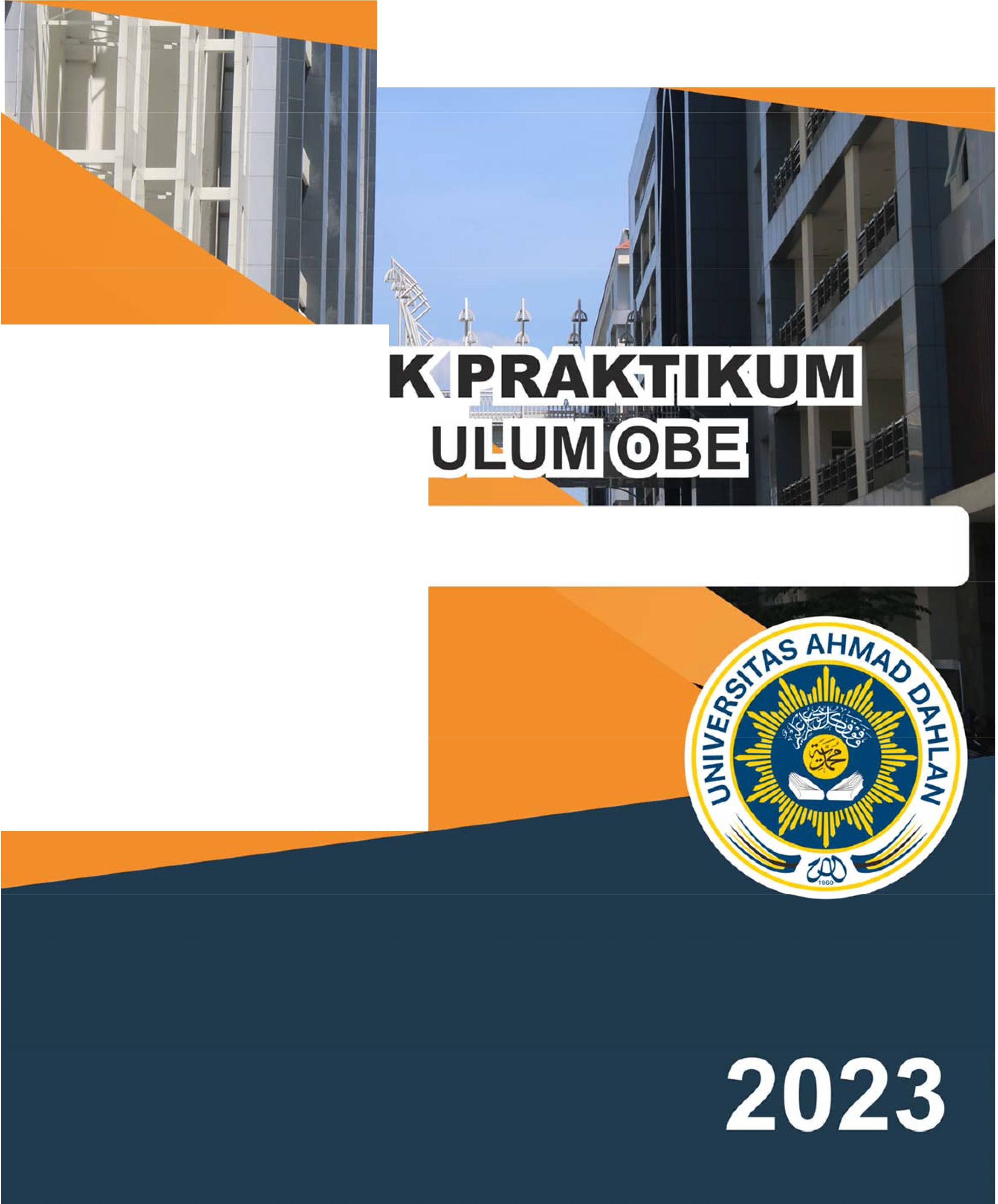


**PP/018/V/R4**



AAT LABORATORIUM

**r**

**,=t•1 S1 INFORMATIKA**

*.=::=- IJ* **FAKULTAS TEKNOLOGI INDUSTRI**

'--....,-.-/ **UNIVERSITAS AHMAD DAHLAN**

. ,

**F!ETiUN U**

E01S1 KURIK

**STRUKTUR DATA**

**Penyusun:**

**Drs. Wahyu Pujiyono, M.Kom**

# HAK CIPTA

## PETUNJUK PRAKTIKUM STRUKTUR DATA

Copyright© 2023,

Drs. Wahyu Pujiyono, M.Kom.

#### Hak Cipta dilindungi Undang‐Undang

Dilarang mengutip, memperbanyak atau mengedarkan isi buku ini, baik sebagian maupun seluruhnya, dalam bentuk apapun, tanpa izin tertulis dari pemilik hak cipta dan penerbit.

Diterbitkan oleh:

**Program Studi Informatika** Fakultas Teknologi Industri Universitas Ahmad Dahlan

Jalan Ring Road Selatan, Tamanan, Banguntapan, Bantul Yogyakarta 55166

**Penulis** : Drs. Wahyu Pujiyono, M.Kom.

**Editor** : Laboratorium S1 Informatika, Universitas Ahmad Dahlan **Desain sampul** : Laboratorium S1 Informatika, Universitas Ahmad Dahlan **Tata letak** : Laboratorium S1 Informatika, Universitas Ahmad Dahlan

**Ukuran/Halaman** : 21 x 29,7 cm /99 halaman

Didistribusikan oleh:

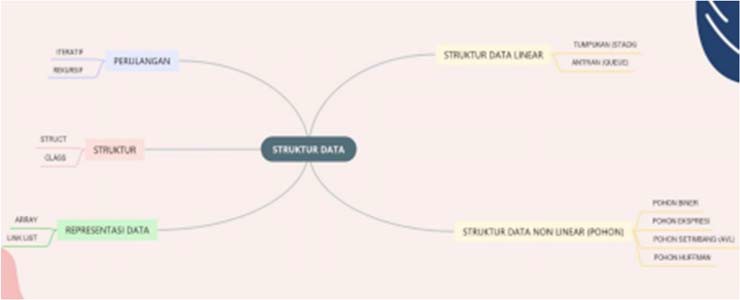
Laboratorium S1 Informatika

Universitas Ahmad Dahlan

Jalan Ring Road Selatan, Tamanan, Banguntapan, Bantul Yogyakarta 55166 Indonesia

# KATA PENGANTAR

Materi praktikum Struktur Data dikonstruksikan sebagaimana gambar dibawah ini:



Mahasiswa mempelajari Mata Kuliah Struktur Data dibagi menjadi lima bagian. Pada materi perulangan akan mempelajari bentuk iterative dan rekursif selanjutnnya pada materi struktur data, mahasiswa akan mempelajari struct dan class. Kemudian pada materi struktur data linear, mahasiswa akan mempelajari tumpukan (stack) dan antrian (queue). Dan yang terakhir yaitu pada materi struktur data non linear (pohon), akan mempelajari pohon biner, pohon ekspresi, pohon setimbang (avl), dan pohon Huffman.

Yogyakarta, 10 Juli 2023

Penyusun

Drs. Wahyu Pujiyono, M.Kom.

# DAFTAR PENYUSUN

Drs. Wahyu Pujiyono, M.Kom.



Drs. Wahyu Pujiyono, M.Kom. atau akrab di panggil Pak Wahyu, merupakan lektor kepala dan juga dosen pengampu Mata Kuliah Struktur Data, Algoritma Pemrograman, dan Kewirausahaan di Universitas Ahmad Dahlan Yogyakarta. Beliau lahir di Banyumas pada tanggal 4 November tahun 1966. Beliau menyelesaikan S‐1 nya di Universitas Gadjah Mada dengan jurusan Matematika serta S‐2 di Universitas Indonesia dengan jurusan Ilmu Komputer. Beliau juga telah mempublikasikan beberapa artikel ilmiah serta melakukan beberapa pengabdian kepada masyarakat, salah satunya yaitu pembuatan media pembelajaran di SD Muhammadiyah Ambarketawang 3 Yogyakarta.

# KONTRIBUSI PENULIS

|  |  |
| --- | --- |
| **Nomor Bab** | **Daftar Penulis** |
| Bab I | Drs. Wahyu Pujiyono, M.Kom. |
| Bab II | Drs. Wahyu Pujiyono, M.Kom. |
| Bab III | Drs. Wahyu Pujiyono, M.Kom. |
| Bab IV | Drs. Wahyu Pujiyono, M.Kom. |
| Bab V | Drs. Wahyu Pujiyono, M.Kom. |
| Bab VI | Drs. Wahyu Pujiyono, M.Kom. |
| Bab VII | Drs. Wahyu Pujiyono, M.Kom. |
| Bab VIII | Drs. Wahyu Pujiyono, M.Kom. |
| Bab IX | Drs. Wahyu Pujiyono, M.Kom. |
| Bab X | Drs. Wahyu Pujiyono, M.Kom. |

# HALAMAN REVISI

Yang bertanda tangan di bawah ini:

Nama : Drs. Wahyu Pujiyono, M.Kom.

NIPM 19661104 199107 111 0677540

Jabatan : Dosen Pengampu Mata Kuliah **Struktur Data**

Dengan ini menyatakan pelaksanaan Revisi Petunjuk Praktikum **Struktur Data** untuk Program Studi Informatika telah dilaksanakan dengan penjelasan sebagai berikut:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| No | Keterangan Revisi |  |  |  | Tanggal Revisi | Nomor Modul |
| 1 | a. Memperbaiki  sesuai standar | template | petunjuk | praktikum | 23 Agustus 2019 | PP/018/III/R2 |
| 2 | 1. Menambahkan materi video youtube untuk dipelajar sebelum mengerjakan pretest 2. Mengubah soal postest dari pertemuan satu   sampai delapan | | | | 25 Agustus 2020 | PP/018/III/R2 |
| 3 | 1. Penyesuaian template baru 2. Penambahan CPL dan CPMK | | | | 5 Agustus 2021 | PP/018/III/R3 |
| 4 | 1. Pergantian template OBE 2. Penambahan Studi Kasus dari pertemuan satu sampai empat | | | | 10 Juli 2023 | PP/018/V/R4 |

Yogyakarta, 10 Juli 2023 Penyusun

Drs. Wahyu Pujiyono, M.Kom.

NIPM. 19661104 199107 111 0677540

# HALAMAN PERNYATAAN

Yang bertanda tangan di bawah ini:

Nama : Drs. Wahyu Pujiyono, M.Kom.

NIK/NIY 60910095

Jabatan : Dosen Pengampu Mata Kuliah **Struktur Data**

Menerangkan dengan sesungguhnya bahwa Petunjuk Praktikum ini telah direview dan akan digunakan untuk pelaksanaan praktikum di Semester Gasal Tahun Akademik 2023/2024 di Laboratorium Praktikum S1 Informatika, Program Studi Informatika, Fakultas Teknologi Industri, Universitas Ahmad Dahlan.

Yogyakarta, 10 Juli 2023

Mengetahui,

Ketua Kelompok Keilmuan



Dr. Ardiansyah, S.T., M.Cs.

NIPM : 19790723 200309 111 0932301

Kepala Laboratorium Praktikum S1 Informatika



Murein Miksa Mardhia S.T., M.T.

NIPM. 19891019 201606 011 1236278

# VISI DAN MISI PRODI INFORMATIKA

#### VISI

Menjadi program studi yang unggul dan inovatif dalam bidang rekayasa perangkat lunak dan sistem cerdas dengan dijiwai nilai‐nilai Islam

#### MISI

1. Mengimplementasikan nilai‐nilai AIK pada semua aspek kegiatan.
2. Memajukan ilmu pengetahuan dan teknologi Rekayasa Perangkat Lunak dan Sistem cerdas melalui pendidikan, penelitian, dan pengabdian kepada masyarakat.
3. Mengembangkan kerjasama dalam pendidikan, penelitian, dan pengabdian kepada masyarakat di tingkat lokal, nasional, maupun internasional.
4. Menyelenggarakan tata kelola program studi yang unggul dan inovatif.
5. Berperan aktif dalam kegiatan yang menunjang profesi dosen.

# TATA TERTIB LABORATORIUM S1 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 yang akan mengikuti inhal diwajibkan mendaftarkan diri dan membayar administrasi inhal kepada laboran inhal paling lambat H‐1 jadwal inhal.
4. Praktikan harus berpakaian rapi sesuai dengan ketentuan Universitas, sebagai berikut:
   1. Tidak boleh memakai Kaos Oblong, termasuk bila ditutupi Jaket/Jas Almamater (Laki‐laki / Perempuan) dan Topi harus Dilepas.
   2. Tidak Boleh memakai Baju ketat, Jilbab Minim dan rambut harus tertutup jilbab secara sempurna, tidak boleh kelihatan di jidat maupun di punggung (khusus Perempuan).
   3. Tidak boleh memakai baju minim, saat duduk pun pinggang harus tertutup rapat (Laki‐laki / Perempuan).
   4. Laki‐laki tidak boleh memakai gelang, anting‐anting ataupun aksesoris Perempuan.
5. 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).
6. Praktikan dilarang meninggalkan kegiatan praktikum tanpa seizin Asisten atau Laboran.
7. Praktikan harus meletakkan sepatu dan tas pada rak/loker yang telah disediakan.
8. Selama praktikum dilarang *NGENET/NGE‐GAME*, kecuali mata praktikum yang membutuhkan atau menggunakan fasilitas Internet.
9. Praktikan dilarang melepas kabel jaringan atau kabel power praktikum tanpa sepengetahuan laboran
10. Praktikan harus memiliki FILE Petunjuk praktikum dan digunakan pada saat praktikum dan harus siap sebelum praktikum berlangsung.
11. Praktikan dilarang melakukan kecurangan seperti mencontek atau menyalin pekerjaan praktikan yang lain saat praktikum berlangsung atau post‐test yang menjadi tugas praktikum.
12. 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.
13. 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.
14. 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:
   1. Tidak boleh memakai Kaos Oblong, termasuk bila ditutupi Jaket/Jas Almamater (Laki‐laki / Perempuan) dan Topi harus Dilepas.
   2. Tidak Boleh memakai Baju ketat, Jilbab Minim dan rambut harus tertutup jilbab secara sempurna, tidak boleh kelihatan di jidat maupun di punggung (khusus Perempuan).
   3. Tidak boleh memakai baju minim, saat duduk pun pinggang harus tertutup rapat (Laki‐laki / Perempuan).
   4. 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 mencatat dan merekap praktikan dengan status INHAL setiap minggu serta wajib mengumumkan mekanisme INHAL di awal pertemuan praktikum.
7. Asisten harus dapat bertindak secara profesional sebagai seorang asisten praktikum dan dapat menjadi teladan bagi praktikan.
8. 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.
9. Asisten tidak diperkenankan mengobrol sendiri apalagi sampai membuat gaduh.
10. Asisten dimohon mengkoordinasikan untuk meminta praktikan agar mematikan komputer untuk jadwal terakhir dan sudah dilakukan penilaian terhadap hasil kerja praktikan.
11. Asisten wajib untuk mematikan LCD Projector dan komputer asisten/praktikan apabila tidak digunakan.
12. 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. Pelaksanaan (waktu dan metode) INHAL sama seperti praktikum mingguan/reguler.
4. Izin praktikum mengikuti aturan izin SIMERU/KULIAH.
5. Yang tidak berkepentingan dengan praktikum dilarang mengganggu praktikan atau membuat keributan/kegaduhan.
6. Penggunaan lab diluar jam praktikum maksimal sampai pukul 21.00 dengan menunjukkan surat ijin dari Kepala Laboratorium Prodi Informatika.

Yogyakarta, 10 Juli 2023 Kepala Laboratorium Praktikum

S1 Informatika



Murein Miksa Mardhia S.T., M.T.

