PRAKTIKUM 1

Mengenal Variabel dan Konstanta

Tujuan Praktikum :

1. Mempraktekkan variabel dan konstanta
2. Membuat file kerja dengan Bahasa C++ dan Pascal
3. Memulai membuat program baru, meng-compile, menyimpan dan menjalankan program yang telah dibuat
4. Memanggil/membuka kembali program yang telah disimpan
5. Membuat program menjadi program yang execute

Petunjuk : Mulailah pekerjaan anda dengan niat yang ikhlas sesuai dengan agama dan kepercayaan masing-masing.

(1). Tulislah source program berikut ini :

C ++ :

```
/* Program Pertamaku */
#include <iostream.h>
#include <conio.h>

#define sapa1 "\t\tSelamat Datang di Labotratorium Pemrograman Dasar\n";
#define sapa2 "\t\tSistem Komputer FTI Universitas Andalas\n"
#define nama "....................."; /*Tuliskan Nama Anda */
#define alamat "....................."; /*Tuliskan Alamat Anda */
#define sekolah "....................."; /*Tuliskan Nama Sekolah Asal Anda */

using namespace std;
int main ()
{
    cout<<sapa1;cout<<sapa2;
    cout<<"\n\nNama Anda Adalah  : ";cout<<nama;
    cout<<"\nAnda Beralamat di : ";cout<<alamat;
    cout<<"\nSekolah Asal Anda : ";cout<<sekolah;
    cout<<"\n\n\t\tSELAMAT BELAJAR BAHASA C++";
getch();
}
```

1. Simpan program
   - ✓ Tanyakan kepada asisten kemana program anda disimpan.
   - ✓ Beri nama file :**Prak-11.cpp**
   - ✓ Compile dan perbaiki program sampai benar.

- ✓ Jika belum benar tanyakan pada asisten.

2. Jalankan program.
3. Keluar dari lingkungan Editor C++.
4. Aktifkan kembali Compiler C++ dan buka kembali program yang anda simpan tadi.
5. Bentuk file executable (berekstensi .EXE) dan jalankan dari lingkungan Windows tanpa menggunakan Editor C++.

PASCAL :

```
(* Program pertamaku *)
Program pertamaku;
Uses crt;
Const sapa1='Selamat datang di Laboratorium Pemrograman Dasar ';
Const sapa2='Sistem Komputer FTI Universitas Andalas";
Const nama='……..'; (* tuliskan nama anda *)
Const alamat='……..'; (*tuliskan alamat anda *)
Const sekolah='……..'; (*tuliskan sekolah asal anda *)

Begin
        Clrscr();
        Writeln(sapa1);
        Writeln(sapa2);
        Writeln;
        Writeln('Nama anda adalah ',nama);
        Writeln;
        Writeln('Anda beralamat di ',alamat);
        Writeln;
        Writeln('Sekolah Asal Anda ',sekolah);
        Writeln;
        Writeln('Selamat menggunakan Turbo Pascal 6.0, Terima Kasih');
        Repeat until keypressed;
End.
```

6. Simpan program
   - ✓ Tanyakan kepada asisten kemana program anda disimpan.
   - ✓ Beri nama file :**Prak-11.pas**
   - ✓ Compile dan perbaiki program sampai benar.
   - ✓ Jika belum benar tanyakan pada asisten.
7. Jalankan program.
8. Bentuk file executable (berekstensi .EXE), tanyakan caranya sama asisten.
9. Keluar dari lingkungan Turbo Pascal.
10. Jalankan program dari lingkungan Windows tanpa menggunakan Turbo Pascal.
11. Aktifkan kembali Turbo Pascal dan buka kembali program yang anda simpan tadi.

KASUS

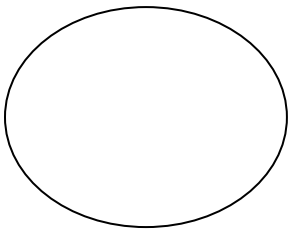
(2). Diketahui variabel A=10 dan B=50. Tukarkan letak dari kedua nilai variabel tersebut sehingga sewaktu dicetak nilai A=50 dan B=10. Buatlah programnya **(Prak-12.cpp** dan **Prak-12.pas)**

(3). Buat program untuk menghitung luas bangun dan ruang bserikut ini :

a. Luas Persegi Panjang (**Prak-13a.cpp** dan **Prak-13a.pas**)

Luas = panjang x lebar

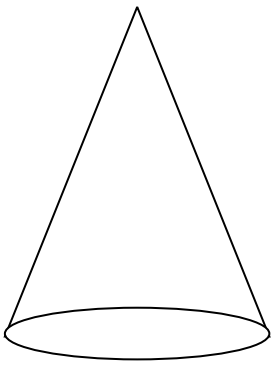
b. Luas Lingkaran**(Prak-13b.cpp** dan **Prak-13b.pas)**



Luas = Phi x r x r

Keliling = Phi x D atau 2 x Phi x r

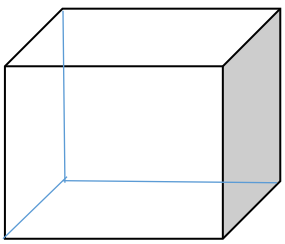
c. Volume Kerucut **(Prak-13c.cpp** dan **Prak-13c.pas)**



