Contoh Program Kecil
Dalam Bahasa C

Oleh :

Inggriani Liem

Program Studi Teknik Informatika

Institut Teknologi Bandung

Mei 1999

```
/* Lembar ini sengaja dibiarkan kosong */
```

DAFTAR ISI

PROGRAM C

```
/* File hello.c */
void
main()
{
      printf("hello\n ");

}
```

```
/* File : hello1.c */
/* menuliskan hello ke layar */

int
main ()
{
/* KAMUS */


/* ALGORITMA */
  printf ("hello\n");
  return 0;

}
```

ASSIGNMENT, INPUT & OUTPUT

```
/* File : ASIGN.C */
/* Assignment nilai integer dan print */
int
main ()
{/* Kamus */
  int i;
  /* Program */
  printf ("hello\n");
  i = 5;
  printf ("Ini nilai i : %d \n", i);
  return 0;
}
```

```
/* File asgdll.c */
int main()
{ /* Kamus */
  float f;
  long double fll;
/* Algoritma */
   f= 20.0f;
   fll=10.0L;
return 0;
}
```

```
/* File : ASIGNi.C */
/* Assignement dan print */
#include <limits.h>
int
main ()
{
/* Kamus */
  int i;
  long int ii;
  /* Program */
  printf ("hello\n");
  i = 1234;
  ii = 123456;
  printf ("Ini nilai i=1234  = : %d \n", i);
  printf ("Ini nilai ii=123456 : : %10d \n", ii);
/* print nilai batas integer */
  printf (“Min dan Max integer : %d, %d \n”, INT_MIN, INT_MAX);
  printf (“Max long integer : %ld, %ld \n”,  LONG_MAX);
  return 0;
}
```

```
/* File : asign2.c */
/* Deskripsi :                                                    */
/* Program ini berisi contoh sederhana untuk mendefinisikan        */
/* variabel-variabel bilangan bulat (short int, int, long int),      */
/* karakter, bilangan riil,                          */

/*-----------------------------------------------------------------*/
int
main ()
{
/* KAMUS */
  short ks = 1;
  int ki = 1;
  long kl = 10000;
  char c = 65;                /* inisialisasi karakter dengan integer */
  char c1 = 'Z';        /* inisialisasi karakter dengan karakter */
  float x = 1.55;

/* Algoritma */
/* penulisan karakter sebagai karakter */
  printf ("Karakter = %c\n", c);
  printf ("Karakter = %c\n", c1);

/* penulisan karakter sebagai integer */
  printf ("Karakter = %d\n", c);
  printf ("Karakter = %d\n", c1);


  printf ("Bilangan integer (short) = %d\n", ks);
  printf ("\t\t(int) = %d\n", ki);
  printf ("\t\t(long) = %ld\n", kl);      /* perhatikan format %ld */
  printf ("Bilangan Real = %f8.3\n", x);
  return 0;
}
```

OPERATOR & EKSPRESI

```
/* File : incr.c */
/* Effek dari operator ++ */
int
main ()
{
/* Kamus */
  int i, j;
  /* Program */
  i = 3;
  j = i++;
  printf ("Nilai i : %d\nNilai j : %d\n", ++i, j);
  return 0;
}
```

```
/* File : oper1.c */
/*  pemakaian beberapa operator terhadap bit */
int
main ()
{
/* KAMUS */
  int n = 10;                /* 1010 */
  int x = 1;                 /* 1 */
  int y = 2;                 /* 10 */
/* ALGORITMA */
  printf ("n = %d \n", n);
  printf ("x = %d \n", x);
  printf ("y = %d \n", y);
  printf ("n &  8  =  %d \n", n & 8);     /* 1010 AND 1000 */
  printf ("x & ~ 8 = %d \n", x & ~8);     /* 1  AND 0111 */
  printf ("y << 2  = %d \n", y << 2);   /* 10 ==> 1000 = 8 */
  printf ("y >> 3  = %d \n", y >>3);;  /* 10 ==> 0000 = 0 */
  return 0;
}
```

```
/* File : oprator.c */
/* Contoh pengoperasian variabel bertype dasar */
int main ()
#include "boolean.h"
{/* Kamus */
   boolean Bool1, Bool2, TF ;
   int i,j, hsl ;
   float x,y,res;
/*  algoritma */
 printf ("Utk program ini, baca teksnya dan tambahkan output");
  Bool1 = true; Bool2 = false;
  TF = Bool1 && Bool2 ;  /* Boolean AND */
  TF = Bool1 || Bool2 ;  /* Boolean OR */
  TF = ! Bool1 ;         /* NOT */
  TF = Bool1  ^Bool2;    /* XOR */
/*  operasi numerik  */
  i = 5; j = 2 ;
  hsl = i+j; hsl = i - j; hsl = i / j; hsl = i * j;
  hsl = i /j ;  /* pembagian bulat */
  hsl = i%j ; /* sisa. modulo */
/*  operasi numerik  */
  x = 5.0 ; y = 2.0 ;
  res = x + y; res = x - y; res = x / y; res = x * y;
/*  operasi relasional numerik */
  TF = (i==j); TF = (i!=j);
  TF = (i < j); TF = (i > j); TF = (i <= j); TF = (i >= j);
/*  operasi relasional numerik */
  TF = (x != y);
  TF = (x < y); TF = (x > y); TF = (x <= y); TF = (x >= y);
return 0;
}
```

```
/* File : oper2.c */
/*  pemakaian beberapa operator terhadap RELATIONAL DAN bit */
int main ()
{/* KAMUS */
  char i, j;
/* ALGORITMA */
  i = 3;                /* 00000011 dalam biner */
  j = 4;                /* 00000100 dalam biner */

  printf ("i = %d \n", i);
  printf ("j = %d \n", j);

  printf (" i && j  =  %d \n", i && j);/* 1:op lojik : true and true=> true */
  printf (" i & j   = %d \n", i & j);  /* 0: 00000000 dalam biner */
  printf (" i|| j   = %d \n", i || j); /* 1 : OR LOJIK, True or true => true */
  printf (" i| j    = %d \n", i | j);  /* 7: 00000111 biner */
  printf (" i^j     = %d \n", i ^ j);  /* 7: 00000111 biner */
  printf (" ~i      = %d \n", ~i);  /* -4: 11111100 biner */
  return 0;
}
```

Perhatikanlah operator boolean dibandingkan operator bit jika dipakai dalam ekspresi kondisional:

```
/* KAMUS */
  char i, j;
  i=3; j=4;
/* Algoritma */
  if (i&&j) {…};  /* true = true and true  */
  if (i&j) {…};   /* false = 00000000 */
  if (i||j) {…};  /* true = true and true  */
  if (i|j) {…};   /* true = 00000111 */
  if (i^j) {…};   /* true = 00000111 */
  if (~i) {…};    /* true = 11111000 */
```

```
/* File : exp.c */
/*  pemakaian  operator kondisional */
int
main ()
{
/* KAMUS */
  int x = 1;
  int y = 2;
/* ALGORITMA */
  printf ("x = %d \n", x);
  printf ("y = %d \n", y);
  printf ("hasil ekspresi  = (x<y)?x:y  =  %d \n", (x < y) ? x : y);
  return 0;
}
```

```
/* File : exp1.c */
/*  pembagian integer, casting */
int main ()
{/* KAMUS */
  int x = 1;
  int y = 2;
  float fx;
  float fy;
/* ALGORITMA */
  printf ("x/y (format integer) = %d \n", x/y);
  printf ("x/y (format float) = %f \n", x/y);
/* supaya hasilnya tidak nol */
  fx=x;
  fy=y;
  printf ("x/y (format integer) = %d \n", fx/fy);
  printf ("x/y (format float) = %f \n", fx/fy);
/* casting */
  printf ("float(x)/float(y) (format integer) = %d \n", (float)x/(float)y);
  printf ("float(x)/float(y) (format float) = %f \n", (float)x/(float)y);
  x = 10;
  y = 3;
  printf ("x/y (format integer) = %d \n", x/y);
  printf ("x/y (format float) = %f \n", x/y);
  return 0;
```

```
}
```

```
/* File : oper3.c */
/*  Operator terner */
/*  Ekspresi ditulis sebagai makro */
#define max(a,b) ((a>b) ? a: b)
int
main ()
{
/* KAMUS */
  int i = 0;  /* perhatikan int i,j=0 bukan seperti ini */
  int j = 0;
  char c = 8;
  char d = 10;
  char e = max (c, d);
  int k = max (i, j);
/* ALGORITMA */
  printf ("Nilai e = %d \n", e);
  printf ("Nilai k = %d \n", k);
  i = 2;
  j = 3;
  k = max (i++, j++);
  printf ("Nilai k = %d \n", k);
  return 0;
}
```

PEMBACAAN NILAI

```
/* File : BACA.C */
/* contoh membaca integer */
/* kemudian menuliskan nilai yang dibaca */
int main ()
{/* Kamus */
  int a;
  /* Program */
  printf ("Contoh membaca dan menulis, ketik nilai integer: ");
  scanf ("%d", &a); /* coba ketik : scanf ("%d", &a); Apa akibatnya ?*/
  printf ("Nilai yang dibaca : %d \n", a);
  return 0;
}
```

```
/* file : bacakar.c */
int main()
{/* Kamus */
char cc;
/* Algoritma */
printf ("hello\n");
printf("baca 1 kar : ");scanf ("%c ", &cc);
printf ("%c", cc);
printf ("bye \n");
return 0;
}
```

```
/* File : BACASTR.C */
/* deklarasi dan alokasi string, kemudian mengisinya dengan membaca */
int
main ()
{
/* Kamus */
  char *str;
  char *str1;
  /* Program */
  printf ("\nBaca string, maks 20 karakter: ");
  str = (char *) malloc (20 * sizeof (char));
  printf("masukkan sebuah string, max 20 kar: "); scanf("%s",str);
  printf ("String yang dibaca : %s\n", str);
  str1 = (char *) malloc (20 * sizeof (char));
  strcpy (str1, str);
  printf ("String yang disalin : %s\n", str1);
  return 0;
}
```

```
/* File : asgSTR.C */
/* deklarasi dan alokasi string, kemudian mengisinya dengan membaca */
int
main ()
{
/* Kamus */
  char *str;
  char *str1;
  /* Program */
  printf ("\nBaca string, maks 20 karakter: ");
  str = (char *) malloc (20 * sizeof (char));
  strcpy(str,"Ini string..");
  printf ("String yang diisikan : %s\n", str);
  str1 = (char *) malloc (20 * sizeof (char));
  strcpy (str1, str);
  printf ("String yang disalin : %s\n", str1);
  return 0;
}
```

