Pemrograman Dasar

Pertemuan IX

Rekursi

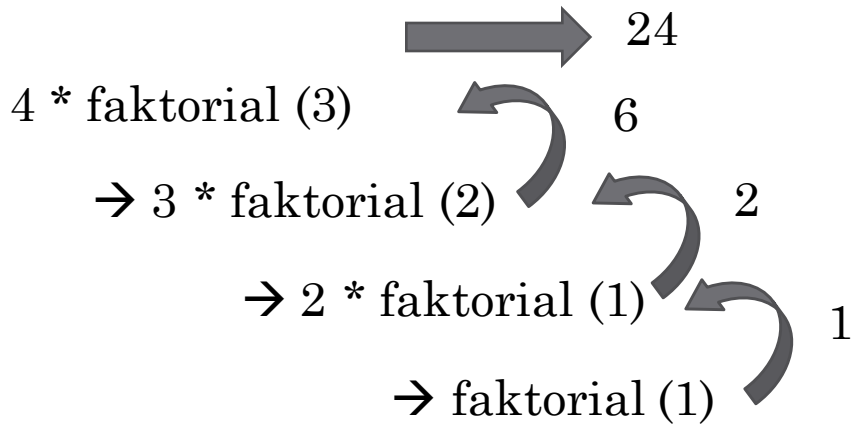
- Rekursi adalah suatu kemampuan subrutin untuk memanggil dirinya sendiri.
- Pada beberapa kondisi atau persoalan, kemampuan memanggil diri sendiri tersebut **akan sangat berguna** karena mempermudah solusi.
- Kelemahan rekursi satu-satunya adalah terjadi stack overflow (stack tidak mampu lagi menangani permintaan pemanggilan rekursif karena telah kehabisan memori).

  → *Stack : area memori / RAM yang dipakai untuk variabel lokal dan untuk pengalokasian memori ketika suatu subrutin dipanggil*

- Cara menangani kelemahan tersebut adalah : programmer harus memastikan bahwa proses rekursi akan selesai pada suatu titik / kondisi tertentu.

Contoh Konsep Rekursif

- Persoalan rekursif biasa dijumpai pada matematika → contoh : menghitung faktorial
- Faktorial m!
  m! = 1 x 2 x 3 x … x m
- $m!\ \begin{cases} 1, jika\ m=0\ atau\ m=1 \\ m * ((m-1)!), jika\ m>0 \end{cases}$
- Misal :
  Faktorial (4) diperoleh dari :



Implementasi Faktorial dengan Rekursi

- Algoritma :

```
SUBRUTIN faktorial (n)
    JIKA n=0 ATAU 1 MAKA
        NILAI-BALIK 1
     SEBALIKNYA
        NILAI-BALIK n * faktorial (n-1)
      AKHIR-JIKA
 AKHIR-SUBRUTIN
```

- Algoritma :

```
#include <stdio.h>

long int faktorial (int n)
{
        if (n == 0 || n == 1)
                return 1;
        else
                return n * faktorial (n-1);
}

int main ()
{
        int bil, n;
        long int hasil;

        printf("n = ");
        scanf("%d", &n);

        hasil = faktorial (n);
        printf("%d! = %ld", n, hasil);

        return 0;
} // fakt.c
```

Contoh Rekursif – 1

- Skema Fibonacci
  fib(n) = 0 untuk n = 0
  fin(n) = 1 untuk n = 1
  fib(n) = fib(n-1) + fib(n-2), untuk n > 1

Fibonacci :
n = 0 → fib(n) = 0
n = 1 → fib(n) = 1
n = 2 → fib(n) = 1
n = 3 → fib(n) = 2
n = 4 → fib(n) = 3
n = 5 → fib(n) = 5
n = 6 → fib(n) = 8
n = 7 → fib(n) = 13
n = 8 → fib(n) = 21
n = 9 → fib(n) = 35
…

Contoh Rekursif – 1 (Lanjutan)

```
ALGORITMA

SUBRUTIN fib(n)
    JIKA n=0 MAKA
        NILAI-BALIK 0
    SEBALIKNYA JIKA n = 1 MAKA
        NILAI-BALIK 1
    SEBALIKNYA
        NILAI-BALIK fib(n-1) + fib(n-2)
    AKHIR-JIKA
AKHIR-SUBRUTIN
```

