MODUL PRAKTIKUM

"BAHASA C++"

Untuk Mata Kuliah :

Solusi Problem Mengunakan Bahasa C

Pemograman Komputer

2020

LABORATORIUM KOMPUTER FAKULTAS TEKNIK

UNIVERSITAS PANCASILA

LEMBAR PENGESAHAN

MODUL PRAKTIKUM
Bahasa C++

Oleh:

Agung saputra ST.MT

Dengan ini telah diperiksa dan disetujui untuk digunakan pada praktikum Bahasa C++ di Laboratorium Komputer Fakultas Teknik Universitas Pancasila, dan dinyatakan berlaku mulai semester Genap Tahun Akademik 2019/2020.

Jakarta, 4 Maret 2020

Disahkan oleh	Diperiksa Oleh
Ketua Laboratorium Komputer	Koordinator Asisten Praktikum
Agung Saputra ST.MT	Veni Apriani

KATA PENGANTAR

Puji syukur ke hadirat Tuhan Yang Maha Kuasa, yang telah memberikan rahmat-Nya sehingga Modul Praktikum Bahasa C++ untuk mahasiswa/i Jurusan Teknik Industri, dan Teknik Elektro S1 Fakultas Teknik Universitas Pancasila ini dapat diselesaikan dengan sebaik-baiknya.

Modul praktikum ini dibuat sebagai pedoman dalam melakukan kegiatan praktikum untuk mata kuliah Solusi Problem Mengunakan Bahasa C mahasiswa/i Jurusan Teknik Elektro, Pemograman Komputer mahasiswa/i jurusan Teknik Industri yang merupakan kegiatan penunjang mata kuliah. Modul praktikum ini diharapkan dapat membantu mahasiswa/i dalam mempersiapkan dan melaksanakan praktikum dengan lebih baik, terarah, dan terencana. Pada setiap topik telah ditetapkan tujuan pelaksanaan praktikum dan semua kegiatan yang harus dilakukan oleh mahasiswa/i serta teori singkat untuk memperdalam pemahaman mahasiswa/i mengenai materi yang dibahas.

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

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

Jakarta, 4 Maret 2020

Tim Penyusun

DAFTAR ISI

PRAKTIKUM MINGGU 1

PENGENALAN PROGRAM C

STRUKTUR RUNTUNAN (SEQUENCE)
LANGKAH-LANGKAH PEMBUATAN PROGRAM

a. Untuk menulis program baru caranya tekan tombol ALT+ F pilih NEW terlebih dahulu sebelum Anda mengetik program Anda.
b. Ketikkan program Anda.
c. Simpan program Anda dengan cara tekan tombol F2 atau ALT + F pilih Save kemudian ketikkan nama program Anda, misalnya `Contoh1.c` <enter>
d. Jalankan program dengan menekan tombol ALT + R pilih RUN atau CTRL + F9
e. Bila ada kesalahan perbaiki kesalahan tersebut kemudian Anda ulangi langkah 3 dan 4
f. Untuk membuat program baru ulangi langkah 1.
g. Bila Anda ingin mencek apakah program Anda sudah tersimpan atau ingin menampilkan program Anda kembali ke layar, maka tekan tombol ALT + F pilih LOAD atau OPEN kemudian enter.
h. Jangan lupa, sebelumnya set isi menu OPTIONS > DIRECTORIES sesuai lokasi directory Turbo C dan file program Anda.

Selanjutnya Anda kerjakan contoh-contoh program dibawah ini beserta tugasnya.

1. BARIS KOMENTAR

Baris komentar adalah baris-baris yang menjelaskan maksud dari perubah yang digunakan atau maksud dari program itu sendiri. Hal ini dimaksudkan untuk memudahkan pelacakan atas perubah yang digunakan apabila program yang digunakan cukup besar atau memudahkan orang lain memahami program yang kita buat. Dalam program, baris komentar diletakkan diantara tanda /* dan */ dan baris ini tidak dikerjakan oleh komputer, hanya dianggap sebagai baris kosong.

2. BENTUK / STRUKTUR PROGRAM C

Bentuk program C mirip dengan kebanyakan program bahasa tingkat tinggi lainnya. Bentuk programnya adalah :

Judul Program
Daftar Header File
Deklarasi
Deskripsi

2.1 Judul Program

Judul program sifatnya sebagai dokumentasi saja, tidak signifikan terhadap proses program. Ditulis dalam bentuk baris komentar.

Contoh :

```
/* Program Menghitung Rata-Rata */
```

2.2 Header File

C menyediakan sejumlah file judul (header file) yaitu file yang umumnya berisi prototipe fungsi, definisi makro, variabel dan definisi tipe. File ini mempunyai ciri yaitu namanya diakhiri dengan extension .h.

Contoh :

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

Keterangan : menyatakan bahwa agar membaca file bernama stdio.h saat pelaksanaan kompilasi.

2.3 Deklarasi

Deklarasi adalah bagian untuk mendefinisikan semua nama yang dipakai dalam program. Nama tersebut dapat berupa nama tetapan (konstanta), nama variabel, nama tipe, nama prosedur, nama fungsi.

2.4 Deskripsi

Bagian inti dari suatu program yang berisi uraian langkah-langkah penyelesaian masalah. Program C pada hakekatnya tersusun atas sejumlah blok fungsi. Sebuah program minimal mengandung sebuah fungsi. Setiap fungsi terdiri dari satu atau beberapa pernyataan, yang secara keseluruhan dimaksudkan untuk melaksanakan tugas khusus.

Bagian pernyataan fungsi (disebut tubuh fungsi) diawali dengan tanda { dan diakhiri dengan tanda }

3. VARIABEL

Variabel dalam program digunakan untuk menyimpan suatu nilai tertentu dimana nilai tersebut dapat berubah-ubah. Setiap variabel mempunyai tipe dan hanya data yang bertipe sama dengan tipe variabel yang dapat disimpan di dalam variabel tersebut. Setiap variabel mempunyai nama. Pemisahan antar variabel dilakukan dengan memberikan tanda koma.

Contoh :

```
int jumlah;
float harga_per_unit, total_biaya;
```

Dari contoh diatas,variabel `jumlah` hanya boleh menerima data yang bertipe integer (bulat), tidak boleh menerima data bertipe lainnya. Variabel `harga_per_unit` dan `total_biaya` hanya bisa diisi dengan bilangan float (pecahan).

4. KONSTANTA

Berbeda dengan variabel yang isinya bisa berubah selama eksekusi program berlangsung, nilai suatu konstanta tidak bisa berubah.