NIPM. 19891019 201606 011 1236278

# DAFTAR ISI

[HAK CIPTA 1](#_TOC_250021)

[KATA PENGANTAR 2](#_TOC_250020)

[DAFTAR PENYUSUN 3](#_TOC_250019)

[KONTRIBUSI PENULIS 4](#_TOC_250018)

[HALAMAN REVISI 5](#_TOC_250017)

[HALAMAN PERNYATAAN 6](#_TOC_250016)

[VISI DAN MISI PRODI INFORMATIKA 7](#_TOC_250015)

[TATA TERTIB LABORATORIUM S1 INFORMATIKA 8](#_TOC_250014)

[DAFTAR ISI 11](#_TOC_250013)

[DAFTAR GAMBAR 12](#_TOC_250012)

[DAFTAR TABEL 13](#_TOC_250011)

[SKENARIO PRAKTIKUM SECARA DARING 14](#_TOC_250010)

[PRAKTIKUM 1: CLASS DAN REKURSI 16](#_TOC_250009)

[PRAKTIKUM 2: ARRAY STATIS 28](#_TOC_250008)

[PRAKTIKUM 3: ARRAY DINAMIS 35](#_TOC_250007)

[PRAKTIKUM 4: LINK LIST 44](#_TOC_250006)

[PRAKTIKUM 5: VARIASI LINK LIST 52](#_TOC_250005)

[PRAKTIKUM 6: STACK DENGAN ARRAY 56](#_TOC_250004)

[PRAKTIKUM 7: SIMULASI QUEUE DENGAN ARRAY CIRCULAR 64](#_TOC_250003)

PRAKTIKUM 8: ANTRIAN BERPRIORITAS MENGGUNAKAN DOUBLY LINK LIST 74

[PRAKTIKUM 9: POHON 78](#_TOC_250002)

[PRAKTIKUM 10: SEKUEN 91](#_TOC_250001)

[DAFTAR PUSTAKA 99](#_TOC_250000)

# DAFTAR GAMBAR

# DAFTAR TABEL

# SKENARIO PRAKTIKUM SECARA DARING

Nama Mata Praktikum : Struktur Data Jumlah Pertemuan : 10 + 1 responsi

###### TABEL SKENARIO PRAKTIKUM DARING

|  |  |  |  |
| --- | --- | --- | --- |
| **Pertemuan ke** | **Judul Materi** | **Waktu *(Lama praktikum sampai pengumpul***  ***an posttest)*** | **Skenario Praktikum**  ***(Dari pemberian pre‐test, post‐test dan pengumpulannya serta mencantumkan metode yang digunakan misal video, whatsapp group, Google meet atau lainnya)*** |
| 1 | Class dan Rekursi | 4 hari | Pretest dikerjakan h‐1 pelaksanaan praktikum dan dikumpulkan pada hari H pelaksanaan praktikum. Kegiatan dilaksanakan via google meet selama  +‐ 1 jam. Postest pengerjaan maksimal  3 hari. |
| 2 | Array Statis | 4 hari | Pretest dikerjakan h‐1 pelaksanaan praktikum dan dikumpulkan pada hari H pelaksanaan praktikum. Kegiatan dilaksanakan via google meet selama  +‐ 1 jam. Postest pengerjaan  maksimal 3 hari. |
| 3 | Array Dinamis | 4 hari | Pretest dikerjakan h‐1 pelaksanaan praktikum dan dikumpulkan pada hari H pelaksanaan praktikum. Kegiatan dilaksanakan via google meet selama  +‐ 1 jam. Postest pengerjaan  maksimal 3 hari. |
| 4 | Linked List | 4 hari | Pretest dikerjakan h‐1 pelaksanaan praktikum dan dikumpulkan pada hari H pelaksanaan praktikum. Kegiatan dilaksanakan via google meet selama  +‐ 1 jam. Postest pengerjaan  maksimal 3 hari. |
| 5 | Variasi Linked List | 4 hari | Pretest dikerjakan h‐1 pelaksanaan praktikum dan dikumpulkan pada hari H pelaksanaan praktikum. Kegiatan dilaksanakan via google meet selama  +‐ 1 jam. Postest pengerjaan  maksimal 3 hari. |
| 6 | Stack dengan Array | 4 hari | Pretest dikerjakan h‐1 pelaksanaan praktikum dan dikumpulkan pada hari H pelaksanaan praktikum. Kegiatan dilaksanakan via google meet selama  +‐ 1 jam. Postest pengerjaan  maksimal 3 hari. |
| 7 | Simulasi Queue dengan Array  Circular | 4 hari | Pretest dikerjakan h‐1 pelaksanaan |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  | praktikum dan dikumpulkan pada hari H pelaksanaan praktikum. Kegiatan dilaksanakan via google meet selama  +‐ 1 jam. Postest pengerjaan  maksimal 3 hari. |
| 8 | Antrian Berprioritas Menggunakan Doubly Linked List | 4 hari | Pretest dikerjakan h‐1 pelaksanaan praktikum dan dikumpulkan pada hari H pelaksanaan praktikum. Kegiatan dilaksanakan via google meet selama  +‐ 1 jam. Postest pengerjaan  maksimal 3 hari. |
| 9 | Pohon | 4 hari | Pretest dikerjakan h‐1 pelaksanaan praktikum dan dikumpulkan pada hari H pelaksanaan praktikum. Kegiatan dilaksanakan via google meet selama  +‐ 1 jam. Postest pengerjaan  maksimal 3 hari. |
| 10 | Pohon Setimbang | 4 hari | Pretest dikerjakan h‐1 pelaksanaan praktikum dan dikumpulkan pada hari H pelaksanaan praktikum. Kegiatan dilaksanakan via google meet selama  +‐ 1 jam. Postest pengerjaan  maksimal 3 hari. |
| 11 | Responsi | 3 Jam | Pengerjaan soal |

# PRAKTIKUM 1: CLASS DAN REKURSI

Pertemuan ke 1

**Total Alokasi Waktu** : 90 menit

* Materi : 10 menit
* Pre‐Test : 10 menit
* Praktikum : 40 menit
* Post‐Test : 30 menit

**Total Bobot Penilaian** : 100%

* Pre‐Test : 30 %
* Praktik : 30 %
* Post‐Test : 40 %

Pemenuhan CPL dan CPMK:

|  |  |
| --- | --- |
| CPL‐03 | Mampu menerapkan konsep teoritis bidang area Informatika terkait matematika dasar  dan ilmu komputer untuk memodelkan masalah dan meningkatkan produktivitas |
| CPMK‐01 | Mampu memahami, menjelaskan, dan menerapkan konsep Dasar‐dasar Pemrograman  Menggunakan C++ serta Struct dan Class |

### DESKRIPSI CAPAIAN PEMBELAJARAN

Setelah mengikuti praktikum ini mahasiswa diharapkan mampu:

* + 1. Membuat program C++ menggunakan Class
    2. Membuat fungsi dengan perulangan rekursi

### INDIKATOR KETERCAPAIAN PEMBELAJARAN

Indikator ketercapaian diukur dengan:

|  |  |  |
| --- | --- | --- |
| CPL‐03 | CPMK‐01 | Kemampuan mahasiswa dalam menerapkan konsep Dasar‐ dasar Pemrograman Menggunakan C++ serta Struct dan Class |

### TEORI PENDUKUNG

###### LANDASAN TEORI CLASS

Bentuk umum konstruksi struct

Sebelum menggunakan kelas kita harus tahu struktur struct dalam bahasa C. Struct adalah kumpulan tipe data bentukan yang berisi kumpulan deklarasi variabel menjadi sebuah nama dalam kesatuan yang utuh. Sebagai contoh sebuah titik dalam dua dimensi terdiri dari absis (sumbu x) dan koordinat (sumbu y) bila kita mendeklarasikan variabel biasa:

int x; int y;

variabel x dan y yang masing masing bertipe int bisa di artikan untuk menampung bilangan integer secara umum, tidak terlihat makna nya sebagai representasi. Untuk itu kita kumpulkan dalam sebuah struktur struct.

Bentuk umum dari struct adalah:

Struct namastruct {

Daftar deklarasi variabel;

}

Contoh kasus titik di atas dapat di buat struct berikut ini: struct titik{

int x = 0; int y = 0;

};

Untuk mendeklarasikan titik A dan B sebagai variabel bertipe struct maka kita deklarasikan sebagai berikut:

titik A,B;

Bentuk umum konstruksi kelas: class nama‐kelas {

// bagian public public :

// bagian private private :

// bagian protected protected :

};

Bagian public digunakan sebagai antar muka kelas dengan program (atau kelas) lain, bagian private biasanya digunakan untuk data member. Hal ini sebagai implementasi dari information hiding, artinya data sebaliknya dilindungi (encapsulation) dari program yang tidak berhak untuk mengaksesnya. Namun pada dasarnya, bagian private ini dapat diakses oleh semua anggota kelas. Selain anggota kelas, bagian private bisa juga diakses oleh kelas friend (contohnya friend untuk operator overloading ostream <<).

Bagian protected hanya digunakan apabila suatu saat ingin dibuat kelas turunan dan kelas turunan inilah yang dapat mengakses selain anggota kelas yang bersangkutan. Berikut ini diberikan kelas Bilangan untuk mempelajari akses anggota kelas terhadap

bagian private. Selain itu juga diberikan operator overloading untuk memudahkan manipulasi terhadap data.

###### LANDASAN TEORI REKURSI

Fungsi yang memanggil dirinya sendiri dinamakan fungsi rekursi. Sebagai contoh, perhatikan konstruksi fungsi faktorial berikut. Definisi faktorial adalah :

n! = 1 \* 2 \* 3 \* … (n‐2) \* (n‐1) \* n

Untuk membuat bentuk faktorial secara rekursi, kita tulis ulang menjadi :

n! = n \* (n‐1) \* (n‐2) \* … \* 3 \* 2 \* 1

Ambil satu suku, sisanya menggunakan operasi yang sama dengan masalah awal, yaitu : n! = n \* (n‐1) !

Untuk menghitung faktorial secara rekursi kita butuh defisini :

n! = 1, jika n = 0

(kasus paling sederhana dan menentukan nilai. Digunakan untuk menghentikan fungsi berulang yang memanggil dirinya sendiri)

n! = n \* (n‐1) !, jika n > 0

(kasus pemanggilan fungsi itu sendiri. Cirinya : harus mengarah pada kasus penyetop, yaitu n berkurang mengarah menuju 0)

Di bawah ini disajikan fungsi rekursi menggunakan C++ : int faktorial (int n) {

if (n == 0) return (1); // kasus penyetop else

return (n \* faktorial(n – 1)); // pemanggilan rekursif

}

Mari kita coba tahapan menghitung 4! berikut :

4! = 4 \* 3 !

= 4 \* 3 \* 2 !

= 4 \* 3 \* 2 \* 1 !

= 4 \* 3 \* 2 \* 1 \* 0 !

Karena 0 ! = 1, maka pemanggilan rekursi berhenti dan kita bisa menghitung perkalian nilai‐nilai yang telah ada.

### HARDWARE DAN SOFTWARE

Hardware dan software yang digunakan dalam praktikum ini yaitu:

1. Komputer.
2. Borland C++ atau Kompilator sejenis.

### PRE‐TEST

Jawablah pertanyaan berikut (**Total Skor: 100):**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **CPL** | **CPMK** | **Pertanyaan** | **Skor** |
| 1. | CPL‐03 | CPMK‐01 | Ada 3 jenis atau 3 bagian yang berbeda dari struktur class dan pemrogaman c++ yaitu bagian publik, bagian privat, dan bagian protected. Jelaskan peruntukan dari  masing masing bagian | 20 |
| 2. | CPL‐03 | CPMK‐01 | Buatlah contoh class yang menggunakan ketiga bagian itu | 40 |
| 3. | CPL‐03 | CPMK‐01 | Apa yang di sebut dengan subprogram rekursi | 10 |
| 4. | CPL‐03 | CPMK‐01 | Berikan contoh fungsi rekursi dan prosedur rekursi | 30 |

### LANGKAH PRAKTIKUM

Aturan Penilaian (Total Skor: 100):

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No** | **CPL** | **CPMK** | **Pertanyaan** | **Dokumen**  **Pendukung** | **Skor** |
| 1. | CPL‐03 | CPMK‐01 | Selesaikan langkah praktikum 1  dan 2 | Hasil praktikum  langkah 1 dan 2 | 100 |

#### Langkah‐Langkah Praktikum:

1. Implementasi Class

Contoh implementasi penciptaan kelas diberikan di bawah ini. Kelas Bilangan mempunyai dua data member yaitu a bertipe int dan b bertipe float. Contoh kelas ini menunjukkan bahwa dalam paradigma pemrograman berorientasi objek menggunakan C++, private data member dipandang secara global oleh member function kelas itu sendiri. Dalam kenyataannya, private data member tidak dapat diakses kelas atau program lain di luar kelasnya.

|  |  |
| --- | --- |
|  | Program 1.1. File GLOBAL.CPP |
| 1. | #include <iostream.h> |
| 2. | class Bilangan { |
| 3. | friend ostream& operator<<(ostream&, const Bilangan&); |
| 4. | friend istream& operator>>(istream&, Bilangan&); |
| 5. | public : |

|  |  |
| --- | --- |
| 6. | Bilangan(int a0=0, float b0=0.0) : a(a0), b(b0) { } |
| 7. | void banding\_int(const Bilangan&, const Bilangan&); |
| 8. | Bilangan& operator=(const Bilangan&); |
| 9. | Bilangan operator+(const Bilangan&) const; |
| 10. | Bilangan operator‐()const; |
| 11. | private: |
| 12. | int a; |
| 13. | float b; |
| 14. | }; |
| 15. | ostream& operator<<(ostream& out, const Bilangan& x) |
| 16. | { out << "Bagian integer : " << x.a << endl; |
| 17. | out << "Bagian float : " << x.b << endl; |
| 18. | return out; |
| 19. | } |
| 20. | void Bilangan::banding\_int(const Bilangan& x, const Bilangan& y) |
| 21. | { |
| 22. | if (x.a > y.a) cout << x.a << "::x lebih besar dari " << y.a << "::y"; |
| 23. | else cout << x.a << "::x lebih kecil dari " << y.a << "::y"; |
| 24. | } |
| 25. | Bilangan& Bilangan::operator=(const Bilangan& x) |
| 26. | { a = x.a; |
| 27. | b = x.b; |
| 28. | return \*this; |

|  |  |
| --- | --- |
| 29. | } |
| 30. | istream& operator>>(istream& in, Bilangan& x) |
| 31. | { |
| 32. | cout <<"\nMasukkan bagian integer : "; |
| 33. | in >> x.a; |
| 34. | cout <<"Masukkan bagian float : "; |
| 35. | in >> x.b; |
| 36. | return in; |
| 37. | } |
| 38. | Bilangan Bilangan::operator+(const Bilangan& x) const |
| 39. | { Bilangan cc; |
| 40. | cc.a = a + x.a; |
| 41. | cc.b = b + x.b; |
| 42. | return cc; |
| 43. | } |
| 44. | Bilangan Bilangan::operator‐() const |
| 45. | { Bilangan x; |
| 46. | x.a = ‐a; |
| 47. | x.b = ‐b; |
| 48. | return x; |
| 49. | } |
| 50. | void main() { |
| 51. | Bilangan s, t(‐2,3.14), d; |

|  |  |
| --- | --- |
| 52. | cout << "Nilai awal s\n" << s; // operator biner << output |
| 53. | cout << "Nilai awal t dari deklarasi\n" << t; |
| 54. | s = t; // operator biner = |
| 55. | cout << "Setelah s di‐assign t\n"; |
| 56. | cout << "Nilai s\n" << s; |
| 57. | cout << "Masukkan nilai‐nilai objek d"; |
| 58. | cin >> d; // operator >> input |
| 59. | cout << "Setelah d + t => \n" << d+t; // operator biner + |
| 60. | cout << "Nilai d dinegatifkan\n" << ‐d; // operator uner ‐ |
| 61. | } |

Kelas Bilangan mempunyai dua data member yaitu a dan b masing‐masing bertipe int dan float. Ada 1 konstruktor, dan 5 operator overloading serta sebuah method. Perhatikan bagaimana membentuk konstruktor, membuat operator overloading dan memanipulasi data lewat method.