KONSTANTA & KALKULASI SEDERHANA

```
/* File : KONSTANT.C */
/* Membaca jari-jari, menghitung luas lingkaran */
/* latihan pemakaian konstanta */
int main ()
{/* Kamus */
  const float pi = 3.1415;
  float r;
  /* program */
 /* baca data */
  printf ("Jari-jari lingkaran =");
  scanf ("%f", &r);
 /* Hitung dan tulis hasil */
  printf ("Luas lingkaran = %f\n", pi * r * r);
  printf ("Akhir program \n");
  return 0;
```

```
}
```

```
/* File : KONSTAN2.C */
/* Menghitung luas lingkaran, dari jari-jari yang dibaca */
/* latihan pemakaian konstanta dengan define */
#define pi 3.1415
int main ()
{/* Kamus */
  float r;
  float luas;
  /* program */
/* Baca data */
  printf ("Jari-jari lingkaran =");
  scanf ("%f", &r);
/* Hitung dengan rumus */
  luas = pi * r * r;
/* Print hasil */
  printf ("Luas lingkaran = %6.2f\n", luas);
  printf ("Akhir program \n");
  return 0;
}
```

INSTRUKSI KONDISIONAL

```
/* File : IF1.C */
/* contoh pemakaian IF satu kasus */
/* membaca nilai integer, menuliskan nilainya jika positif */
int
main ()
{
/* Kamus */
  int a;
  /* Program */
  printf ("Contoh IF satu kasus \n");
  printf ("Ketikkan suatu nilai integer ");
  scanf ("%d", &a);
  if (a >= 0)
    {
      printf ("Nilai a positif %d \n", a);
    };

  return 0;
}
```

```
/* File : IF2.C */
/* contoh pemakaian IF dua kasus komplementer */
/* Membaca sebuah nilai, */
/* menuliskan 'Nilai a positif , nilai a', jika a >=0 */
/*            'Nilai a negatif , nilai a', jika a <0  */
int
main ()
{
/* Kamus */
  int a;
  /* Program */
  printf ("Contoh IF dua kasus \n");
  printf ("Ketikkan suatu nilai integer :");
  scanf ("%d", &a);
  if (a >= 0)
    {
      printf ("Nilai a positif %d \n", a);
    }
  else /* a< 0 */
    {
      printf ("Nilai a negatif %d \n", a);
    }
  return 0;
}
```

```
/* File : IF3.C */
/* contoh pemakaian IF tiga kasus */
/* Membaca sebuah nilai, */
/* menuliskan 'Nilai a positif , nilai a', jika a >0 */
/*            'Nilai Nol , nilai a', jika a = 0  */
/*            'Nilai a negatif , nilai a', jika a <0  */
int
main ()
{
/* Kamus */
  int a;
  /* Program */
  printf ("Contoh IF tiga kasus \n");
  printf ("Ketikkan suatu nilai integer :");
  scanf ("%d", &a);
  if (a > 0)
    {
      printf ("Nilai a positif %d \n", a);
    }
  else if (a == 0)
    {
      printf ("Nilai Nol %d \n", a);
    }
  else /* a > 0 */
    {
      printf ("Nilai a negatif %d \n", a);
    }
  return 0;
}
```

```
/* File : tempair.c */
/* contoh pemakaian IF tiga kasus : wujud air */
int
main ()
{
/* Kamus : */
  int T;
  /* Program */
  printf ("Contoh IF tiga kasus \n");
  printf ("Temperatur (der. C) = ");
  scanf ("%d", &T);
  if (T < 0)
    {
      printf ("Wujud air beku %d \n", T);
    }
  else if ((0 <= T) && (T <= 100))
    {
      printf ("Wujud air cair %d \n", T);
    }
  else if (T > 100)
    {
      printf ("Wujud air uap/gas %d \n", T);
    };
  return 0;
}
```

```
/* File : KASUS.C */
/* Contoh kasus dengan switch */
#include <stdio.h>
int
main ()
{
/* Kamus */
  char cc;
  /* Program */
  printf ("Ketikkan sebuah huruf, akhiri dengan RETURN \n");
  scanf ("%s", &cc);
  switch (cc)
    {
    case 'a':
      {
      printf (" Yang anda ketik adalah a \n");
      break;
      }
    case 'u':
      {
      printf (" Yang anda ketik adalah u \n");
      break;
      }
    case 'e':
      {
      printf (" Yang anda ketik adalah e \n");
      break;
      }
    case 'i':
      {
      printf (" Yang anda ketik adalah i \n");
      break;
      }
    default:
      printf (" Yang anda ketik adalah huruf mati \n");
    }
  return 0;
}
```

```
/* File dll.c */
/* Eksrpesi kondisional dengan boolean */
#include "boolean.h"
int main()
{ /* Kamus */
   boolean bool;
/* Algoritma */
  bool= true;
  if(bool) { printf("true\n");} else printf("false\n");
  if(!bool) { printf("salah\n");} else printf("benar\n");
return 0;
}
```

```
/* File : MAX2.C */
/* Maksimum dua bilangan yang dibaca */
int
main ()
{
/* Kamus */
  int a, b;
  /* Program */
  printf ("Maksimum dua bilangan : \n");
  printf ("Ketikkan dua bilangan, pisahkan dg RETURN : \n");
  scanf ("%d %d", &a, &b);
  printf ("Ke dua bilangan : a=%d, b=%d \n", a, b);
  if (a >= b)
    {
      printf ("Nilai a yang maksimum %d \n", a);
    }
  else /* a > b */
    {
      printf ("Nilai b yang maksimum: %d \n", b);
    };
  return 0;
}
```

PENGULANGAN

```
/* File : PRIFOR.C */
/* Baca N, Print 1 s/d N dengan FOR */
int
main ()
{
/* Kamus */
  int i;
  int N;
  /* program */
  printf ("Baca N, print 1 s/d N ");
  printf ("N = ");
  scanf ("%d", &N);
  for (i = 1; i <= N; i++)
    {
      printf ("%d \n", i);
    };
  printf ("Akhir program \n");
  return 0;
}
```

```
/* File : PRIREP.C */
/* contoh baca N, */
/* print 1 s/d N dengan REPEAT */
int
main ()
{
/* Kamus : */
  int N;
  int i;
  /* Program */
  printf ("Nilai N >0 = "); /* Inisialisasi */
  scanf ("%d", &N);
  i = 1; /* First Elmt */
  printf ("Print i dengan REPEAT: \n");
  do
    {
      printf ("%d \n", i); /* Proses */
      i++;  /* Next Elmt */
    }
  while (i <= N); /* Kondisi pengulangan */
  return 0;
}
```

```
/* File : PRXREP.C */
/* contoh baca nilai x, */
/* Jumlahkan nilai yang dibaca dengan ITERATE */
int
main ()
{
/* Kamus : */
  int Sum;
  int x;
  /* Program */
  printf ("Masukkan nilai x (int), akhiri dg 999 : ");
  scanf ("%d", &x);             /* First Elmt */
  if (x == 999)
    {
      printf ("Kasus kosong \n");
    }
  else
    {                           /* MInimal ada satu data yang dijumlahkan */
      Sum = 0;                  /* Inisialisasi; invariant !! */
      do
        {
          Sum = Sum + x; /* Proses */
          printf ("Masukkan nilai x (int), akhiri dg 999 : ");
          scanf ("%d", &x);     /* Next Elmt */
        } while (x != 999);         /* Kondisi pengulangan */
      printf ("Hasil penjumlahan = %d \n", Sum);       /* Terminasi */
    }
  return 0;
}
```

```
/* file : PRITER.C */
/* Baca N, */
/* Print i = 1 s/d N dengan ITERATE */
int
main ()
{
/* Kamus : */
  int N;
  int i;
  /* Program */
  printf ("Nilai N >0 = "); /* Inisialisasi*/
  scanf ("%d", &N); /
  i = 1;  /* First Elmt */
  printf ("Print i dengan ITERATE : \n");
  for (;;)
    {
      printf ("%d \n", i); /* Proses */
      if (i == N) /* Kondisi Berhenti */
      break;
      else
      {       i++; /* Next Elmt */  }
    } /* (i == N) */
  return 0;
}
```

```
/* File : PRXITER.C */
/* contoh baca nilai x, */
/* Jumlahkan nilai yang dibaca dengan ITERATE */
int main ()
{/* Kamus : */
  int Sum;
  int x;
  /* Program */
  printf ("Masukkan nilai x (int), akhiri dg 999 : ");
  scanf ("%d", &x);             /* First Elmt */
  if (x == 999)
    {      printf ("Kasus kosong \n");
    }
  else
    {     /* MInimal ada satu data yang dijumlahkan */
      Sum = x;                  /* Inisialisasi; invariant !! */
      for (;;)
        {
            printf ("Masukkan nilai x (int), akhiri dg 999 : ");
            scanf ("%d", &x);   /* Next Elmt */
         if (x == 999)
            break;
          else
            {
              Sum = Sum + x;    /* Proses */
            }
        }
      printf ("Hasil penjumlahan = %d \n", Sum);        /* Terminasi */
    }
  return 0;
}
```

```
/* File : PRIW.C */
/* Baca N, Print i = 1 s/d N dengan WHILE */
int main ()
{/* Kamus : */
  int N;
  int i;
  /* Program */
  printf ("Nilai N >0 = "); /* Inisialisasi */
  scanf ("%d", &N);
  i = 1;  /* First Elmt */
  printf ("Print i dengan WHILE: \n");
  while (i <= N) /* Kondisi pengulangan */
    { printf ("%d \n", i); /* Proses */
      i++;                 /* Next Elmt */
    }; /* (i > N) */
  return 0;
}
```

```
/* File : PRIWHILE1.C */
/* Baca N, */
/* Print i = 1 s/d N dengan while (ringkas) */
int main ()
{/* Kamus : */
  int N;
  int i = 1;
  /* Program */
  printf ("Nilai N >0 = ");
  scanf ("%d", &N);
  printf ("Print i dengan WHILE (ringkas): \n");
  while (i <= N)
    {
      printf ("%d \n", i++);
    } /* (i > N)*/
  return 0;
}
```

```
/* File : PRXWHILE.C */
/* contoh baca nilai x, */
/* Jumlahkan nilai yang dibaca dengan WHILE */
int main ()
{/* Kamus : */
  int Sum;
  int x;
  /* Program */
  Sum = 0;  /* Inisialisasi */
  printf ("Masukkan nilai x (int), akhiri dg 999 : ");
  scanf("%d", &x);/* First Elmt*/
  while (x != 999)  /* Kondisi berhenti */
    {
      Sum = Sum + x;  /* Proses */
      printf ("Masukkan nilai x (int), akhiri dg 999 : ");
      scanf("%d", &x);/* First Elmt*/
    }
  printf ("Hasil penjumlahan = %d \n", Sum);  /* Terminasi */
  return 0;
}
```

```
/* File : FOREVER.C */
/* Loop terus menerus */
int main ()
{/* Kamus : */
#define true 1
/* Program */
  printf("Program akan looping, akhiri dengan ^c ");
  while (true)
    {      printf ("Print satu baris ....\n");
    }
  return 0;
}
```