Volume = Phi/3 x r x r x Tinggi

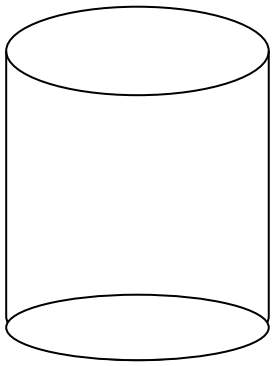
Luas Alas = π x r x r

d. Volume Kubus **(Prak-13d.cpp** dan **Prak-13d.pas)**



Volume = rusuk x rusuk x rusuk

e. Volume Tabung **(Prak-13e.cpp** dan **Prak-13e.pas)**



Volume = Phi x r x r x t

*** selamat mencoba ***

PRAKTIKUM 2

Pemilihan Kondisi

Tujuan Praktikum :

1. Memahami lebih lanjut struktur bahasa pemrograman C++ dan Pascal
2. Mempraktekkan statement menerima input data (cin pada C++) dan (readln pada Pascal)
3. Mempraktekkan struktur dasar algoritma pemilihan kondisi :
   - IF .. THEN
   - IF .. THEN ELSE ..
   - IF .. THEN .. ELSEIF
   - CASE .. OF ..
   - SWITCH .. CASE ..

Petunjuk : Mulailah pekerjaan anda dengan niat yang ikhlas sesuai dengan agama dan kepercayaan masing-masing.

KASUS :

(1). Diinputkan sebuah bilangan sembarang dari keyboard, tentukanlah apakah bilangan tersebut GENAP atau GANJIL !. Buat program C++ dan Pascalnya **(Prak-21.cpp** dan **Prak-21.pas)**

(2). Diinputkan dua buah bilangan sembarang dari keyboard, tentukanlah bilangan terbesar dari kedua bilangan-bilangan tersebut.Buat program C++ dan Pascalnya **(Prak-22.cpp** dan **Prak-22.pas)**

(3). Diinputkan tiga buah bilangan sembarang dari keyboard, tentukanlah bilangan terbesar dari ketiga bilangan-bilangan tersebut. Buat program C++ dan Pascalnya **(Prak-23.cpp** dan **Prak-23.pas)**

(4). Seorang mahasiswa telah melaksanakan ujian, nilai yang diperoleh adalah UTS dan UAS. Untuk mendapat Nilai akhir (NA) ditentukan rumus : NA = 40%*UTS + 60%*UAS.

Kemudian tentukan Nilai Huruf (NH) dengan ketentuan :

NA	NH
$0 \leq NA \leq 40$	E
$41 \leq NA \leq 55$	D
$56 \leq NA \leq 65$	C
$66 \leq NA \leq 79$	B
$80 \leq NA \leq 100$	A

Buat program C++ dan Pascalnya **(Prak24.cpp** dan **Prak24.pas)**

(5). Sebuah perusahaan PT. ABC mempekerjakan karyawan mingguan, jam kerja wajib setiap minggu adalah 48 jam dan dibayarkan gaji pokoknya berdasarkan golongan. Apabila karyawan bekerja melebihi jam kerja wajib, maka kelebihan tersebut dianggap lembur dan dibayarkan Rp. 20,000 per jam. Besar gaji pokok ditentukan oleh golongan, yakni :

- Jika gol = 1 maka Gaji Pokok adalah 100,000
- Jika gol = 2 maka Gaji Pokok adalah 150,000
- Jika gol = 3 maka Gaji Pokok adalah 200,000

Tentukanlah Lembur dan Total Gaji yang akan diterima oleh karyawan tersebut !. Buat program C++ dan Pascalnya **(Prak25.cpp** dan **Prak25.pas)**

(6). Buat program C++ dan Pascal (menggunakan SWITCH ... CASE) untuk menentukan nama hari dari kode hari yang diinputkan dari keyboard. **(Prak26.cpp** dan **Prak26.pas)**

*** selamat bekerja ***

PRAKTIKUM 3

Perulangan For dan While

Tujuan Praktikum :

1. Memperkenalkan kepada mahasiswa struktur kontrol perulangan/looping
2. Memahami dan mengerti penggunaan struktur kontrol perulangan FOR ..., WHILE
3. Dapat menerapkan struktur kontrol perulangan for ..., while ... untuk menyelesaikan masalah dalam kehidupan sehari-hari

Petunjuk : Mulailah pekerjaan anda dengan niat yang ikhlas sesuai dengan agama dan kepercayaan masing-masing.

KASUS :

(1). Buat program C++ dan Pascal untuk menampilkan deret bilangan berikut :

a. 1 4 9 16 25 36 49**(Prak31a.cpp** dan **Prak31a.pas)**

b. 1 2 4 7 11 16 22 29**(Prak31b.cpp** dan **Prak 31b.pas)**

c. 50 45 40 35 30 25 20 15 10 5 0 **(Prak31c.cpp** dan **Prak31c.pas)**

(2). Buat program C++ dan Pascal untuk menampilkan bilangan dengan layout :

```
      X       Y       XY
 -----------------------------
      0       5       ...
      1       10      ...
 2    15      ...
 3    20      ...
      ..      ..        ...
     10       50       ,,,
 -----------------------------
```

(Prak32.cpp dan **Prak32.pas)**

(3). Diinputkan 10 buah bilangan bulat sembarang dari keyboard, tentukan berapa buah bilangan yang GENAP-GENAP saja !. Buat program C++ dan Pascalnya. **(Prak33.cpp** dan **Prak33.pas)**