1. Implementasi Rekursi

Di bawah ini diberikan beberapa contoh program dengan perulangan secara rekursi. Coba dan perhatikan hal‐hal baru yang bisa anda peroleh dari setiap contoh.

|  |  |
| --- | --- |
|  | Program 2.1 |
| 1. | void Tampil(int n) { |
| 2. | static int i=0; |
| 3. | if (n<=0) return; |
| 4. | cout << “Pemanggilan ke : “ << ++I << endl; |
| 5. | Tampil(n‐1); |
| 6. | } |

|  |  |
| --- | --- |
| 7. | int main(int argc, char \*argv[]) { |
| 8. | int n=3; |
| 9. | Tampil(n); |
| 10. | system(“PAUSE”); |
| 11. | return EXIT\_SUCCESS; |
| 12. | } |

Cobalah mengganti dengan nilai n yang berbeda‐beda.

Program berikut adalah program untuk mencari bilangan x dari sederetan bilang pemanggilan rekursi (atau boleh dikatakan sederetan bilangan) dari bilangan n sampai 1.

|  |  |
| --- | --- |
|  | Program 2.2 |
| 1. | void cari(int x, int n) { |
| 2. | static int i; |
| 3. | if (n == 0) cout << "Bilangan tidak ditemukan\n"; |
| 4. | else if (x == n) cout << "Bilangan ditemukan setelah perulangan ke  \n" << i; |
| 5. | else { |
| 6. | i++; |
| 7. | cari(x, n‐1); |
| 8. | } |
| 9. | } |
| 10. | int main(int argc, char \*argv[]) { |
| 11. | cari(14, 9); |
| 12. | system("PAUSE"); |

|  |  |
| --- | --- |
| 13. | return EXIT\_SUCCESS; |
| 14. | } |

Cobalah dengan berbagai variasi pemanggilan fungsi dengan mengubah baris 11, misalnya : cari (4, 10);

Kadang diperlukan beberapa pernyataan kondisional, oleh karena secara matematik memang dibutuhkan. Sebagai contoh adalah menghitung nilai pangkat n dari x berikut :

x0 = 1

x1 = x

xn = (x2)n/2, untuk n genap

Bagaimana untuk n ganjil? Bukankah bila n ganjil bila dibagi 2 akan menghasilkan bilangan pecahan. Maka untuk itu, kita ambil satu suku, kemudian sisanya menggunakan kaidah bilangan genap : xn = xn‐1x, untuk n ganjil.

Perhatikan bahwa nilai n‐1 akan genap karena n ganjil. Dengan mudah kita implementasikan menjadi fungsi dalam C++ berikut :

|  |  |
| --- | --- |
|  | Program 2.3 |
| 1. | float Pangkat(float x, int n) { |
| 2. | if (n==0) return 1; |
| 3. | if (n==1) return x; |
| 4. | if (n % 2 == 1) { |
| 5. | cout << "Masuk bagian ganjil\n"; |
| 6. | return Pangkat(x, n‐1) \* x; |
| 7. | } |
| 8. | else { |
| 9. | cout << "Masuk bagian genap\n"; |
| 10. | return Pangkat(x\*x, n/2); |

|  |  |
| --- | --- |
| 11. | } |
| 12. | } |

Cobalah amati perbedaan genap dan ganjil dengan menelusuri hasil eksekusinya menggunakan:

Pangkat(2, 5); Dan Pangkat(2, 6);

Tentu saja anda harus membuat fungsi main‐nya sendiri.

### POST TEST

Jawablah pertanyaan berikut (**Total Skor: 100):**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **CPL** | **CPMK** | **Pertanyaan** | **Skor** |
| 1. | CPL‐03 | CPMK‐01 | Cetaklah secara terbalik bilangan INTEGER maksimal  5 digit secara iteratif dan rekursif. Sebagai contoh : input : 12345  output : 54321  Gunakan class untuk menampung data integernya dan operasi (fungsi). |  |
| Program jalan (Screenshot input output) | 30 |
| Iteratif | 20 |
| Rekursif | 30 |
| Class | 20 |

Jawablah pertanyaan berikut (**Total Skor: 100):**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **CPL** | **CPMK** | **Pertanyaan** | **Skor** |
| 1. | CPL‐03 | CPMK‐01 | 1. Analisislah komponen data yang ada dalam KTP diatas, lalu buatlah struktur (Struct) dari data KTP diatas. Buatlah sebuah variabel untuk menampung data KTP, siapkan 3 data ktp dari teman 1 tim. Masukkan ketiga data KTP teman tersebut melalui keyboard dan disimpan ke file. | 100 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  | 2. Diskusikan dengan teman 1 tim dan pilihlah topik atau judul tugas proyek, posting ke Facebook dengan thread  yang telah disediakan. |  |

### HASIL CAPAIAN PRAKTIKUM

Diisi oleh asisten setelah semua assessment dinilai.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **No** | **Bentuk**  **Assessment** | **CPL** | **CPMK** | **Bobot** | **Skor (0‐100)** | **Nilai Akhir**  **(Bobot x Skor)** |
| 1. | Pre‐Test | CPL‐03 | CPMK‐01 | 30% |  |  |
| 2. | Praktik | CPL‐03 | CPMK‐01 | 30% |  |  |
| 3. | Post‐Test | CPL‐03 | CPMK‐01 | 40% |  |  |
| **Total Nilai** | | | | | |  |

###### LEMBAR JAWABAN PRE‐TEST DAN POST‐TEST PRAKTIKUM

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

# PRAKTIKUM 2: ARRAY STATIS

Pertemuan ke 2

**Total Alokasi Waktu** : 90 menit

* Materi : 10 menit
* Pre‐Test : 10 menit
* Praktikum : 40 menit
* Post‐Test : 30 menit

**Total Bobot Penilaian** : 100%

* Pre‐Test : 30 %
* Praktik : 30 %
* Post‐Test : 40 %

Pemenuhan CPL dan CPMK:

|  |  |
| --- | --- |
| CPL‐03 | Mampu menerapkan konsep teoritis bidang area Informatika terkait matematika dasar  dan ilmu komputer untuk memodelkan masalah dan meningkatkan produktivitas |
| CPMK‐01 | Mampu memahami, menjelaskan, dan menerapkan konsep Konsep Antrian  lebih khusus Priority Queue diimplementasikan dengan Array (statis dan dinamis) dan Linklist |

### DESKRIPSI CAPAIAN PEMBELAJARAN

Setelah mengikuti praktikum ini mahasiswa diharapkan mampu:

* + 1. Mengamati data member array statis
    2. Mengamati cara kerja konstruktor array statis
    3. Mengamati cara pembuatan operator overloading input dan output untuk array statis

### INDIKATOR KETERCAPAIAN PEMBELAJARAN

Indikator ketercapaian diukur dengan:

|  |  |  |
| --- | --- | --- |
| CPL‐03 | CPMK‐04 | Kemampuan mahasiswa dalam menerapkan konsep Konsep Antrian  lebih khusus Priority Queue diimplementasikan dengan Array (statis dan dinamis) dan Linklist |

### TEORI PENDUKUNG

Array adalah struktur data yang mengandung type data yang mempunyai type sama. Suatu array adalah sekelompok memori yang berhubungan. Array mempunyai nama dan type yang sama. Untuk merujuk lokasi tertentu atau elemen dalam array; nama array dan angka posisi (disebut subscript atau indeks) dari elemen tersebut dalam array. Pada dasarnya, array menyimpan data secara statis artinya pada saat dikompilasi sudah

ditentukan berapa memori yang diperlukan dan pada saat program dieksekusi kita tidak dapat mengubah banyaknya memori yang dibutuhkan.

Secara umum, array statis dideklarasikan dalam bentuk :

Tipe\_data var\_array[banyaknya tempat yang dipesan];

Sebagai contoh untuk memesan tempat sebanyak 10 satuan memori pada array x yang menyimpan data bertipe int dideklarasikan menggunakan pernyataan :

int x[10];

yang perlu diingat indeks array dalam C++ dimulai dari 0 sampai dengan n‐1, jika kita memesan memori sebanyak n tempat.

### HARDWARE DAN SOFTWARE

Hardware dan software yang digunakan dalam praktikum ini yaitu:

1. Komputer.
2. Borland C++ atau Kompilator sejenis.

### PRE‐TEST

Jawablah pertanyaan berikut (**Total Skor: 100):**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **CPL** | **CPMK** | **Pertanyaan** | **Skor** |
| 1. | CPL‐03 | CPMK‐04 | Jelaskan pengertian array ! | 20 |
| 2. | CPL‐03 | CPMK‐04 | Gambarkan array dengan 7 elemen, yaitu  (5,1,2,3,6,4,7) dengan cara menambah elemen di depan tiap kali menambah elemen baru. | 40 |
| 3. | CPL‐03 | CPMK‐04 | Kemudian, hapuslah elemen di depan 3 kali  dan gambarkan prosesnya. Setiap gambar dalam bentuk tabel beserta index dan nilainya | 40 |

### LANGKAH PRAKTIKUM

Aturan Penilaian (Total Skor: 100):

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No** | **CPL** | **CPMK** | **Pertanyaan** | **Dokumen**  **Pendukung** | **Skor** |
| 1. | CPL‐03 | CPMK‐04 | Selesaikan langkah praktikum | Hasil praktikum | 100 |

#### Langkah‐Langkah Praktikum:

Untuk mengetahui konsep array statis, perhatikan program di bawah ini. Kelas Array 1D merupakan tipe data abstrak untuk array statis. Perhatikan pada bagian data member, kita akan mengamati perilaku array yang menyimpan data bertipe char.

|  |  |
| --- | --- |
|  | geser.cpp |
| 1. | #include <iostream.h> |
| 2. | #define maks 5 |
| 3. | class Array1D { |
| 4. | friend ostream& operator<<(ostream&, const Array1D&); |
| 5. | friend istream& operator>>(istream&, Array1D&); |
| 6. | public : |
| 7. | Array1D(); |
| 8. | void cetak(); |
| 9. | void geser\_kiri(); |
| 10. | void geser\_kanan(); |
| 11. | private : |
| 12. | char A[maks]; |
| 13. | }; |

Kelas Array1D mempunyai 1 konstruktor, dan 3 method. Method geser\_kiri dan geser\_kanan digunakan untuk memanipulasi array, sedangkan method cetak digunakan untuk mencetak array. Perhatikan bahwa method cetak sebenarnya sama dengan operator overloading output <<. Perhatikan pula dialek cara membuat operator overloading.

|  |  |
| --- | --- |
| 14. | Array1D::Array1D() { |
| 15. | for (int i=0; i<maks; i++) |
| 16. | A[i] = '0'; |
| 17. | } |
| 18. | void Array1D::cetak() { |
| 19. | for (int i=0; i<maks; i++) |

|  |  |
| --- | --- |
| 20. | cout << A[i] << " "; |
| 21. | } |
| 22. | ostream& operator<<(ostream& out, const Array1D& x) { |
| 23. | for (int i=0; i<maks; i++) |
| 24. | cout << x.A[i] << " "; |
| 25. | cout << endl; |
| 26. | return out; |
| 27. | } |
| 28. | istream& operator>>(istream& in, Array1D& x) |
| 29. | { int posisi; |
| 30. | cout << "Mengisi array pada posisi ke : "; |
| 31. | in >> x.A[posisi‐1]; |
| 32. | } |
| 33. | else |
| 34. | cout << "Anda memasukkan posisi di luar range ... "; |
| 35. | return in; |
| 36. | } |
| 37. | void Array1D::geser\_kanan() |
| 38. | { |
| 39. | int n = maks; |
| 40. | int temp = A[n‐1]; |
| 41. | for (int i=n‐1; i >= 0; i‐‐) |
| 42. | A[i+1] = A[i]; |

|  |  |
| --- | --- |
| 43. | A[0] = temp; |
| 44. | } |
| 45. | void Array1D::geser\_kiri() |
| 46. | { |
| 47. | int n = maks; |
| 48. | int temp = A[0]; |
| 49. | for (int i=0; i < n; i++) |
| 50. | A[i] = A[i+1]; |
| 51. | A[n‐1] = temp; |
| 52. | } |
| 53. | main() { |
| 54. | Array1D x; |
| 55. | cout << "Array masih kosong : " << x; |
| 56. | cin >> x; |
| 57. | cout << "Isi Array saat ini : " << x; |
| 58. | x.geser\_kiri(); |
| 59. | cout << "Isi Array setelah digeser ke kiri : " << x; |
| 60. | x.geser\_kanan(); |
| 61. | cout << "Isi Array setelah digeser ke kanan : " << x; |
| 62. | return 0; |
| 63. | } |

Bila kita melakukan operasi penghapusan elemen indeks 0, maka akan terjadi pergeseran elemen ke kiri. Sebaliknya, bila kita menambah elemen di depan maka akan terjadi pergeseran elemen ke kanan.

### POST TEST

Jawablah pertanyaan berikut (**Total Skor: 100):**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **CPL** | **CPMK** | **Pertanyaan** | **Skor** |
| 1. | CPL‐03 | CPMK‐04 | Gunakan struct untuk data KTP dan class untuk operasi input dan output (Sesuaikan dengan urutan KTP) data yang dimaksudkan dalam array. Gunakan data  anggota kelompok untuk program anda. |  |
|  |  |  | Program jalan (Screenshot input output) | 30 |
|  |  |  | Input data dari file | 20 |
|  |  |  | Struct | 20 |
|  |  |  | Class | 30 |

Jawablah pertanyaan berikut (**Total Skor: 100):**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **CPL** | **CPMK** | **Pertanyaan** | **Skor** |
| 1. | CPL‐03 | CPMK‐04 | Dari program KTP pertemuan pertama, buatlah program menggunakan array statis dengan menambahkan 7 data KTP . Lakukan sorting didasarkan nama. Algoritma sorting  bebas. | 100 |

### HASIL CAPAIAN PRAKTIKUM

Diisi oleh asisten setelah semua assessment dinilai.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **No** | **Bentuk Assessment** | **CPL** | **CPMK** | **Bobot** | **Skor (0‐100)** | **Nilai Akhir (Bobot x Skor)** |
| 1. | Pre‐Test | CPL‐03 | CPMK‐04 | 30% |  |  |
| 2. | Praktik | CPL‐03 | CPMK‐04 | 30% |  |  |
| 3. | Post‐Test | CPL‐03 | CPMK‐04 | 40% |  |  |
| **Total Nilai** | | | | | |  |

###### LEMBAR JAWABAN PRE‐TEST DAN POST‐TEST PRAKTIKUM

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

# PRAKTIKUM 3: ARRAY DINAMIS

Pertemuan ke 3

**Total Alokasi Waktu** : 90 menit

* Materi : 10 menit
* Pre‐Test : 10 menit
* Praktikum : 40 menit
* Post‐Test : 30 menit

**Total Bobot Penilaian** : 100%

* Pre‐Test : 30 %
* Praktik : 30 %
* Post‐Test : 40 %

Pemenuhan CPL dan CPMK:

|  |  |
| --- | --- |
| CPL‐03 | Mampu menerapkan konsep teoritis bidang area Informatika terkait matematika dasar  dan ilmu komputer untuk memodelkan masalah dan meningkatkan produktivitas |
| CPMK‐04 | Mampu memahami, menjelaskan, dan menerapkan konsep Konsep Antrian lebih khusus Priority Queue diimplementasikan dengan Array (statis dan dinamis)  dan Linklist |

### DESKRIPSI CAPAIAN PEMBELAJARAN

Setelah mengikuti praktikum ini mahasiswa diharapkan mampu:

* + 1. Mampu mengamati data member array dinamis
    2. Mampu membandingkan perilaku array statis dengan array dinamis

### INDIKATOR KETERCAPAIAN PEMBELAJARAN

Indikator ketercapaian diukur dengan:

|  |  |  |
| --- | --- | --- |
| CPL‐03 | CPMK‐04 | Kemampuan mahasiswa dalam menerapkan konsep Konsep Antrian lebih khusus Priority Queue diimplementasikan dengan Array (statis  dan dinamis) dan Linklist |

### TEORI PENDUKUNG

Pada praktikum sebelumnya telah diketahui bahwa array statis tidak dapat diubah lagi banyaknya tempat penyimpannya. Ada kondisi di mana kita ingin memanfaatkan keunggulan array yaitu kemudian akses elemennya, namun di sisi lain, kita juga ingin suatu saat bisa mengubah banyaknya elemen yang dapat disimpan. Untuk itulah kita

ciptakan array dinamis. Pada dasarnya, array dalam C++ adalah pointer. Namun array statis merupakan pointer konstan. Dengan karakter seperti itu, kita dapat mengembangkan array dinamis dengan menggunakan pointer. Perhatikan kelas Array 1D di bawah ini khususnya pada bagian data member.