PROSEDUR & FUNGSI

```
/* File : subprg.C */
/* Contoh program yang mengandung prosedur dan fungsi */
/*** Ada dua cara untuk deklarasi Prototype. Yang dipakai cara kedua ***/
/* int maxab(int, int);        */
/* void tukar (int*, int*);     */
int maxab (int a, int b);
void tukar (int *a, int *b);
/*** Program Utama ***/
int main ()
{/* Membaca dua bilangan integer */
/* Menuliskan maksimum dua bilangan yang dibaca dg memanggil fungsi */
/* Menukar kedua bilangan dengan 'prosedur' */
  int a, b;
  printf ("Maksimum dua bilangan : \n");
  printf ("Ketikkan dua bilangan, pisahkan dg RETURN : \n");
  scanf ("%d %d", &a, &b);
  printf ("Ke dua bilangan : a=%d, b=%d \n", a, b);
  printf ("Maksimum = %d \n", maxab (a, b));
  printf ("Tukar kedua bilangan... \n");
  tukar (&a, &b);
  printf ("Ke dua bilangan setelah tukar: a=%d, b=%d \n", a, b);
  return 0;
}

/* BODY/REALISASI prosedur/fungsi */
int
maxab (int a, int b)
{                       /* mencari maksimum dua bilangan bulat */
  return ((a >= b) ? a : b);
}

void
tukar (int *a, int *b)
{                       /* menukar dua bilangan bulat */
  int temp;
  temp = *a;
  *a = *b;
  *b = temp;
}
```

```
/* File : calljam.c */
/* memanfaatkan time.h,primitf yg disediakan bahasa C untuk manipulasi waktu */
#include <sys/time.h>
#include <stdio.h>
int
main ()
{
/* KAMUS */
/* typedef long time_t; */

/*   struct tm
   /* {  */
  /*  int     tm_sec; *//* seconds */
  /*  int tm_min;     *//* minutes */
  /*  int tm_hour;    *//* hours */
  /*  int tm_mday;    *//* day of the month */
  /*  int tm_mon;     *//* month */
  /*  int tm_year;    *//* year */
  /*  int tm_wday;    *//* day of the week */
  /*  int tm_yday;    *//* day in the year */
  /*  int tm_isdst;   *//* daylight saving time */

/* Definisi variabel yang diperlukan untuk mengambil date dan time */
  time_t now;
  struct tm *T;

/* ALGORITMA */
  printf ("Memanggil jam dan date dari sistem \n");
  now = time (NULL);
  T = localtime (&now);
  printf ("Jam = %d:%d:%d\n ", T->tm_hour, T->tm_min, T->tm_sec);
  printf ("Tgl = %d-%d-%d\n ", T->tm_mday, T->tm_mon + 1, T->tm_year);
  printf ("local time %s\n", asctime (T));
  return 0;
}
```

TYPE ENUMERASI

```
/* File : enum.c */
/* Deklarasi dan pemakaian type enumerasi */
int
main ()
{
/* KAMUS */
  enum hari  /* “type” */
    {
      senin, selasa, rabu, kamis, jumat, sabtu
    }
  hariku;  /* hariku : variabel */
  enum
    {
      satu, dua, tiga
    }
  angka;  /* variabel */
  enum
    {
      KEYWORD = 01, EXTERNAL = 03, STATIC = 04
    };   /* sekedar enumerasi “konstanta” bernama “, mengelompokkan */
  typedef enum
    {
      merah, putih, kuning
    }
  warna; /* nama type */

  unsigned int flags;
  warna w = kuning;

/* ALGORITMA */

  angka = tiga;
  printf ("Angka %d \n ", angka);

  hariku = 0;
  printf ("Hari %d \n ", hariku);

  printf ("Masukkan sebuah angka [0..2] ");
  scanf ("%d", &angka);
  printf ("Angka %d \n ", angka);

  flags = EXTERNAL;
  printf ("flags %d \n ", flags);

  printf ("Warna = %d\n", w);
  return 0;
}
```

TYPE KOMPOSISI

```
/* File : STRUK.C */
/* contoh pendefinisian dan pengisian struktur: Titik, mhs, meter */
int main ()
{/* Kamus */
/* cara 1 */
  struct
    {
      char nama[20];
      int nim;
      int nilai;
    }  Mhs;  /* nama variabel berupa struct */
  /* cara 2 */
  struct meter     /* tag, utk menyebut struct … */
    {
      int m;
      int cm;
    };
  struct meter M1; /* M1 adalah variabel */
  /* cara 3 : INI yang membuat TYPE BARU */
  typedef struct
    {
      float x;
      float y;
    }  Point; /* nama type */
  Point P1;
  Point *P2;  /* pointer ke struct, akan dibahas pada pointer */
  Point P3;             /* latihan baca */
/* Algoritma */
  printf ("Contoh mengisi struktur dengan assignment : \n");
  printf ("Titik P1, dengan P1.x dan P1.y:\n");
  P1.x = 1.1;
  P1.y = 2.5;
  printf ("P1.x = %4.1f\nP1.y = %4.1f\n", P1.x, P1.y);

  /* mengacu pointer ke struct, akan dibahas setelah pointer */
  printf ("Titik P2, dengan P2->x dan P2->y :\n");
  P2 = (Point *) malloc (sizeof (Point));
  P2->x = 9.12;
  P2->y = 2.567;
  printf ("P2.x = %f \nP2.y = %f\n", P2->x, P2->y);

  printf ("Baca Titik P3\n");
  scanf ("%f %f", &P3.x, &P3.y);
  printf ("P3.x = %f \nP3.y = %f \n", P3.x, P3.y);

  strcpy (Mhs.nama, "Juliette");
  Mhs.nim = 7473;
  Mhs.nilai = 80;
  printf ("Hasil assignment thd Mhs \n");
  printf ("Nama = %s\nNim = %d\nNilai = %d\n", Mhs.nama,
  Mhs.nim, Mhs.nilai);
  return 0;
}
```

```
/* File struk2.c */
/* contoh pendefinisian dan pengisian struktur yang mengandung string */
#include <stdlib.h>
int main ()
{
struct
    { char *nama;
      int nim;
      int nilai;
    }  Mhs;
  printf ("\nNama = ");
  Mhs.nama = (char *) malloc (20 * sizeof(char)); /* alokasi */
  gets (Mhs.nama);
  printf ("\nNIM dan nilai= ");
  scanf ("%d %d", &Mhs.nim, &Mhs.nilai);
  printf ("Hasil assignment thd Mhs \n");
  printf ("Nama = %s\nNim = %d\nNilai = %d\n", Mhs.nama, Mhs.nim, Mhs.nilai);
  return 0;
}
```

```
/* File : struk3.c */
/**** Definisi TYPE & Variabel GLOBAL */
/* contoh membedakan variabel, tag dan nama type */
typedef struct point  /* nama tag */
  {    float x;
    float y;};
typedef struct point *pointprt;  /* pointptr : nama type */
typedef struct tagx  /* nama tag */
  { int i;
    int j;
  }namat;     /* nama type */
typedef struct tagx *tpoint;
/* typedef struct namat *tnm; --> undefined type */
typedef struct
  { int k;
    int l;
  } nama; /* type */
typedef nama cc;
typedef nama *cp;
/*--> ok sebab tidak pakai TAG */
typedef int infotype;
typedef struct tElmtlist *address; /* pointer dituliskan ke “tag” */
typedef struct tElmtlist
  {infotype info;
    struct tElmtlist *next;
  } ElmtList;
ElmtList elmq;
address P;
 /* tElmtlist el;   ---> error, t Elmtlist bukan nama type, tapi nama tag */
cp pcp;
pointprt ptr;
/*** Program Utama ****/
int main ()
{  printf ("Hello, hanya coba deklarasi");
  return 0;
}
```

UNION

```
/* File union.c */
int main()
{
      /* KAMUS */
      /* Definisi Type */
      typedef struct {
            float           x;
            float           y;
      }               Point;
      typedef struct {
            Point           P1;
            Point           P2;
      }               Garis;
      typedef struct {
            Point           TopLeft;
            Point           BottomRight;
      }               BS;
      typedef  union  {
            Garis           G;
            BS              S4;
      } Gambar;

      typedef struct {
            int             jenis;
            Gambar          Gb;
      }               Geometrik;

      /* Definisi variabel */
      Point           P;
      Garis           G;
      BS              MyBS;
      Geometrik       MyGGrs1, MyGBS1;
      /* Algoritma */
      printf("hello\n ");
      /* Cara mengacu komponen */
      /* Mengisi Titik P */
      P.x = 0.0;
      P.y = 0.5;
      /* Mengisi Garis G */
      G.P1.x = 0.5;
      G.P1.y = 1.5;
      G.P2 = P;
      /* Mengisi Bujur sangkar MyBS */
      MyBS.TopLeft = P;
      MyBS.BottomRight.x = 5.0;
      MyBS.BottomRight.y = 5.0;
      /* mengisi Geometrik MyG */
      MyGGrs1.jenis = 1;      /* Garis */
      MyGGrs1.Gb.G = G;
      MyGBS1.jenis = 2; /* Bujur sangkar */
      MyGBS1.Gb.S4 = MyBS;
      return 0;
}
```

```
/* File : RecVar.c */
/* Record Varian */
int main()
/* Kamus */
/* Cell adalah sebuah sel spread sheet, yang mungkin isinya :*/
/*    Formula : string; integer atau real */
{ /* KAMUS */
typedef enum { rumus, integer, riil } Trec;
typedef union {
                    char * form ;
                    int  nili  ;
                   float nilf ; } IsiSel;

     typedef struct {
              char  adrbrs ;
              int  adrkol ;
               Trec  Tsel;
               IsiSel   Isel ;
                   } Cell;
/* variabel */
  Cell FCell, ICell, RCell;

/* Algoritma */
  /* Cara mengisi nilai */
  /* Type cell adalah formula */
  FCell.adrbrs = 'A';
  FCell.adrkol = 1;
  FCell.Tsel = rumus;

  FCell.Isel.form = (char *) malloc(15*sizeof(char));
  strcpy (FCell.Isel.form, "XYZ12");
  /* Type cell adalah integer */
  ICell.adrbrs = 'A';
  ICell.adrkol = 2;
  ICell.Tsel = integer;
  ICell.Isel.nili = 10;

/* Type cell adalah bilangan */
  RCell.adrbrs = 'A';
  RCell.adrkol = 3;
  RCell.Tsel = riil;
  RCell.Isel.nilf = 10.55;

return 0;

}
```

```
 /* File : RecVarx.c */
int main()
{

/* Record Varian dengan type bentukan*/
/* Kamus */
/* Gambar adalah bentuk yang dapat berupa garis, segi empat */

    typedef  enum  {garis, segi4} trec ;
```

```
    typedef struct {
             int   x;
             int   y ; } Point;

    typedef  struct {
             Point  Pawal ; /* titik awal */
             Point  Pakhir ; /* titik akhir*/  } TGrs;

    typedef  struct { /* Segi empat */
             Point  TopLeft ; /* Kiri atas */
              Point BottRight ; /* Kanan bawah */ } TS4;

    typedef      union {
               TGrs   G ;
               TS4  S4  ; } Geo;
    typedef struct { int id ; /* identitas gambar */
                   trec TBentuk; /* penentu bentuk gambar */
                  Geo G; } Gambar ;

/* Variabel */
 Gambar G1, G2 ;
 Gambar G3 ;

TGrs  Grs;
TS4 Segi4;

/* Algoritma */
/* Cara mengisi nilai type pembangun */
 Grs.Pawal.x = 0; Grs.Pakhir.x = 0;
 Grs.Pakhir.x = 0; Grs.Pakhir.y = 5;

Segi4.TopLeft.x = 0; Segi4.TopLeft.y = 10;
Segi4.BottRight.x = 10; Segi4.BottRight.y = 10;

  /* Cara mengisi nilai */
  /* Gambar adalah garis */
  G1.id = 1;
  G1.TBentuk = garis;
   G1.G.G.Pawal.x = 10;
  G1.G.G.Pawal.y = 10;
  G1.G.G.Pakhir.x = 10;
  G1.G.G.Pakhir.y = 10;

  /* Gambar adalah segiempat */
  G2.id = 99;
  G2.TBentuk = segi4;
  G2.G.S4.TopLeft.x = 0;
  G2.G.S4.TopLeft.y = 0;
  G2.G.S4.BottRight.x = 10;
  G2.G.S4.BottRight.y = 10;

  /******* HATI- HATI *******/
  /* Perhatikan apa yang terjadi saat kompilasi */
  /* dengan assignment berikut*/
  G3.id = 99;
  G3.TBentuk = garis;
  G3.G.S4.TopLeft.x = 0;
  G3.G.S4.TopLeft.y = 0;
  G3.G.S4.BottRight.x = 10;
  G3.G.S4.BottRight.y = 10;
  /* Komentar anda ???*/
return 0;
}
```