(4). Suatu kelas memiliki 40 orang mahasiswa yang berumur 18 th, 19 th, 20 th, 21 thn dan umur lainnya. Buatlah program C++ dan Pascal untuk menghitung dan menampilkan rekapitulasi mahasiswa dengan umur-umur tersebut !. **(Prak34.cpp** dan **Prak34.pas)**

*** selamat bekerja ***

PRAKTIKUM 4

Nested Loop

Tujuan Praktikum :

1. Memperkenalkan kepada mahasiswa tentang Nested Loop *(Loop Bersarang)*
2. Dapat menerapkan Neted Loop untuk menyelesaikan masalah sehari-hari

Petunjuk : Mulailah pekerjaan anda dengan niat yang ikhlas sesuai dengan agama dan kepercayaan masing-masing.

KASUS :
Buat program C++ dan Pascal untuk mencetak tampilan berikut :

(1).
```
   1  1  1  1
   2  2  2  2
3  3  3  3
   4  4  4  4
```
(Prak41.cpp dan **Prak41.pas)**

(2).
```
2  4  6
8  10 12
14 16 18
20 22 24
```
(Prak42.cpp dan **Prak42.pas)**

(3).
```
1 0 0 0
1  1  0  0
1  1  1 0
1 1  11
```
(Prak43.cpp dan **Prak43.pas)**

(4).
```
   1
2  2
  3  3  3
  4  4  4  4
  5  5  5  5  5
  6  6  6  6  6  6
```
(Prak44.cpp dan **Prak45.pas)**

(5).
```
   7  7  7  7  7  7  7
      6  6  6  6  6  6
 5  5  5  5  5
  4  4  4  4
  3  3  3
  2  2
```
(Prak44.cpp dan **Prak45.pas)**

1***** selamat bekerja *****

PRAKTIKUM 5

Larik/Array

Tujuan Praktikum :

1. Memperkenalkan kepada mahasiswa penggunaan larik Dimensi Satu dan Dua
2. Mempraktekkan pemakaian larik dimensi satu untuk elemen yang diinputkan
3. Mempraktekkan pemakaian larik dimensi satu untuk elemen yang didefinsikan
4. Menugaskan mahasiswa untuk membuat program menggunakan larik sebagai pemecahan masalah dalam kehidupan sehari-hari

Petunjuk : Mulailah pekerjaan anda dengan niat yang ikhlas sesuai dengan agama dan kepercayaan masing-masing.

KASUS :

(1). Buatlah program C++ dan Pascal menggunakan Larik Dimensi Satu untuk menyelesaikan permasalahan-permasalahan berikut :
   a. Mencetak 10 elemen larik A yang elemennya didefinisikan didalam program. **Prak51a.cpp** dan **Prak51a.pas)**
   b. Mencetak 10 elemen larik B yang elemennya diinputkan dari keyboard. (**Prak51b.cpp** dan **Prak51b.pas)**
   c. Mencetak elemen terkecil dan terbesar dari data yang ada dalam larik C. (**Prak51c.cpp** dan **Prak51c.pas)**

(2). Buatlah program C++ dan Pascal menggunakan Larik Dimensi Dua untuk menyelesaikan permasalah-permasalahan berikut :
   a. Menginputkan dan mencetak Matriks A dengan ordo 3x4. **(Prak52a.cpp** dan **Prak52a.pas)**
   b. Mennjumlahkan Matrik A dan Matriks B dan kemudian hasilnya simpan ke dalam Matriks C. (**Prak52b.cpp** dan **Prak52b.pas)**

*** selamat bekerja ***

PRAKTIKUM 6

Sorting dan Searching

Tujuan Praktikum :

1. Memperkenalkan kepada mahasiswa metoda pengurutan *(sorting)*
2. Memperkenalkan kepada mahasiswa metoda pencarian *(searching)*
3. Mempraktekkan pemakaian metoda *bubblesort* dan *selection sort*
4. Mempraktekkan pemakaian metoda *sequential search* dan *binary search*

Petunjuk : Mulailah pekerjaan anda dengan niat yang ikhlas sesuai dengan agama dan kepercayaan masing-masing.

KASUS :

(1). Diketahui sebuah lariks A dengan elemen {10, 2, 100, 50, 35, 25, 70, 65, 85}. Buatlah program C++ dan Pascal untuk menyelesaiakan masalaha-masalah berikut :
   a. Mengurutkan elemen data secara Ascending menggunakan metoda Bubble Sort. **(Prak61a.cpp** dan **Prak61a.pas)**
   b. Mengurutkan elemen data secara Descending menggunakan metoda Selection Sort. **(Prak61b.cpp** dan **Prak61b.pas)**

(2). Diketahui sebuah lariks B dengan elemen {3, 25, 75, 5, 10, 1, 18, 50, 35, 25}. Buatlah program C++ dan Pascal untuk mencari data tertentu menggunakan metoda Sequential Search. **(Prak62.cpp** dan **Prak62.pas)**

(3). Diketahui sebuah lariks B dengan elemen {1, 3, 5, 10, 18, 25, 35, 75, 85, 90, 105}. Buatlah program C++ dan Pascal untuk mencari data tertentu menggunakan metoda Binary Search. **(Prak63.cpp** dan **Prak63.pas)**

*** selamat bekerja ***

PRAKTIKUM 7

Fungsi dan Prosedur

Tujuan Praktikum :

1. Memperkenalkan kepada mahasiswa konsep pemrograman modular
2. Memperkenalkan kepada mahasiswa pemakaian fungsi/function
3. Memperkenalkan kepada mahasiswa pemakaian prosedur
4. Mempraktekkan pemakaian fungsi
5. Mempraktekkan pemakaian prosedur

Petunjuk : Mulailah pekerjaan anda dengan niat yang ikhlas sesuai dengan agama dan kepercayaan masing-masing.

