Pemrograman Bahasa C dengan Turbo C

Achmad Solichin
Sh-001@plasa.com

Bab VII
Fungsi

1 PENGERTIAN FUNGSI

Fungsi merupakan suatu bagian dari program yang dimaksudkan untuk mengerjakan suatu tugas tertentu dan letaknya terpisah dari program yang memanggilnya. Fungsi merupakan elemen utama dalam bahasa C karena bahasa C sendiri terbentuk dari kumpulan fungsi-fungsi. Dalam setiap program bahasa C, minimal terdapat satu fungsi yaitu fungsi main(). Fungsi banyak diterapkan dalam program-program C yang terstruktur. Keuntungan penggunaan fungsi dalam program yaitu program akan memiliki struktur yang jelas (mempunyai readability yang tinggi) dan juga akan menghindari penulisan bagian program yang sama.

Dalam bahasa C fungsi dapat dibagi menjadi dua, yaitu fungsi pustaka atau fungsi yang telah tersedia dalam Turbo C dan fungsi yang didefinisikan atau dibuat oleh programmer.

2 BEBERAPA FUNGSI PUSTAKA DALAM BAHASA C

- **Fungsi Operasi String (tersimpan dalam header file "string.h")**
  - **strcpy()**
    - ☑ Berfungsi untuk menyalin suatu string asal ke variable string tujuan.
    - ☑ Bentuk umum : strcpy(var_tujuan, string_asal);
  - **strlen()**
    - ☑ berfungsi untuk memperoleh jumlah karakter dari suatu string.
    - ☑ Bentuk umum : strlen(string);

Contoh Program :

```
#include "stdio.h"
#include "conio.h"
#include "string.h"
void main()
{   char nama[25];
    strcpy(nama, "Achmad Solichin");
    printf("Nama : %s", nama);
    printf("Banyaknya karakter nama Anda adalah : %i", strlen(nama));
    getch();
}
```

- **strcat()**
  - ☑ Digunakan untuk menambahkan string sumber ke bagian akhir dari string tujuan.
  - ☑ Bentuk umum : strcat(tujuan, sumber);

- **strupr()**
  - ☑ Digunakan untuk mengubah setiap huruf dari suatu string menjadi huruf capital.
  - ☑ Bentuk umum : strupr(string);
- **strlwr()**
  - ☑ Digunakan untuk mengubah setiap huruf dari suatu string menjadi huruf kecil semua.
  - ☑ Bentuk umum : strlwr(string);

Contoh Program :

```
#include "stdio.h"
#include "conio.h"
#include "string.h"
void main()
{   char satu[30] = "Fakultas Teknologi Informasi";
    char dua[30] = "Universitas Budi Luhur";
    clrscr();
    strcat(satu, dua);
    printf("Hasil penggabungannya : %s\n", satu);
    printf("Jika diubah menjadi huruf kapital semua :\n");
    printf("%s", strupr(satu));
    printf("Jika diubah menjadi huruf kecil semua :\n");
    printf("%s", strlwr(satu));
    getch();
}
```

- **strcmp()**
  - ☑ Digunakan untuk membandingkan dua buah string.
  - ☑ Hasil dari fungsi ini bertipe integer dengan nilai :
    - (a) Negative, jika string pertama kurang dari string kedua.
    - (b) Nol, jika string pertama sama dengan string kedua
    - (c) Positif, jika string pertama lebih besar dari string kedua.
  - ☑ Bentuk umum : strcmp(string1, string2);

Contoh Program :

```
#include "stdio.h"
#include "conio.h"
#include "string.h"
```

```
void main()
{   char string1[5], string2[5];
    int hasil;
    clrscr();

    printf("Masukkan string 1 : "); scanf("%s", &string1);
    printf("Masukkan string 2 : "); scanf("%s", &string2);
    hasil = strcmp(string1, string2);
    if(hasil > 0)
        printf("%s > %s", string1,string2);
    else
        if(hasil == 0)
            printf("%s = %s", string1, string2);
        else
            printf("%s < %s", string1, string2);
    getch();
}
```