BITFIELD

```
/* File : bitf.c */
/* Deklarasi dan pemakaian type bitfield */
int
main ()
{
/* KAMUS */
  struct flags
    {
      unsigned int B0:1;      /* LSB */
      unsigned int B1:1;
      unsigned int B2:1;
      unsigned int B3:1;      /* MSB */
    }
  keystatus;

/* ALGORITMA */
  printf ("Keystatus.B1 : %d \n", keystatus.B1);
  if (keystatus.B1 == 1)
    {
      keystatus.B1 = 0;
    }
  else
    {
      keystatus.B1 = 1;
    }
  return 0;
}
```

POINTER

```
/* File : pointa1.c */
/* Pointer ke type dasar,mendeklarasi dan alokasi variabel dinamik */
#include <stdlib.h>
int main ()
{/* KAMUS */
  int i = 5; /* deklarasi, inisialisasi nilai variabel statik */
  int *Ptri = (int *) malloc (4);    /* deklarasi, alokasi */
  int *Ptrj = (int *) malloc (sizeof (int)); /* deklarasi, alokasi */
   float *fx=3; /* deklarasi, alokasi, inisialisasi variabel dinamik */
/* ALGORITMA */
  *Ptri = 8;  /* mendefinisikan isi */
  *Ptrj = 0;  /* mendefinisikan isi */
  printf ("Nilai yang diacu Ptri : %d \n", *Ptri);
  printf ("Nilai yang diacu Ptrj : %d \n", *Ptrj);
   return 0;
}
```

```
/* File : STRUKptr.C */
/* contoh pendefinisian struct dan pointer ke struct */
int
main ()
{
/* cara 3 : INI yang membuat TYPE BARU */
  typedef struct
    {
      float x;
      float y;
    }  Point; /* nama type */

  Point *P2;  /* deklarasi pointer ke struct */
  Point *P3=(Point *) malloc (sizeof (Point));  /* deklarasi, alokasi */

  /* Cara I:mengacu  elemen pointer ke struct */
  printf ("Titik P2, dengan P2->x dan P2->y :\n");
  P2 = (Point *) malloc (sizeof (Point));  /* Alokasi */
  P2->x = 9.12;                            /* Isi nilai komponen */
  P2->y = 2.567;

  printf ("P2.x = %f \nP2.y = %f\n", P2->x, P2->y);

  /* Cara kedua : Perhatikanlah tanda kurung, lihat prioritas () dibdk . */
  (*P3).x = 0.5; /* Mendefinisikan isi */
  (*P3).x = 10.5; /* Mendefinisikan isi */

  return 0;
}
```

```
/* File : list1.c */
/* contoh deklarasi list dan pemakaian makro */
#include <stdlib.h>
/* Definisi akses komponen type, standard kuliah Algoritma & Pemrograman */
#define info(P) (P)->info
#define next(P) (P)->next
#define Nil NULL

/** Definisi TYPE Global (sebenarnya utk soal ini tidak perlu global */
/* Elemen list linier */
typedef int infotype;
typedef struct tElmtlist *address;
typedef struct tElmtlist
  {
    infotype info;
    address next;
  } ElmtList;

/*** PROGRAM UTAMA */
int
main ()
{/* Kamus */
  address First;
  address P, Q;

  /* Create list kosong */
  First = Nil;
  /* alokasi, insert as first element */
  P = (address) malloc (sizeof (ElmtList));
  info (P) = 10;
  next (P) = Nil;
  First = P;

  /* alokasi, insert as first element */
  Q = (address) malloc (sizeof (ElmtList));
  info (Q) = 20;
  next (Q) = Nil;
  next (Q) = First;
  First = Q;
  /* alokasi, insert as first element */
  P = (address) malloc (sizeof (ElmtList));
  info (P) = 30;
  next (P) = Nil;
  next (P) = First;
  First = P;
  printf ("%d\n", info (next (next (First))));
  return 0;
}
```