Contoh :

```
const int m = 8;
#define pajak 0.05
```

5. FUNGSI `main()`

Fungsi main() harus ada pada program, karena fungsi inilah yang menjadi titik awal dan titik

akhir eksekusi program. Tanda { di awal fungsi menyatakan awal tubuh fungsi sekaligus awal eksekusi program, sedangkan tanda } di akhir fungsi merupakan akhir tubuh fungsi dan sekaligus akhir eksekusi program.

6. FUNGSI `printf()`

Merupakan fungsi yang digunakan untuk menampilkan data ke layar. Dengan menggunakan fungsi ini, tampilan dapat diatur (diformat) dengan mudah.

Bentuk umum dari fungsi ini :

```
printf("string kontrol", argumen1, argumen2, ....);
```

String kontrol dapat berupa keterangan beserta penentu format (seperti %d, %f). Argumen adalah data yang akan ditampilkan, dapat berupa variabel, konstanta, maupun ungkapan.

Contoh :

```
/* Program Satu */
#include
<stdio.h>main()
   {
Printf("Belajar Pemrograman Komputer");
}
```

7. FUNGSI `scanf()`

Merupakan fungsi yang digunakan untuk menampilkan data yang dimasukkan dari keyboard.

CONTOH PERMASALAHAN 1 :

Menghitung luas dan keliling lingkaran dengan besar jari-jari lingkaran dimasukkan melalui keyboard.

```
/* Program Dua */
/* Menghitung Luas dan Keliling Lingkaran */
```
Judul program

```
#include<stdio.h>
 #include<conio.h>
#define phi 3.14
main()
{
float jari,luas,keliling
printf("Masukan jari-jari lingkaran = ");
scanf("%f", &jari);
luas=phi*jari*jari;
keliling=2*phi*jari;
printf("Luas lingkaran = %f \n",luas);
```

Header File yg digunakan

Deklarasi

Deskripsi

```
printf(”Keliling lingkaran = %f \n”,keliling);

     }
```

CONTOH PERMASALAHAN 2 :

Menukarkan dua buah nilai dari dua buah variabel. Misalnya, sebelum pertukaran nilai a=5, nilai b=3, maka setelah pertukaran nilai a=3, nilai b=5.

```
   /* Program Tiga */
   /* Menukarkan nilai a dengan b */
   #include<stdio.h>
   #include<conio.h>
   main()
{
int a,b,c;
printf(”Program Menukar 2 Buah Nilai \n\n”);
printf(”Sebelum ditukar \n”); printf(”--------------- \n”);
printf(”Bilangan pertama = ”); scanf(”%i”,&a); printf(”Bilangan
kedua = ”); scanf(”%i”,&b);
 c=a;
 a=b;
 b=c;
 printf(“Setelah ditukar \n”);
 printf(”--------------- \n”);
 printf(”Bilangan pertama = %i\n”,a);
 printf(”Bilangan kedua = %i\n”,b);

 }
```

CONTOH PERMASALAHAN 3 :

Mengkonversikan total detik menjadi jam menit detik.

Petunjuk : 1 menit = 60 detik

1 jam = 3600 detik

```
   /* Program Empat */
   /* Mengkonversi total detik menjadi jam menit detik */
   #include<stdio.h>
   #include<conio.h>
   main()
{
int totdet,jam,sisa,menit,detik;
printf("Program Konversi Detik Menjadi Jam Menit Detik \n");

printf("------------------------------------------- \n");
printf("Masukkan total detik = ");
scanf("%i",&totdet);
jam=totdet/3600;
sisa=totdet%3600;
menit=sisa/60;
detik=sisa%60;
printf("Jumlah jam = %i jam \n",jam);
printf("Jumlah menit = %i menit \n",menit);
printf("Jumlah detik = %i detik \n",detik);
```

TUGAS

Seorang user di warnet mulai menggunakan internet pada pukul J1 dan selesai pada pukul J2. Bila tarip penggunaan di warnet tersebut 1 jam Rp. 5000,- maka **buat program billing warnet** untuk menghitung lama pemakaian (dalam jam menit detik) dan biaya yang harus dibayar user.

PRAKTIKUM MINGGU 2
PEMILIHAN KONDISI (SELECTION)

1. PILIHAN TUNGGAL Bentuk paling sederhana pilihan tunggal adalah jika hanya ada satu pilihan kondisi yang disediakan.

```
if kondisi
{
true statement
}
```

CONTOH PERMASALAHAN 4 :

Menentukan huruf vokal dari suatu huruf yang dimasukkan dari keyboard.

```
/* Program Lima */
/* Program menentukan huruf vokal */
#include<stdio.h>
#include<conio.h>
main()
{
        char huruf;
        int main();
        printf("Program Menentukan Huruf Vokal \n");
        printf("------------------------------ \n \n");
        printf("Masukan huruf : ");
        scanf("%c",&huruf);
        if(huruf=='a'||huruf=='i'||huruf=='u'||huruf=='e'||huruf=='o')
        printf("Huruf %c adalah huruf vokal", huruf);
}
```

2. PILIHAN GANDA

Digunakan untuk menentukan tindakan yang akan digunakan bila kondisi bernilai benar dan salah.

```
if kondisi
{
true statement
}
else
{
false statement
}
```

CONTOH PERMASALAHAN 5 :

Menentukan bilangan terbesar dari dua buah bilangan.

```
/* Program Enam */
/* Program menentukan bilangan terbesar dari dua bilangan */
#include<stdio.h>
#include<conio.h>
main()
{
    int a,b;
    int main();
    printf("Masukan bilangan pertama = ");
    scanf("%i",&a);
    printf("Masukan bilangan kedua = ");
    scanf("%i", &b);
    if(a>b)
    printf("Bilangan terbesar adalah %i",a);
    else
    printf("Bilangan terbesar adalah %i",b);}
```

3. PILIHAN MAJEMUK Untuk menentukan tindakan yang akan digunakan disediakan lebih dari 2 alternatif. Merupakan bentuk statement if dengan statement if lain di dalam if sebelumnya.

```
if kondisiA
{
if kondisiB
{
true statementB
}
else
{
false statementB
}
}

else
{
false statementA
}
```

CONTOH PERMASALAHAN 6 :

Mengelompokan nilai dengan ketentuan :

Jika nilai angka >= 90, maka nilai huruf = A

Jika nilai angka >= 80, maka nilai huruf = B

Jika nilai angka >= 70, maka nilai huruf = C

Jika nilai angka >= 60, maka nilai huruf = D

Jika nilai angka < 60, maka nilai huruf = E

```
/* Program Tujuh */
/* Program mengelompokan nilai */
#include<stdio.h>
#include<conio.h>
main()
{
int nilai;
printf("PROGRAM PENGELOMPOKAN NILAI \n");
printf("-------------------------- \n");
printf("Masukkan nilai (0 - 100) : ");
scanf("%i",&nilai);
if(nilai>=90)
printf("Nilai = A");else if(nilai>=80) printf("Nilai = B"); else
if(nilai>=70)
printf("Nilai = C");
else
if(nilai>=60)
printf("Nilai = D");
else
printf("Nilai = E");

}
```