- **Fungsi Operasi Karakter (tersimpan dalam header "ctype.h")**
  - **islower()**
    - ☑ Fungsi akan menghasilkan nilai benar (bukan nol) jika karakter merupakan huruf kecil.
    - ☑ Bentuk umum : islower(char);
  - **isupper()**
    - ☑ Fungsi akan menghasilkan nilai benar (bukan nol) jika karakter merupakan huruf kapital.
    - ☑ Bentuk umum : isupper(char);
  - **isdigit()**
    - ☑ Fungsi akan menghasilkan nilai benar (bukan nol) jika karakter merupakan sebuah digit.
    - ☑ Bentuk umum : isdigit(char);
  - **tolower()**
    - ☑ Fungsi akan mengubah huruf capital menjadi huruf kecil.
    - ☑ Bentuk umum : tolower(char);
  - **toupper()**
    - ☑ Fungsi akan mengubah huruf kecil menjadi huruf kapital.
    - ☑ Bentuk umum : toupper(char);

Contoh Program :

```
#include "stdio.h"
#include "conio.h"
#include "ctype.h"
void main()
{   char karakter;
    clrscr();
    printf("Masukkan sebuah karakter : ");  karakter = getche();
        if(isupper(karakter))     //periksa apakah "karakter" adalah huruf kapital
    {           puts(" adalah huruf besar");
                printf("Huruf kecilnya adalah : %c", tolower(karakter));
    }
    else
        if(islower(karakter))     //periksa apakah "karakter" adalah huruf kecil
        {   puts(" adalah huruf kecil");
            printf("Huruf besarnya adalah : %c", toupper(karakter));
        }
```

```
        else
            if(isdigit(karakter))  //periksa apakah "karakter" adalah digit
              puts(" adalah karakter digit");
            else
              puts(" bukan huruf besar, huruf kecil atau digit");

    getch();
}
```

- **Fungsi Operasi Matematik (tersimpan dalam header "math.h" dan "stdlib.h")**
  - **sqrt()**
    - ☑ Digunakan untuk menghitung akar dari sebuah bilangan.
    - ☑ Bentuk umum : sqrt(bilangan);
  - **pow()**
    - ☑ Digunakan untuk menghitung pemangkatan suatu bilangan.
    - ☑ Bentuk umum : pow(bilangan, pangkat);

Contoh Program :

```
#include "stdio.h"
#include "conio.h"
#include "math.h"
void main()
{   int x, y;
    float z;
    clrscr();
    printf("Menghitung x pangkat y\n");
    printf("x = "); scanf("%i", &x);
    printf("y = "); scanf("%i", &y);
    printf(" %i dipangkatkan dengan %i adalah %7.2lf", x, y, pow(x, y));
    getch();

    clrscr();
    printf("Menghitung akar suatu bilangan z\n");
    printf("z = "); scanf("%f", &z);
    printf("Akar dari %f adalah %7.2lf", z, sqrt(z));
    getch();
}
```

- **sin(), cos(), tan()**
  - ☑ Masing-masing digunakan untuk menghitung nilai sinus, cosinus dan tangens dari suatu sudut.
  - ☑ Bentuk umum :

```
sin(sudut);
cos(sudut);
tan(sudut);
```

Contoh Program :

```
#include "stdio.h"
#include "conio.h"
#include "math.h"
void main()
{   float sudut;
    clrscr();
    printf("Menghitung nilai sinus, cosinus dan tangens\n");
    printf("Masukkan sudut : "); scanf("%f", &sudut);
    printf("Nilai sinus %.2f derajat adalah %.3f", sudut, sin(sudut));
```

```
    printf("Nilai cosinus %.2f derajat adalah %.3f", sudut, cos(sudut));
    printf("Nilai tangens %.2f derajat adalah %.3f", sudut, tan(sudut));
    getch();
}
```

- **atof()**
  - ☑ Digunakan untuk mengkonversi nilai string menjadi bilangan bertipe double.
  - ☑ Bentuk umum : atof(char x);
- **atoi()**
  - ☑ Digunakan untuk mengkonversi nilai string menjadi bilangan bertipe integer.
  - ☑ Bentuk umum : atoi(char x);

Contoh Program :

```
#include "stdio.h"
#include "conio.h"
#include "math.h"
void main()
{   char x[4] = "100", y[5] ="10.3";
    int a;
    float b;
    clrscr();
    a = atoi(x);  b = atof(y);
    printf("Semula A = %s B = %s\n", x,y);
    printf("Setelah dikonversi A = %i B = %.2f", a,b);
    getch();
}
```