ARRAY

```
/* File : tabel.c */
/* latihan array statis : mengisi dg assignment, menulis */
int
main ()
{
/* Kamus */
  int i;
  int tab[10]; /* Cara mengacu elemen ke-i: tab[i] */
  int N;
  /* Program */
  N = 5;
  printf ("Isi dan print tabel untuk indeks 1..5 \n");
  /* isi dengan assignment */
  for (i = 1; i <= N; i++)
    {
      tab[i] = i;
    };

  /* traversal: print */
  for (i = 1; i <= N; i++)
    {
      printf ("i=%d  tab[i]=%d \n", i, tab[i]);
    };

  return 0;
}
```

```
/* File : tabdin.c */
/* mendeklarasi dan alokasi variabel dinamik */
#include <stdlib.h>
int
main ()
{
/* KAMUS */
  int i;
  int *tab = (int *) malloc (10 * sizeof (int)); /* deklarasi, alokasi */
  /* Cara mengacu elemen ke-i: tab[i], atau *(tab+i) */
/* ALGORITMA */
  printf ("Tabel sudah dialokasi secara dinamik\n");
  /* mengisi dengan assignment dan menuliskan tabel */
  for (i = 0; i < 10; i++)
    {
      *(tab + i) = i;
      printf ("Nilai tabel ke %2d : %d\n", i, *tab + i);
    }
  return 0;
}
```

```
/* File : tabel1.c */
/* latihan array dinamik. Ukuran tabel ditentukan dari pembacaan */
#include <stdlib.h>
int main ()
{/* Kamus */
/* definisi tabel integer */
  int *tab; /* deklarasi array; perhatikan tanpa komentar */
            /* belum tampak bedanya dengan pointer ke integer biasa*/
  int N;
/* definisi indeks */
  int i;
 /* program */
  printf ("Contoh mengisi array dinamik berukuran 0..N : \n");
  printf (" N = ");
  scanf ("%d", &N);
  printf ("Alokasi setelah mengetahui ukuran tabel \n");
  tab = (int *) malloc ((N + 1) * sizeof (int)); /* alokasi: ukuran! */
  if (tab != NULL)
    { /* Mendefinisikan elemen tabel */
      for (i = 0; i <= N; i++)
      { *(tab + i) = i;
        printf ("i=%d  tab[i]=%d \n", i, *(tab + i));
      };
      printf ("Akhir program \n");
      /* dealloc */
      free (tab);  /* dealokasi */
      return 0;
    }
  else
    { printf ("alokasi gagal ...\n");
      return 1;
    }
}
```

```
/* File : tabel2.c */
/* latihan array statis multidimensi : mengisi dg assignment, menulis */
int main ()
{/* Kamus */
/* definisi matriks dan isi dengan agregat */
  int tab[3][4] =
  { {1, 2, 3, 4},
    {2, 3, 4, 5},
    {3, 4, 5, 6},
  };
  int i;                /* indeks baris */
  int j;                /* indeks kolom */
  /* Program */
  printf (" Print matriks 3 x 4 \n");
  /* traversal: print */
  for (i = 0; i < 3; i++)
    { for (j = 0; j < 4; j++)
      {       printf ("i,j=%d,%d  tab[i,j]=%d \n", i, j, tab[i][j]);
      }
    };
  return 0;
}
```

```
/* File : tabel3.c */
/* latihan array statis multidimensi : mengisi dg assignment, menulis */
int
main ()
{
/* Kamus */
/* definisi tabel 3 dimensi  */
  int tab[3][4][2];
/* definisi indeks */
  int i;                /* indeks baris */
  int j;                /* indeks kolom */
  int k;                /* indeks ??? */
  /* Program */
  printf ("Isi dan print tabel 3 dimensi, kemudian print \n");
  for (i = 0; i < 3; i++)
    {
      for (j = 0; j < 4; j++)
      {
        for (k = 0; k < 2; k++)
          {
            if ((i == j) && (j == k) && (i == k))
            {
              tab[i][j][k] = 1;
            }
            else
            {
              tab[i][j][k] = 0;
            }
            printf ("i,j,k=%d,%d,%d  tab[i,j,k]=%d \n", i, j, k, tab[i][j][k]);
          }
      }
    };

  return 0;
}
```

```
/* File : arrstru.c */
/* Mencoba array of a structure : array dengan elemen type terstruktur */
int main ()
{/* Kamus */
  int i;
  typedef struct
    {
      int x;
      int y;
    }  Point;
  Point P;
  Point arrp[10]; /* definisi, alokasi tabel statik */
  /* Definisikan elemen tabel */
  for (i = 0; i <= 9; i++)
    {
      arrp[i].x = i;
      arrp[i].y = i + 1;
    }
  /* tulis isi tabel */
  for (i = 0; i <= 9; i++)
    {
      printf ("\n %d/%d", arrp[i].x, arrp[i].y);
    }

  return 0;
}
```

```
/* File : tarrstr.c */
/* Array of string: pendefinisian dan pengaksesan */

#define STRING char*
int main ()
{/* KAMUS */
/* definisi array yang elemennya string, statik, dan seklaigus mengisi */
  static STRING s[3] =  {"the", "time", "is"};
/* definisi array yang elemennya string, dinamik */
  STRING (*TabStr); /* deklarasi array of string */
  int i;
/* PROGRAM */
/* print isi s */
  for (i = 0; i < 3; i++)
    {
      printf ("%s\n ", s[i]);
    }
  /* alokasi TabStr sebanyak 3 */
  TabStr = (STRING *) malloc (3 * sizeof (STRING));
  for (i = 0; i < 3; i++)
    {  /* alokasi string yang merupakan elemen tabel */
      *(TabStr + i) = (STRING)  malloc (5 * sizeof (char));
      printf ("\nInput Str[%d], maksimum 5 karakter : ", i);
      scanf ("%5s", *(TabStr + i)); /* mengisi nilai string */
      printf ("\n Nilai Str[%d] : %5s\n ", i, *(TabStr + i));
    }
  return 0;
}
```

```
/* File : arrstr.c */
/* Array of string: pendefinisian dan pengaksesan */
/* perhatikanlah permasalahannya */
int
main ()
{
/* KAMUS */
/* definisi array yang elemennya string, statik, dan sekaligus mengisi */
  static char *s[3] =  {"the", "time", "is"};
/* definisi array yang elemennya string, dinamik */
  char *(*TabStr);

int i;
/* PROGRAM */
/* print isi s */
  for (i = 0; i < 3; i++)
    {
      printf ("%s\n ", s[i]);
    }

  /* alokasi TabStr sebanyak 3 */
  TabStr = (char **) malloc (3 * sizeof (char *));
  for (i = 0; i < 3; i++)
    {
      *(TabStr + i) = (char *) malloc (5 * sizeof (char));
      printf ("\nInput Str[%d], maksimum 5 karakter : ", i);
      scanf ("%5s", *(TabStr + i));
      printf ("\n Nilai Str[%d] : %5s\n ", i, *(TabStr + i));
    }

  return 0;
}
```

```
/* File : tabparam.c */
/* tabel sebagai parameter prosedur. Perhatikan cara passing parameter aktual */
#include <stdio.h>
/* prototype */
int maxab (int a, int b);
void tukar (int *a, int *b);
void Offsettab (int *T, int Awal, int Akhir);
/* Melakukan increment terhadap setiap elemen tabel */
/* dengan indeks  Awal s/d Akhir */
void printtab (int *T, int Awal, int Akhir);
/* print isi tabel indeks Awal s/d Akhir */
int main ()
{ /* KAMUS */
  int tab[10];                    /* definisi tabel integer */
  int N = 10;
  int i;                   /* indeks */
  /* Algoritma */
  /* Mengisi tabel */
  for (i = 0; i < N; i++)
    { tab[i] = i;
      printf ("%d\n", tab[i]);    }
  /* print */
  printtab (tab, 0, N - 1);
/* passing parameter aktual berupa integer */
  tukar (&tab[1], &tab[5]);
  printf ("tab[1]= %d; tab[5] = %d\n", tab[1], tab[5]);
  printf ("Maks dari elemen 1 dibdk 10  = %d\n", maxab (tab[1], tab[2]));
  /* passing parameter input output (aktual) berupa tabel: tanpa & */
  Offsettab (tab, 0, N - 1);
  printtab (tab, 0, N - 1);
  return 0;
}
/* BODY/REALISASI prosedur/fungsi */
int maxab (int a, int b)
{            /* mencari maksimum dua bilangan bulat */
  return ((a >= b) ? a : b);
}
void tukar (int *a, int *b)
{/* menukar dua bilangan bulat */
  int temp;
  temp = *a;
  *a = *b;
  *b = temp;
}
void Offsettab (int *T, int Awal, int Akhir)
{/* Melakukan increment terhadap setiap elemen tabel */
/* dengan indeks  Awal s/d Akhir */
/* kamus lokal */
  int i;
  for (i = Awal; i <= Akhir; i++)
    { T[i] = T[i] + 1;           /* karena T[i] identik dg *T+i */
      printf ("Dalam prosedur offset T[%d] = %d\n", i, T[i]);
    }
}
void printtab (int *T, int Awal, int Akhir)
{/* print isi tabel indeks Awal s/d Akhir */
/* Kamus */
  int i;
/* Algoritma */
  for (i = Awal; i <= Akhir; i++)
    {      printf ("tab[%d] = %d\n", i, T[i]);    }
}
```

```
/* File : tabstr.c */
/* latihan array dengan def type : mengisi dg assignment, menulis */
int
main ()
{
/* Kamus */

/* definisi tabel integer */
  typedef struct
    {
      int T[10];              /* array integernya */
      int N;                  /* ukuran efektif */
    }
  tabint;
  int i;
  tabint Mytab;

  /* Program */
  printf ("Tentukan ukuran tabel, maks 10 = ");
  scanf ("%d%", &Mytab.N);

  /* isi dengan assignment */
  for (i = 0; i < Mytab.N; i++)
    {
      Mytab.T[i]  = i;
      printf ("i=%d  Mytab.T =%d \n", i, Mytab.T[i]);
    };
  return 0;
}
```

```
/* File : tabstrin.c */
/* latihan array dengan def type : mengisi dg assignment, menulis */
int
main ()
{/* Kamus */
/* definisi tabel integer */
  typedef struct
    {
      int T[10];              /* array integernya */
      int N;                  /* ukuran efektif */
    }
  tabint;
  int i;
  tabint Mytab;
  /* Program */
  printf ("Tentukan ukuran tabel, maks 10 = ");
  scanf ("%d%", &Mytab.N);
  /* isi dengan assignment */
  for (i = 0; i < Mytab.N; i++)
    {
      Mytab.T[i]  = i;
      printf ("i=%d  Mytab.T =%d \n", i, Mytab.T[i]);
    };
  return 0;
}
```

```
/* File : tabstru.c */
/* latihan array dengan def type : mengisi dg assignment, menulis */
int main ()
{
/* Kamus */

/* definisi tabel integer */
  typedef struct
    {
      int *T;                /* array integernya */
      int N;                 /* ukuran efektif */
    }  tabint;
  tabint Mytab;
  int i;
  /* Program */
  printf ("Tentukan ukuran tabel = ");
  scanf ("%d%", &Mytab.N);

  Mytab.T= (int *) malloc (Mytab.N * sizeof (int));

  /* isi dengan assignment */
  for (i = 0; i < Mytab.N; i++)
    {
      *(Mytab.T + i) = i;  /* dapat juga ditulis MyTab.T[i] ***/
      printf ("i=%d  Mytab.T =%d \n", i, *(Mytab.T + i));
    };
  return 0;
}
```

```
/* File : bacatab1.c */
/* latihan array dinamis dan statis : mengisi dg baca, menulis */
/* Latihan passing parameter tabel */
  typedef struct
    {
      int tab[10];
      int N;
    }  TabInt;

/* prototype */
void incTab (TabInt *T );     /* increment setiap element tabel */
void printTab (TabInt T);     /* print tabel */
int main ()
{/* Kamus */
  int i;
  TabInt T;
  /* Program */
  T.N = 3;
  printf ("Isi dan print tabel untuk indeks 1..5 \n");
  /* isi dari pembacaan */
  for (i = 0; i < T.N; i++)
    {
      printf ("Input Tabel ke-[%d] = ", i);
      scanf ("%d", &(T.tab[i]));
    };
  /* print : perhatikan passing parameter */
  printTab (T);
/* Increment : perhatikan passing parameter */
  incTab (&T);
  printTab (T);
  return 0;
}

/* Body prototype */
void incTab (TabInt * T)            /* increment setiap element tabel */
{     /* Kamus lokal */
  int i;
/* Algoritma */
  /* traversal: print */
  for (i = 0; i < (*T).N; i++)
    {
      (*T).tab[i] = (*T).tab[i] + 1;
    }
}

void printTab (TabInt T)            /* print tabel */
{
/* Kamus lokal */
  int i;
/* Algoritma */
  /* traversal: print */
  for (i = 0; i < T.N; i++)
    {
      printf ("T[%d]=%d \n", i, T.tab[i]);
    }
}
```

```
/* File : bacatab2.c */
/* latihan array dinamis dan statis : mengisi dg baca, menulis */
/* Latihan passing parameter tabel */
  typedef struct
    {
      int *tab;
      int N;
    }  TabInt;

/* prototype */
void incTab (TabInt *T );     /* increment setiap element tabel */
void printTab (TabInt T);     /* print tabel */
int main ()
{ /* Kamus */
  int i;
  TabInt T;
  /* Program */
  T.tab = (int *) malloc(3*sizeof(int));
  T.N = 3;
  printf ("Isi dan print tabel untuk indeks 1..5 \n");
  /* isi dari pembacaan */
  for (i = 0; i < T.N; i++)
    {
      printf ("Input Tabel ke-[%d] = ", i);
      scanf ("%d", &(T.tab[i]));
    };
  /* print : perhatikan passing parameter */
  printTab (T);
/* Increment : perhatikan passing parameter */
  incTab (&T);
  printTab (T);
  return 0;
}

/* Body prototype */
void incTab (TabInt * T)            /* increment setiap element tabel */
{     /* Kamus lokal */
  int i;
/* Algoritma */
  /* traversal: print */
  for (i = 0; i < (*T).N; i++)
    {
      (*T).tab[i] = (*T).tab[i] + 1;
    }
}
void printTab (TabInt T)            /* print tabel */
{/* Kamus lokal */
  int i;
/* Algoritma */
  /* traversal: print */
  for (i = 0; i < T.N; i++)
    {
      printf ("T[%d]=%d \n", i, T.tab[i]);
    }
}
```

```
/* File : bacatab2.c */
/* latihan array dinamis dan statis : mengisi dg baca, menulis */
/* Latihan passing paramter tabel */
  typedef struct
    {
      int *tab;
      int N;
    }  TabInt;
/* prototype */
void incTab (TabInt *T );     /* increment setiap element tabel */
void printTab (TabInt T);     /* print tabel */
int main ()
{/* Kamus */
  int i;
  TabInt T;
  /* Program */
  T.tab = (int *) malloc(3*sizeof(int));
  T.N = 3;
  printf ("Isi dan print tabel untuk indeks 1..5 \n");
  /* isi dari pembacaan */
  for (i = 0; i < T.N; i++)
    {
      printf ("Input Tabel ke-[%d] = ", i);
/*      scanf ("%d", (T.tab)+i   ); */

        scanf ("%d", & (*(T.tab +i))   );
    };
  /* print : perhatikan passing parameter */
  printTab (T);
/* Increment : perhatikan passing parameter */
  incTab (&T);
  printTab (T);
  return 0;
}
/* Body prototype */
void incTab (TabInt * T)            /* increment setiap element tabel */
{     /* Kamus lokal */
  int i;
/* Algoritma */
  /* traversal: print */
  for (i = 0; i < (*T).N; i++)
    {
      (*T).tab[i] = (*T).tab[i] + 1;
    }
}
void printTab (TabInt T)            /* print tabel */
{/* Kamus lokal */
  int i;
/* Algoritma */
  /* traversal: print */
  for (i = 0; i < T.N; i++)
    {
      printf ("T[%d]=%d \n", i, T.tab[i]);
    }
}
```