KASUS :

(1). Buatlah program C++ dan Pascal menggunakan fungsi untuk menyelesaikan masalah-masalah berikut:

a. Diinputkan sebuah bilangan dari keyboard, apakah bilangan tersebut GENAP atau GANJIL. **(Prak71a.cpp** dan **Prak71a.pas)**

b. Diinputkan dua buah bilangan dari dari keyboard, tentukan bilangan terbesar dari kedua bilangan tersebut. **(Prak71b.cpp** dan **Prak71b.pas)**

c.Menghitung deret S = 1+2+3+4+5+...+n. **(Prak71c.cpp** dan **Prak71c.pas)**

d. Menghitung nilai fungsi $f(x)=2x^2 + 5x - 8$ dan menampilkannya dalam bentuk tabel yang berisi nilai-nilai x dan F(x) di dalam selang [10,15] dengan delta x = 0.2. **(Prak71d.cpp** dan **Prak71d.pas).**

(2). Buatlah program C++ dan Pascal menggunakan prosedur untuk menyelesaikan masalah-masalah berikut:

a. Diinputkan dua buah bilangan A dan B dari dari keyboard, tukarkan letak dari kedua bilangan tersebut. **(Prak72a.cpp** dan **Prak72a.pas)**

b. Menentukan luas segitiga dari alas dan tinggi yang diinputkan dari keyboard. **(Prak72b.cpp** dan **Prak72b.pas)**

c. Menghitung faktorial dari suatu N yang diinputkan dari keyboard.**(Prak72c.cpp** dan **Prak72c.pas).**

*** selamat bekerja ***

PRAKTIKUM 8

Pointer

Tujuan Praktikum :

1. Memperkenalkan kepada mahasiswa tentang konsep pointers
2. Mempraktekkan pemakaian pointer

Apa itu Pointers?

Pointer adalah variable yang berisi alamat memori sebagai nilainya dan berbedadengan variabel biasa yang berisi nilai tertentu. Dengan kata lain, pointer berisialamat dari variabel yang mempunyai nilai tertentu.

Adapun bentuk umum dari pernyataan variable pointer dalam C adalah :

```
type *var-name;
```

Dengan :

- type adalah tipe dasar pointer.
- variable_name adalah nama variable pointer.
- * adalah operator memori yang fungsinya untuk mengembalikan nilai variabel pada alamatnya yang ditentukan oleh operand.

Contoh deklarasi pointer yang valid :

```
int *ip;    /* pointer to an integer */
double *dp; /* pointer to a double */
float *fp;  /* pointer to a float */
char *ch    /* pointer to a character */
```

Tipe data sebenarnya dari nilai semua pointer baik integer, float char atau yanglainnya, adalah sama, yaitu serangkaian bilangan hexadecimal yangmerepresentasikan sebuah alamat memory. Perbedaannya pointer dari tipe data yangberbeda hanya terletak pada perbedaan tipe data variable atau konstan yang ditunjukoleh pointer tersebut.

Contoh Program

Program-1 :

```
/* Source code to demonstrate, handling of pointers in C program */
#include <iostream>
#include <conio.h>
```

```
using namespace std;

int main(){
int *pc,c;
c=22;
cout<<"Alamat variabel c : "<<&c;cout<<"\n";
cout<<"Nilai variabel c : "<<c;cout<<"\n\n";
pc=&c;
cout<<"Alamat dari pointer pc : "<<pc;cout<<"\n";
cout<<"Nilai dari pointer pc : "<<*pc;cout<<"\n\n";
c=11;
cout<<"Alamat dari pointer pc: "<<pc;cout<<"\n";
cout<<"Nilai dari pointer pc: "<<*pc;cout<<"\n\n";
*pc=2;
cout<<"Alamat dari variabel c: "<<&c;cout<<"\n";
cout<<"Nilai dari variabel c: "<<c;cout<<"\n\n";
getch();
}
```

Simpan dengan nama file : **Pointer-1.cpp**

Program-2 :

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

using namespace std;

int main ()
{
int var = 20;
int *ip;
ip = &var;
cout<<"Alamat variabel var : "<<&var;cout<<"\n";
cout<<"Alamat penyimpanan variabel ip di : "<<ip;cout<<"\n";
cout<<"Nilai dari variebel ip : "<<*ip;cout<<"\n";
getch();
}
```

Simpan dengan nama file : **Pointer-2.cpp**

Pointers and Arrays

Array sangat erat kaitannya dengan pointer. Array dan pointer merupakan sinonimdalam hal bagaimana keduanya dipakai untuk mengakses memory. Akan tetapi,perbedaan penting diantara keduanya adalah variable pointer dapat menerima alamatyang berbeda sebagai nilai, sedangkan pada array bersifat tetap. Perhatikan contohberikut :

Program-3 :

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

using namespace std;
int main()
{
char c[4];
int i;
for(i=0;i<4;++i)
{
printf("Address of c[%d]=%x\n",i,&c[i]);
}
  getch();
}
```

Pointers pada Struct

Pointer dapat digunakan bersamaan dengan structures. Sebuah variable pointer darisebuah struktur dapat diciptakan dengan cara berikut :

```
struct name {
member1;
member2;
.
.
};
-------- dalam function -------
struct name *ptr;
```

Pada potongan program diatas, variable pointer bertipe **struct name** telahdiciptakan.