4. STRUKTUR CASE (STATEMENT SWITCH) Untuk masalah dengan dua pilihan atau lebih, struktur CASE dapat menyederhanakan penulisan IF yang bertingkat-tingkat.

```
Switch(kondisi)
{
case konstanta1 : {Statement-statement ; break}
case konstanta1 : {Statement-statement ; break}
case konstanta1 : {Statement-statement ; break}
case konstanta1 : {Statement-statement ; break}
      .....
      }
```

CONTOH PERMASALAHAN 7 :

Pemilihan kode jurusan dengan struktur case

```
/* Program Delapan */
/* Program pemilihan kode jurusan */
#include<stdio.h>
#include<conio.h>
#include<string.h>
main()
{
char nama[15],ket[30],kode;
```

```
   printf("Masukkan nama mahasiswa: ");
   scanf("%s",&nama);
   printf("Pilih kode jurusan [A/B/C/D] : ");
   kode=getche();
   switch (kode)
   {
   case 'A' : {
   strcpy(ket,"Jurusan Teknik Informatika");
   break;
   }
   case 'B' : {
   strcpy(ket,"Jurusan Manajemen Informatika");
   }
   case 'C' : {strcpy(ket,"Jurusan Sistem Informasi");
   break;
   }
   case 'D' : {
   strcpy(ket,"Jurusan Teknik Komputer");
   break;
   }
   }
   printf("\n \n");
   printf("Nama mahasiswa : %s \n",nama);
   printf("Kode jurusan : %c \n",kode);
   printf("Nama jurusan : %s \n",ket);

}
```

TUGAS

2. Misalkan karyawan PT. Makmur dikelompokkan berdasarkan golongannya. Upah per jam tiap karyawan bergantung pada golongannya, dengan ketentuan :

   Golongan Upah per jam

   A. Rp. 5000,-
   B. Rp. 7000,-
   C. Rp. 8000,-
   D. Rp. 10.000,-

   Jumlah jam kerja normal selama 1 minggu adalah 48 jam. Kelebihan jam kerja dianggap lembur dengan upah lembur adalah Rp.4000,- per jam untuk semua golongan karyawan.
   Buat program menghitung gaji karyawan mingguan. Data yang dimasukan dari keyboard adalah nama karyawan, golongan, jumlah jam kerja. Data yang dicetak adalah nama karyawan dan gajinya.
   Program dibuat dengan menggunakan struktur IF dan CASE.

3. Diinginkan sebuah program menghitung segitiga siku-siku yang mempunyai tampilan menu dan fasilitas sebagai berikut dan pada setiap pilihan menu, dimasukkan data alas (sisi siku-siku pertama) dan tinggi (sisi siku-siku kedua). Program dibuat dengan menggunakan struktur CASE.

MENU SEGITIGA SIKU-SIKU

1. Hitung panjang sisi miring
2. Hitung luas
3. Hitung keliling
4. Keluar program

PRAKTIKUM MINGGU 3
PENGULANGAN (LOOPING)

1. STRUKTUR FOR Struktur ini digunakan bila kita mengetahui secara pasti banyaknya pengulangan yang akan dilakukan. Pernyataan FOR mempunyai 3 parameter yaitu :

1. Nilai awal (initial value)
2. Test kondisi yang menentukan akhir loop (condition expression)
3. penentu perubahan nilai (incremental expression)

Bentuk FOR :

```
for (initial value; condition expression; incremental expression)
```

Keterangan :

Initial value : memberikan nilai awal pada variabel kontrol

Condition expression : ekspresi yang menyatakan berhentinya pengulangan. Jika tes kondisi bernilai salah maka loop akan berhenti.

Incremental expression : berfungsi menaikkan/menurunkan nilai dari variabel kontrol. Dapat berupa nilai positif (penaikan) / nilai negatif (penurunan)
Penaikan : setiap loop operator ++ akan menambah nilai 1 ke variable kontrol
Penurunan : setiap operator -- akan menurunkan nilai 1 pada variabel kontrol

CONTOH PERMASALAHAN 8 :

```
/* Program Sembilan */
/* Program Mencetak Angka Urut (Penaikan) */
#include<stdio.h>
#include<conio.h>
main()
{
int i;
clrscr();
for (i=1;i<=10;i++)
printf("%d.Hallo,Selamat belajar,Aku yakin Aku pasti bisa...
\n",i);
}
```

CONTOH PERMASALAHAN 9 :

```
/* Program Sepuluh */
/* Program Mencetak Angka Urut (Penurunan) */
#include<stdio.h>
#include<conio.h>
main()
{
```

```
int i;
clrscr();
for (i=10;i>=1;i--)
printf("%d.Hallo,Selamat belajar,Aku yakin Aku pasti bisa...
\n",i);
}
```

CONTOH PERMASALAHAN 10 :

```
/* program sebelas */
/* program mencari data terbesar terkecil */
#include <stdio.h>
#include <conio.h>
main ()
{
     int n,i,max,min,bil;
     printf ("program mencari data terbesar dan terkecil \n\n");
     printf ("masukkan banyaknya data = ");
     scanf ("%d",&bil);
     max=bil;
     min=bil;
     for (i=2;i<=n;i++)
     {
          printf ("masukkan bilangan ke-%d : ",i);
          scanf ("%d",&bil);
          if (bil>max)
          max=bil;
          if(bil<min)
          min=bil;
     }
     printf ("\n");
     printf ("data terbesar %d \n",max);
     printf ("data terkecil %d \n",min);
}
```

CONTOH PERMASALAHAN 11 :

```
/* Program Duabelas */
/* Program Mencetak Jumlah Bilangan */
#include<stdio.h>
#include<conio.h>
main()
{
int i,awal,akhir,jumlah;
clrscr();
printf("Masukkan bilangan awal : ");
scanf("%d",&awal);
printf("Masukkan bilangan akhir : ");
scanf("%d",&akhir);
jumlah=0;
```

```
for (i=awal;i<=akhir;i++)
{
jumlah=jumlah+i;
printf("Jumlah bilangan dari %d sampai %d adalah
     %d",awal,akhir,jumlah);}
}
```

CONTOH PERMASALAHAN 12 :

```
/* Program Tigabelas */
/* Program Mencetak Bilangan Kuadrat, Akar Bilangan dan Jumlahnya
*/
#include<stdio.h>
#include<conio.h>
#include<math.h>
main()
{
int i,kuadrat,jumbil,jumkuadrat;
float akar,jumakar;
clrscr();
printf("TABEL KUADRAT DAN AKAR BILANGAN \n");
printf("--------------------------------- \n");
printf("Bilangan Kuadrat Akar Bilangan \n");
printf("--------------------------------- \n");
jumbil=0;
jumkuadrat=0;
jumakar=0;
for (i=1;i<=10;i++)
{ kuadrat=i*i; akar = sqrt(i);
jumbil=jumbil+i;
jumkuadrat=jumkuadrat+kuadrat;
jumakar = jumakar+akar;
printf(" %2d %3d %7.4f \n",i,kuadrat,akar);
}
printf("================================= \n");
printf(" %2d %3d %7.4f\n",jumbil,jumkuadrat,jumakar);
}
```

TUGAS

4. Buat progran menghitung jumlah 6 suku pertama dari barisan $1^3,2^3,3^3,4^3$
5. Buat program menghitung jumlah dari deret 3,7,11, 15, ...