- **div()**
  - ☑ Digunakan untuk menghitung hasil pembagian dan sisa pembagian.
  - ☑ Bentuk umum : **div_t div(int x, int y)**
  - ☑ Strukturnya :

```
typedef struct
{  int qout;          // hasil pembagian
   int rem            // sisa pembagian
} div_t;
```

Contoh Program :

```
#include "stdio.h"
#include "conio.h"
#include "stdlib.h"
void main()
{   int x, y;
    div_t hasil;
    clrscr();
    printf("Menghitung sisa dan hasil pembagian x dengan y\n");
    printf("x = "); scanf("%i", &x);
    printf("y = "); scanf("%i", &y);
    hasil = div(x,y);
    printf("\n\n %3i div %3i = %3i sisa %3i", x, y, hasil.quot, hasil.rem);
    getch();
}
```

- **max()**
  - ☑ Digunakan untuk menentukan nilai maksimal dari dua buah bilangan.
  - ☑ Bentuk umum : max(bilangan1, bilangan2);

- **min()**
  - ☑ Digunakan untuk menentukan bilangan terkecil dari dua buah bilangan.
  - ☑ Bentuk umum : min(bilangan1, bilangan2);

Contoh Program :

```
#include "stdio.h"
#include "conio.h"
#include "stdlib.h"
void main()
{   int x, y, z;
    clrscr();

    printf("Menentukan bilangan terbesar dan terkecil\n");
    printf("X = "); scanf("%i", &x);
    printf("Y = "); scanf("%i", &y);
    printf("Z = "); scanf("%i", &z);
    printf("\nBilangan terbesar : %i", max(max(x, y), z));
    printf("\nBilangan terkecil : %i", min(min(x, y), z));
    getch();
}
```

3 MEMBUAT FUNGSI SENDIRI

Deklarasi Fungsi

Sebelum digunakan (dipanggil), suatu fungsi harus dideklarasikan dan didefinisikan terlebih dahulu. Bentuk umum pendeklarasian fungsi adalah :

```
tipe_fungsi nama_fungsi(parameter_fungsi);
```

Sedangkan bentuk umum pendefinisian fungsi adalah :

```
Tipe_fungsi nama_fungsi(parameter_fungsi)
{   statement
    statement
    ……...
    ……...
}
```

Hal-hal yang perlu diperhatikan dalam penggunaan fungsi :

- Kalau tipe fungsi tidak disebutkan, maka akan dianggap sebagai fungis dengan nilai keluaran bertipe integer.
- Untuk fungsi yang memiliki keluaran bertipe bukan integer, maka diperlukan pendefinisian penentu tipe fungsi.
- Untuk fungsi yang tidak mempunyai nilai keluaran maka dimasukkan ke dalam tipe void
- Pernyataan yang diberikan untuk memberikan nilai akhir fungsi berupa pernyataan return.
- Suatu fungsi dapat menghasilkan nilai balik bagi fungsi pemanggilnya.

Contoh Program 1 :

```
#include "stdio.h"
#include "conio.h"
float tambah(float x, float y);              /* prototype fungsi tambah(), ada titik koma */
void main()
{   float a, b, c;
    clrscr();
    printf("A = ");  scanf("%f", &a);
```

```
    printf("B = ");  scanf("%f", &b);
    c = tambah(a, b);            /* pemanggilan fungsi tambah() */
    printf("A + B = %.2f", c);
    getch();
}

float tambah(float x, float y)    /* Definisi fungsi , tanpa titik koma */
{   return (a+b);                 /* Nilai balik fungsi */
}
```

Contoh Program 2 :

```
/* Program menghitung nilai factorial */
#include "stdio.h"
#include "conio.h"
long int faktorial(int N);              /* prototype fungsi factorial() */

void main()
{   int N;
    long int fak;

    printf("Berapa factorial ? "); scanf("%i", &N);
    fak = faktorial(N);          /* pemanggilan fungsi factorial() */
    printf("%i factorial = %ld\n", N, fak);
    getch();
}

long int faktorial(int N)        /* definisi fungsi factorial */
{   int I;
    long int F = 1;
    if(N<=0)
    return(0);
    for(I=2; I<=N; I++)
    F = F * I;
    return(F);
}
```

- **Parameter Formal dan Parameter Aktual**
  - **Parameter Formal** adalah variabel yang ada pada daftar parameter dalam definisi fungsi.
  - **Parameter Aktual** adalah variabel (parameter) yang dipakai dalam pemanggilan fungsi.