##### Kelas Array 1D mempunyai dua data member, yang pertama adalah size yang menyatakan besarnya tempat memori yang dapat digunakan. Pointer element digunakan untuk menunjuk elemen pertama dari array. C++ menjamin letak dari elemen array bersambungan di memori. Jadi kita tak perlu ragu untuk memperlakukan pointer seperti array.

|  |  |
| --- | --- |
| 1. | #include <iostream.h> |
| 2. | #include "xcept.h" |
| 3. | template<class T> |
| 4. | class Array1D { |
| 5. | friend ostream& operator<<(ostream&, const Array1D<T>&); |
| 6. | public: |
| 7. | Array1D(int size = 0); |
| 8. | Array1D(const Array1D<T>& v); |
| 9. | ~Array1D() {delete [] element;} |
| 10. | T& operator[](int i) const; |
| 11. | int Size() {return size;} |
| 12. | Array1D<T>& operator=(const Array1D<T>& v); |
| 13. | Array1D<T> operator+() const; |
| 14. | Array1D<T> operator+(const Array1D<T>& v) const; |
| 15. | Array1D<T> operator‐() const; |
| 16. | Array1D<T> operator‐(const Array1D<T>& v) const; |
| 17. | Array1D<T> operator\*(const Array1D<T>& v) const; |

|  |  |
| --- | --- |
| 18. | Array1D<T>& operator+=(const T& x); |
| 19. | Array1D<T>& ReSize(int sz); |
| 20. | private: |
| 21. | int size; |
| 22. | T \*element; |
| 23. | }; |

Perlu diperhatikan terdapat 2 konstruktor kelas Array1D, yang pertama dengan ukuran 0 (7) dan konstruktor salinan (copy constructor) yang berfungsi untuk mendeklarasikan sekaligus memberikan nilai dari objek lain (8). Sebagai ilustrasi, pernyataan :

int a = 2; int b = a;

akan menyalin nilai objek (variabel) a ke objek (variabel) b.

### HARDWARE DAN SOFTWARE

Hardware dan software yang digunakan dalam praktikum ini yaitu:

1. Komputer.
2. Borland C++ atau Kompilator sejenis.

### PRE‐TEST

Jawablah pertanyaan berikut (**Total Skor: 100):**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **CPL** | **CPMK** | **Pertanyaan** | **Skor** |
| 1. | CPL‐03 | CPMK‐04 | Sebutkan karakteristik Array dinamis! | 25 |
| 2. | CPL‐03 | CPMK‐04 | Tunjukkan fungsi dasar pembuatan array secara  dinamis (pada program yang sudah ada) | 25 |
| 3. | CPL‐03 | CPMK‐04 | Apa perbedaan array dinamis dengan array statis? | 25 |
| 4. | CPL‐03 | CPMK‐04 | Buat program sederhana memakai array dinamis  (boleh intinya saja tidak harus lengkap) | 25 |

### LANGKAH PRAKTIKUM

Aturan Penilaian (Total Skor: 100):

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No** | **CPL** | **CPMK** | **Pertanyaan** | **Dokumen**  **Pendukung** | **Skor** |
| 1. | CPL‐03 | CPMK‐04 | Selesaikan langkah praktikum | Hasil praktikum | 100 |

#### Langkah‐Langkah Praktikum:

|  |  |
| --- | --- |
| 24. | template<class T> |
| 25. | Array1D<T>::Array1D(int sz) |
| 26. | if (sz < 0) throw BadInitializers(); |
| 27. | size = sz; |
| 28. | element = new T[sz]; |
| 29. | } |
| 30. | template<class T> |
| 31. | Array1D<T>::Array1D(const Array1D<T>& v) |
| 32. | size = v.size; |
| 33. | element = new T[size]; // get space |
| 34. | for (int i = 0; i < size; i++) // copy elements |
| 35. | element[i] = v.element[i]; |
| 36. | } |

Perhatikan perilaku dari konstruktor salinan di atas, dan amati perbedaannya dengan operator overloading assignment = di bawah ini.

|  |  |
| --- | --- |
| 37. | template<class T> |
| 38. | T& Array1D<T>::operator[](int i) const { |
| 39. | if (i < 0 || i >= size) throw OutOfBounds(); |
| 40. | return element[i]; |
| 41. | } |
| 42. | template<class T> |
| 43. | Array1D<T>& Array1D<T>::operator=(const Array1D<T>& v) { |
| 44. | if (this != &v) { |
| 45. | size = v.size; |

|  |  |
| --- | --- |
| 46. | delete [] element; |
| 47. | element = new T[size]; |
| 48. | for (int i = 0; i < size; i++) |
| 49. | element[i] = v.element[i]; |
| 50. | } |
| 51. | return \*this; |
| 52. | } |
| 53. | template<class T> |
| 54. | Array1D<T> Array1D<T>:: operator+(const Array1D<T>& v) const { |
| 55. | if (size != v.size) throw SizeMismatch(); |
| 56. | Array1D<T> w(size); |
| 57. | for (int i = 0; i < size; i++) |
| 58. | w.element[i] = element[i] + v.element[i]; |
| 59. | return w; |
| 60. | } |
| 61. | template<class T> |
| 62. | Array1D<T> Array1D<T>:: operator‐(const Array1D<T>& v) const { |
| 63. | if (size != v.size) throw SizeMismatch(); |
| 64. | Array1D<T> w(size); |
| 65. | for (int i = 0; i < size; i++) |
| 66. | w.element[i] = element[i] ‐ v.element[i]; |
| 67. | return w; |
| 68. | } |
| 69. | template<class T> |
| 70. | Array1D<T> Array1D<T>::operator‐() const { |
| 71. | Array1D<T> w(size); |
| 72. | for (int i = 0; i < size; i++) |

|  |  |
| --- | --- |
| 73. | w.element[i] = ‐element[i]; |
| 74. | return w; |
| 75. | } |
| 76. | template<class T> |
| 77. | Array1D<T> Array1D<T>::operator\*(const Array1D<T>& v) const { |
| 78. | if (size != v.size) throw SizeMismatch(); |
| 79. | Array1D<T> w(size); |
| 80. | for (int i = 0; i < size; i++) |
| 81. | w.element[i] = element[i] \* v.element[i]; |
| 82. | return w; |
| 83. | } |
| 84. | template<class T> |
| 85. | Array1D<T>& Array1D<T>::operator+=(const T& x) { |
| 86. | for (int i = 0; i < size; i++) |
| 87. | element[i] += x; |
| 88. | return \*this; |
| 89. | } |
| 90. | template<class T> |
| 91. | ostream& operator<<(ostream& out, const Array1D<T>& x) { |
| 92. | for (int i = 0; i < x.size; i++) |
| 93. | out << x.element[i] << " "; |
| 94. | return out; |
| 95. | } |
| 96. | template<class T> |
| 97. | Array1D<T>& Array1D<T>::ReSize(int sz) { |
| 98. | if (sz < 0) throw BadInitializers(); |
| 99. | delete [] element; |

|  |  |
| --- | --- |
| 100. | size = sz; |
| 101. | element = new T [size]; |
| 102. | return \*this; |
| 103. | } |

Fungsi main disajikan berikut ini.

|  |  |
| --- | --- |
| 1. | #include <iostream.h> |
| 2. | #include "array1d.h" |
| 3. | void main(void) { |
| 4. | try { |
| 5. | Array1D<int> X(10), Y, Z; |
| 6. | for (int i=0; i < 10; i++) |
| 7. | X[i] = i; |
| 8. | cout << "X[3] = " << X[3] << endl; |
| 9. | cout << "X is " << X << endl; |
| 10. | Y = X; |
| 11. | cout << "Y is " << Y << endl; |
| 12. | X += 2; |
| 13. | cout << "X incremented by 2 is " << X << endl; |
| 14. | Z = (Y + X) \* Y; |
| 15. | cout << "(Y + X) \* Y is " << Z << endl; |
| 16. | cout << "‐(Y + X) \* Y is " << ‐Z << endl; |
| 17. | } |
| 18. | catch (...) { |
| 19. | cerr << "An exception has occurred" << endl;} |
| 20. | } |

### POST TEST

Jawablah pertanyaan berikut (**Total Skor: 100):**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **CPL** | **CPMK** | **Pertanyaan** | **Skor** |
| 1. | CPL‐03 | CPMK‐04 | Ubahlah postes pertemuan 2 dengan array |  |
|  |  |  | dinamis. Tambahkan untuk mengubah maksimal data |
|  |  |  | yang bisa ditampung dalam array KTP itu. Gunakan data |
|  |  |  | teman dari kelompok lain. |
|  |  |  | Program jalan (Screenshot input output) | 30 |
|  |  |  | Input data dari file | 20 |
|  |  |  | Struct | 20 |
|  |  |  | Class | 30 |

Jawablah pertanyaan berikut (**Total Skor: 100):**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **CPL** | **CPMK** | **Pertanyaan** | **Skor** |
| 1. | CPL‐03 | CPMK‐04 | Menggunakan program pertemuan kedua, tambahkan subprogram untuk mengekstrak data didasarkan pada jenis kelamin menggunakan implementasi array dinamis. Perhatikan kita tidak pernah tahu berapa jumlah atau banyaknya data sehingga array benar benar harus mampu menampung data sebanyak data yang ada (tidak boleh memori melebihi data yang ada). Tambahkan aplikasi atau subprogram pencarian data berdasarkan jenis kelamin bila pria dia akan mengakses array pertama, jika wanita dia  akan mengakses array kedua. | 100 |

### HASIL CAPAIAN PRAKTIKUM

Diisi oleh asisten setelah semua assessment dinilai.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **No** | **Bentuk**  **Assessment** | **CPL** | **CPMK** | **Bobot** | **Skor (0‐100)** | **Nilai Akhir**  **(Bobot x Skor)** |
| 1. | Pre‐Test | CPL‐03 | CPMK‐04 | 30% |  |  |
| 2. | Praktik | CPL‐03 | CPMK‐04 | 30% |  |  |
| 3. | Post‐Test | CPL‐03 | CPMK‐04 | 40% |  |  |
| **Total Nilai** | | | | | |  |

###### LEMBAR JAWABAN PRE‐TEST DAN POST‐TEST PRAKTIKUM

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

# PRAKTIKUM 4: LINK LIST

Pertemuan ke 4

**Total Alokasi Waktu** : 90 menit

* Materi : 10 menit
* Pre‐Test : 10 menit
* Praktikum : 40 menit
* Post‐Test : 30 menit

**Total Bobot Penilaian** : 100%

* Pre‐Test : 30 %
* Praktik : 30 %
* Post‐Test : 40 %

Pemenuhan CPL dan CPMK:

|  |  |
| --- | --- |
| CPL‐03 | Mampu menerapkan konsep teoritis bidang area Informatika terkait matematika dasar  dan ilmu komputer untuk memodelkan masalah dan meningkatkan produktivitas |
| CPMK‐02 | Mampu memahami, menjelaskan, dan menerapkan konsep Pengantar Pointer dan Link  list beserta berbagai macam bentuknya |

### DESKRIPSI CAPAIAN PEMBELAJARAN

Setelah mengikuti praktikum ini mahasiswa diharapkan mampu:

* + 1. Menerapkan konsep link list beserta operasi‐operasinya

### INDIKATOR KETERCAPAIAN PEMBELAJARAN

Indikator ketercapaian diukur dengan:

|  |  |  |
| --- | --- | --- |
| CPL‐03 | CPMK‐02 | Kemampuan mahasiswa dalam menerapkan konsep Pengantar  Pointer dan Link list beserta berbagai macam bentuknya |

### TEORI PENDUKUNG

Link list atau daftar berantai adalah suatu struktur data yang tersusun atas paling tidak dua elemen yaitu info dan link (biasanya berupa pointer) yang gunanya untuk merangkai elemen (atau biasa disebut sebagai node) berikutnya. Oleh karena merupakan kesatuan, maka kita dapat menggunakan kelas sebagaimana contoh berikut :

template <class T> class Node {

friend class List<T>;

friend ostream& operator<<(ostream&, const List<T>&); // ??? public :

Node(T& t, Node<T>\* p) : info(t), berikut(p) { } protected :

T info;

Node \*berikut;

};

Field **info** akan menampung informasi (bisa digeneralisasi) sedangkan field **berikut** berupa pointer yang akan menunjuk (me‐*link*) elemen/node berikutnya. Gambar logiknya adalah sebagai berikut :

A

null

C

Gambar di atas adalah A yang dideklarasikan dengan : Node A;

yang menunjuk ke alamat yang isinya (infonya) adalah karakter C dan field berikutnya tidak menunjuk ke manapun (diberi nilai konstanta **null**).