2. STRUKTUR FOR BERSARANG CONTOH PERMASALAHAN

13 : Membuat tampilan seperti berikut :

```
*****
*****
****
```

```
/*program empatbelas*/
/*program mencetak bintang persegipanjang*/
#include<stdio.h>
#include<conio.h>
main()
{
      int i,j;
      int main();
      printf("contoh loop bersarang-->bintang persegipanjang\n\n");
      for(i=1;i<=3;i++)

      {
            for(j=1;j<=5;j++)
            {
                  printf("*");
            }
            printf("\n");
      }
}
```

CONTOH PERMASALAHAN 14 :

Membuat tampilan seperti berikut :

```
/*program empatbelas*/
/*program mencetak bintang persegipanjang*/
#include<stdio.h>
#include<conio.h>
main()
{
      int i,j;
      int main();
      printf("contoh loop bersarang-->bintang persegipanjang\n\n");
      for(i=1;i<=3;i++)

      {
            for(j=1;j<=5;j++)
            {
                  printf("*");
            }
            printf("\n");
```

```
    }
}
```

TUGAS

6. Buat program tabel perkalian : *

*	1	2	3	4	5	6	7	8	9	10
1	.	.	.	.	.	.	.	.	.	.
2	.	.	.	.	.	.	.	.	.	.
3	.	.	.	.	.	.	.	.	.	.
4	.	.	.	.	.	.	.	.	.	.
5	.	.	.	.	.	.	.	.	.	.
6	.	.	.	.	.	.	.	.	.	.
7	.	.	.	.	.	.	.	.	.	.
8	.	.	.	.	.	.	.	.	.	.
9	.	.	.	.	.	.	.	.	.	.
10	.	.	.	.	.	.	.	.	.	.

7. Buat tampilan seperti berikut :

```
Masukkan segitiga angka : 5
1
12
123
1234
12345
1234
123
12
1
```

3. STRUKTUR WHILE

Struktur ini digunakan bila kita belum mengetahui secara pasti berapakali banyaknya pengulangan yang akan dilakukan. Berakhirnya proses pengulangan ditentukan oleh suatu kondisi. Selama kondisi terpenuhi, maka pengulangan terus dilakukan, dan sebaliknya, bila kondisinya tidak terpenuhi maka pengulangan dihentikan.

Bentuk WHILE :

```
while (condition expression)
{
statement-statement;
}
```

CONTOH PERMASALAHAN 16 :

```
/* Program Tujuhbelas*/
/* Program Menghitung Rata-rata Bilangan */
#include<stdio.h>
#include<conio.h>
main()
{
int n=0;
float jumlah=0,bil,rata;
char lagi='Y';
clrscr();
while ((lagi=='Y')||(lagi=='y'))
{
printf("Masukkan bilangan : ");
scanf("%f",&bil);
jumlah=jumlah+bil;
n=n+1;
printf("Apakah Anda akan memasukkan data lagi [Y/T]: ");
scanf("%s",&lagi);
printf("\n");
}
rata=jumlah/n;
printf("\n");
printf("Banyaknya bilangan : %i \n",n);
printf("Rata-rata bilangan : %.2f \n",rata);
getch();
}
```

3. STRUKTUR DO-WHILE

DO-WHILE pada dasarnya sama dengan instruksi WHILE. Perbedaannya hanya terletak pada penempatan ekspresi kondisi. Untuk DO-WHILE, kondisi diletakkan pada bagian bawah. Jadi statement-statement yang berada dalam loop akan dikerjakan dahulu baru dilakukan tes terhadap kondisi.

Bentuk DO-WHILE :

```
do
{
statement-statement;
}
while (condition expression);
```

CONTOH PERMASALAHAN 17 :

```
/*program delapanbelas*/
/*program menghitung rata-rata bilangan*/
#include<stdio.h>
#include<conio.h>
main()
{
        int n=0;
        float jumlah=0,bil,rata;
        char lagi;
        int main();
        do
        {
                printf ("masukkan bilangan:");
                scanf("%f",&bil);
                jumlah=jumlah+bil;
                n=n+1;
                printf("apakah anda akan memasukkan data lagi [Y/T]:");
                scanf("%s",&lagi);
                printf("\n");
        }
        while((lagi=='Y')||(lagi=='y'));
        rata=jumlah/n;



        printf("\n");
        printf("banyaknya bilangan :%i\n",n);
        printf("rata-rata bilangan :%2f\n",rata);
}
```

CONTOH PERMASALAHAN 18 :

```
/*program sembilanbelas*/
/*program membuat menu pengulangan*/
#include<stdio.h>
#include<conio.h>
main()
{
    int pilih;
    do
```

```
    {
          int main();
          printf("daftar menu tombo-ngeleh\n");
          printf("------------------------\n");
          printf("1. pecel lele\n");
          printf("2. tempe bakar penyet\n");
          printf("3. rica-rica ayam\n");
          printf("4. bakso sapi/kakap\n");
          printf("\n");
          printf("masukkan pilihan anda! (0=selesai)");
          scanf("%i",&pilih);
          switch(pilih)
          {
               case 1:
                    printf("harga pecel lele rp.4000,-");
                    break;
                    case 2:
                         printf("harga tempe bakar penyet
rp.3000,-");
                         break;
                         case 3:
                              printf("harga rica-rica ayamm
rp.5000,-");
                              break;
                              case 4:
                                   printf("harga bakso sapi/kakap
rp.5000,-");
                                   break;
                                   case 0:
                                        printf("selesai");
                                        break;
          }
    }
    while (pilih!=0);
}
```

PRAKTIKUM MINGGU 4
ARRAY & ANIMASI HURUF

Array merupakan koleksi data dimana setiap elemen memakai nama dan tipe yang sama serta setiap elemen diakses dengan membedakan indeks arraynya.