Contoh penggunaan

```
#include <iostream>
#include <conio.h>
using namespace std;
typedef struct
{
int nim;
int umur;
float ipk;
```

```
} Mahasiswa;
Mahasiswa m;
Mahasiswa *p = &m;

int main()
{
//struct biasa
    m.nim=123;
    m.ipk=3.2;
    m.umur=23;
    cout<<"\nnim = "<<m.nim;
    cout<<"\nipk = "<<m.ipk;
    cout<<"\numur ="<<m.umur;
//struct pointer
    p->ipk = 3.5;
    p->nim = 321;
    p->umur = 32;
    cout<<"\n\nnim = "<<p->nim;
    cout<<"\nipk = “<<p->ipk;
    cout<<"\numur = “<<p->umur;
//mengacu pada variabel aslinya
    cout<<"\n\nnim = “<<m.nim;
    cout<<"\nipk = “<<m.ipk;
    cout<<"\numur = “<<m.umur;
    getch();
}
```

KASUS :

Buatlah program Dev C++ untuk :

(1). Merubah huruf besar ke kecil dengan menggunakanpointer!**(Prak81.cpp)**

(2). Menampilkan kalimat terbalik dari suatu kalimat yangdiinputkan menggunakan pointer dan array !**(Prak82.cpp)**

(3). Mengecek apakah suatu kata palindrom ataubukan, tanpa memperhatikan spasi dan huruf besar/kecilnya denganmenggunakan struct dan pointer.**(Prak83.cpp)**

*** selamat bekerja

PRAKTIKUM 9

Linked List

Tujuan Praktikum :

1. Memahami konsep linked list dan mengerti kegunaannya
2. Memahami instruksi-instruksi dan cara pembuatan program yang dapatdigunakan untuk menerapkan linked list.
3. Mampu mempraktekkan linked list untuk menyelesaikan masalah

Linked List

Linked list adalah sebuah struktur data kompleks yang sangat bermanfaat dalamsistem dan applications programming. Sebuah linked list terdiri atas serangkaiannodes, dimana setiap node berisi sebuah elemen data, serta sebuah pointer ke nodeberikutnya.

Struktur yang berisi sebuah elemen data dan sebuah pointer ke node berikutnyadiciptakan dengan cara :

```
struct list {
int value;
struct list *next;
};
```

Statement diatas menciptakan sebuah struktur data yang disebut dengan list yangberisi dua members (anggota). Anggota pertama adalah sebuah integer yangbernama value. Anggota kedua bernama next, yang merupakan pointer ke strukturlist lainnya atau node.

Dua struktur yang memiliki tipe yang sama yaitu list dapat dideklarasikan sebagai berikut :

```
struct list n1, n2;
```

Pointer next dari n1 dapat di-set untuk menunjuk ke n2 dengan cara :

```
n1.next = &n2;
```

Linked List terbagi 2 :

1. Singly Linked List
2. Doubly Linked List

Singly Linked List

Program-1 :

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

using namespace std;
struct list
{
   int value;
   struct list *next;
};

main()
{
    struct list n1, n2, n3;
int i;
n1.value = 100;
n2.value = 200;
n3.value = 300;
n1.next = &n2;
n2.next = &n3;
i = n1.next->value;
cout<<i; cout<<"\n";
 cout<<n1.next->value; cout<<"\n";
cout<<n2.next->value;
}
```

Simpan dengan nama file : **Linked-list1.cpp**

Program-2 :

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

using namespace std;

struct list {
  int value;
  struct list *next;
};
```

```
main()
{
  struct list n1, n2, n3, n4;
  struct list *list_pointer = &n1;
  n1.value = 100;
  n1.next = &n2;
  n2.value = 200;
  n2.next = &n3;
  n3.value = 300;
  n3.next = &n4;
  n4.value = 400;
  n4.next = 0;

while( list_pointer != 0 )
  {
    cout<<"\n"<<list_pointer->value;
    list_pointer = list_pointer->next;
  }
getch();
}
```

Simpan dengan nama file : **Linked-list2.cpp**

KASUS :

Buatlah program Dev C++ untuk melakukan hal-hal berikut :

(1). Memasukkan beberapa data dalam sebuah linked list,jika akan mengakhiri, tekan n maka akan tampil node yang sudah dimasukkanke dalam linked list tersebut. **(Prak91.cpp)**

(2). Mengimplementasikan beberapa operasi pada linkedlist berikut :**(Prak92.cpp)**

a. Operasi penambahan pada linked list (buatlah sebuah link list yang menyimpan karakter/huruf nama lengkapanda).

b. Operasi penghapusan pada linked list (hapuslah huruf ke 3,5,6 dari data nama anda yang ada pada linked listtersebut).

c. Operasi penghapusan pada linked list (tampilkan rangkaian karakter penyusun nama yang masih tersisa).

