for (i=0; i<A.baris; i++)
74.     {  for (j=0; j<A.kolom; j++)
75.             cout << setw(5) << A.A[i][j] << " ";
76.         cout << endl;
```

```
77.     }
78.     return out;
79.   }
80.   main() {
81.     Matriks matriks1, matriks2;
82.     Matriks jumlah;
83.     cout << "Memasukkan data matriks I \n";
84.     cin >> matriks1;
85.     cout << "Memasukkan data matriks II\n";
86.     cin >> matriks2;
87.     jumlah.matriks_jumlah(matriks1,matriks2);
88.     cout << "Hasil Penjumlahan : \n";
89.     jumlah.cetak_matriks();
90.     jumlah.perkalian_matriks(matriks1,matriks2);
91.     cout << "Hasil Perkalian : \n";
92.     jumlah.cetak_matriks();
93.     getch();
94.     return 0;
95.   }
```

8.6 POST TEST

1. Buat lah flowchart untuk mengetahui 2 buah array 2 dimensi 3x3 dan gabungkan 2 array tersebut menjadi satu dimensi kemudian lakukan penyortingan (gunakan algoritma yang anda sukai) di mulai dari yang terkecil. Buatlah algoritma hingga program C++nya.
2. Seperti nomor 1, gunakan subprogam dalam flowchart untuk mengetahui 2 buah array 2 dimensi 3x3 dan gabungkan 2 array tersebut menjadi satu dimensi kemudian lakukan penyortingan (gunakan algoritma yang anda sukai) di mulai dari yang terkecil. Buatlah algoritma hingga program C++nya.
3. Konversikan hasil dari flowchart nomor 1 dan 2 menjadi progam C++.
4. Buat lah flowchart dan konversikan ke C++ untuk mengecek matrik bujur sangkar apakah matrik identitas atau bukan.

LEMBAR JAWABAN PRE-TEST / POST-TEST / EVALUASI PRAKTIKUM 8: ARRAY DUA DIMENSI

Nama : **NIM :**	**Asisten:** **Paraf Asisten:**	**Tanggal:** **Nilai:**

PRAKTIKUM 9: ARRAY 1-2 DIMENSI

Pertemuan ke 9

Total Alokasi Waktu : 180 menit (Alokasi waktu disesuaikan dengan RPS)
- Pre-Test : 15 menit
- Praktikum : 60 menit
- Post-Test : 45 menit
- Uji Kompetensi : 60 menit

Total Skor Penilaian : 100%
- Pre-Test : 20 %
- Praktikum : 30 %
- Post-Test : 50 %
- Uji Kompetensi : Kompeten/Belum Kompeten

9.1 TUJUAN DAN INDIKATOR CAPAIAN

1. Mahasiswa dapat memanipulasi indeks array 1 menjadi array 2 dimensi

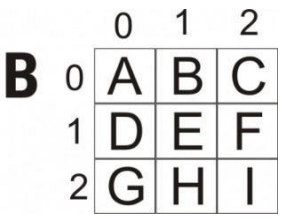
9.2 TEORI PENDUKUNG

Perhatikan konstruksi array 1 dimensi (sebagai input) dan array 2 dimensi (sebagai output) berikut :

Input :

Gambar 1 Posisi huruf di dalam array satu dimensi a

Output :



Gambar 2 Posisi huruf di dalam array 2 dimensi b

Prosesnya dapat menggunakan ilustrasi berikut ini :

Gambar 3 Proses pembentukan indeks kedua dari array 2 dimensi

Perhatikan gambar 3. Setiap bagian masuk baris ke 0 sampai dengan masuk baris ke 2 itu menjadi indeks yang baru untuk array 2 dimensi.

9.3 PRE TEST

1. Buatlah algoritma mengurutkan data array 2 dimensi berukuran 3x3 dengan menggunakan algoritma bubble sort.

Petunjuk : konversikan array 2 dimensi menjadi array 1 dimensi, urutkan menggunakan buble sort lalu kembalikan menjadi array 2 dimensi lagi.

9.4 ALAT DAN BAHAN

Alat dan bahan yang digunakan dalam praktikum ini yaitu:

1. Komputer.
2. DevC++.
3. OpenGL Library.

9.5 LANGKAH PRAKTIKUM

Berdasarkan dari landasan teori, konstruksikan algoritmanya dan buatlah flowchart lalu telusuri setiap pembentukan baris demi baris dari indeks yang baru.

9.6 POST TEST

1. Buat lah flowchart untuk mengurutkan data array 2 dimensi berukuran 3x3 dengan menggunakan algoritma bubble sort.

2. Seperti nomor 1, gunakan subprogam dalam flowchart untuk mengurutkan data array 2 dimensi berukuran 3x3 dengan menggunakan algoritma bubble sort.

3. Konversikan hasil dari flowchart nomor 1 dan 2 menjadi progam C++.

LEMBAR JAWABAN PRE-TEST / POST-TEST / EVALUASI PRAKTIKUM 9: ARRAY 1-2 DIMENSI

Nama : **NIM :**	**Asisten:** **Paraf Asisten:**	**Tanggal:** **Nilai:**

PRAKTIKUM 10:POINTER

Pertemuan ke 10

Total Alokasi Waktu : 180 menit (Alokasi waktu disesuaikan dengan RPS)
- Pre-Test : 15 menit
- Praktikum : 60 menit
- Post-Test : 45 menit
- Uji Kompetensi : 60 menit

Total Skor Penilaian : 100%
- Pre-Test : 20 %
- Praktikum : 30 %
- Post-Test : 50 %
- Uji Kompetensi : Kompeten/Belum Kompeten
-

10.1 TUJUAN DAN INDIKATOR CAPAIAN

1. Mahasiswa dapat membuat variabel dinamis
2. Mahasiswa dapat mengarahkan pointer ke tampat yang dikehendaki
3. Mahasiswa dapat membuat aplikasi pointer dalam bentuk link list

10.2 TEORI PENDUKUNG

Terdapat dua jenis variabel menurut kebutuhan program untuk menyimpan data, yaitu variabel statis dan variabel dinamis. Variabel statis adalah variabel yang harus dipesan dahlu sebelum digunakan dan banyaknya tidak dapat diubah saat program dieksekusi. Variabel dinamis adalah variabel yang dapat dipesan (dibuat) saat dibutuhkan waktu program dieksekusi, namun dapat pula dihancurkan (atau dikembalikan ke sistem operasi) saat tidak dibutuhkan.

Cara mendeklarasikan variabel dinamis cukup menambahkan tanda * di antara tipe data dan nama variabelnya. Sebagai contoh :