- Algoritma :

```
#include <stdio.h>

long int fib (unsigned int n)
{
        if (n == 0) return 0;
        else if (n ==1) return 1;
        else return fib (n-1) + fib (n-2);
}

int main ()
{
        int bil, n;
        long int hasil;

        printf("n = ");
        scanf("%d", &n);

        hasil = fib (n);
        printf("fib(%d) = %ld", n, hasil);

        return 0;
} // fib.c
```

Contoh Rekursif – 2

- Pemangkatan bilangan Y^n, dengan n > 0, dapat dilakukan dengan rekursif menggunakan pola sbb
  - $Y^n = Y$, untuk n = 1
  - $Y^n = Y \times Y^{n-1}$ untuk n > 1
- ALGORITMA :

```
SUBRUTIN pangkat (y,n)
        JIKA n=1 MAKA
                NILAI-BALIK y
        SEBALIKNYA
                NILAI-BALIK y * pangkat(y, n-1)
        AKHIR-JIKA
AKHIR-SUBRUTIN
```

- Algoritma :

```
#include <stdio.h>

long int pangkat (unsigned int y, unsigned int n)
{
        if (n == 1) return y;
        else if (n ==1) return y * pangkat (y, n-1);
}

int main ()
{
        int y, n;
        long int hasil;

        printf("y = ");
        scanf("%d", &y);

        printf("n = ");
        scanf("%d", &n);

        hasil = pangkat(y, n);
        printf("%d ^ %d = %ld", y, n, hasil);

        return 0;
} // pangkat.c
```

Contoh Rekursif – 3

- Buatlah rekursif subrutin untuk membalik sebuah bilangan. Misal :
  Input : 7895
  Output : 5987

- ALGORITMA :
  SUBRUTIN balik (bil)
  tampilkan (sisaPembagian(n, 10))
  digitTersisaDiKiri ← bil / 10
  JIKA digitTersisaDiKiri != 0 MAKA
  balik(digitTersisaDiKiri)
  AKHIR-JIKA
  AKHIR-SUBRUTIN

- Algoritma :

```
#include <stdio.h>

void balik (long int bil)
{
        long int digitTersisaDiKiri;
        printf("%d", bil % 10);
        digitTersisaDiKiri = bil / 10;
        if (digitTersisaDiKiri != 0)
                balik(digitTersisaDiKiri);
}

int main ()
{
        int bil;

        printf("bilangan bulat = ");
        scanf("%d", &bil);

        balik(bil);
        return 0;
} //balik.c
```

*rekursif berupa void → tidak ada nilai-balik

Contoh Rekursif – 4

- FUNGSI ACKERMANN
- Diciptakan oleh Wilhelm Ackermann – 1928
- Ackermann Function merupakan fungsi pertama dan sederhana dari total computable function
- Ackermann Function sangat populer, karena dapat menghasilkan bilangan yang sangat-sangat besar bahkan dari inputan yang kecil

$$A(m,n) = \begin{cases} n+1 & \text{if } m = 0 \\ A(m-1,1) & \text{if } m > 0 \text{ and } n = 0 \\ A(m-1, A(m,n-1)) & \text{if } m > 0 \text{ and } n > 0. \end{cases}$$

- Bahkan dengan A(4,2), hasilnya adalah 19,729
- Untuk A(4,3) hasilnya adalah $2^{2^{65536}} - 3$

$$\begin{aligned} A(1,2) &= A(0, A(1,1)) \\ &= A(0, A(0, A(1,0))) \\ &= A(0, A(0, A(0,1))) \\ &= A(0, A(0,2)) \\ &= A(0,3) \\ &= 4. \end{aligned}$$