EXTERNAL FILE

```
/* File : filetxt2.c */
/* membaca teks memanfaatkan EOF untuk mengakhiri pembacaan */
/* teks dapat dibuat dengan editor teks */
/* pola pembacaan YANG DIPAKAI DI KULIAH */
#include <stdio.h>
int main ()
{ /* KAMUS */
  static char CC;  /* karakter yang sedang dibaca, Current character */
  static char FILE_NAME[63] = "pitakar.txt";
  int retval;
  FILE *fileku;
 /*  Algoritma */
  fileku = fopen (FILE_NAME, "r");
  retval = fscanf (fileku,"%c", &CC);
  while (retval != EOF)
    {
      printf ("karakter yang dibaca : %c \n", CC);
      retval = fscanf (fileku,"%c", &CC);
    }
  return 0;
}
```

```
/* File : filetxt.c */
/* membaca teks yang diakhiri titik, tidak memanfaatkan EOF */
/* pola pembacaan yang lain */
#include <fcntl.h>
int main ()
{/* KAMUS */
  static char CC;       /* karakter yang sedang dibaca, Current character */
  static char FILE_NAME[63] = "pitakar.txt";
  static int _handle;         /* file handle   */
  int retval;
  _handle = open (FILE_NAME, O_RDONLY);
  retval = read (_handle, &CC, 1);
  while (CC != '.')
    {
      printf ("karakter yang dibaca : %c \n", CC);
      retval = read (_handle, &CC, 1);
    }
  return 0;
}
```

```
/* File : filetxt1.c */
/* membaca teks , diakhiri dengan EOF */
#include <fcntl.h>
int main ()
{/* KAMUS */
  static char CC;       /* karakter yang sedang dibaca,current character */
  static char FILE_NAME[63] = "pitakar.txt";
  static int _handle;         /* file handle   */
  int retval;
/* ALGORITMA */
  _handle = open (FILE_NAME, O_RDONLY);
  retval = read (_handle, &CC, 1);  /* baca 1 byte */
  while (retval == 1)         /* berhasil baca 1 byte */
    {
      printf ("karakter yang dibaca : %c \n", CC);
      retval = read (_handle, &CC, 1);
    }
  return 0;
}
```

```
/* fileint.c */
/* membuat dan membaca kembali file yang setiap rekamannya adalah integer */
#include <stdio.h>
int main ()
{/* KAMUS */
  int i;
  int rek;
  int retval;
  FILE *fileku;
  /* ALGORITMA */
  /* Membuat */
  fileku = fopen ("filein.dat", "w");
  for (i = 1; i < 5; i++)
    { /* perhatikan format penulisan */
      retval = fprintf (fileku, "%d ", i);
    }
  printf ("\nSelesai membuat ...");
  fclose (fileku);
  /* Membaca kembali */
  fileku = fopen ("filein.dat", "r");
  retval = fscanf (fileku, "%d", &rek); /* harus sama dg ketika dibuat*/
  printf ("%d ", rek);
  while (retval != EOF)
    {
      printf ("Yang dibaca : %d \n", rek);
      retval = fscanf (fileku, "%d ", &rek);
    }
  printf ("\nbye...");
  fclose (fileku);
  return 0;
}
```

```
/* filerek.c */
/* membuat dan membaca kembali sebuah file yang rekamannya type terstruktur  */
#include <stdio.h>
int main ()
{
  typedef
  struct
    {
      int NIM;
      float Nilai;
    }
  TMhs;
  int i;
  TMhs Mhs;
  int retval;
  FILE *fileku;
  /* ALGORITMA */
  fileku = fopen ("filerek.dat", "w");
  for (i = 1; i < 5; i++)
    {
      Mhs.NIM = i;
      Mhs.Nilai = i * 0.8;
      retval = fprintf (fileku, "%d %f", Mhs.NIM, Mhs.Nilai);
      printf ("retval %d", retval);
    }
  printf ("\n selesai membuat...");
  fclose (fileku);

  fileku = fopen ("filerek.dat", "r");
  retval = fscanf (fileku, "%d %f", &Mhs.NIM, &Mhs.Nilai);
  printf ("retval %d \n", retval);
  while (retval != EOF)
    {
      printf ("Yang dibaca : %d  %f \n", Mhs.NIM, Mhs.Nilai);
      retval = fscanf (fileku, "%d %f", &Mhs.NIM, &Mhs.Nilai);
      printf ("retval %d \n", retval);
    }
  printf ("\nbye...");
  fclose (fileku);
  return 0;
}
```

```
/* filekoma.c */
/* membaca text file */
/* separator data adalah koma */
#include <stdio.h>
int
main ()
{
/* kamus */
  int x;
  char *cc[10];
  int retval;
  /* program */
  FILE *fileku;
  fileku = fopen ("koma.dat", "r");
  retval = fscanf (fileku, " %d,%s ", &x, cc); /* cc krn pointer*/
  while (retval != EOF)
    {
      printf ("\n%d,%s", x,cc);
      retval = fscanf (fileku, "%d,%s ", &x, cc);

    }
  printf ("\nbye...");
  fclose (fileku);
  return 0;
}
```

Isi koma.dat

```
1,data
2, mydata
3, dataku
```

```
/* filedat.c */
/* membaca dan menulis text file  */
/* separator adalah blank */
/* Data berupa string tidak boleh mengandung blank!!! */
#include <stdio.h>
int main ()
{/* Kamus */
  int n;
  char nama[21];
  float persen;
  int retval;
  FILE *fileku;
/* ALgoritma */
  fileku = fopen ("filedat.txt", "r");
  retval = fscanf (fileku, "%d %s %f", &n, nama, &persen);
  while (retval != EOF)
    {
      printf ("Data : %d %s %f \n", n, nama, persen);
      retval = fscanf (fileku, "%d %s %f", &n, nama, &persen);
    }
  printf ("\nbye...");
fclose (fileku);
  return 0;
}
```

Isi filedat.txt

```
1 Namaku 10.2
  2 Namamu 67.4
```

PROGRAM DALAM BEBERAPA MODUL

Modul berikut adalah untuk manipulasi jam :

File	Deskripsi isi
jam.h	Type dan prototype Jam
jam.c	Realisasi (body) dari jam.h
mjam.c	Main program untuk mentest beberapa fungsi/prosedur pada jam.c

```
/* File : jam.h */
/* deklarasi TYPE dan prototype type jam */
#ifndef jam_H
#define jam_H

typedef struct
  {
    int HH;
    int MM;
    int SS;
  }
jam;

/* prototype */
void ResetJam (jam * J);
/* Mengisi sebuah jam J dengan 00:00:00 */

void TulisJam (jam J);
/* menulis sebuah jam */

int JamToDetik (jam J);
/* konversi jam ke detik */

jam DetikToJam (int d);
/* konversi dari detik menjadi jam */
#endif
```

```
/* File : jam.c */
/* Body prototype type jam */

#include "jam.h"

/* BODY prototype */
void
ResetJam (jam * J)
/* Mengisi sebuah jam J dengan 00:00:00 */
{                        /* KAMUS LOKAL */

/* ALGORITMA */
  (*J).HH = 0;
```

```
  (*J).MM = 0;
  (*J).SS = 0;
}

void TulisJam (jam J)
/* menulis sebuah jam */
{                       /* KAMUS LOKAL */

/* ALGORITMA */
  printf ("Jam : %2d:%2d:%2d\n", J.HH, J.MM, J.SS);
}

int
JamToDetik (jam J)
/* konversi jam ke detik */
{                       /* KAMUS LOKAL */

/* ALGORITMA */
  return (J.HH * 3600 + J.MM * 60 + J.SS);
}

jam
DetikToJam (int d)
/* konversi dari detik ke struktur jam */
{                       /* KAMUS LOKAL */
  jam J;
  int sisa;

/* ALGORITMA */
  J.HH = d / 3600;
  sisa = d % 3600;
  J.MM = sisa / 60;
  J.SS = sisa % 60;
  return J;

}
```

```
/* File : mjam.c */
/* memanfaatkan primitif jam  */
#include "jam.h"
int
main ()
{/* KAMUS */
  jam J1;
  jam J2;
  int dt=1000;
/* PROGRAM */
  printf ("hello\n");
  ResetJam (&J1);
  TulisJam (J1);
  printf("Konversi jam ke detik: %d\n",JamToDetik(J1));
  J2=DetikToJam(dt);
  TulisJam(J2);
  return 0;
}
```

Modul berikut adalah untuk manipulasi jam dengan representasi type yang berbda. Perhatikanlah kode jam1.c dan bandingkanlah dengan jam.c.
Program utama untuk mentest tidak diberikan dalam catatan ini.