```
int *P;
```

adalah pernyataan yang mendeklarasikan variabel pointer yang menunjuk data yang bertipe int. Data ini bisa saja ditandai oleh variabel statis.

```
int x = 5;
```

Pernyataan P = &x; berarti bahwa pointer P digunakan untuk menunjuk data bertipe int yaitu 5 yang disimpan pada variabel x.

Dalam penggunaannya, variabel dinamis dapat menunjuk ke manapun, asal tipe data yang ditunjuk sesuai dengan tipe pointer ketika dideklarasikan. Tentu saja kita tidak bisa pernyataan :

char C = ‘x’;

P = &C;

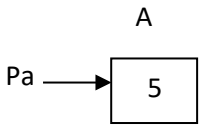
akan merupakan pernyataan yang tidak valid, oleh karena variabel C menyimpan data bertipe char, sadangkan P adalah pointer yang menunjuk data bertipe int.

Program di bawah ini memberikan cara memindahkan pointer :

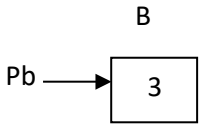
```
1.  #include <iostream.h>
2.
3.  void main() {
4.          int A = 5;
5.          int B = 3;
6.          cout << "Nilai A : " << A << endl;
7.          cout << "Nilai B : " << B << endl;
8.          int *Pa = &A;
9.          cout << "Nilai yang ditunjuk pointer Pa (var. A) : " << *Pa << endl;
10.         int *Pb = &B;
11.         cout << "Nilai yang ditunjuk pointer Pb (var. B) : " << *Pb << endl;
12.         Pa = Pb;  // ekivalen dengan Pa = &B
13.         cout << "Setelah tugas Pa diubah : \n";
14.         cout << "Nilai yang ditunjuk pointer Pb (var. B) : " << *Pb << endl;
15.         cout << "Nilai yang ditunjuk pointer Pa (var. B) : " << *Pa << endl;
16. }
```

Baris 4 dan 5 :

Baris 8 :

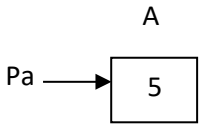


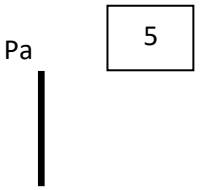
Baris 10 :



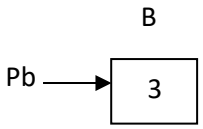
Baris 12 : Semula menjadi

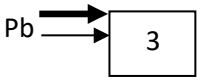
A





B

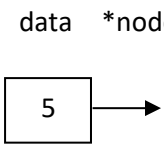




Bagaimana untuk membuat rantai data ? Diberikan definisi node (setiap mata rantai) sebagai berikut :