*** selamat bekerja ***

PRAKTIKUM 10

Stack

Tujuan Praktikum :

1. Memahami konsep stack dan mengetahui kegunaannya.
2. Mengimplementasikan struktur data stack dalam bahasa pemrograman Dev C++
3. Mengidentifikasi permasalahan pemrograman yang cocok diselesaikan dengan stack dan mampu menyelesaikannya

Stack adalah struktur data yang umum digunakan untuk merepresentasikan sesuatuyang perlu diorganisasikan dengan urutan tertentu. Misalnya, ketika sebuah fungsimemanggil fungsi lain, dimana fungsi lain tersebut mamanggil fungsi berikutnya.Sangatlah penting untuk memastikan bahwa setelah fungsi ketiga selesai, fungsitersebut kembali ke fungsi kedua, bukan ke fungsi pertama.

Secara konsep, sebuah stack bersifat sederhana, dimana stack merupakan strukturdata yang mengizinkan untuk menambah dan mengurangi elemen dengan urutan yangsudah ditentukan. Setiap kali sebuah elemen ditambahkan, maka elemen tersebutakan berada pada top dari stack tersebut. Elemen juga hanya bisa dihapus jika elementersebut berada pada posisi top. Konsekuensinya, stack disebut memiliki prilaku “lastin first out”.

Beberapa terminologi dasar stack “
• Push untuk menambahkan sebuah elemen ke stack.
• Pop untuk menghapus sebuah elemen dari stack.
• Peek untuk memeriksa elemen dalam stack tanpa menghapusnya.

Stack dapat diimplementasikan dengan dua cara :
1. Array based implementation
2. Linked list based implementation

Contoh Program

Program-1 :

```
/* Program of stack using array*/
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
#define MAX 5

int top = -1;
int stack_arr[MAX];
```

```
void push();
void pop();
void display();
int main()
{
  int choice;
while(1)
  {
    printf("\n1.Push\n");
    printf("2.Pop\n");
    printf("3.Display\n");
    printf("4.Quit\n\n");
printf("Enter your choice : ");
    scanf("%d",&choice);
    switch(choice)
    {
      case 1 :
push();
       break;
      case 2:
pop();
       break;
      case 3:
display();
       break;
      case 4:
exit(1);
      default:
printf("\nPILIHAN TIDAK TERSEDIA\n");
   }
  }
getch();
}

void push()
{
  int pushed_item;
if(top == (MAX - 1))
printf("Stack Overflow\n");
  else
  {
printf("\nEnter the item to be pushed in stack : ");
    scanf("%d",&pushed_item);
    top = top + 1;
    stack_arr[top] = pushed_item;
  }
```

```
}

void pop()
{
if(top == -1)
printf("Stack Underflow\n");
else{
printf("Popped element is : %d\n",stack_arr[top]);
  top = top - 1;
  }
}

void display()
{
  int i;
if(top == -1)
printf("Stack is empty\n");
  else {
printf("\nStack elements :\n");
for(i = top; i >=0; i--)
printf("%d\n", stack_arr[i] );
     }
}
```

Simpan dengan nama file : **Stack-1.cpp**

Program-2

```
#include<stdio.h>
#include<stdlib.h>
#include <conio.h>
typedef struct node
{
   int data;
   struct node *link;
}NODE;
NODE *top = NULL;
void push();
void pop();
void display();

int main()
{
    int choice = 0;
   while(1)
```

```
 {
    printf("\n1.Push\n");
     printf("2.Pop\n");
     printf("3.Display\n");
     printf("4.Quit\n\n");
     printf("Enter your choice : ");
     scanf("%d",&choice);

    switch(choice)
    {
     case 1 :
        push();
        break;
    case 2:
      pop();
      break;
    case 3:
      display();
      break;
    case 4:
       exit(0);
    default:
      printf("\nPILIHAN TIDAK TERSEDIA\n");
    }
 }
 getch();
}

void push()
{
  NODE *temp;
   int info = 0;
   printf("Enter data to be pushed (0-9999) : ");
   scanf("%d",&info);
   temp = (NODE *) malloc(sizeof(NODE));
   if (temp == NULL)
        printf("\nMemory Allocation Failed");
    else {
        temp->data = info;
        temp->link = top;
        top = temp;
        printf(" Node has been inserted at Top(Front) Successfully !!\n");
        }
 }

void pop()
```

```
{
  NODE *tmp;
  if(top == NULL)
    printf("Stack is empty!!\n");
 else {
    tmp = top;
    printf("Popped data : %d\n",tmp->data);
    top = top->link;
    free(tmp);
  }
}

void display()
{
   NODE *temp;
   if(top == NULL)
   {
     printf("Empty Stack !!!\n");
  }
  else
  {
     temp = top;
     printf("Stack :- \n");
     while(temp != NULL)
    {
       printf("\n ____\n");
       printf("|%4d|",temp->data);
       temp = temp->link;
    }
  }
}
```

Simpan dengan nama file : **Stack-2.cpp**

PRAKTIKUM 11

Queue

Tujuan Praktikum :

1. Memahami konsep queue dan mengetahui kegunaannya.
2. Mengimplementasikan struktur data queue menggunakan array.
3. Mengimplementasikan struktur data queue menggunakan linked list.
4. Mengidentifikasi permasalahan pemrograman yang cocok diselesaikan dengan queue dan memanfaatkan queue untuk menyelesaikan masalah tersebut

Queue
adalah bentuk khusus dari struktur data abstrak dimana elemen-elemendisimpan secara berurutan. Operasi dasar pada struktur data ini adalahpenambahan entity pada bagian ekor (rear) yang disebut dengan enqueue danpenghapusan elemen pada bagian kepala (head), disebut dengan dequeue. Modeloperasi ini membuat struktur data queue bersifat FIFO (First In First Out).