Bentuk : `tipe nama_var[ukuran];`

Keterangan :

Tipe : menyatakan jenis elemen array misal int, char, dll

Ukuran : menyatakan jumlah maksimal elemen array

Contoh : int nilai [5]

1. ARRAY DIMENSI SATU

CONTOH PERMASALAHAN 19 :

```
/* Program Duapuluh */
/* Program membuat animasi huruf */
   #include <stdio.h>
   int main(){
     int array[10] =
{9,1,16,5,7,9,2,4,6,8};
     printf("%d \n",
array[9]);//di dalam kotak angka
ke
   }
```

2. ARAY DUA DIMENSI

CONTOH PERMASALAHAN 20 :

```
/* Program Duapuluh satu */
/* Program menginput data dengan array */
#include <stdio.h>

int main(void)
{
  int bilangan[2][2];

  bilangan[0][0] = 100;
  bilangan[0][1] = 101;
  bilangan[1][0] = 110;
  bilangan[1][1] = 111;

  printf("Isi array bilangan: \n");
  printf("%d, %d \n",bilangan[0][0],bilangan[0][1]);
  printf("%d, %d \n",bilangan[1][0],bilangan[1][1]);
```

```
}
```

CONTOH PERMASALAHAN 22 :

```
 /* Program Duapuluh tiga */
#include <stdio.h>
#include <conio.h>


main()
{
     int i,s;
    char h=64, nama[5][4][22] =
{"A","B","C","D","E","F","G","H","I","J","K",
"L","M","N","O","P","Q","S","T","U" };//
[jumlah baris][jumlah kolom][jumlah karakter]
printf("urutan huruf : \n\n");
for(i=0; i<5; i++)
    {
        ++h;
        printf("Grup %c  \n", h);

        for(s=0; s<4; s++)
        {
            printf("      %d. %s \n", s+1,nama[i][s]);
        }
        printf("\n");
    }
```

PRAKTIKUM MINGGU 5
SUBPROGRAM / FUNGSI

Program komputer yang dibuat untuk menyelesaikan permasalahan umumnya berukuran besar.

- Cara terbaik untuk menangani program besar adalah menyusunnya dari potongan-potongan program yang berukuran kecil-kecil (disebut modul) merupakan konsep dari pemrograman terstruktur yaitu pemrograman yang menitikberatkan pada pemecahan masalah yang kompleks menjadi masalah yang sederhana.
- Program yang terdiri dari modul/subprogram/prosedur/routine lebih mudah ditangani dibanding dengan program yang terdiri dari banyak sekali baris.
- Modul program dalam C disebut fungsi (function)
- Fungsi adalah blok dari kode yang dirancang untuk melakukan tugas khusus.
- Tujuan pembuatan fungsi :
  - program menjadi terstruktur
  - menghemat kode program karena dapat mengurangi duplikasi kode
  - fungsi dapat dipanggil dari program atau fungsi yang lain
  - mempersingkat/memperpendek panjang program
  - mempermudah cek kesalahan

PENDEKLARASIAN DAN PENDEFINISIAN FUNGSI

- Fungsi harus dideklarasikan di dalam program pemanggil/program utama, dengan tujuan supaya program pemanggil mengenal nama fungsi tersebut serta cara mengaksesnya.
- Deklarasi fungsi diakhiri ;
- Fungsi didefinisikan dengan diawali tipe data keluaran fungsi (di depan nama fungsi), defaultnya adalah integer.
- Pendefinisian fungsi tidak diakhiri ;
- Aturan pemberian nama fungsi sama dengan aturan penulisan variabel
- Blok fungsi diawali dengan { dan diakhiri dengan }

Bentuk :

```
tipe_data nama_fungsi (daftar parameter)
```

Keterangan :

daftar parameter : berisi variabel dan tipe variabel yang berfungsi sebagai masukan untuk fungsi tersebut. Masukan tersebut akan diproses untuk menghasilkan nilai tertentu sesuai dengan tipe data fungsi.

contoh : `int tukar (int x, int y)`

VARIABEL GLOBAL, VARIABEL LOKAL, VARIABEL STATIK

VARIABEL GLOBAL

- Variabel yang dideklarasikan di luar blok fungsi dan bersifat dikenali oleh semua bagian program.
- Data-data yang tersimpan dalam sebuah variabel dapat diakses di setiap blok fungsi

• Disarankan untuk tidak digunakan, karena variabel ini dapat men-sharing-kan data dan dapat diubah secara tidak sengaja oleh suatu blok fungsi, sehingga nilainya bisa berubah.

VARIABEL LOKAL

• Variabel yang dideklarasikan dalam suatu blok fungsi tertentu dan hanya dikenal oleh blok fungsi tersebut.
• Variabel lokal akan dihapus dari memori jika proses sudah meninggalkan blok letak variabel lokalnya.

VARIABEL STATIK

• Variabel statik sering dipakai sebagai variabel lokal.
• Beda variabel lokal dan variabel statik adalah variabel lokal akan dihapus dari memori jika proses sudah meninggalkan blok letak variabel lokalnya, sedangkan variabel statik akan dihapus dari memori jika program dimatikan.

Contoh :

```
int i,j; /* variabel global */
main ()
{
int k,l; /* variabel lokal */
}
fungsi()
{
static int m,n; /* variabel statik */
}
```

PARAMETER FORMAL & PARAMETER AKTUAL (NYATA)

• Parameter formal adalah variabel yang ada pada daftar parameter dalam definisi fungsi.
• Parameter aktual (nyata) adalah parameter yang dapat berupa variabel atau konstanta maupun ungkapan yang dipakai dalam pemanggilan fungsi.

Cara melewatkan/mengirim parameter ke dalam fungsi :

• pengiriman parameter dengan nilai (paramater by value) disebut juga parameter masukan
- Nilai dari parameter aktual akan disalin ke parameter formal.
- Nilai parameter aktual tidak dapat berubah sekalipun nilai parameter formal berubah-ubah.

Contoh :

A,B parameter aktual
x, y parameter formal
A x

B y
Pada saat pemanggilan suatu fungsi, misal :
A bernilai 20 x juga bernilai 20
B bernilai 30 y juga bernilai 30

- pengiriman parameter secara acuan (parameter by reference) disebut juga parameter masukan/keluaran
  - perubahan – perubahan yang terjadi pada nilai parameter formal di fungsi akan mempengaruhi nilai parameter aktual.
  - merupakan upaya untuk melewatkan alamat dari suatu variabel ke dalam fungsi
  - dipakai untuk mengubah isi suatu varabel di luar fungsi dengan pelaksanaan pengubahan dilakukan di dalam fungsi

FUNGSI YANG MENGEMBALIKAN NILAI

- Pada dasarnya semua fungsi mengembalikan nilai, tetapi untuk fungsi yang tidak mengembalikan nilai disebut fungsi bertipe **void**.
- Untuk mengembalikan nilai sebuah fungsi, digunakan kata **return** yang diikuti dengan nilai yang akan dikembalikan.
- Dalam sebuah fungsi return bisa mengembalikan beberapa nilai, tetapi setiap kata return hanya bisa mengembalikan sebuah nilai saja.