```
class node {
public :
        int data;
        int *node;
};
```

Gambar logik dari class node di atas adalah :



10.3 PRE TEST

1. Buatlah algoritma untuk menghitung banyaknya elemen linklist.

10.4 ALAT DAN BAHAN

Alat dan bahan yang digunakan dalam praktikum ini yaitu:

1. Komputer.
2. Dev C++.
3. OpenGL Library.

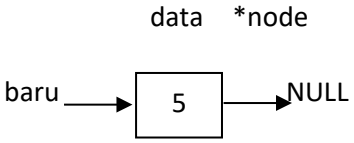
10.5 LANGKAH PRAKTIKUM

Dari teori pendukung di atas kita akan menyambung node-node di atas sehingga akan menjadi rantai data. Ketiklah progam berikut dan amati setiap output dengan logika pernyataan sebelumnya.

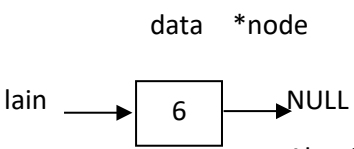
```
1.  #include <iostream.h>
2.
3.  class node {
4.  public :
5.          int data;
6.          node *berikut;
7.  };
8.
9.  void main() {
10.         // langkah satu
11.         node *baru;
12.         baru = new node;
13.         baru->data = 5;
14.         baru->berikut = NULL;
15.         cout << "Isi data node baru adalah : " << baru->data << endl;
16.         // langkah dua
17.         node *lain;
18.         lain = new node;
19.         lain->data = 6;
20.         lain->berikut = NULL;
21.         cout << "Isi data node lain adalah : " << lain->data << endl;
22.         // langkah tiga : menyambung rantai
23.         baru->berikut = lain;
24.         cout << "Isi data node lain dicetak dari node baru adalah : ";
```

```
25.         cout << baru->berikut->data << endl;
26.         // langkah empat
27.         node *kepala = baru;
28.         cout << "Mencetak node pertama dari pointer kepala : ";
29.         cout << kepala->data << endl;
30.         cout << "Mencetak node kedua dari pointer kepala : ";
31.         cout << kepala->berikut->data << endl;
32.         // langkah lima : pointer yang jalan-jalan
33.         cout << "Menggunakan perulangan untuk mencetak setiap data pada rantai\n";
34.         node *jalan = kepala;
35.         int i = 1;
36.         while (jalan != NULL) {
37.                 cout << "Data ke-" << i << " > " << jalan->data << endl;
38.                 i++;
39.                 jalan = jalan->berikut;
40.         }
41.         // langkah enam : bukti bahwa pointer kepala tidak kehilangan data
42.         cout << "Mencetak node pertama dari pointer kepala : ";
43.         cout << kepala->data << endl;
44.         cout << "Mencetak node kedua dari pointer kepala : ";
45.         cout << kepala->berikut->data << endl;
46. }
```

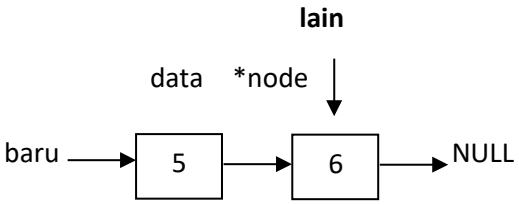
Langkah satu (baris 10) :



Langkah dua (baris 17) :



Langkah tiga (baris 23) :



10.6 POST TEST

1. Ketiklah progam dari algoritma saat pretest dan jalankan.

LEMBAR JAWABAN PRE-TEST / POST-TEST / EVALUASI PRAKTIKUM 10: POINTER

Nama : **NIM :**	**Asisten:** **Paraf Asisten:**	**Tanggal:** **Nilai:**

DAFTAR PUSTAKA

Cormen, H. Thomas, dkk. *Introduction to Algorithms Third Edition*, London, The MIT Press, 2009

Farrel, Mary, *Computer Programming for Teens*, USA, Thomson Learning Inc.,2008

Gozali, William. dan Alham Fikri Aji, *Pemrogaman Kompetitif Dasar Versi 1.4*, Indonesia, Ikatan Alumni TOKI

Hanly, Jeri. Hall and Koffmann, Elliot B. *Problem Solving and Program Design in C*, Addison Wesley, 2007

R.G. Dromey, *How Solve It by Computer*, London, Prentice Hall, 1982

Sahni, Sartaj, Data Structures, *Algorithms and Applications in C++ Second Edition*, India, Universities Press, 2005

Schildt, Herbert, *C++ A Beginner's Guide Second Edition*, Osborn, McGraw Hill, 2004

Schildt, Herb, *C++Progamming Cookbook*, Osborn, McGraw Hill, 2008


LABORATORIUM
TEKNIK INFORMATIKA
UNIVERSITAS AHMAD DAHLAN


UNIVERSITAS AHMAD DAHLAN