  Dalam contoh program pertambahan di atas parameter formal terdapat pada pendefinisisan fungsi :

```
float tambah(float x, float y)  //parameter formal
{  return (a+b);
}
```

Sedangkan parameter aktual terdapat pada pemanggilan fungsi :

```
void main()
{  ……………
   ……..
   c = tambah(a, b);            //parameter aktual
   ……..
}
```

- **Cara Melewatkan Parameter**

  Cara melewatkan suatu parameter dalam Bahasa C ada dua cara yaitu :

  - Pemanggilan Secara Nilai (*Call by Value*)
    - ☑ Call by value akan menyalin nilai dari parameter aktual ke parameter formal.
    - ☑ Yang dikirimkan ke fungsi adalah nilai dari datanya, bukan alamat memori letak dari datanya.
    - ☑ Fungsi yang menerima kiriman nilai akan menyimpannya di alamat terpisah dari nilai aslinya yang digunakan oleh bagian program yang memanggil fungsi.
    - ☑ Perubahan nilai di fungsi (parameter formal) tidak akan merubah nilai asli di bagian program yang memanggilnya.
    - ☑ Pengiriman parameter secara nilai adalah pengiriman searah, yaitu dari bagian program yang memanggil fungsi ke fungsi yang dipanggil.
    - ☑ Pengiriman suatu nilai dapat dilakukan untuk suatu ungkapan, tidak hanya untuk sebuah variabel, elemen array atau konstanta saja.

    Contoh Program :

```
#include "stdio.h"
#include "conio.h"
void tukar(int x, int y);              /* pendeklarasian fungsi */

void main()
{   int a,b;
    clrscr();
    a = 15;
    b = 10;
    printf("Nilai sebelum pemanggilan fungsi\n");
    printf("a = %i b = %i\n\n", a, b);     // a dan b sebelum pemanggilan fungsi

    tukar(a,b);            /* pemanggilan fungsi tukar() */

    printf("Nilai setelah pemanggilan fungsi\n");
    printf("a = %i b = %i\n\n", a, b);     // a dan b setelah pemanggilan fungsi
    getch();
}
void tukar(int x, int y)  /* Pendefinisian fungsi tukar() */
{   int z;                             /* variabel sementara */
    z = x;
    x = y;
    y = z;
    printf("Nilai di akhir fungsi tukar()\n");
    printf("x = %i y = %i\n\n", x, y);
}
```

  - Pemanggilan Secara Referensi (*Call by Reference*)
    - ☑ Pemanggilan secara Referensi merupakan upaya untuk melewatkan alamat dari suatu variabel ke dalam fungsi.
    - ☑ Yang dikirimkan ke fungsi adalah alamat letak dari nilai datanya, bukan nilai datanya.
    - ☑ Fungsi yang menerima kiriman alamat ini makan menggunakan alamat yang sama untuk mendapatkan nilai datanya.
    - ☑ Perubahan nilai di fungsi akan merubah nilai asli di bagian program yang memanggil fungsi.
    - ☑ Pengiriman parameter secara referensi adalah pengiriman dua arah, yaitu dari fungsi pemanggil ke fungsi yang dipanggil dan juga sebaliknya.
    - ☑ Pengiriman secara acuan tidak dapat bdilakukan untuk suatu ungkapan.

Contoh Program :

```
#include "stdio.h"
#include "conio.h"
void tukar(int *px, int *py);

void main()
{   int a,b;
    clrscr();
    a = 15;
    b = 10;
    printf("Nilai sebelum pemanggilan fungsi\n");
    printf("a = %i b = %i\n\n", a, b);

    tukar(&a,&b);                    /* parameter alamat a dan alamat 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;              /* variabel sementara */
    z = *px;
    *px = *py;
    *py = z;
    printf("Nilai di akhir fungsi tukar()\n");
    printf("*px = %i *py = %i\n\n", *px, *py);
}
```

- **Penggolongan Variabel berdasarkan Kelas Penyimpanan (Storage Class)**
  - Variabel lokal
    Variabel lokal adalah variabel yang dideklarasikan di dalam fungsi.
    Sifat-sifat variabel lokal :
    - ☑ Secara otomatis akan diciptakan ketika fungsi dipanggil dan akan lenyap ketika proses eksekusi terhadap fungsi berakhir.
    - ☑ Hanya dikenal oleh fungsi tempat variabel dideklarasikan
    - ☑ Tidak ada inisialisasi secara otomatis (saat variabel diciptakan nilainya random).
    - ☑ Dideklarasikan dengan menambahkan kata "**auto**" (opsional).