Secara umum operasi link list adalah sebagai berikut :

* + 1. init link list, fungsinya untuk memodelkan link list yang dibuat. Misalnya circular link list, headed link list, double link list, dan sebagainya.
    2. tambah node di depan, fungsinya untuk menambah elemen/node di depan.
    3. tambah node di tengah. Operasi menambah elemen di tengah ini maksudnya adalah bukan menambah di depan ataupun di belakang. Jadi misalnya ada elemen b. Elemen ini ditempatkan pada posisi dengan syarat tertentu, misalnya pada posisi setelah nilai a dan sebelum nilai c di mana a < b< c.
    4. tambah node di belakang, yaitu operasi menambah elemen di akhir link list.
    5. menghapus node di depan, yaitu operasi untuk menghapus elemen/node paling depan
    6. menghapus node di tengah, yaitu operasi untuk menghapus elemen/node dengan syarat yang ditentukan
    7. menghapus di belakang, yaitu operasi untuk menghapus elemen/node paling belakang

Gambar Logik

Misalkan kita asumsikan bahwa struktur datanya adalah sebagaimana didefinisikan di atas. Gambar logik untuk konstruktor kelas Node adalah sebagai berikut :

baru

Null

C

Node(T& t, Node<T>\* p) : info(t), berikut(p) { }

Dieksekusi dengan pernyataan : Node<T>\* baru(‘C’, null);

### HARDWARE DAN SOFTWARE

Hardware dan software yang digunakan dalam praktikum ini yaitu:

1. Komputer.
2. Borland C++ atau Kompilator sejenis.

### PRE‐TEST

Jawablah pertanyaan berikut (**Total Skor: 100):**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **CPL** | **CPMK** | **Pertanyaan** | **Skor** |
| 1. | CPL‐03 | CPMK‐02 |  |  |
|  |  |  | Buat gambar logic step by step untuk penambahan node baru hingga terbentuk | 50 |
|  |  |  | Selanjutnya lakukan penyisipan node X diantara B dan C | 50 |

### LANGKAH PRAKTIKUM

Aturan Penilaian (Total Skor: 100):

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No** | **CPL** | **CPMK** | **Pertanyaan** | **Dokumen**  **Pendukung** | **Skor** |
| 1. | CPL‐03 | CPMK‐02 | Selesaikan langkah praktikum | Hasil praktikum | 100 |

#### Langkah‐Langkah Praktikum:

Ketik dan buatlah program implementasi kelas List selengkapnya yang diberikan berikut ini.

|  |  |
| --- | --- |
| 1. | #include <iostream.h> |
| 2. | template <class T> |
| 3. | class Node { |
| 4. | friend class List<T>; |
| 5. | friend ostream& operator<<(ostream&, const List<T>&); |
| 6. | public : |
| 7. | Node(T& t, Node<T>\* p) : info(t), berikut(p) { } |
| 8. | protected : |
| 9. | T info; |
| 10. | Node \*berikut; |
| 11. | }; |
| 12. | template <class T> |
| 13. | class List { |
| 14. | friend ostream& operator<<(ostream&, const List<T>&); |
| 15. | public : |
| 16. | List() : kepala(0) { } |
| 17. | ~List(); |
| 18. | void sisip(T t); |
| 19. | int hapus(T& t); |
| 20. | int kosong() { return (kepala == 0); } |
| 21. | void cetak(); |
| 22. | protected: |

|  |  |
| --- | --- |
| 23. | Node<T>\* kepala; |
| 24. | Node<T>\* nodeBaru(T& t, Node<T>\* p) |
| 25. | { Node<T>\* q = new Node<T>(t,p); return q; } |
| 26. | }; |
| 27. | template <class T> |
| 28. | ostream& operator<<(ostream& out, const List<T>& k) { |
| 29. | for (Node<T>\* p = k.kepala; p; p=p‐>berikut) |
| 30. | out << p‐>info << " ‐> "; |
| 31. | out << "\*\n"; |
| 32. | return out; |
| 33. | } |
| 34. | template <class T> |
| 35. | List<T>::~List() { |
| 36. | Node<T>\* temp; |
| 37. | for (Node<T>\* p = kepala; p;) |
| 38. | { |
| 39. | temp = p; |
| 40. | p = p‐>berikut; |
| 41. | delete temp; |
| 42. | } |
| 43. | } |
| 44. | template <class T> |
| 45. | void List<T>::sisip(T t) |
| 46. | { cout << t << " masuk list : "; |
| 47. | Node<T>\* p = nodeBaru(t,kepala); |
| 48. | kepala = p; |
| 49. | } |
| 50. | template <class T> |
| 51. | int List<T>::hapus(T& t) |

|  |  |
| --- | --- |
| 52. | { |
| 53. | if (kosong()) return 0; // penghapusan gagal |
| 54. | t = kepala‐>info; |
| 55. | Node<T>\* p = kepala; |
| 56. | kepala = kepala‐>berikut; |
| 57. | delete p; |
| 58. | return 1; // penghapusan sukses |
| 59. | } |
| 60. | template <class T> |
| 61. | void List<T>::cetak() |
| 62. | { |
| 63. | for (Node<T>\* p = kepala; p; p=p‐>berikut) |
| 64. | cout << p‐>info << " ‐> "; |
| 65. | cout << "\*\n"; |
| 66. | } |
| 67. | void main() { |
| 68. | List<char> x; |
| 69. | char data; |
| 70. | x.sisip('a'); |
| 71. | cout << x; |
| 72. | x.sisip('b'); |
| 73. | cout << x; |
| 74. | x.sisip('c'); |
| 75. | cout << x; |
| 76. | x.sisip('d'); |
| 77. | cout << x; |
| 78. | for (int i=0; i<5; i++) { |
| 79. | x.hapus(data); cout << data << " dihapus dari list : "; |
| 80. | cout << x; |

|  |  |
| --- | --- |
| 81. | } |
| 82. | } |

### POST TEST

Jawablah pertanyaan berikut (**Total Skor: 100):**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **CPL** | **CPMK** | **Pertanyaan** | **Skor** |
| 1. | CPL‐03 | CPMK‐02 | Gunakan struktur dari postes pertemuan 3 (data KTP) sebagai data link list. Buat operasi tambah dan hapus data   1. no ganjil : di belakang 2. no genap di depan |  |
|  |  |  | Program jalan (Screenshot input output) | 30 |
|  |  |  | Operasi menambah data | 20 |
|  |  |  | Operasi menghapus data | 20 |
|  |  |  | Class | 30 |

Jawablah pertanyaan berikut (**Total Skor: 100):**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **CPL** | **CPMK** | **Pertanyaan** | **Skor** |
| 1. | CPL‐03 | CPMK‐02 | Buatlah menggunakan data pertemuan ketiga sebuah fungsi untuk menampung data KTP menggunakan linklist. Spesifikasi data dibaca melalui file disimpan di linklist dan ditampilkan dilayar.  Tim Ganjil : cara menambah elemen linklist didepan Tim Genap: cara menambah elemen linklist dibelakang | 100 |

### HASIL CAPAIAN PRAKTIKUM

Diisi oleh asisten setelah semua assessment dinilai.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **No** | **Bentuk**  **Assessment** | **CPL** | **CPMK** | **Bobot** | **Skor (0‐100)** | **Nilai Akhir**  **(Bobot x Skor)** |
| 1. | Pre‐Test | CPL‐03 | CPMK‐02 | 30% |  |  |
| 2. | Praktik | CPL‐03 | CPMK‐02 | 30% |  |  |
| 3. | Post‐Test | CPL‐03 | CPMK‐02 | 40% |  |  |
| **Total Nilai** | | | | | |  |

###### LEMBAR JAWABAN PRE‐TEST DAN POST‐TEST PRAKTIKUM

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

# PRAKTIKUM 5: VARIASI LINK LIST

Pertemuan ke 5

**Total Alokasi Waktu** : 90 menit

* Materi : 10 menit
* Pre‐Test : 10 menit
* Praktikum : 40 menit
* Post‐Test : 30 menit

**Total Bobot Penilaian** : 100%

* Pre‐Test : 30 %
* Praktik : 30 %
* Post‐Test : 40 %

Pemenuhan CPL dan CPMK:

|  |  |
| --- | --- |
| CPL‐03 | Mampu menerapkan konsep teoritis bidang area Informatika terkait matematika dasar  dan ilmu komputer untuk memodelkan masalah dan meningkatkan produktivitas |
| CPMK‐02 | Mampu memahami, menjelaskan, dan menerapkan konsep Pengantar Pointer dan Link  list beserta berbagai macam bentuknya |

### DESKRIPSI CAPAIAN PEMBELAJARAN

Setelah mengikuti praktikum ini mahasiswa diharapkan mampu:

* + 1. Mengembangkan struktur untuk operasi link list yang lebih komplek

### INDIKATOR KETERCAPAIAN PEMBELAJARAN

Indikator ketercapaian diukur dengan:

|  |  |  |
| --- | --- | --- |
| CPL‐03 | CPMK‐02 | Kemampuan mahasiswa dalam menerapkan konsep Pengantar  Pointer dan Link list beserta berbagai macam bentuknya |

### TEORI PENDUKUNG

Teori yang sudah di bahas di kelas telah di jelaskan karakteristik circular link list dan doubly link list. Dalam diktat bab 7 tentang variasi link list telah di jelaskan gambar logic beserta pernyataan C++ untuk eksekusi di komputer.

### HARDWARE DAN SOFTWARE

Hardware dan software yang digunakan dalam praktikum ini yaitu:

1. Komputer.
2. Borland C++ atau Kompilator sejenis.

### PRE‐TEST

Jawablah pertanyaan berikut (**Total Skor: 100):**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **CPL** | **CPMK** | **Pertanyaan** | **Skor** |
| 1. | CPL‐03 | CPMK‐02 | Buatlah gambar logic untuk operasi menambah elemen dan menghapus elemen:   1. no ganjil : double link list 2. no genap : circular link list | 100 |

### LANGKAH PRAKTIKUM

Aturan Penilaian (Total Skor: 100):

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No** | **CPL** | **CPMK** | **Pertanyaan** | **Dokumen**  **Pendukung** | **Skor** |
| 1. | CPL‐03 | CPMK‐02 | Selesaikan langkah praktikum 1‐ 7 Circular link list | Hasil praktikum  langkah 1‐7 Circular link list | 50 |
| 2. | CPL‐03 | CPMK‐02 | Selesaikan langkah praktikum 1‐ 7 Doubly link list | Hasil praktikum  langkah 1‐7 Doubly link list | 50 |

#### Langkah‐Langkah Praktikum:

Sesuai dengan pilihan diskusi siapkan pretest gambar logic dari operasi:

Circular link list

* + 1. Buatlah skenario membuat deklarasi variabel circular link list beserta inisialisasi karakter circular link list
    2. Tambahkan 1 pointer yang akan bertugas menunjuk elemen link list paling belakang
    3. Berturut turut tambahkan di depan data berikut: 3 2 1. Perhatikan setiap kali menambah data di depan maka elemen pointer yang menunjuk akan selalu ada pada penunjuk elemen belakang (dalam hal ini akan selalu menunjuk data 1)
    4. Selanjutnya buatlah skenario menambah elemen di belakang berturut turut data: 6 7 8
    5. Pastikan pointer akan selalu menunjuk data paling belakang (selalu menunjuk elemen yang di masukan terakhir)
    6. Dari penunjuk pointer awal sisipkan data 4 yang berada di antara data 1 dan 6 7. Buatlah sebuah fungsi untuk mencetak semua elemen melingkar 2 kali tercetak

Doubly link list

1. Buatlah skenario membuat deklarasi variabel doubly link list beserta inisialisasi karakter doubly link list
2. Tambahkan 1 pointer yang akan bertugas menunjuk elemen link list paling belakang
3. Berturut turut tambahkan di depan data berikut: 3 2 1. Perhatikan setiap kali menambah data di depan maka elemen pointer yang menunjuk akan selalu ada pada penunjuk elemen belakang (dalam hal ini akan selalu menunjuk data 1)
4. Selanjutnya buatlah skenario menambah elemen di belakang berturut turut data: 6 7 8
5. Pastikan pointer akan selalu menunjuk data paling belakang (selalu menunjuk elemen yang di masukan terakhir)
6. Dari penunjuk pointer awal sisipkan data 4 yang berada di antara data 1 dan 6
7. Buatlah sebuah fungsi untuk mencetak semua elemen secara maju dan secara mundur dari posisi awal maju mencetak dari pointer awal, mundur mencetak mulai dari pointer akhir

### POST TEST

Jawablah pertanyaan berikut (**Total Skor: 100):**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **CPL** | **CPMK** | **Pertanyaan** | **Skor** |
| 1. | CPL‐03 | CPMK‐02 | Ubah konstruktor dan operasi dari link list postes pertemuan 4 dengan model link list berikut :   1. no ganjil : double link list 2. no genap : circular link list |  |
| Program jalan (Screenshot input output) | 30 |
| Operasi menambah data | 25 |
| Operasi menghapus data | 25 |
| Class | 20 |

### HASIL CAPAIAN PRAKTIKUM

Diisi oleh asisten setelah semua assessment dinilai.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **No** | **Bentuk**  **Assessment** | **CPL** | **CPMK** | **Bobot** | **Skor (0‐100)** | **Nilai Akhir**  **(Bobot x Skor)** |
| 1. | Pre‐Test | CPL‐03 | CPMK‐02 | 30% |  |  |
| 2. | Praktik | CPL‐03 | CPMK‐02 | 30% |  |  |
| 3. | Post‐Test | CPL‐03 | CPMK‐02 | 40% |  |  |
| **Total Nilai** | | | | | |  |

###### LEMBAR JAWABAN PRE‐TEST DAN POST‐TEST PRAKTIKUM

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

# PRAKTIKUM 6: STACK DENGAN ARRAY

Pertemuan ke 6

**Total Alokasi Waktu** : 90 menit

* Materi : 10 menit
* Pre‐Test : 10 menit
* Praktikum : 40 menit
* Post‐Test : 30 menit

**Total Bobot Penilaian** : 100%

* Pre‐Test : 30 %
* Praktik : 30 %
* Post‐Test : 40 %

Pemenuhan CPL dan CPMK:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| CPL‐04 CPL‐08 | Mampu berpikir logis, kritis, sistematis dan inovatif, dan mampu mengambil keputusan secara tepat di bidang keahliannya dan mampu merancang dan mengimplementasikan algoritma/metode dalam mengidentifikasi dan memecahkan masalah yang melibatkan  perangkat lunak dan pemikiran komputasi | | | | | | |
| CPMK‐03 | Mampu memahami,  dan Aplikasinya | menjelaskan, | dan | menerapkan | konsep | Konsep | Stack |

### DESKRIPSI CAPAIAN PEMBELAJARAN

Setelah mengikuti praktikum ini mahasiswa diharapkan mampu:

1. Menerapkan penggunaan array untuk mengimplementasikan struktur data stack

### INDIKATOR KETERCAPAIAN PEMBELAJARAN

Indikator ketercapaian diukur dengan:

|  |  |  |
| --- | --- | --- |
| CPL‐04 | CPMK‐03 | Kemampuan mahasiswa dalam menerapkan konsep konsep Stack |
| CPL‐08 |  | dan Aplikasinya |

### TEORI PENDUKUNG

Ada dua operasi yang digunakan pada stack, yaitu push (memasukkan elemen ke stack) dan pop (mengeluarkan elemen dari stack). Misalkan deklarasi stack terdiri dari 5 elemen

dengan menggunakan kelas Stack.