$$\begin{aligned} A(4,3) &= A(3, A(4,2)) \\ &= A(3, A(3, A(4,1))) \\ &= A(3, A(3, A(3, A(4,0)))) \\ &= A(3, A(3, A(3, A(3,1)))) \\ &= A(3, A(3, A(3, A(2, A(3,0))))) \\ &= A(3, A(3, A(3, A(2, A(2,1))))) \\ &= A(3, A(3, A(3, A(2, A(1, A(2,0)))))) \\ &= A(3, A(3, A(3, A(2, A(1, A(1,1)))))) \\ &= A(3, A(3, A(3, A(2, A(1, A(0, A(1,0))))))) \\ &= A(3, A(3, A(3, A(2, A(1, A(0, A(0,1))))))) \\ &= A(3, A(3, A(3, A(2, A(1, A(0,2)))))) \\ &= A(3, A(3, A(3, A(2, A(1,3))))) \\ &= A(3, A(3, A(3, A(2, A(0, A(1,2)))))) \\ &= A(3, A(3, A(3, A(2, A(0, A(0, A(1,1))))))) \\ &= A(3, A(3, A(3, A(2, A(0, A(0, A(0, A(1,0)))))))) \\ &= A(3, A(3, A(3, A(2, A(0, A(0, A(0, A(0,1)))))))) \\ &= A(3, A(3, A(3, A(2, A(0, A(0, A(0,2))))))) \\ &= A(3, A(3, A(3, A(2, A(0, A(0,3)))))) \\ &= A(3, A(3, A(3, A(2, A(0,4))))) \\ &= A(3, A(3, A(3, A(2,5)))) \\ &= \dots \\ &= A(3, A(3, A(3,13))) \\ &= \dots \\ &= A(3, A(3,65533)) \\ &= \dots \\ &= A(3, 2^{65536} - 3) \\ &= \dots \\ &= 2^{2^{65536}} - 3. \end{aligned}$$

Contoh Rekursif – 4 (Lanjutan)

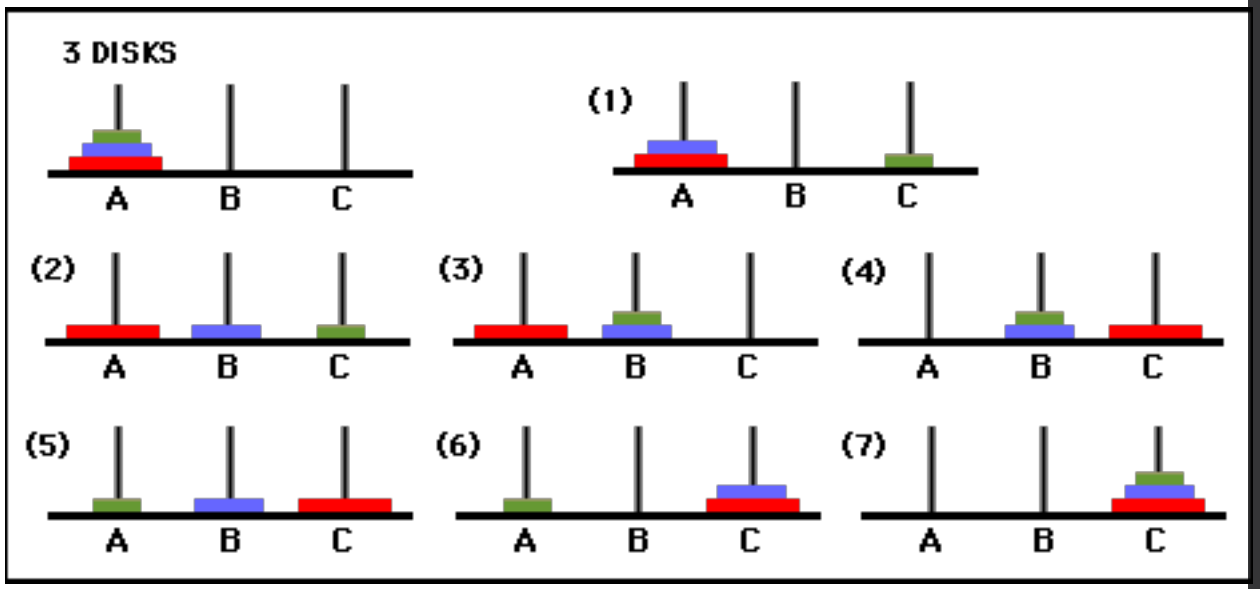
```
ALGORITMA :
SUBRUTIN acker(m,n)
   JIKA m=0 MAKA
         NILAI-BALIK n+1
    SEBALIKNYA JIKA n = 0 MAKA
         NILAI-BALIK acker(m-1,1)
    SEBALIKNYA
         NILAI-BALIK acker(m-1, acker(m, n-1))
     AKHIR-JIKA
AKHIR-SUBRUTIN
```

- Algoritma :

```
#include <stdio.h>

long int acker (int m, int n)
{
        if (m == 0) return n+1;
        else if (n == 0)
              return acker (m-1, 1);
        else
              return acker (m-1, acker (m, n-1));
}

int main ()
{
        printf("%ld\n", acker(0,5));

         printf("%ld\n", acker(1,0));

         printf("%ld\n", acker(3,1));

        return 0;
} // acker.c
```