  - Variabel global (eksternal)
    Variabel global (eksternal) adalah variabel yang dideklarasikan di luar fungsi.
    Sifat-sifat variabel global :
    - ☑ Dikenal (dapat diakses) oleh semua fungsi.
    - ☑ Jika tidak diberi nilai awal secara otomatis berisi nilai nol.
    - ☑ Dideklarasikan dengan menambahkan kata "**extern**" (opsional).

Contoh Program :

```
#include "stdio.h"
#include "conio.h"
void tampil(void);
int i = 25;                      /* variabel global */
void main()
{   clrscr();
    printf("Nilai variabel i dalam fungsi main() adalah %i\n\n", i);
```

```
    tampil();
    i = i * 4;                    /* nilai i yang dikali 4 adalah 25 (global) bukan 10  */
    printf("\nNilai variabel i dalam fungsi main() sekarang adalah %i\n\n", i);

    getch();
}

void tampil(void)
{   int i = 10;             /* variabel lokal */
   printf("Nilai variabel i dalam fungsi tampil() adalah %i\n\n", i);
}
```

- Variabel Statis
  Variabel statis adalah variabel yang nilainya tetap dan bisa berupa variabel lokal (internal) dan variabel global (eksternal).
  Sifat-sifat variabel statis :
  - ☑ Jika bersifat internal (lokal), maka variabel hanya dikenal oleh fungsi tempat variabel dideklarasikan.
  - ☑ Jika bersifat eksternal (global), maka variabel dapat dipergunakan oleh semua fungsi yang terletak pada program yang sama.
  - ☑ Nilai variabel statis tidak akan hilang walau eksekusi terhadap fungsi telah berakhir.
  - ☑ Inisialisasi hanya perlu dilakukan sekali saja, yaitu pada saat fungsi dipanggil pertama kali.
  - ☑ Jika tidak diberi nilai awal secara otomatis berisi nilai nol.
  - ☑ Dideklarasikan dengan menambahkan kata "**static**".

- Variabel Register
  Variabel Register adalah variabel yang nilainya disimpan dalam resister dan bukan dalam memori RAM.
  Sifat-sifat variabel register :
  - ☑ Hanya dapat diterapkan pada variabel lokal yang bertipe int dan char.
  - ☑ Digunakan untuk mengendalikan proses perulangan (looping).
  - ☑ Proses perulangan akan lebih cepat karena variabel register memiliki kecepatan yang lebih tinggi dibandingkan variabel biasa.
  - ☑ Dideklarasikan dengan menambahkan kata "**register**".

Contoh Program :

```
#include "stdio.h"
#include "conio.h"
void main()
{   register int x;                /* variabel register */
    int jumlah;
    clrscr();
    for(i=1; i<=100; i++)
    jumlah = jumlah + I;
    printf("1+2+3+…+100 = %i\n", jumlah);
    getch();
}
```

- **Fungsi Rekursif**
  Fungsi rekursif adalah fungsi yang memanggil dirinya sendiri.

Contoh Program :

```
#include "stdio.h"
```

```
#include "conio.h"
long int faktorial(int N);                    /* prototype fungsi faktorial */
void main()
{    int N;

     printf("Berapa factorial ? "); scanf("%i", &N);

     printf("Faktorial dari %i = %ld\n", N, faktorial(N));
     getch();
}

long int faktorial(int N)          /* definisi fungsi factorial */
{
     if(N==0)
        return(1);
     else
        return(N * faktorial(N - 1)); // fungsi faktorial() memanggil fungsi factorial()
}
```

LATIHAN 7

1. Buat fungsi untuk menentukan apakah suatu bilangan bulat bersifat ganjil atau genap. Jika genap maka fungsi menghasilkan nilai 1, dan 0 untuk selainnya.
2. Buatlah fungsi menjumlahkan bilangan 1,2,3, ….., n secara rekursif.
3. Buatlah Program untuk menghitung jarak maksimum (xmax) dan ketinggian maksimum (hmax) dari sebuah peluru yang ditembakkan dengan sudut elevasi A. Anggap g = 10 m/s2 (Gunakan fungsi sin() dan cos())