#define maks 5 class Stack {

friend ostream& operator<<(ostream&, const Stack&);

public :

Stack();

int penuh(int); int kosong(int); void cetak(); void push(char); char pop();

private :

char A[maks]; int banyak;

};

Gambar logiknya adalah sebagai berikut : (bayangkan stacknya rubuh)

operasi selalu diterapkan pada puncak stack

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| a | b | c | d | e |  |

1 2 3 4 5 atas

Operasi Push

Cara memasukkan elemen ke stack adalah dengan cara melihat status puncak stack. Penambahan elemen berarti banyak elemen bertambah 1. Jadi variabel **banyak** akan selalu bertambah 1 bila ada penambahan elemen seperti di bawah ini. Namun tentu saja, bila stack penuh, kita tidak dapat melakukan operasi push.

void Stack::push(char x) {

cout << "\nElemen masuk : " << x;

if (penuh(banyak)) cout << "Stack penuh"; else if (A[0]=='0') {

A[0] = x;

banyak++;

}

else {

for (int i=banyak; i>=0; i‐‐) A[i+1] = A[i];

A[0] = x;

banyak++;

}

}

Operasi Pop

Berkebalikan dengan operasi push, operasi pop status variabel **banyak** akan berkurang dengan 1. Namun kadang diperlukan pengecekan terhadap kondisi stack, yaitu bila stack kosong, tidaklah mungkin melakukan operasi pop.

char Stack::pop()

{ cout << "\nPop stack, elemen yang di‐pop : " << A[0]; char temp=A[0];

for (int i=0; i<banyak; i++) A[i] = A[i+1];

A[banyak] = '0'; banyak‐‐; return temp;

}

### HARDWARE DAN SOFTWARE

Hardware dan software yang digunakan dalam praktikum ini yaitu:

1. Komputer.
2. Borland C++ atau Kompilator sejenis.

### PRE‐TEST

Jawablah pertanyaan berikut (**Total Skor: 100):**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **CPL** | **CPMK** | **Pertanyaan** | **Skor** |
| 1. | CPL‐04 | CPMK‐03 | [KATEGORI 1] Diberikan **tumpukan** ( a,b,c,d,e,f ). | 100 |
|  | CPL‐08 |  | Jelaskan dengan contoh cara kerja membagi sebuah |  |
|  |  |  | tumpukan menjadi buah tumpukan. Gambarkan |  |
|  |  |  | proses pembagiannya dengan operasi pop |  |
|  |  |  | (mengeluarkan elemen) dan push (memasukkan elemen) |  |
|  |  |  | ! Kata dengan huruf tebal menunjukkan variabel. |  |
| 2. | CPL‐04 | CPMK‐03 | Diberikan tumpukan **tumpuk1** berisi ( a,b,c } dan **tumpuk2** | 100 |
|  | CPL‐08 |  | berisi (d,e,f ). Jelaskan cara kerja menggabungkan dua |  |
|  |  |  | tumpukan tersebut menjadi sebuah **tumpukan (**pastikan |  |
|  |  |  | **tumpuk1** dan **tumpuk2** sudah kosong! Kata dengan huruf |  |
|  |  |  | tebal menunjukkan variabel. |  |

### LANGKAH PRAKTIKUM

Aturan Penilaian (Total Skor: 100):

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No** | **CPL** | **CPMK** | **Pertanyaan** | **Dokumen**  **Pendukung** | **Skor** |
| 1. | CPL‐04  CPL‐08 | CPMK‐03 | Selesaikan langkah praktikum | Hasil praktikum | 100 |

#### Langkah‐Langkah Praktikum:

Ketik dan buatlah program dari omplementasi selengkapnya dari kelas Stack sebagai berikut.

|  |  |
| --- | --- |
| 1. | #include <iostream.h> |
| 2. | #define maks 5 |
| 3. | class Stack { |
| 4. | friend ostream& operator<<(ostream&, const Stack&); |
| 5. | public : |
| 6. | Stack(); |
| 7. | int penuh(int); |
| 8. | int kosong(int); |
| 9. | void cetak(); |
| 10. | void push(char); |
| 11. | char pop(); |
| 12. | private : |
| 13. | char A[maks]; |
| 14. | int banyak; |
| 15. | }; |
| 16. | ostream& operator<<(ostream& out, const Stack& s) |
| 17. | { cout << "\nIsi stack : "; |
| 18. | for (int i=0; i< s.banyak; i++) |
| 19. | out << s.A[i] << " "; |
| 20. | } |
| 21. | Stack::Stack() { |

|  |  |
| --- | --- |
| 22. | banyak = 0; |
| 23. | for (int i=0; i<maks; i++) |
| 24. | A[i] = '0'; |
| 25. | } |
| 26. | int Stack::penuh(int s) |
| 27. | { return s == maks ? 1 : 0; } |
| 28. | int Stack::kosong(int s) |
| 29. | { return s == 0 ? 1 : 0; } |
| 30. | void Stack::cetak() |
| 31. | { cout << "\nIsi stack : "; |
| 32. | for (int i=0; i< banyak; i++) |
| 33. | cout << A[i] << " "; } |
| 34. | void Stack::push(char x) |
| 35. | { cout << "\nElemen masuk : " << x; |
| 36. | if (penuh(banyak)) cout << "Stack penuh"; |
| 37. | else if (A[0]=='0') { |
| 38. | A[0] = x; |
| 39. | banyak++; } |
| 40. | else { |
| 41. | for (int i=banyak; i>=0; i‐‐) |
| 42. | A[i+1] = A[i]; |
| 43. | A[0] = x; |
| 44. | banyak++; } |
| 45. | } |
| 46. | char Stack::pop() |
| 47. | { cout << "\nPop stack, elemen yang di‐pop : " << A[0]; |
| 48. | char temp=A[0]; |

|  |  |
| --- | --- |
| 49. | for (int i=0; i<banyak; i++) A[i] = A[i+1]; |
| 50. | A[banyak] = '0'; |
| 51. | banyak‐‐; |
| 52. | return temp; |
| 53. | } |
| 54. | void main() { |
| 55. | Stack stack; |
| 56. | for (char c='a'; c<'d'; c++) { |
| 57. | stack.push(c); |
| 58. | stack.cetak(); |
| 59. | } |
| 60. | char p = stack.pop(); |
| 61. | stack.cetak(); |
| 62. | cout << "\n\nCetak pakai ovrloading : " << stack; |
| 63. | } |

### POST TEST

Jawablah pertanyaan berikut (**Total Skor: 100):**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **CPL** | **CPMK** | **Pertanyaan** | **Skor** |
| 1. | CPL‐04 CPL‐08 | CPMK‐03 | Buatlah model stack menggunakan array :   1. no ganjil : array statis 2. no genap : array dinamis |  |
| Program jalan (Screenshot input output) | 30 |
| Operasi menambah data (push) | 25 |
| Operasi menghapus data (pop) | 25 |
| Class | 20 |

### HASIL CAPAIAN PRAKTIKUM

Diisi oleh asisten setelah semua assessment dinilai.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **No** | **Bentuk**  **Assessment** | **CPL** | **CPMK** | **Bobot** | **Skor (0‐100)** | **Nilai Akhir**  **(Bobot x Skor)** |
| 1. | Pre‐Test | CPL‐04  CPL‐08 | CPMK‐03 | 30% |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 2. | Praktik | CPL‐04  CPL‐08 | CPMK‐03 | 30% |  |  |
| 3. | Post‐Test | CPL‐04  CPL‐08 | CPMK‐03 | 40% |  |  |
| **Total Nilai** | | | | | |  |

###### LEMBAR JAWABAN PRE‐TEST DAN POST‐TEST PRAKTIKUM

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

# PRAKTIKUM 7: SIMULASI QUEUE DENGAN ARRAY CIRCULAR

Pertemuan ke 7

**Total Alokasi Waktu** : 90 menit

* Materi : 10 menit
* Pre‐Test : 10 menit
* Praktikum : 40 menit
* Post‐Test : 30 menit

**Total Bobot Penilaian** : 100%

* Pre‐Test : 30 %
* Praktik : 30 %
* Post‐Test : 40 %

Pemenuhan CPL dan CPMK:

|  |  |
| --- | --- |
| CPL‐04 CPL‐08 | Mampu berpikir logis, kritis, sistematis dan inovatif, dan mampu mengambil keputusan secara tepat di bidang keahliannya dan mampu merancang dan mengimplementasikan algoritma/metode dalam mengidentifikasi dan memecahkan masalah yang melibatkan  perangkat lunak dan pemikiran komputasi |
| CPMK‐04 | Mampu memahami, menjelaskan, dan menerapkan konsep Konsep Antrian  lebih khusus Priority Queue diimplementasikan dengan Array (statis dan dinamis) dan Linklist |

### DESKRIPSI CAPAIAN PEMBELAJARAN

Setelah mengikuti praktikum ini mahasiswa diharapkan mampu:

* + 1. Menerapkan penggunaan array untuk mengimplementasikan struktur data queue

### INDIKATOR KETERCAPAIAN PEMBELAJARAN

Indikator ketercapaian diukur dengan:

|  |  |  |
| --- | --- | --- |
| CPL‐04 | CPMK‐04 | Kemampuan mahasiswa dalam menerapkan konsep Konsep Antrian |
| CPL‐08 |  | lebih khusus Priority Queue diimplementasikan dengan Array (statis |
|  |  | dan dinamis) dan Linklist |

### TEORI PENDUKUNG

Misalkan dipunyai array yang terdiri dari 6 elemen sebagai berikut :

|  |  |  |  |
| --- | --- | --- | --- |
| indeks | 1 | 2 | 3 |
|  |  |  |  |

banyak\_elemen = 0, depan = 0, belakang = 0

Untuk memonitor kondisi array, dibutuhkan tiga variabel lagi, yaitu depan, belakang, dan banyak\_elemen. Variabel depan digunakan untuk memonitor posisi elemen yang berada paling depan. Variabel belakang digunakan untuk memonitor posisi elemen yang berada paling belakang, sedangkan variabel banyak\_elemen digunakan untuk menghitung banyaknya elemen antrian. Dari gambar di atas, kondisi awal antrian adalah antrian dalam kondisi kosong yaitu nilai depan = belakang = banyak\_elemen = 0.

Misalkan diasumsikan struktur data dari queue adalah sebagai berikut :

const maks\_elemen = 6 ;

type antri = array[1..maks\_elemen] of char; var

antrian : antri;

banyak\_elemen, depan, belakang : integer; elemen : char;

Untuk menginisialisasi antrian dapat dibuat prosedur sebagai berikut :

Queue::Queue() { banyak = 0;

for (int i=0; i<maks; i++) A[i] = '0';

}

Untuk mengecek apakah antrian dalam keadaan kosong atau penuh, variabel banyak\_elemen dapat digunakan. Antrian penuh jika banyak\_elemen = maks\_elemen, antrian kosong jika banyak\_elemen = 0. Implementasinya dapat menggunakan fungsi sebagai berikut :

int Queue::kosong(int s) { return s == 0 ? 1 : 0;

}

int Queue::penuh(int s) { return s == maks ? 1 : 0;

}

Untuk menjelaskan operasi enqueue dan dequeue, diasumsikan :

* + 1. jika elemen masuk (operasi enqueue), maka nilai variabel depan bertambah 1 dan bila sampai maksimum elemen depan diset sama dengan 1.
    2. jika elemen keluar (operasi dequeue), maka nilai variabel belakang bertambah 1 dan bila sampai maksimum elemen depan diset sama dengan 1.

Operasi Enqueue (Memasukkan Elemen ke Queue)

kondisi awal

|  |  |  |  |
| --- | --- | --- | --- |
| indeks | 1 | 2 | 3 |
|  |  |  |  |

banyak\_elemen = 0, depan = 0, belakang = 0

enqueue A

|  |  |  |  |
| --- | --- | --- | --- |
| indeks | 1 | 2 | 3 |
|  | A |  |  |

enqueue B

enqueue C

enqueue D

dequeue

banyak\_elemen = 1, depan = 1, belakang = 1

|  |  |  |  |
| --- | --- | --- | --- |
| indeks | 1 | 2 | 3 |
|  | A | B |  |

banyak\_elemen = 1, depan = 1, belakang = 2

|  |  |  |  |
| --- | --- | --- | --- |
| indeks | 1 | 2 | 3 |
|  | A | B | C |

banyak\_elemen = 3, depan = 1, belakang = 3

Antrian penuh, …

|  |  |  |  |
| --- | --- | --- | --- |
| indeks | 1 | 2 | 3 |
|  |  | B | C |

banyak\_elemen = 2, depan = 2, belakang = 3 elemen yang keluar : A

|  |  |  |  |
| --- | --- | --- | --- |
| indeks | 1 | 2 | 3 |
|  |  |  | C |

dequeue

enqueue D

banyak\_elemen = 1, depan = 3, belakang = 3 elemen yang keluar : B

|  |  |  |  |
| --- | --- | --- | --- |
| indeks | 1 | 2 | 3 |
|  | D |  | C |

banyak\_elemen = 2, depan = 3, belakang = 1

terlihat bahwa masuknya elemen D ditempatkan “memutar”. dequeue

|  |  |  |  |
| --- | --- | --- | --- |
| indeks | 1 | 2 | 3 |
|  | D |  |  |

dequeue

banyak\_elemen = 1, depan = 1, belakang = 1 elemen yang keluar : C

|  |  |  |  |
| --- | --- | --- | --- |
| indeks | 1 | 2 | 3 |
|  |  |  |  |

banyak\_elemen = 0, depan = 0, belakang = 0 elemen yang keluar : D

Bila antrian kosong, kondisi antrian seperti pada saat awal. Dari perilaku langkah‐langkah di atas dapat dibuat prosedur enqueue sebagai berikut :

void Queue::enqueue(char x)

{ cout << "\nElemen : " << x << " masuk antrian";

if (penuh(banyak)) cout << "queue penuh"; else if (A[0]=='0') {

A[0] = x;

banyak++; } else {

for (int i=banyak; i>=0; i‐‐) A[i+1] = A[i];

A[0] = x;

banyak++;

}

}

Operasi Dequeue (Mengeluarkan Elemen dari Queue)

Bila pada prosedur enqueue banyak\_elemen bertambah 1, pada dequeue berkurang

1. Demikian pula, bila pada prosedur enqueue variabel yang aktif adalah belakang (bertambah 1 kecuali bila sudah pada akhir indeks ‐ memutar), maka pada dequeue yang aktif adalah variabel depan (bertambah 1 kecuali bila sudah pada akhir indeks ‐ memutar). Dequeue sengaja dibuat dalam bentuk fungsi agar elemen yang dikeluarkan dari antrian bisa digunakan oleh program lain.

char Queue::dequeue()

{

char temp=A[‐‐banyak];

cout << "\nDequeue elemen ‐‐> " << temp; A[banyak] = '0';

return temp;

}

### HARDWARE DAN SOFTWARE

Hardware dan software yang digunakan dalam praktikum ini yaitu:

1. Komputer.
2. Borland C++ atau Kompilator sejenis.

### PRE‐TEST

Jawablah pertanyaan berikut (**Total Skor: 100):**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **CPL** | **CPMK** | **Pertanyaan** | **Skor** |
| 1. | CPL‐04 | CPMK‐04 | Apa yang dimaksud dengan antrian (queue)? | 30 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | CPL‐08 |  |  |  |
| 2. | CPL‐04  CPL‐08 | CPMK‐04 | Jelaskan cara kerja menambah dan mengeluarkan elemen  antrian dengan posisi elemen terdepan | 70 |
| 3. | CPL‐04  CPL‐08 | CPMK‐04 | Apa yang dimaksud dengan antrian (queue)? | 30 |
| 4. | CPL‐04  CPL‐08 | CPMK‐04 | Jelaskan cara kerja menambah dan mengeluarkan elemen  antrian dengan posisi elemen paling belakang | 70 |

### LANGKAH PRAKTIKUM

Aturan Penilaian (Total Skor: 100):

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No** | **CPL** | **CPMK** | **Pertanyaan** | **Dokumen Pendukung** | **Skor** |
| 1. | CPL‐04  CPL‐08 | CPMK‐04 | Selesaikan langkah praktikum | Hasil praktikum | 100 |

#### Langkah‐Langkah Praktikum:

Ketik dan jalankan program implementasi selengkapnya dari kelas Queue berikut.

|  |  |
| --- | --- |
| 1. | #include <iostream.h> |
| 2. | #define maks 5 |
| 3. | class Queue { |
| 4. | friend ostream& operator<<(ostream&, const Queue&); |
| 5. | public : |
| 6. | Queue(); |
| 7. | int penuh(int); |
| 8. | int kosong(int); |
| 9. | void cetak(); |
| 10. | void enqueue(char); |
| 11. | char dequeue(); |
| 12. | private : |
| 13. | char A[maks]; |
| 14. | int banyak; |
| 15. | }; |
| 16. | ostream& operator<<(ostream& out, const Queue& s) |

|  |  |
| --- | --- |
| 17. | { cout << "\nIsi Queue : "; |
| 18. | for (int i=0; i< s.banyak; i++) |
| 19. | out << s.A[i] << " "; |
| 20. | } |
| 21. | Queue::Queue() { |
| 22. | banyak = 0; |
| 23. | for (int i=0; i<maks; i++) |
| 24. | A[i] = '0'; |
| 25. | } |
| 26. | int Queue::penuh(int s) |
| 27. | { return s == maks ? 1 : 0; } |
| 28. | int Queue::kosong(int s) |
| 29. | { return s == 0 ? 1 : 0; } |
| 30. | void Queue::cetak() |
| 31. | { cout << "\nIsi Queue : "; |
| 32. | for (int i=0; i< banyak; i++) |
| 33. | cout << A[i] << " "; |
| 34. | } |
| 35. | void Queue::enqueue(char x) |
| 36. | { |
| 37. | cout << "\nElemen : " << x << " masuk antrian"; |
| 38. | if (penuh(banyak)) cout << "queue penuh"; |
| 39. | else if (A[0]=='0') { |
| 40. | A[0] = x; |
| 41. | banyak++; |
| 42. | } |
| 43. | else { |

|  |  |
| --- | --- |
| 44. | For (int i=banyak; i>=0; i‐‐) |
| 45. | A[i+1] = A[i]; |
| 46. | A[0] = x; |
| 47. | banyak++; |
| 48. | } |
| 49. | } |
| 50. | char Queue::dequeue() { |
| 51. | char temp=A[‐‐banyak]; |
| 52. | cout << "\nDequeue elemen ‐‐> " << temp; |
| 53. | A[banyak] = '0'; |
| 54. | return temp; |
| 55. | } |
| 56. | main() { |
| 57. | Queue q; |
| 58. | for (char c='a'; c<'d'; c++) { |
| 59. | q.enqueue(c); |
| 60. | cout << q; |
| 61. | } |
| 62. | char p = q.dequeue(); |
| 63. | q.cetak(); |
| 64. | cout << "\n\nCetak pakai overloading : " << q; |
| 65. | return 0; |
| 66. | } |