Queue dapat diimplementasikan dengan 2 cara :
1. Array based implementation of queue
2. Linked list implementation of queue

Contoh program

Program-1:

```
/*Program of queue using array*/
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
#define MAX 5
int rear = -1;
int front = -1;
int queue_arr[MAX];

void enqueue();
void dequeue();
void display();

main()
```

```
{
  int choice;
  while(1)
 {
   printf("1.Insert\n");
   printf("2.Delete\n");
   printf("3.Display\n");
   printf("4.Quit\n");
   printf("Enter your choice : ");
   scanf("%d",&choice);
   switch(choice)
  {
      case 1 :
         enqueue();
         break;
      case 2 :
         dequeue();
         break;
      case 3:
         display();
         break;
      case 4:
         exit(1);
      default:
         printf("PILIHAN TIDAK TERSEDIA\n");
   }
 }
 getch();
}

void enqueue()
{
  int added_item;
  if (rear == MAX - 1)
    printf("Queue Overflow\n");
  else
  {
   if (front == -1)
     front = 0;
    printf("Input the element for adding in queue : ");
    scanf("%d", &added_item);
    rear = rear + 1;
    queue_arr[rear] = added_item ;
    }
}
```

```
void dequeue()
{
  if (front == -1 || front > rear)
 {
    printf("Queue Underflow\n");
    return ;
 }
 else
 {
   printf("Element deleted from queue is : %d\n",
   queue_arr[front]);
   front = front + 1;
  }
}

void display()
{
   int i;
   if (front == -1)
     printf("Queue is empty\n");
  else
 {
    printf("Queue is :\n");
    for(i = front; i <= rear; i++)
       printf("%d ",queue_arr[i]);
      printf("\n");
  }
}
```

Simpan dengan nama file : **Queue-1.cpp**

Program-2 :

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

typedef struct node
{
int data;
struct node *link;
} NODE;
NODE *front, *rear = NULL;

void enqueue();
void dequeue();
```

```
void display();

int main()
{
int choice = 0;
while(1)
{
printf("\n1.Push\n");
printf("2.Pop\n");
printf("3.Display\n");
printf("4.Quit\n\n");
printf("Enter your choice : ");
scanf("%d",&choice);
switch(choice)
{
case 1 :
enqueue();
break;
case 2:
dequeue();
break;
case 3:
display();
break;
case 4:
exit(0);
default:
printf("\nPILIHAN TIDAK TERSEDIA\n");
}
}
getch();
}

void enqueue()
{
NODE *temp;
int info = 0;
printf("Enter data to be enqueued (0-9999) : ");
scanf("%d",&info);
temp = (NODE *) malloc(sizeof(NODE));
if (temp == NULL)
printf("\nMemory Allocation Failed");
else
{
temp->data = info;
temp->link = NULL;
```

```
if (front == NULL)
{
front = rear = temp;
}
else
        {
rear->link = temp;
rear = temp;
}
printf(" Node has been inserted at End Successfully !!");
}
}

void dequeue()
{
NODE *temp;
int info;
if (front == NULL)
{
printf(" Underflow!!!");
}
else
{
temp = front;
info = front->data;
if (front == rear)
{
rear = NULL;
}

front = front->link;
temp->link = NULL;
printf(" Deleted Node(From Front)with the Data: %d\n", info);
free(temp);
}
}

void display()
{
NODE *temp;
if (front == NULL)
printf("Empty Queue\n");
else
{
temp = front;
printf("Front->");
```

```
while (temp)
{
printf("[%d]->", temp->data);
temp = temp->link;
}
printf("Rear\n");
}
}
```

KASUS :

Lengkapi menu dan implementasi program-2 diatas dengan 3 hal berikut :
1. Check if empty**(Queue111.cpp)**
2. Get the first element of the queue**(Queue112.cpp)**
3. Get the number of entries in the queue**(Queue113.cpp)**

PRAKTIKUM 12

Tree

Tujuan Praktikum :

1. Memahami konsep tree dan mengetahui kegunaannya.
2. Mengimplementasikan struktur data tree menggunakan linked list.
3. Mengidentifikasi permasalahan pemrograman yang cocok diselesaikan dengan tree dan memanfaatkan tree untuk menyelesaikan masalah tersebut.

Tree
adalah kumpulan node yang saling terhubung satu sama lain dalam suatu kesatuanyang membentuk layakya struktur sebuah pohon.Struktur pohon adalah suatu cara merepresentasikan suatu struktur hirarki(one-to-many) secara grafis yang mirip sebuah pohon, walaupun pohontersebut hanya tampak sebagai kumpulan node-node dari atas ke bawah.

Suatu struktur data yang tidak linier yang menggambarkan hubungan yang hirarkis (one-to-many) dan tidak linier antara elemen-elemennya.

Operasi-operasi Pada Tree

1.Insert: menambah node ke dalam Tree secara rekursif. Jika data yang akandimasukkan lebih besar daripada elemen root, maka akan diletakkan di nodesebelah kanan, sebaliknya jika lebih kecil maka akan diletakkan di node sebelahkiri. Untuk data pertama akan menjadi elemen root.

2.Find: mencari node di dalam Tree secara rekursif sampai node tersebutditemukan dengan menggunakan variable bantuan ketemu. Syaratnya adalahtree tidak boleh kosong.

3.Traverse: yaitu operasi kunjungan terhadap node-node dalam pohon dimanamasing-masing node akan dikunjungi sekali.

4.Count: menghitung jumlah node dalam Tree.
5.Height : mengetahui kedalaman sebuah Tree.
6. Find Min dan Find Max : mencari nilai terkecil dan terbesar pada Tree.
7. Child : mengetahui anak dari sebuah node (jika punya).

Jenis Traverse
a.PreOrder: cetak node yang dikunjungi, kunjungi left, kunjungi right.
b.InOrder: kunjungi left, cetak node yang dikunjungi, kunjungi right.
c.PostOrder: kunjungi left, kunjungi right, cetak node yang dikunjungi.

Contoh Program

Program-1 :