File	Deskripsi isi
jam1.h	Type dan prototype Jam (definisi type berbeda dengan jam.h)
jam1.c	Realisasi (body) dari jam1.h

```
/* File : jam1.c */
/* Body proto type jam dengan representasi am/pm   */
#include "jam1.h"
/* BODY prototype */
void
ResetJam (jam * J)
/* Mengisi sebuah jam J dengan 00:00:00 */
{                       /* KAMUS LOKAL */

/* ALGORITMA */
  (*J).HH = 0;
  (*J).MM = 0;
  (*J).SS = 0;
  (*J).pasi = pm;
}

void TulisJam (jam J)
/* menulis sebuah jam */
{                       /* KAMUS LOKAL */

/* ALGORITMA */
  if (J.pasi == pm)
    {
      printf ("Jam : %2d:%2d:%2d PM\n", J.HH, J.MM, J.SS);
    }
  else
    {
      printf ("Jam : %2d:%2d:%2d AM\n", J.HH, J.MM, J.SS);
    };
}

int JamToDetik (jam J)
/* konversi jam ke detik */
{                       /* KAMUS LOKAL */

/* ALGORITMA */

  if (J.pasi == pm)
    {
      return (J.HH * 3600 + J.MM * 60 + J.SS);
    }
  else
    {
      return ((J.HH + 12) * 3600 + J.MM * 60 + J.SS);
    };
}
```

```
jam DetikToJam (int d)
/* konversi dari detik ke struktur jam */
{                        /* KAMUS LOKAL */
  jam J;
  int sisa;

/* ALGORITMA */
  J.HH = d / 3600;
  if (J.HH > 12)
    {
      J.HH = J.HH - 12;
      J.pasi = pm;
    }
  else
    {
      J.pasi = am;
    };
  sisa = d % 3600;
  J.MM = sisa / 60;
  J.SS = sisa % 60;
  return J;
}
```

MACRO KONDISIONAL

Program utama jamm.c berikut memanfaatkan modul jam atau jam1, dengan makro kondisional

```
/* File : jamm.c */
/* memanfaatkan primitif jam1 atau jam dengan macro kondisional */
/* tuliskan #define REP untuk memakai jam.h */
#define REP
#ifdef (REP)
#include "jam.h"
#else
#include "jam1.h"
#endif
int main ()
{/* KAMUS */
  jam J1;
  jam J2;
  int dt=1000;
/* PROGRAM */
  printf ("hello\n");
  ResetJam (&J1);
  TulisJam (J1);
  printf("Konversi jam ke detik: %d\n",JamToDetik(J1));
  J2=DetikToJam(dt);
  TulisJam(J2);
  return 0;
}
```

PROGRAM DENGAN PARAMETER

```
/* File : prgparam.c */
/* Program yang mengandung parameter */

#include <stdio.h>
int
main(int argc, char *argv[])
{
        /* Kamus */
        int             x;
        FILE           *FXINT;
        const int       AKHIR = 9999;

        /* Algoritma */

        /* periksa nama file yang diberikan */
        /* pada saat pemanggilan */
        if (argc < 2) {
                fprintf(stderr, "sintaks : %s <nama-file>\n", argv[0]);
                exit(1);
        } else {
                /* Buka file untuk penulisan nilai yang dibaca */
                FXINT = fopen(argv[1], "w");
                if (!FXINT) {
fprintf(stderr, "kesalahan dalam membuka file %s\n", argv[1]);
                        exit(1);
                }
        }

        /* pengulangan untuk membaca bilangan diakhiri dengan 9999 */
        do {
                printf("Masukkan nilai integer, akhiri dengan %d : ", AKHIR);
                scanf("%d", &x);
                fprintf(FXINT, "%d\n", x);
        } while (x != AKHIR);
        fclose(FXINT);
        return 0;
}
```

POINTER TO FUNCTION

```
/* File : pointf.c */
/* Pointer ke function */
/* prototype */
void f1 (void);
void f2 (void);
void f3 (void);
void f4 (void);

int main ()
{
/* KAMUS LOKAL */
#define true 1
#define false 0
/* DEklarasi variabel */
  int quit = false;

/* ALGORITMA */
  printf ("Pointer to function \n");
/* Menu * */
  do
    {
      printf ("Pilih salah satu :\n");
      printf ("1. Buka File hanya untuk baca \n");
      printf ("2. Tutup file \n");
      printf ("3. Edit File \n");
      printf ("4. Quit\n");
      switch (getchar ())
      {
      case '1':
        f1 ();
        break;
      case '2':
        f2 ();
        break;
      case '3':
        f3 ();
        break;
      case '4':
        f4 ();
        quit = true;
        break;
      default:
        printf ("Pilihan di luar yang ditentukan...\n");
        break;
      }
      getchar ();       /* untuk membuang return */
    }
  while (quit != true);

  return 0;
}
```

```
/* BODY FUNGSI */
void f1 ()
{
  printf ("Ini Fungsi F1 \n");
}

void f2 ()
{
  printf ("Ini Fungsi F2 \n");
}

void f3 ()
{
  printf ("Ini Fungsi F3 \n");
}

void f4 ()
{
  printf ("Quit... \n");
}
```

```
/* File : pointf1.c */
/* Pointer ke function */
/* prototype */
void f1 (void);
void f2 (void);
void f3 (void);
void f4 (void);
/* KAMUS GLOBAL */
#define true 1
#define false 0
int quit = false;
int main ()
{/* KAMUS LOKAL */
/* Definisi tabel yang elemennya adalah pointer ke fungsi */
/* Elemen tabel yang ke-i akan mengakses fungsi ke-i */
/* Pilihan menjadi Indeks tabel, dan dipakai untuk mengaktifkan fungsi yang */
/* sesuai */
  void (*tab[4]) () =   {f1, f2, f3, f4};
  printf ("Pointer to function \n");
/* Menu * */
  do
    {
      printf ("Pilih salah satu :\n");
      printf ("1. Buka File hanya untuk baca \n");
      printf ("2. Tutup file \n");
      printf ("3. Edit File \n");
      printf ("4. Quit\n");
      tab[getchar () - '1'] ();
      getchar ();/* membuang return */
    }
  while (!quit);
  return 0;
}

/* BODY FUNGSI */
void f1 ()
{
  printf ("Ini Fungsi F1 \n");
}

void f2 ()
{
  printf ("Ini Fungsi F2 \n");
}

void f3 ()
{
  printf ("Ini Fungsi F3 \n");
}

void f4 ()
{
  quit = true;
  printf ("Quit... \n");
}
```

```
/* File : pointf2.c */
/* Pointer ke function */
/* function sebagai parameter */
/* Melakukan offset terhadap tabel tergantung fungsi f */
/* KAMUS GLOBAL */
/* prototype */

typedef struct
  {
    int T[10];                /* isi tabel */
    int N;              /* ukuran efektif */
  }
TabInt;

/* fungsi yang diberikan sebagai parameter aktual */
int succ (int i);       /* suksesor */
int pred (int i);       /* predesesor */

/* prosedur dengan parameter sebuah fungsi */
void geser (TabInt * TT, int (*f) (int));

/* PROGRAM UTAMA */
int main ()
{
/* KAMUS LOKAL */
  TabInt MyTab;
  int i;

  MyTab.N = 10;
  for (i = 0; i < MyTab.N; i++)
    {
      MyTab.T[i] = i;
    }
  printf ("Nilai asal tabel dalam main \n");
  for (i = 0; i < MyTab.N; i++)
    {
      printf (" %d ", MyTab.T[i]);
    }
  printf ("\n");
  /* Pakai geser dengan parameter succ */
  printf ("Geser dengan succ \n");
  geser (&MyTab, succ);
  printf ("dalam main \n");
  for (i = 0; i < MyTab.N; i++)
    {
      printf (" %d ", MyTab.T[i]);
    }
  printf ("\n");

  printf ("Nilai tabel dalam main setelah aplikasi succ \n");
  for (i = 0; i < MyTab.N; i++)
    {
      printf (" %d ", MyTab.T[i]);
    }
  printf ("\n");
```

```
  /* Pakai geser dengan parameter pred */
  printf ("Geser dengan pred \n");
  geser (&MyTab, pred);
  printf ("dalam main setelah aplikasi pred \n");
  for (i = 0; i < MyTab.N; i++)
    {
      printf (" %d ", MyTab.T[i]);
    }
  printf ("\n");
  return 0;
}
   /* BODY FUNGSI */
int succ (int i)
{
  return (i + 1);
}

int pred (int i)
{
  return (i - 1);
}

void geser (TabInt * TT, int (*f) (int))
{
  int i;

  printf ("dalam geser \n");
  for (i = 0; i < (*TT).N; i++)
    {
      (*TT).T[i] = f ((*TT).T[i]);
      printf (" %d ", (*TT).T[i]);
    }
  printf ("\n");
}
```

```
/* File : pointf3.c */
/* Pointer ke function , function sebagai parameter */
/* Melakukan offset terhadap tabel tergantung fungsi f */
/* KAMUS GLOBAL */
int N;                          /* ukuran efektif */

/* fungsi yang diberikan sebagai parameter aktual */
int succ (int i);       /* suksesor */
int  pred (int i);              /* predesesor */

/* prosedur dengan parameter sebuah fungsi */
void geser (int *TT, int (*f) (int ));

/* PROGRAM UTAMA */
int main ()
{/* KAMUS LOKAL */
  int MyTab[10];
  int i;
/* Algoritma */
  N = 10;
```

```
  for (i = 0; i < N; i++)
    {
      MyTab[i] = i;
    }
  printf ("Isi tabel dalam main sebelum pemanggilan \n");
  for (i = 0; i < N; i++)
    {
      printf (" %d ", MyTab[i]);
    }
  printf ("\n");
  geser (MyTab, succ);
  printf ("dalam main \n");
  for (i = 0; i < N; i++)
    {
      printf (" %d ", MyTab[i]);
    }
  printf ("\n");
  return 0;
}
   /* BODY FUNGSI */
int succ (int i)
{
  return (i + 1);
}

int pred (int i)
{
  return ( i - 1);
}

void geser (int *TT, int (*f) (int ))
{
  int i;
  printf ("dalam geser \n");
  for (i = 0; i < N; i++)
    {
      TT[i] =  f (TT[i]);
      printf (" %d ", TT[i]);
    }
  printf ("\n");
}
```

```
/* File : pointf4.c */
/* Pointer ke function */
/* Prosedur dengan parameter input/output sebagai parameter */
/* Melakukan offset terhadap tabel tergantung fungsi f */
/* KAMUS GLOBAL */
  int N;                      /* ukuran efektif */
/* prototype */
/* fungsi yang diberikan sebagai parameter aktual */
void succ (int *i);           /* suksesor */
void pred (int *i);           /* predesesor */
/* prosedur dengan parameter sebuah fungsi */
void geser (int *TT, void (*f) (int*));
/* PROGRAM UTAMA */
int main ()
{/* KAMUS LOKAL */
  int MyTab[10];
  int i;
  N = 10;
  for (i = 0; i < N; i++)
    {
      MyTab[i] = i;
    }
  printf ("Isi tabel dalam main sebelum pemanggilan \n");
  for (i = 0; i < N; i++)
    {
      printf (" %d ", MyTab[i]);
    }
  printf ("\n");
  geser (MyTab, succ);
  printf ("dalam main \n");
  for (i = 0; i < N; i++)
    {
      printf (" %d ", MyTab[i]);
    }
  printf ("\n");
  return 0;
}
 /* BODY FUNGSI */
void succ (int *i)
{
  *i = *i + 1;
}
void pred (int *i)
{
  *i = *i - 1;
}
void geser (int *TT, void (*f) (int *))
{  int i;
   printf ("dalam geser \n");
  for (i = 0; i < N; i++)
    {
      f (&TT[i]);
      printf (" %d ", TT[i]);
    }
  printf ("\n");
}
```

```
/* File : pointf5.c */
/* Pointer ke function */
/* Prosedur dengan parameter input/output sebagai parameter */
/* Melakukan offset terhadap tabel tergantung fungsi f */
/* KAMUS GLOBAL */
int N;                        /* ukuran efektif */
enum MyType
  {
    bulat, karakter
  };
/* prototype */
/* fungsi yang diberikan sebagai parameter aktual */
void succI (int *i);          /* suksesor */
void predI (int *i);          /* predesesor */
void succC (char *i);         /* suksesor */
void predC (char *i);         /* predesesor */
void printtab (void *T);      /* print tabel yang belum ketahuan typenya */
/* prosedur dengan parameter sebuah fungsi yang argumennya belum jelas */
void geser (int *TT, void (*f) (void *));

/* PROGRAM UTAMA */
int main ()
{
/* KAMUS LOKAL */
  void *MyTabInt;
  void *MyTabC;
  int i;

  /* Algoritma */
  N = 10;
  MyTabInt = (int *) malloc (N * sizeof (int));
  MyTabC = (char *) malloc (N * sizeof (char));
   *MyTabInt= 1 ;
  for (i = 0; i < N; i++)
    {
      *(MyTabInt + i) = i;
      *(MyTabC + i) =  'Z';
    }
  printf ("Isi tabel dalam main sebelum pemanggilan \n");
  printf ("table integer \n ");
  printtab ((int *) MyTabInt);

  printf ("table character \n ");
  printtab ((char *) MyTabC);
  printf ("\n");
  geser ((int *) MyTabInt, (int *) succI);
  geser ((char *) MyTabC, (char *) succC);
  printf ("dalam main \n");
  printf ("table integer \n ");
  printtab ((int *) MyTabInt);
  printf ("table character \n ");
  printtab ((char *) MyTabC);
  printf ("\n");
  return 0;
}
```

```
 /* BODY FUNGSI */
void succI (int *i)
{
  *i = *i + 1;
}

void predI (int *i)
{
  *i = *i - 1;
}
void succC (char *c)
{
  *c = *c + 1;
}
void predC (char *c)
{
  *c = *c - 1;
}
void geser (void *TT, void (*f) (void *))
{ /* Kamus */
  int i;
/* Algoritma */
  printf ("dalam geser \n");
  for (i = 0; i < N; i++)
    {
      f (&TT[i]);
      printf (" %d ", TT[i]);
    }
  printf ("\n");
}
void printtab (void *T, enum MyType Ty)
{
  for (i = 0; i < N; i++)
    {
      switch (Ty)
      {
      case bulat:
        printf ("%d ", (int *) *(T + i));
        break;
      case karakter:
        printf ("%c ", (char *) *(T + i));
        break;
      }
    }
}
```