CONTOH PERMASALAHAN 23 : menggunakan fungsi yang tidak mengembalikan nilai

```
Algoritma fungsi_garis
Deklarasi
    function garis () → integer
Deskripsi
     garis()
     write("NO NIM NAMA NILAI")
     garis()


function garis() → integer
Deklarasi
Deskripsi
     write("====================")
```

```
Output program :

====================
NO NIM NAMA NILAI
====================
```

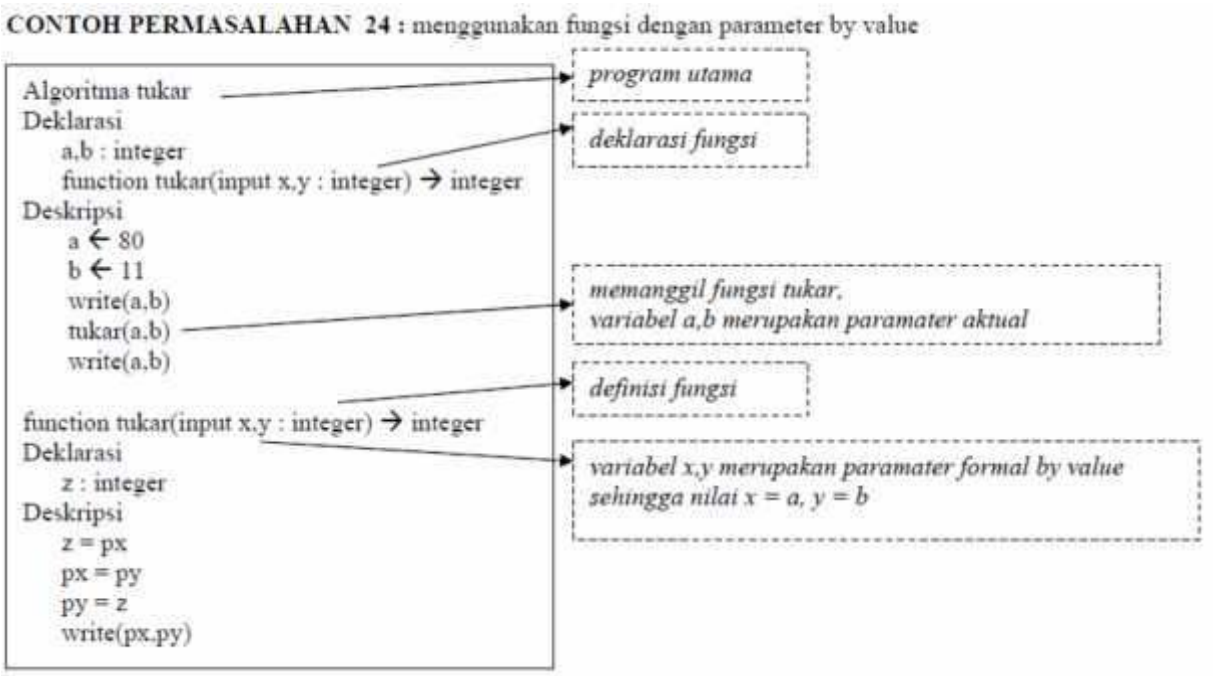
```
/* Program Duapuluh empat */
/* Program sederhana menggunakan fungsi */
/* yang tidak mengembalikan nilai */
#include<stdio.h>
#include<conio.h>

void garis();          /* deklarasi fungsi */
main()
{
  clrscr();
  garis();
  printf("NO   NIM   NAMA      NILAI \n");
  garis();
  getch();
}
void garis()          /* definisi fungsi */
{
```

program utama

memanggil fungsi garis

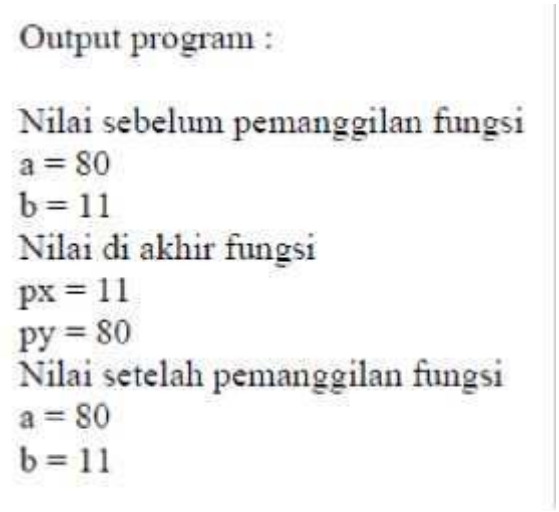
CONTOH PERMASALAHAN 24 : menggunakan fungsi dengan parameter by value




```
/* Program Duapuluh lima */
/* Program tukar menggunakan fungsi */
/* dengan parameter by value */
#include<stdio.h>
#include<conio.h>
void tukar(int x,int y); /*deklarasi fungsi*/
main()
{
int a,b;
clrscr();
a=80;
b=11;
printf("Nilai sebelum pemanggilan fungsi \n");
printf("a = %i b = %i \n",a,b);
tukar(a,b);
printf("Nilai setelah pemanggilan fungsi \n");
printf("a = %i b = %i \n",a,b);
getch();
}
void tukar(int px,int py) /* definisi fungsi */
{
int z;
z=px;
px=py;
py=z;
printf("Nilai diakhir fungsi \n");
printf("px = %i py = %i \n",px,py);
```

```
}
```

Output program :
Nilai sebelum pemanggilan fungsi
a = 80 b = 11
Nilai di akhir fungsi
px = 11
py = 80
Nilai setelah pemanggilan fungsi
a = 80
b = 11