```
#include <stdlib.h>
#include <stdio.h>
#include <conio.h>
typedef struct tnode
{
int data;
struct tnode *right, *left;
} TNODE;

TNODE *CreateBST(TNODE *, int);
void Inorder(TNODE *);
void Preorder(TNODE *);
void Postorder(TNODE *);

main()
{
TNODE *root = NULL;
int opn, elem, n, i;
do
{
//clrscr();
printf("\n ### Binary Search Tree Operations ### \n\n");
printf("\n Press 1-Creation of BST");
printf("\n 2-Traverse in Inorder");
printf("\n 3-Traverse in Preorder");
printf("\n 4-Traverse in Postorder");
printf("\n 5-Exit\n");
printf("\n Your option ? ");
scanf("%d", &opn);
switch (opn)
{
case 1:
root = NULL;
printf("\n\nBST for How Many Nodes ?");
scanf("%d", &n);
for (i = 1; i <= n; i++)
{
printf("\nRead the Data for Node %d ?", i);
scanf("%d", &elem);
root = CreateBST(root, elem);
}
printf("\nBST with %d nodes is ready to Use!!\n", n);
break;
```

```
case 2:
printf("\n BST Traversal in INORDER \n");
Inorder(root);
break;
case 3:
printf("\n BST Traversal in PREORDER \n");
Preorder(root);
break;
case 4:
printf("\n BST Traversal in POSTORDER \n");
Postorder(root);
break;
case 5:
printf("\n\n Terminating \n\n");
break;
default:
printf("\n\nInvalid Option !!! Try Again !! \n\n");
break;
}
printf("\n\n\n\n Press a Key to Continue . . . ");
getch();
}
while (opn != 5);
}

TNODE *CreateBST(TNODE *root, int elem)
{
if (root == NULL)
{
root = (TNODE *) malloc(sizeof(TNODE));
root->left = root->right = NULL;
root->data = elem;
return root;
}
else
{
if (elem < root->data)
root->left = CreateBST(root->left, elem);
else if (elem > root->data)
root->right = CreateBST(root->right, elem);
else
printf(" Duplicate Element !! Not Allowed !!!");
return (root);
}
}
```

```
void Inorder(TNODE *root)
{
if (root != NULL)
{
Inorder(root->left);
printf(" %d ", root->data);
Inorder(root->right);
}
}

void Preorder(TNODE *root)
{
if (root != NULL)
{
printf(" %d ", root->data);
Preorder(root->left);
Preorder(root->right);
}
}

void Postorder(TNODE *root)
{
if (root != NULL)
{
Postorder(root->left);
Postorder(root->right);
printf(" %d ", root->data);

}
}
```

KASUS :

Lengkapi contoh program tersebut dengan menyediakan menu untuk : **(Tree12.cpp)**
a. Count: menghitung jumlah node dalam Tree.
b.Height : mengetahui kedalaman sebuah Tree.
c.Find Min dan Find Max : mencari nilai terkecil dan terbesar pada Tree.
d.Find: mencari node di dalam Tree.

Sumber/Referensi:

1) *Algorithm + Data Structures = Programs* by Wirth, Niklaus, New Jersey: Prentice-Hall.
2) *Fundamentals of Data Structures in C* by Ellis Horowitz, Sartaj Sahni and Susan Anderson-Freed; W. H. Freeman, 1992
3) *Struktur Data Menggunakan Turbo Pascal 6.6,* Santoso, P. Insap, Andi Ofset Yogyakarta, 1993.
4) *Data Structures and Program Design in C, Second Edition* by Robert Kruse et al.; Prentice Hall, 1997
5) *Algorithms in C, Third Edition Parts 1 - 4* by Robert Sedgewick; Addison-Wesley, 1998
6) *Konsep dan Implementasi Struktur Data,* Zakaria, T.M., Prijono, A,Informatika Bandung, 2006
7) *Data Structures and Other Objects Using Java by* Main, M.. *Pearson.* Third Edition, 2006
8) *Konsep dan Implementasi Struktur Data*,Marcus, Zakaria, 2006. Bandung : Informatika.
9) *Modul Pembelajaran Struktur Data,* Rosa A.S,Penerbit Modula, Bandung (2010)
10) *Algoritma dan Pemrograman,* Suarga, Math M.,2010, Andi Yogyakarta