### POST TEST

Jawablah pertanyaan berikut (**Total Skor: 100):**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **CPL** | **CPMK** | **Pertanyaan** | **Skor** |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1. | CPL‐04 CPL‐08 | CPMK‐04 | Gunakan postes Pertemuan 5 untuk membuat model antrian menggunakan link list :   1. no ganjil : double link list 2. no genap : circular link list |  |
| Program jalan (Screenshot input output) | 30 |
| Operasi menambah data (enqueue) | 25 |
| Operasi menghapus data (dequeue) | 25 |
| Class | 20 |

### HASIL CAPAIAN PRAKTIKUM

Diisi oleh asisten setelah semua assessment dinilai.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **No** | **Bentuk**  **Assessment** | **CPL** | **CPMK** | **Bobot** | **Skor (0‐100)** | **Nilai Akhir**  **(Bobot x Skor)** |
| 1. | Pre‐Test | CPL‐04  CPL‐08 | CPMK‐04 | 30% |  |  |
| 2. | Praktik | CPL‐04  CPL‐08 | CPMK‐04 | 30% |  |  |
| 3. | Post‐Test | CPL‐04  CPL‐08 | CPMK‐04 | 40% |  |  |
| **Total Nilai** | | | | | |  |

###### LEMBAR JAWABAN PRE‐TEST DAN POST‐TEST PRAKTIKUM

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

PRAKTIKUM 8: ANTRIAN BERPRIORITAS MENGGUNAKAN DOUBLY

LINK LIST

Pertemuan ke 8

**Total Alokasi Waktu** : 90 menit

* Materi : 10 menit
* Pre‐Test : 10 menit
* Praktikum : 40 menit
* Post‐Test : 30 menit

**Total Bobot Penilaian** : 100%

* Pre‐Test : 30 %
* Praktik : 30 %
* Post‐Test : 40 %

Pemenuhan CPL dan CPMK:

|  |  |
| --- | --- |
| CPL‐04 CPL‐08 | Mampu berpikir logis, kritis, sistematis dan inovatif, dan mampu mengambil keputusan secara tepat di bidang keahliannya dan mampu merancang dan mengimplementasikan algoritma/metode dalam mengidentifikasi dan memecahkan masalah yang melibatkan  perangkat lunak dan pemikiran komputasi |
| CPMK‐04 | Mampu memahami, menjelaskan, dan menerapkan konsep Konsep Antrian lebih khusus Priority Queue diimplementasikan dengan Array (statis dan dinamis)  dan Linklist |

### DESKRIPSI CAPAIAN PEMBELAJARAN

Setelah mengikuti praktikum ini mahasiswa diharapkan mampu:

* + 1. Membuat gambar logic operasi dalam antrian berprioritas yaitu operasi menambah dan menghapus elemen
    2. Mempraktekan gambar logic menggunakan salah satu bahasa pemrograman

### INDIKATOR KETERCAPAIAN PEMBELAJARAN

Indikator ketercapaian diukur dengan:

|  |  |  |
| --- | --- | --- |
| CPL‐04 | CPMK‐04 | Kemampuan mahasiswa dalam menerapkan konsep Konsep Antrian |
| CPL‐08 |  | lebih khusus Priority Queue diimplementasikan dengan Array (statis |
|  |  | dan dinamis) dan Linklist |

### TEORI PENDUKUNG

Teori pendukung bisa dibaca di Diktat Struktur Data.

### HARDWARE DAN SOFTWARE

Hardware dan software yang digunakan dalam praktikum ini yaitu:

1. Komputer.
2. Borland C++ atau Kompilator sejenis

### PRE‐TEST

Jawablah pertanyaan berikut (**Total Skor: 100):**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **CPL** | **CPMK** | **Pertanyaan** | | | | | **Skor** |
| 1. | CPL‐04 | CPMK‐04 | Andaikan terdapat 4 elemen dalam doubly link list | | | | | 25 |
|  | CPL‐08 |  | D yaitu:2 5 8 9. Buatlah struktur kelas dari Node | | | | |  |
|  |  |  | yang memiliki 2 pointer dan 1 data | | | | |  |
| 2. | CPL‐04 | CPMK‐04 | Buat gambar logic untuk menambah elemen | | | | | 25 |
|  | CPL‐08 |  | 10 menggunakan doubly link list. Pastikan dalam doubly | | | | |  |
|  |  |  | link list itu ada pointer **akhir** yang menunjuk elemen | | | | |  |
|  |  |  | terakhir | | | | |  |
| 3. | CPL‐04 | CPMK‐04 | Buat gambar logic untuk menambah elemen | | | | | 25 |
|  | CPL‐08 |  | 10 menggunakan doubly link list. Pastikan dalam doubly | | | | |  |
|  |  |  | link list itu ada pointer **akhir** yang menunjuk elemen | | | | |  |
|  |  |  | terakhir | | | | |  |
| 4. | CPL‐04 | CPMK‐04 | Buat gambar | logic | menghapus | elemen | pertama | 25 |
|  | CPL‐08 |  | (paling depan) |  |  |  |  |  |

### LANGKAH PRAKTIKUM

Aturan Penilaian (Total Skor: 100):

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No** | **CPL** | **CPMK** | **Pertanyaan** | **Dokumen Pendukung** | **Skor** |
| 1. | CPL‐04  CPL‐08 | CPMK‐04 | Selesaikan langkah praktikum | Hasil praktikum | 100 |

#### Langkah‐Langkah Praktikum:

Langkah praktikum bisa dibaca di Diktat Struktur Data

### POST TEST

Jawablah pertanyaan berikut (**Total Skor: 100):**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **CPL** | **CPMK** | **Pertanyaan** | **Skor** |
| 1. | CPL‐04 CPL‐08 | CPMK‐04 | Gunakan postes Pertemuan 5 untuk membuat model antrian berprioritas menggunakan link list :   1. no ganjil : double link list 2. no genap : circular link list |  |
| Program jalan (Screenshot input output) | 30 |
| Operasi menambah data (enqueue) | 25 |
| Operasi menghapus data (dequeue) | 25 |
| Class | 20 |

### HASIL CAPAIAN PRAKTIKUM

Diisi oleh asisten setelah semua assessment dinilai.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **No** | **Bentuk**  **Assessment** | **CPL** | **CPMK** | **Bobot** | **Skor (0‐100)** | **Nilai Akhir**  **(Bobot x Skor)** |
| 1. | Pre‐Test | CPL‐04  CPL‐08 | CPMK‐04 | 30% |  |  |
| 2. | Praktik | CPL‐04  CPL‐08 | CPMK‐04 | 30% |  |  |
| 3. | Post‐Test | CPL‐04  CPL‐08 | CPMK‐04 | 40% |  |  |
| **Total Nilai** | | | | | |  |

###### LEMBAR JAWABAN PRE‐TEST DAN POST‐TEST PRAKTIKUM

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

# PRAKTIKUM 9: POHON

Pertemuan ke 9

**Total Alokasi Waktu** : 90 menit

* Materi : 10 menit
* Pre‐Test : 10menit
* Praktikum : 40 menit
* Post‐Test : 30 menit

**Total Bobot Penilaian** : 100%

* Pre‐Test : 30 %
* Praktik : 30 %
* Post‐Test : 40 %

Pemenuhan CPL dan CPMK:

|  |  |
| --- | --- |
| CPL‐04 CPL‐08 | Mampu berpikir logis, kritis, sistematis dan inovatif, dan mampu mengambil keputusan secara tepat di bidang keahliannya dan mampu merancang dan mengimplementasikan algoritma/metode dalam mengidentifikasi dan memecahkan masalah yang melibatkan  perangkat lunak dan pemikiran komputasi |
| CPMK‐05 | Mampu memahami, menjelaskan, dan menerapkan konsep Struktur Data Pohon  dengan penelusuran dengan menggunakan cara Rekursi, implementasinya Huffman Coding dan AVL |

### DESKRIPSI CAPAIAN PEMBELAJARAN

Setelah mengikuti praktikum ini mahasiswa diharapkan mampu:

1. Menerapkan konsep pohon biner

### INDIKATOR KETERCAPAIAN PEMBELAJARAN

Indikator ketercapaian diukur dengan:

|  |  |  |
| --- | --- | --- |
| CPL‐04 | CPMK‐05 | Kemampuan mahasiswa dalam menerapkan konsep Struktur Data Pohon |
| CPL‐08 |  | dengan penelusuran dengan menggunakan cara Rekursi, |
|  |  | implementasinya Huffman Coding dan AVL |

### TEORI PENDUKUNG

Struktur data pohon merupakan struktur data tak linier. Setiap pohon biner akan mempunyai node yang anaknya paling banyak 2, yaitu anak kiri (left child) dan anak kanan (right child). Bila kita menggunakan link list, maka selain info akan terdapat dua pointer,

yaitu pointer kiri dan pointer kanan yang masing‐masing mewakili anak dari node tersebut. Model dari node dapat dirancang seperti kelas BinaryTreeNode di bawah ini.

|  |  |
| --- | --- |
|  | File btnode.h |
| 1. | template <class T> class BinaryTree; |
| 2. | template <class T> |
| 3. | class BinaryTreeNode { |
| 4. | friend void Visit(BinaryTreeNode<T> \*); |
| 5. | friend void InOrder(BinaryTreeNode<T> \*); |
| 6. | friend void PreOrder(BinaryTreeNode<T> \*); |
| 7. | friend void PostOrder(BinaryTreeNode<T> \*); |
| 8. | friend void LevelOrder(BinaryTreeNode<T> \*); |
| 9. | friend void main(void); |
| 10. | public: |
| 11. | BinaryTreeNode() {LeftChild = RightChild = 0;} |
| 12. | BinaryTreeNode(const T& e) |
| 13. | {data = e; |
| 14. | LeftChild = RightChild = 0;} |
| 15. | BinaryTreeNode(const T& e, BinaryTreeNode \*l, |
| 16. | BinaryTreeNode \*r) |
| 17. | {data = e; |
| 18. | LeftChild = l; |

|  |  |
| --- | --- |
| 19. | RightChild = r;} |
| 20. | private: |
| 21. | T data; |
| 22. | BinaryTreeNode<T> \*LeftChild, |
| 23. | \*RightChild; |
| 24. | }; |

Terdapat 3 konstruktor kelas Binary Tree Node. Yang pertama (11), node yang tercipta belum mempunyai info **data** maupun anak. Yang kedua (12), node baru dengan info **data** e tanpa anak. Sedangkan yang terakhir (15), node baru dengan info **data** e, dan mempunyai anak kiri l dan anak kanan r.

Gambar logik sebuah node adalah sebagai berikut.

|  |  |  |
| --- | --- | --- |
| LeftChild | Data | RightChild |

Penelusuran Node

Ada 4 cara kunjungan node, yaitu preorder, inorder, Postorder dan levelorder. Penelusuran PreOrder dilakukan secara rekursif dengan urutan :

* + 1. Cetak isi node (5)
    2. Kunjungi anak kiri sampai habis (6)
    3. Kunjungi anak kanan sampai habis (7)

|  |  |
| --- | --- |
|  | Penelusuran PreOrder \*t. |
| 1.  2.  3.  4. | template <class T>  void PreOrder(BinaryTreeNode<T> \*t)  {  if (t) { |

|  |  |
| --- | --- |
| 5.  6.  7.  8.  9. | Visit(t); PreOrder(t‐>LeftChild); PreOrder(t‐>RightChild);  }  } |

Penelusuran InOrder dilakukan secara rekursif dengan urutan :

1. Kunjungi anak kiri sampai habis (5)
2. Cetak isi node (6)
3. Kunjungi anak kanan sampai habis (7)

|  |  |
| --- | --- |
|  | Penelusuran InOrder \*t. |
| 1.  2.  3.  4.  5.  6.  7.  8.  9. | template <class T>  void InOrder(BinaryTreeNode<T> \*t)  {  if (t) { InOrder(t‐>LeftChild); Visit(t); InOrder(t‐>RightChild);  }  } |

Penelusuran PostOrder dilakukan secara rekursif dengan urutan :

1. Kunjungi anak kiri sampai habis (5)
2. Kunjungi anak kanan sampai habis (6)
3. Cetak isi node (7)

|  |  |
| --- | --- |
|  | Penelusuran PostOrder \*t. |
| 1.  2.  3.  4.  5.  6.  7.  8.  9. | template <class T>  void PostOrder(BinaryTreeNode<T> \*t)  {  if (t) { PostOrder(t‐>LeftChild); PostOrder(t‐>RightChild); Visit(t);  }  } |

Penelusuran Level Order menggunakan struktur data antrian untuk implementasinya.

|  |  |
| --- | --- |
|  | Penelusuran LevelOrder \*t. |
| 1.  2.  3.  4.  5.  6.  7.  8.  9.  10.  11.  12. | template <class T>  void LevelOrder(BinaryTreeNode<T> \*t)  {  LinkedQueue<BinaryTreeNode<T>\*> Q; while (t) {  Visit(t);  if (t‐>LeftChild) Q.Add(t‐>LeftChild);  if (t‐>RightChild) Q.Add(t‐>RightChild); try {Q.Delete(t);}  catch (OutOfBounds) {return;} |

|  |  |
| --- | --- |
|  | }  } |

### HARDWARE DAN SOFTWARE

Hardware dan software yang digunakan dalam praktikum ini yaitu:

1. Komputer.
2. Borland C++ atau Kompilator sejenis.

### PRE‐TEST

Jawablah pertanyaan berikut (**Total Skor: 100):**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **CPL** | **CPMK** | **Pertanyaan** | **Skor** |
| 1. | CPL‐04 CPL‐08 | CPMK‐05 | Ada sebuah binary tree kosong, kemudian diinsertkan : J R D G T E M H P A F Q (tata cara pembentukannya dijelaskan  di kelas). |  |
| Gambarkan binary tree nya langkah demi langkah | 55 |
| Tentukan Inorder, Postorder, dan Preordernya | 45 |

### LANGKAH PRAKTIKUM

Aturan Penilaian (Total Skor: 100):

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No** | **CPL** | **CPMK** | **Pertanyaan** | **Dokumen**  **Pendukung** | **Skor** |
| 1. | CPL‐04  CPL‐08 | CPMK‐05 | Selesaikan langkah praktikum | Hasil praktikum | 100 |

#### Langkah‐Langkah Praktikum:

Ketik dan jalankan implementasi selengkapnya dari penelusuran node berikut ini.