CONTOH PERMASALAHAN 26 : menggunakan fungsi dengan parameter by reference

```
Algoritma tukar
Deklarasi a,b
: integer
function tukar(input *px,*py :
integer)
Deskripsi
a 80 b 11
write(a,b)
tukar(a,b)
write(a,b)
function tukar(input *px,*py :
integer)
Deklarasi z
: integer
Deskripsi
z = *px
*px = *py
*py = z
write(*px,*py)
```

```
/* Program Duapuluh enam */
/* Program tukar menggunakan fungsi */
/* dengan parameter by reference */
#include<stdio.h>
#include<conio.h>
void tukar(int *px,int *py);
main()
{
int a,b;
clrscr();
a=80;
b=11;
printf("Nilai sebelum pemanggilan fungsi \n");
printf("a = %i b = %i \n\n",a,b);
tukar(&a,&b);
printf("Nilai setelah pemanggilan fungsi \n");
printf("a = %i b = %i \n\n",a,b);
getch();
```

```
}
void tukar(int *px,int *py)
{
int z;
z=*px;
*px=*py;
*py=z;
printf("Nilai diakhir fungsi \n");
printf("*px = %i *py = %i \n\n",*px,*py);
}
```

```
/* Program Duapuluh tujuh */
#include <stdio.h>
int minimum (int x, int y);
main()
{
     int a,b,kecil;
     printf("Masukan nilai a= ");
     scanf("%d",a);
     printf("Masukan nilai b= ");
     scanf("%d",b);
     kecil=minimum(a,b);
     printf ("\n Bilangan terkecil anatar %d dan %d adalah
%d",a,b,kecil);
}
int minimum (int x,int y)
{
     if (x<y)
     return (x);
     else
     return (y);
}
```

```
/* Program Duapuluh delapan */
#include <stdio.h>
int faktorial (int m);
main()
{
     int x;
     printf("Mencari faktorial dari suatu bilangan\n");
     printf("Masukan nilai x(bilangan positif) ; ");
     scanf("%d",&x);
     printf("faktorial dari %d = %d \n",x,faktorial (x));
}
int faktorial (int m)
{
     if(m==1)
     return (1);
     else
     return (m*faktorial(m-1));
}
```

PRAKTIKUM MINGGU 6
RECORD/STRUCT/STRUKTUR

Struktur adalah koleksi dari variabel yang dinyatakan dengan sebuah nama, dengan sifat setiap variabel dapat memiliki tipe yang berlainan. Struktur bisa dipakai untuk mengelompokkan beberapa informasi yang berkaitan menjadi sebuah kesatuan (Pada Bahasa Pascal, struktur disebut dengan record).

Contoh sebuah struktur adala informasi data tanggal, yang berisi:

- Tanggal
- Bulan, dan
- Tahun.

1. Bentuk umum mendeklarasikan/mendefinisikan struktur

Struct nama_tipe_struktur

```
{tipe field1;

 tipe field2;

 .....

 .....
 tipe fieldN;
}variabel_struktur1,...,variabel_strukturM;
```

Adapun **variabel_struktur1,...,variabel_strukturM** menyatakan bahwa variabel struktur yang dideklarasikan bisa lebih dari satu.

Contoh mendefinisikan struktur

```
struct data_tunggal

{int tanggal;
 int bulan;
 int tahun;
};tgl_lahir;
```

Yang mendefinisikan tipe struktur bernama data_tunggal, yang terdiri dari tiga buah elemen (field) berupa tanggal, bulan dan tahun. Field adalah sebutan untuk elemen struktur. Sedangkan variabel tgl_lahir betipe struktur data_tunggal yang mengandung tiga field yaitu tanggal, bulan dan tahun.

Note: nama_tipe_struktur atau variabel_struktur boleh dihilangkan tetapi tidak boleh kedua-duanya dihilangkan.

2. Pemanggilan elemen struktur

Elemen struktur dapat dipanggil dalam program menggunakan bentuk

Variabel_struktur.nama_field

Antara variabel_struktur dan nama_field dipisahkan dengan operator titik (disebut operator titik anggota struktur).

Sedangkan untuk memberikan data nama ke field nama, pernyataan yang diperlukan misalnya berupa

Contoh Program 27

Program untuk mengakses elemen struktur

```
#include <stdio.h>

struct manusia
{
        char nama[100];
        int umur;

};

struct mahasiswa
{
        char jurusan [20];
        int no_absen;
        struct manusia orang;
```

```
};

int main()
{
	struct mahasiswa veni = {"S1 Elektro 2016", 22, {"Veni Apriani", 20}};
	printf("Nama Mahasiswa : %s \n", veni.orang.nama);
	printf("Umur : %d \n", veni.orang.umur);
	printf("Jurusan : %s \n", veni.jurusan);
	printf("No Absen : %i \n", veni.no_absen);


}
```

3. Array dan struktur

Penggunaan struktur sering dikaitkan dengan array, membentuk array dari struktur. Contoh dari array struktur adalah array yang dipakai untuk menyimpan data rekan. Array yang diperlukan untuk masalah ini berupa

```
#define MAKS 20

struct data_rekan larik_rekan[MAKS];
```

Setelah array

larik_rekan dideklarasikan, ruang yang disediakan akan ditunjukkan pada gambar berikut

Tanggal lahir

No	Nama	Tanggal	Bulan	Tahun
0				
1				
2				
.				
.				
.				
N				

Array dari struktur

Contoh Program 29

Program berikut merupakan contoh mengenai array dari struktur. Mula-mula seluruh data dimasukkan ke dalam array, kemudian ditampilkan sehingga membentuk tabel.

```
#include"stdio.h"

#include"conio.h"
void main()
{struct zodiak

 {char nama[11];

  int   tgl_awal;
  int  bln_awal;
  int  tgl_akhir;
  int bln_akhir;
  };

     printf("Bintang anda bukanlah %s\n",bintang.nama);

   getch();}
```

```
printf("Tanggal lahir anda (XX-XX-XXXX): ");

 scanf("%d-%d-%d",&tg_lhr,&bl_lhr,&th_lhr);



 if((tg_lhr>=bintang.tgl_awal&&
   bl_lhr==bintang.bln_awal)||
   (tg_lhr<=bintang.tgl_akhir&&
   bl_lhr==bintang.bln_akhir))


   printf("Bintang anda adalah %s\n",bintang.nama);

   else

   printf("Bintang anda bukanlah %s\n",bintang.nama);

   getch();

}
```

Output setelah program dijalankan

Tanggal lahir anda (XX-XX-XXXX): 15-05-1985

Bintang anda bukanlah Sagitarius

4. Struktur dan fungsi

Pada bagian ini masalah yang dibahas meliputi

Cara melewatkan elemen struktur meupun struktur ke dalam fungsi

Cara melewatkan elemen struktur ke dalam fungsi dapat dilihat pada contoh program berikut

Contoh program 30

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

void tukar_xy(int *x, int *y); /* deklarasi fungsi */

void main()

{ struct koordinat

  { int x; int y; };

  struct koordinat posisi = { 21, 34 };

  clrscr();

  printf("x, y semula --> %d, %d\n", posisi.x, posisi.y);

  tukar_xy(&posisi.x, &posisi.y);

  printf("x, y kini --> %d, %d\n", posisi.x, posisi.y);

  getch();

}

void tukar_xy(int *x, int *y)