SCOPE & LIFETIME

```
/* File : scope.c */
/* Scope and lifetime */
/* prototype */
void Pros1 (int, int, int *);
int fungsi1 (void);

/* KAMUS GLOBAL */
char CC;

int main ()
{/* KAMUS LOKAL */
  char CX;
  int i, j, k, h;
  float x, y;
  int ix;
  int idx;
/* ALGORITMA */
  CC = 'X';
  CX = CC;
  i = 2;
  j = 3;
  k = 1;
  Pros1 (i, j, &h);
  printf ("Nilai k pada main = %d\n", k);
  printf ("Nilai h pada main = %d\n", h);

  for (idx = 0; idx < 6; idx++)
    {
      printf ("Hasil fungsi : %d\n", fungsi1 ());
    }

  if (i < j)
    {
      int ii = 1;
      printf ("ii dideklarasi dalam blok if\n");
      printf ("Nilai ii= %d\n", ii);
    }
  /* SALAH: tidak dikenal : printf ("Nilai ii= %d\n", ii); */

  printf ("i = %d \n", i);
  return 0;
}

void Pros1 (int i, int j, int *h)
{/* KAMUS LOKAL */
  int k;

/* ALGORITMA */
  k = i + j;
  printf ("Nilai k pada pros1 = %d\n", k);
  *h = i + j;
  printf ("Nilai *h pada pros1 = %d\n", *h);
}
```

```
int fungsi1 ()
{
/* KAMUS LOKAL */
  int i=0;
  static int j = 0;
/* ALGORITMA */
  i= i+ 1;
  j = j + 1;
  printf ("Nilai j pada  fungsi1 = %d\n", j);
  printf ("Nilai i pada  fungsi1 = %d\n", i);
  return j;
}
```

```
/* File : blok.h */
/* Contoh deklarasi extern */
#ifndef blok_H
#define blok_H
typedef struct
  {
    int HH;
    int MM;
    int SS;
  }
jam;

/* prototype */
void ResetJam (jam * J);
/* Mengisi sebuah jam J dengan 00:00:00 */

void TulisJam (jam J);
/* menulis sebuah jam */

int JamToDetik (jam J);
/* konversi jam ke detik */

jam DetikToJam (int d);
/* konversi dari detik menjadi jam */

#endif
```

```
/* File : blok.c */
/* deklarasi dan prototype type jam */
/* dan CurrentJam bertype jam yang akan menjadi variabel global, */
/* dideklarasi extern pada main program */
#include "blok.h"

/* VARIABEL GLOBAL */
jam CurrentJam;

/* BODY prototype */
void ResetJam (jam * J)
/* Mengisi sebuah jam J dengan 00:00:00 */
```

```
{/* KAMUS LOKAL */

/* ALGORITMA */
  (*J).HH = 0;
  (*J).MM = 0;
  (*J).SS = 0;
}

void TulisJam (jam J)
/* menulis sebuah jam */
{                       /* KAMUS LOKAL */

/* ALGORITMA */
  printf ("Jam : %2d:%2d:%2d\n", J.HH, J.MM, J.SS);
}

int JamToDetik (jam J)
/* konversi jam ke detik */
{     /* KAMUS LOKAL */

/* ALGORITMA */
  return (J.HH * 3600 + J.MM * 60 + J.SS);
}

jam DetikToJam (int d)
/* konversi dari detik ke struktur jam */
{                       /* KAMUS LOKAL */
  jam J;
  int sisa;
/* ALGORITMA */
  J.HH = d / 3600;
  sisa = d % 3600;
  J.MM = sisa / 60;
  J.SS = sisa % 60;
  return J;
}
```

```
/* File : blokmain.c */
/* memanfaatkan primitif jam  */
#include "blok.h"
int main ()
{/* KAMUS */
  jam J1;
  jam J2;
  int dt=1000;
  extern jam CurrentJam;
/* PROGRAM */
  printf ("hello\n");
  ResetJam (&CurrentJam);

  ResetJam(&J1);
  TulisJam (J1);
  printf("Konversi jam ke detik: %d\n",JamToDetik(J1));
  J2=DetikToJam(dt);
  TulisJam(J2);
  return 0;
}
```

boolean.h

```
/*File : boolean.h */
#ifndef boolean_H
#define boolean_H
#define true 1
#define false 0
#define boolean unsigned char
#endif
```

Mesin Karakter

```
/* File : mesinkar1.h  */
/* Versi 1 : dengan pembacaan sederhana */
#ifndef MESINKAR1_H
#define MESINKAR1_H
#include "boolean.h"
#ifndef MARK
#define MARK '.'
#endif

void START(void);
/*I.S. sembarang */
/*F.S. CC adalah karakter pertama pita */
/*     Jika CC==MARK, EOP menyala (true) */
/*     Jika CC != MARK, EOP padam (false) */

void ADV(void);
/*I.S. CC != MARK */
/*F.S. CC adalah karakter berikutnya dari CC pada I.S. */
/*     Jika CC==MARK, EOP menyala (true) */

boolean EOP(void);
/* true jika CC == MARK */

#endif
```

```
/* File : mesinkar1.c  */
/* Body mesinkar1.h */
#ifndef MESINKAR_C
#define MESINKAR_C
#include <assert.h>
#include <fcntl.h>
#include <stdio.h>
#include "boolean.h"
#include "mesinkar1.h"

/* definisi states */
char CC;
```

```
/* definisi pita */
static char Pita_karakter[63] ="pitakar.txt";
static FILE * FILEKU;
static int  retval;

void START(void) {
/*I.S. sembarang */
/*F.S. CC adalah karakter pertama pita */
/*     Jika CC==MARK, EOP menyala (true) */
/*     Jika CC != MARK, EOP padam (false) */
 FILEKU = fopen(Pita_karakter,"r" );
  retval = fscanf(FILEKU,"%c",&CC) ;
}
void ADV(void) {
/*I.S. CC != MARK */
/*F.S. CC adalah karakter berikutnya dari CC pada I.S. */
/*     Jika CC==MARK, EOP menyala (true) */
  retval = fscanf(FILEKU,"%c",&CC) ;
 if (CC==MARK) {
     fclose(FILEKU);
}}
boolean EOP() {
/* true jika CC == MARK */
  return (CC==MARK);
}
#endif
```

```
/* file : mainkar.c */
/* driver mesinkar1.* /
/* Menuliskan isi pita ke layar */
#include "mesinkar1.h"
#include "boolean.h"
/* Kamus global */
extern char CC;

int main()
{/* Kamus lokal */

/* ALgoritma : menuliskan isi pita */
  printf("awal pita \n");
  START();
  while (!EOP()) {
     printf("%c",CC);
     ADV ();
  }
  printf ("\n... akhir, Bye \n");
  return 0;
}
```

```
/* File : mesinkar.h  */
```

```
/* versi kedua : dengan asersi dan primitif READ */
#ifndef MESINKAR_H
#define MESINKAR_H
#include <assert.h>
#include <fcntl.h>
#include "boolean.h"
#ifndef MARK
#define MARK '.'
#endif
void START(void);
/*I.S. sembarang */
/*F.S. CC adalah karakter pertama pita */
/*     Jika CC==MARK, EOP menyala (true) */
/*     Jika CC != MARK, EOP padam (false) */
void ADV(void);
/*I.S. CC != MARK */
/*F.S. CC adalah karakter berikutnya dari CC pada I.S. */
/*     Jika CC==MARK, EOP menyala (true) */
boolean EOP(void);
/* true jika CC == MARK */
#endif
```

```
/* File : mesinkar.c  */
```

```
/* Body mesinkar.h */
#ifndef MESINKAR_C
#define MESINKAR_C
#include <assert.h>
#include <fcntl.h>
#include <stdio.h>
#include "boolean.h"
#include "mesinkar.h"
/* definisi states */
char CC;
/* definisi pita */
#define  Pita_karakter  "pitakar.txt"
static int _handle;
void START(void) {
/*I.S. sembarang */
/*F.S. CC adalah karakter pertama pita */
/*     Jika CC==MARK, EOP menyala (true) */
/*     Jika CC != MARK, EOP padam (false) */
 _handle = open(Pita_karakter,O_RDONLY );
 assert(_handle !=-1);
 assert(read(_handle,&CC,1) !=-1);
}
void ADV(void) {
/*I.S. CC != MARK */
/*F.S. CC adalah karakter berikutnya dari CC pada I.S. */
/*     Jika CC==MARK, EOP menyala (true) */
 assert (!EOP());
 assert(read(_handle,&CC,1) !=-1);
 if (CC==MARK) assert (close(_handle)!=-1);
}
boolean EOP() {
/* true jika CC == MARK */
  return (CC==MARK); }
#endif
```