|  |  |
| --- | --- |
|  | File kunjungan.cpp |
| 1. | #include <iostream.h> |
| 2. | #include "lqueue.h" |
| 3. | #include "btnode.h" |
| 4. | #include "xcept.h" |
| 5. | template <class T> |
| 6. | void Visit(BinaryTreeNode<T> \*x) { |
| 7. | cout << x‐>data << ' '; |
| 8. | } |

|  |  |
| --- | --- |
| 9. | template <class T> |
| 10. | void PreOrder(BinaryTreeNode<T> \*t) { |
| 11. | if (t) { |
| 12. | Visit(t); |
| 13. | PreOrder(t‐>LeftChild); |
| 14. | PreOrder(t‐>RightChild); |
| 15. | } |
| 16. | } |
| 17. | template <class T> |
| 18. | void InOrder(BinaryTreeNode<T> \*t) { |
| 19. | if (t) { |
| 20. | InOrder(t‐>LeftChild); |
| 21. | Visit(t); |
| 22. | InOrder(t‐>RightChild); |
| 23. | } |
| 24. | } |
| 25. | template <class T> |
| 26. | void PostOrder(BinaryTreeNode<T> \*t) { |
| 27. | if (t) { |
| 28. | PostOrder(t‐>LeftChild); |
| 29. | PostOrder(t‐>RightChild); |
| 30. | Visit(t); |
| 31. | } |
| 32. | } |
| 33. | template <class T> |
| 34. | void LevelOrder(BinaryTreeNode<T> \*t) { |
| 35. | LinkedQueue<BinaryTreeNode<T>\*> Q; |

|  |  |
| --- | --- |
| 36. | while (t) { |
| 37. | Visit(t); |
| 38. | if (t‐>LeftChild) Q.Add(t‐>LeftChild); |
| 39. | if (t‐>RightChild) Q.Add(t‐>RightChild); |
| 40. | try {Q.Delete(t);} |
| 41. | catch (OutOfBounds) {return;} |
| 42. | } |
| 43. | } |
| 44. | void main(void) { |
| 45. | BinaryTreeNode<int> x, y, z; |
| 46. | x.data = 1; |
| 47. | y.data = 2; |
| 48. | z.data = 3; |
| 49. | x.LeftChild = &y; |
| 50. | x.RightChild = &z; |
| 51. | y.LeftChild = y.RightChild = z.LeftChild = z.RightChild  = 0; |
| 52. | cout << "Kunjungan Inorder : "; |
| 53. | InOrder(&x); |
| 54. | cout << endl; |
| 55. | cout << " Kunjungan Preorder : "; |
| 56. | PreOrder(&x); |
| 57. | cout << endl; |
| 58. | cout << " Kunjungan Postorder : "; |
| 59. | PostOrder(&x); |
| 60. | cout << endl; |
| 61. | cout << " Kunjungan Level order : "; |

|  |  |
| --- | --- |
| 62. | LevelOrder(&x); |
| 63. | cout << endl; |
| 64. | } |

Kelas bantu LinkedQueue

|  |  |
| --- | --- |
|  | File lqueue.h |
| 1. | #include "node.h" |
| 2. | #include "xcept.h" |
| 3. | template<class T> |
| 4. | class LinkedQueue { |
| 5. | public: |
| 6. | LinkedQueue() {front = rear = 0;} |
| 7. | ~LinkedQueue(); |
| 8. | int IsEmpty() const |
| 9. | {return ((front) ? 0 : 1);} |
| 10. | int IsFull() const; |
| 11. | T First() const; |
| 12. | T Last() const; |
| 13. | LinkedQueue<T>& Add(const T& x); |
| 14. | LinkedQueue<T>& Delete(T& x); |
| 15. | private: |
| 16. | Node<T> \*front; |
| 17. | Node<T> \*rear; |
| 18. | }; |
| 19. | template<class T> |
| 20. | LinkedQueue<T>::~LinkedQueue() { |
| 21. | Node<T> \*next; |

|  |  |
| --- | --- |
| 22. | while (front) { |
| 23. | next = front‐>link; |
| 24. | delete front; |
| 25. | front = next; |
| 26. | } |
| 27. | } |
| 28. | template<class T> |
| 29. | int LinkedQueue<T>::IsFull() const { |
| 30. | Node<T> \*p; |
| 31. | try {p = new Node<T>; |
| 32. | delete p; |
| 33. | return 0;} |
| 34. | catch (NoMem) {return 1;} |
| 35. | } |
| 36. | template<class T> |
| 37. | T LinkedQueue<T>::First() const |
| 38. | { if (IsEmpty()) throw OutOfBounds(); |
| 39. | return front‐>data; |
| 40. | } |
| 41. | template<class T> |
| 42. | T LinkedQueue<T>::Last() const { |
| 43. | if (IsEmpty()) throw OutOfBounds(); |
| 44. | return rear‐>data; |
| 45. | } |
| 46. | template<class T> |
| 47. | LinkedQueue<T>& LinkedQueue<T>::Add(const T& x)  { |

|  |  |
| --- | --- |
| 48. | Node<T> \*p = new Node<T>; |
| 49. | p‐>data = x; |
| 50. | p‐>link = 0; |
| 51. | if (front) rear‐>link = p; |
| 52. | else front = p; |
| 53. | rear = p; |
| 54. | return \*this; |
| 55. | } |
| 56. | template<class T> |
| 57. | LinkedQueue<T>& LinkedQueue<T>::Delete(T& x) { |
| 58. | if (IsEmpty()) throw OutOfBounds(); |
| 59. | x = front‐>data; |
| 60. | Node<T> \*p = front; |
| 61. | front = front‐>link; |
| 62. | delete p; |
| 63. | return \*this; |
| 64. | } |

### POST TEST

Jawablah pertanyaan berikut (**Total Skor: 100):**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **CPL** | **CPMK** | **Pertanyaan** | **Skor** |
| 1. | CPL‐04 | CPMK‐05 | Dari hasil pretes di atas, buatlah main function |  |
|  | CPL‐08 |  | (lihat contoh main function di atas). Cocokkan |
|  |  |  | hasil penelusuran Inorder, Postorder, dan Preordernya |
|  |  |  | dari pretes anda. |
|  |  |  | Fungsi main dan Program jalan (Screenshot input output) | 40 |
|  |  |  | Fungsi dan output inorder | 15 |
|  |  |  | Fungsi dan output postorder | 15 |
|  |  |  | Fungsi dan output preorder | 15 |
|  |  |  | Class | 15 |

### HASIL CAPAIAN PRAKTIKUM

Diisi oleh asisten setelah semua assessment dinilai.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **No** | **Bentuk**  **Assessment** | **CPL** | **CPMK** | **Bobot** | **Skor (0‐100)** | **Nilai Akhir**  **(Bobot x Skor)** |
| 1. | Pre‐Test | CPL‐04  CPL‐08 | CPMK‐05 | 30% |  |  |
| 2. | Praktik | CPL‐04  CPL‐08 | CPMK‐05 | 30% |  |  |
| 3. | Post‐Test | CPL‐04  CPL‐08 | CPMK‐05 | 40% |  |  |
| **Total Nilai** | | | | | |  |

###### LEMBAR JAWABAN PRE‐TEST DAN POST‐TEST PRAKTIKUM

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

# PRAKTIKUM 10:SEKUEN

Pertemuan ke 10

**Total Alokasi Waktu** : 90 menit

* Materi : 10 menit
* Pre‐Test : 10 menit
* Praktikum : 40 menit
* Post‐Test : 30 menit

**Total Bobot Penilaian** : 100%

* Pre‐Test : 30 %
* Praktik : 30 %
* Post‐Test : 40 %

Pemenuhan CPL dan CPMK:

|  |  |
| --- | --- |
| CPL‐04 CPL‐08 | Mampu berpikir logis, kritis, sistematis dan inovatif, dan mampu mengambil keputusan secara tepat di bidang keahliannya dan mampu merancang dan mengimplementasikan algoritma/metode dalam mengidentifikasi dan memecahkan masalah yang melibatkan  perangkat lunak dan pemikiran komputasi |
| CPMK‐05 | Mampu memahami, menjelaskan, dan menerapkan konsep Struktur Data Pohon  dengan penelusuran dengan menggunakan cara Rekursi, implementasinya Huffman Coding dan AVL |

### DESKRIPSI CAPAIAN PEMBELAJARAN

Setelah mengikuti praktikum ini mahasiswa diharapkan mampu:

* + 1. Mengamati proses perputaran pohon setimbang
    2. Menganalisis dan menggambar logic proses perputaran
    3. Mengimplementasikan dalam bahasa pemrogaman

### INDIKATOR KETERCAPAIAN PEMBELAJARAN

Indikator ketercapaian diukur dengan:

|  |  |  |
| --- | --- | --- |
| CPL‐04 | CPMK‐05 | Kemampuan mahasiswa dalam menerapkan konsep Struktur Data Pohon |
| CPL‐08 |  | dengan penelusuran dengan menggunakan cara Rekursi, |
|  |  | implementasinya Huffman Coding dan AVL |

### TEORI PENDUKUNG

Left Rotation

|  |  |
| --- | --- |
| State | Action |
|  | Kondisi awal pohon di masukan 7 kemudian di masukan 10. Karna angka 10 lebih besar maka akan di letakan di cabang sebelah kanan. |
|  | Setelah itu di masukan 15. Karna angka 15 lebih besar maka akan di letakan di cabang sebelah kanan 10. |
|  | Selisih kedalaman cabang kiri dan cabang kanan lebih dari satu, di katakan tidak setimbang. Maka perlu di lakukan perputaran. |
|  | Ini lah kondisi terakhir pohon yang sudah setimbang. |

Right Rotation

|  |  |
| --- | --- |
| State | Action |
|  | Kondisi awal pohon di masukan 15 kemudian di masukan 8. Karna angka 8 lebih kecil maka akan di letakan di cabang sebelah kiri. |
|  | Setelah itu di masukan 2. Karna angka 2 lebih kecil maka akan di letakan di cabang sebelah kiri 8. |
|  | Selisih kedalaman cabang kiri dan cabang kanan lebih dari satu, di katakan tidak setimbang. Maka perlu di lakukan perputaran. |
|  | Ini lah kondisi terakhir pohon yang sudah setimbang. |

Left‐Right Rotation

|  |  |
| --- | --- |
| State | Action |
|  | Kondisi awal pohon di masukan 20 di lanjutkan 12 lalu di lanjutkan 15. |
|  | Terlihat ketidak seimbangan maka di lakukan perputaran 1 kali ke kiri. |
|  | Begitu terjadi perputaran, 15 akan berada di tengah. |
|  | Selisih kedalaman cabang kiri dan cabang kanan lebih dari satu, di katakan tidak setimbang.  Maka perlu di lakukan perputaran. |
|  | Ini lah kondisi terakhir pohon yang sudah setimbang. |

Right‐Left Rotation

|  |  |
| --- | --- |
| State | Action |
|  | Kondisi awal pohon di masukan 5 di lanjutkan 17 lalu di lanjutkan 10. |
|  | Terlihat ketidak seimbangan maka di lakukan perputaran 1 kali ke kanan. |
|  | Begitu terjadi perputaran, 10 akan berada di tengah. |
|  | Selisih kedalaman cabang kiri dan cabang kanan lebih dari satu, di katakan tidak setimbang.  Maka perlu di lakukan perputaran. |
|  | Ini lah kondisi terakhir pohon yang sudah setimbang. |

### HARDWARE DAN SOFTWARE

Hardware dan software yang digunakan dalam praktikum ini yaitu:

1. Komputer.
2. Borland C++ atau Kompilator sejenis.

### PRE‐TEST

Jawablah pertanyaan berikut (**Total Skor: 100):**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **CPL** | **CPMK** | **Pertanyaan** | **Skor** |
| 1. | CPL‐04 | CPMK‐05 | Siapkan 1 lembar kertas dan buatlah gambar logic | 100 |
|  | CPL‐08 |  | dari data di bawah ini tanpa menggunakan animasi |  |
|  |  |  | **insert 64** |  |
|  |  |  | **insert 60** |  |
|  |  |  | **insert 86** |  |
|  |  |  | **insert 92** |  |
|  |  |  | **insert 30** |  |
|  |  |  | **insert 95** |  |
|  |  |  | **insert 50** |  |
|  |  |  | **insert 13** |  |
|  |  |  | **insert 38** |  |
|  |  |  | **insert 65** |  |
|  |  |  | **insert 35** |  |
|  |  |  | **insert 40** |  |
|  |  |  | **insert 96** |  |
|  |  |  | **insert 80** |  |
|  |  |  | **insert 21** |  |

### LANGKAH PRAKTIKUM

Aturan Penilaian (Total Skor: 100):

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No** | **CPL** | **CPMK** | **Pertanyaan** | **Dokumen**  **Pendukung** | **Skor** |
| 1. | CPL‐04  CPL‐08 | CPMK‐05 | Selesaikan langkah praktikum | Hasil praktikum | 100 |

#### Langkah‐Langkah Praktikum:

Bukalah tautan dari situs berikut https://visualgo.net/bn/bst. Lakukan langkah Langkah berikut**: insert 50**

insert 19

insert 18

insert 57

insert 95

insert 77

insert 39

insert 61

insert 23

insert 56

### POST TEST

Jawablah pertanyaan berikut (**Total Skor: 100):**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **CPL** | **CPMK** | **Pertanyaan** | **Skor** |
| 1. | CPL‐04 CPL‐08 | CPMK‐05 | Dalam contoh di landasan teori buatlah  implementasi dalam bahasa pemrogaman. |  |
| Program jalan (Screenshot input output) | 30 |
| Input | 40 |
| Output | 30 |

### HASIL CAPAIAN PRAKTIKUM

Diisi oleh asisten setelah semua assessment dinilai.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **No** | **Bentuk Assessment** | **CPL** | **CPMK** | **Bobot** | **Skor (0‐100)** | **Nilai Akhir (Bobot x Skor)** |
| 1. | Pre‐Test | CPL‐04  CPL‐08 | CPMK‐05 | 30% |  |  |
| 2. | Praktik | CPL‐04  CPL‐08 | CPMK‐05 | 30% |  |  |
| 3. | Post‐Test | CPL‐04  CPL‐08 | CPMK‐05 | 40% |  |  |
| **Total Nilai** | | | | | |  |

###### LEMBAR JAWABAN PRE‐TEST DAN POST‐TEST PRAKTIKUM

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

# DAFTAR PUSTAKA

* + 1. Aspnes, James, 2015, *Data Structures and C Programming ‐ Computer Science*
    2. Sen, Priya, 2016, *Data Structures & Algorithms*
    3. Goodrich, Michael T., 2011, *Data Structures and Algorithms in C++ 2e*
    4. James R., Stefan B., dan David W, 2003, *A Laboratory Course in C++ Data*
    5. *Structure*, Jones and Barlett Pub, Subbury USA
    6. Karumanchi, Narasimha, 2016, *Data Structure and Algorithmic Thinking with Python DataStructure and Algorithmic Puzzles,* CareerMonk Publications
    7. Ramesh, Anand, dan Gautam Vasappanavara, ***C & Data Structures by Practice, New AgeInternational***, New Delhi, 2007
    8. Sahni, 5, ***Data Structures, Algoritms and Application in C++,*** McGraww Hill Co, Singapore, 2009
    9. Weiss, Mark Allen, 2013, *Data Structures and Algorithm Analysis in C++ ‐ Manal Helal Site*
    10. <http://frank.mtsu.edu/~csci217/manual>
    11. [http://code.activerstate.com/recipes/577480‐hu](http://code.activerstate.com/recipes/577480)ffman‐data‐compression/
    12. <http://oopweb.com/Algorithms/Documents/AnimatedAlgorithms/VolumeFram> e.html
    13. <http://www.cs.usfca.edu/~galles/visualization/>
    14. <http://webdiis.unizar.es/asignaturas/EDA/AVLTree/avltree.html>
    15. https://visualgo.net/bn/bst

RAT LABORATORIUM



f

**-1,l** S1 INFORMATIKA

\\ *-I/* FAKULTAS TEKNOLOGI INDUSTRI

,-.-/ UNIVERSITAS AHMAD DAHLAN