{ int z; z = *x; *x = *y; *y = z; }
```

Output setelah program dijalankan

```
x, y semula --> 21, 34
x, y kini --> 34, 21
```

Latihan

1. Kembangkanlah contoh program 2 sehingga dapat dipakai untuk menentukan bintang kelahiran. Sebagai acuan gunakan data berikut:

   Aries : 21 Maret-19 April
   Taurus : 20 April-20 Mei
   Gemini : 21 Mei-20 Juni
   Cancer : 21 Juni-22 Juli
   Leo : 23 Juli-22 Agustus

   Virgo : 23 Agustus-22 September
   Libra : 23 September-22 Oktober
   Scorpio : 23 Oktober-21 November
   Sagitarius : 22 November-21 Desember
   Aquarius : 20 Januari-18 Februari
   Pisces : 19 Februari-20 Maret

   Gunakan array untuk mengimplementasikan program.

2. Buatlah program untuk menginputkan data-data nilai nama mahasiswa, QUIS, UTS dan UAS dalam suatu struktur daftar nilai mahasiswa dalam suatu kelas mata kuliah algoritmadanpemrograman

PRAKTIKUM MINGGU KE 7

FILE

File adalah sebuah organisasi dari sejumlah record. Masing-masing record bisa terdiri dari satu atau beberapa field. Setiap field terdiri dari satu atau beberapa byte.

1. Membuka File

Untuk membuka atau mengaktifkan file, fungsi yang digunakan adalah fungsi **fopen().**
File dapat berupa file biner atau file teks.
File biner adalah file yang pola penyimpanan di dalam disk dalam bentuk biner, yaitu seperti bentuk pada memori (RAM) computer.
File teks adalah file yang pola penyimpanan datanya dalam bentuk karakter.
Penambahan yang perlu dilakukan untuk menentukan mode teks atau biner adalah "t" untuk file teks dan "b" untuk file biner.
Prototype fungsi fopen() ada di header fungsi "**stdio.h"**
Bentuk umum :

file *fopen(char *namafile, char *mode);

Keterangan :

namafile adalah nama dari file yang akan dibuka/diaktifkan.
mode adalah jenis operasi file yang akan dilakukan terhadap file.

2. Jenis-jenis operasi file :

r : menyarakan file hanya dapat dibaca (file harus sudah ada)
w : menyatakan file baru akan dibuat/diciptakan (file yang sudah ada akan dihapus)
a : untuk membuka file yang sudah ada dan akan dilakukan proses penambahan data (jika file belum ada, otomatis akan dibuat)
r+ : untuk membuka file yang sudah ada dan akan dilakukan proses pembacaan dan penulisan.
w+ : untuk membuka file dengan tujuan untuk pembacaan atau penulisan. Jika file sudah ada, isinya akan dihapus.
a+ : untuk membuka file, dengan operasi yang akan dilakukan berupa perekaman maupun pembacaan. Jika file sudah ada, isinya akan dihapus.

Contoh :

FILE = fopen(“COBA.TXT”, “w”);

3. Menutup File

Untuk menutup file, fungsi yang digunakan adalah **fcloseØ.**

Prototype fungsi **fclose()** ada di header file **“stdio.h”**

Bentuk Umum :

int fclose(FILE *pf);

atau

int fcloseall(void);

4. MELAKSANAKAN PROSES FILE

- **Menulis Karakter**

  Untuk menulis sebuah karakter, bentuk yang digunakan adalah :

  putc(int ch, file *fp)

  Dimana:

  fp adalah pointer file yang dihasilkan oleh **fopen()**.
  ch adalah karakter yang akan ditulis.

Contoh Program 32

```
#include <stdio.h>

main()
{
	FILE* file;
	file = fopen("arsip.txt","w");
	fprintf(file,"Selamat Menempuh Hidup Baru");
}
```

5. Membaca Karakter

- Untuk membaca karakter dari file, fungsi yang digunakan adalah :
  getc(file *fp);
  Dimana:
  fp adalah pointer file yang dihasilkan oleh **fopen()**
  Fungsi **feof()**, digunakan untuk mendeteksi akhir file.
  Pada saat membaca data foef(file *fp)

Membaca dan Menulis String

Fungsi untuk membaca dan menulis string adalah : **fgets() dan fputs()**

- Bentuk Umum :
  fgets(char *str, int p, file *fp)

  fputs(char *str, file *fp)

6. Membaca dan Menulis Blok Data

- Fungsi untuk membaca dan menulis blok data adalah : **fread() dan fwrite()** Bentuk umum :

  fread(void *buffer, int b_byte, int c, file *fp);

  fwrite(void *buffer, int b_byte, int c, file *fp);

  Keterangan :

  buffer adalah pointer ke sebuah area di memori yang menampung data yang akan dibaca dari file.
  b_byte adalah banyaknya byte yang akan dibaca atau ditulis ke file.

c adalah banyaknya item dibaca/ditulis.

Contoh Program 33

```
#include <stdio.h>

main()
{
        FILE* file;
        char teks1[100];
        file = fopen("arsip.txt","r");

        fscanf(file,"%s",teks1);

        printf("%s",teks1);
}
```

7. Membaca dan Menulis File yang Terformat

- Jika diinginkan, data bilangan dapat disimpan ke dalam file dalam keadaan terformat.
- Fungsi yang digunakan adalah :
  fprintf(ptr_file, "string control", daftar argument);

  fscanf(pts_file, "string control", daftar argument);

8. File Sequensial

File sekuensial berisi rekord-rekord data yang tidak mengenal posisi baris atau nomor rekord pada saat aksesnya, dan setiap record dapat mempunyai lebar yang berbeda-beda. Akses terhadapnya selalu dimulai dari awal file dan berjalan satu persatu menuju akhir dari file. Dengan demikian, penambahan file hanya dapat dilakukan terhadap akhir file, dan akses terhadap baris tertentu harus dimulai dari awal file.

Fungsi baku yang terkait dengan file sekuensial ini antara lain adalah **fprintf**,**fscanf,** dan **rewind** . Program berikut menyajikan penanganan file sekuensial tentang data nasabah yang berisi tiga field, yaitu nomor identitas (*account),* nama (*name),* dan posisi tabungannya (*balance)* untuk (1) menyajikan yang tabungannya bernilai nol, (2) berstatus kredit, dan (3) berstatus debet. File data tersimpan dengan nama klien.dat.

Latihan

1. Buatlah program untuk menambahkan tulisan
   BORLAND INTERNATIONAL

   Ke dalam file COBA.TXT. Caranya ubahlah mode "w" pada contoh program 1 menjadi a dan gantilah komentar "File tak dapat diciptakan!" menjadi "File tak dapat dibuka"

2. Lihatlah isi file COBA.TXT dengan perintah TYPE atau dengan menjalankan contoh
3. 'program 2. Perhatikan hasilnya.