Contoh Rekursif – 5

- Problem Tower of Hanoi / Menara Hanoi
- Adalah persoalan klasik untuk memindahkan susunan piring dari suatu tonggak ke tonggak lain, dengan bantuan sebuah tonggak perantara dan tetap mematuhi aturan.
- Ketentuan :
  1. Dalam satu waktu, hanya satu piring saja yang boleh dipindah
  2. Ketika sebuah piring dipindahkan, piring tersebut harus langsung diletakkan pada salah satu dari ketiga tonggak
  3. Setiap piring harus diletakkan di atas piring yang ukurannya lebih besar



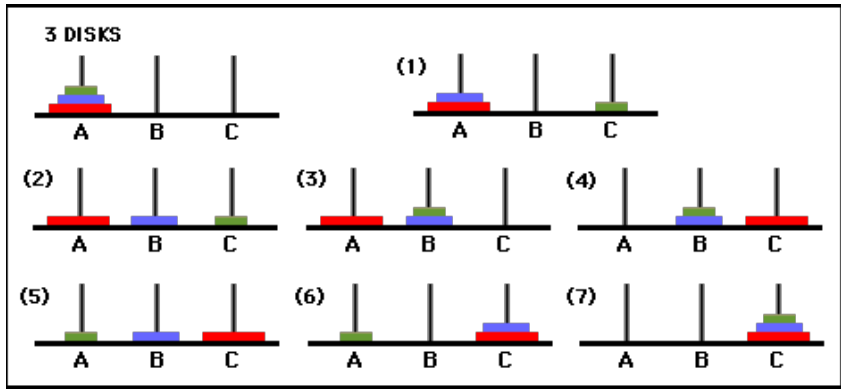
Tower of Hanoi : Memindah susunan piring dari A ke C

Contoh Rekursif 5 - Lanjutan

- Penyelesaian Rekursif :
  1. Pindahkan n-1 piring teratas pada tonggak A ke tonggak B, dengan menggunakan C sebagai perantara. (Gambar : Step 1 – 3)
  2. Pindahkan satu piring tersisa pada tonggak A ke tonggak C (Gambar : Step 4)
  3. Pindahkan n-1 piring teratas pada tonggak B ke tonggak C, dengan menggunakan tonggak A sebagai perantara (Gambar : Step 5– 7)

- ALGORITMA :

```
SUBRUTIN hanoi (n,a,b,c)
   JIKA n=1 MAKA
      tampilkan (“Memindah piring dari”, a, “ke”, c);
   SEBALIKNYA
       //memindahkan n-1 piring dari a ke b dengan
       //c sebagai perantara
       hanoi (n-1, a, c, b)
       //memindah 1 piring tersisa dari a ke c
       hanoi(1, a, b,c)
       //memindah n-1 piring dari b ke c, dengan a sbg perantara
       hanoi (n-1, b, a,c)
    AKHIR-JIKA
AKHIR-SUBRUTIN
```



Contoh Rekursif 5 - Lanjutan

```
#include <stdio.h>

void hanoi (int n, char a, char b, char c)
{
        if (n==1)
                printf("Pindahkan piring dari %c ke %c\n", a, c);
        else {
                hanoi(n-1, a, c, b);
                hanoi(1, a, b, c);
                hanoi(n-1, b, a, c);
        }
}

int main ( )
{
        int jum_piring;
        printf("Jumlah piring : ");
        scanf("%d", &jum_piring);
        hanoi (jum_piring, 'A', 'B', 'C');
        return 0;
} //hanoi.c
```

Latihan Soal

1. Terdapat rumusan seperti berikut :
   f (n) = 1, untuk n = 0
   f (n) = 2 * f (n-1) untuk n > 0
   Berapakah hasil f(4) ?
   Tuangkan dalam bentuk program

2. FPB dapat dibuat dengan rekursif dengan pola berikut :
   fpb(x,y) = y, jika y <= x dan sisaPembagian (x,y) = 0
   fpb(x,y) = fpb (y,x) jika x < y
   fpb (x,y) = fpb(y, sisaPembagian(x,y)) untuk keadaan yang lain

   Tuangkan dalam bentuk program rekursif