PROGRAM STUDI
TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS DIAN NUSWANTORO

Mata Kuliah
Dasar Pemrograman

Rekursif

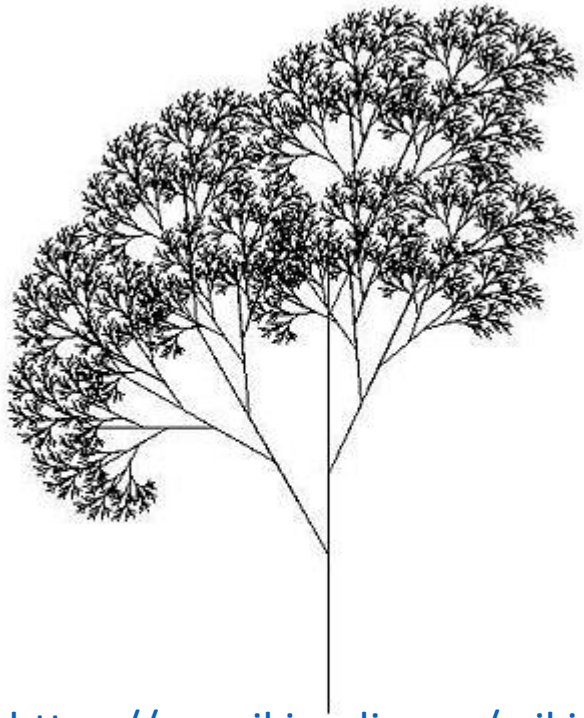
TIM DASAR PEMROGRAMAN
TEKNIK INFORMATIKA S1
UNIVERSITAS DIAN NUSWANTORO

Capaian Pembelajaran

1. Menjelaskan konsep analisis rekurens
2. Membuat fungsi rekursif pada suatu permasalahan komputasional sederhana
3. Menganalisis ekspresi rekursif untuk mengetahui alur kerja fungsi rekursif dalam penyelesaian permasalahan komputasional

Rekursif

- Merupakan proses pengulangan item – item dengan cara yang mirip. (definisi umum)
- Secara algoritmik: suatu cara untuk mendesain solusi dari masalah dengan cara **divide-and-conquer** atau **decrease-and-conquer**
  - Apa itu divide-and-conquer? Mengurangi masalah ke versi yang lebih sederhana dari masalah yang sama
- Secara semantic: suatu Teknik pemrograman dimana fungsi dapat memanggil dirinya sendiri



https://en.wikipedia.org/wiki/Recursion_(computer_science)

Rekursif Humor pada Singkatan

- PHP singkatan dari “PHP Hypertext Preprocessor”
- WINE singkatan dari “WINE Is Not an Emulator”
- GNU singkatan dari “GNU’s Not Unix”
- SPARQL singkatan dari “SPARQL Protocol and RDF Query Language”

Rekursif

- Secara umum, masalah – masalah yang berkaitan dengan rekursif ini juga dapat diselesaikan dengan solusi iterative, tetapi terkadang kita perlu untuk mengidentifikasi dan mengindeks permasalahan yang lebih kecil pada saat memrogram
- Didalam pemrograman, rekursif tidak boleh memiliki tujuan akhir yang tidak terhingga
- Setidaknya dia harus memiliki 1 atau lebih **kasus pokok/basis** (base case)
- Harus menyelesaikan permasalahan **yang sama pada input lain** dengan tujuan dari penyederhanaan input masalah yang lebih besar

Sejauh ini… solusi iteratif

- Solusi iterative dilakukan saat kita mengkonstruksi pengulangan (misalnya dengan for loop atau while loop) atau kita kenal dengan istilah **iterative algorithm**
- Dia dapat menangkap komputasi suatu set variable state yang diperbaharui pada setiap iterasi melalui perulangan

Contoh Perkalian: solusi Iteratif

- "kalikan a * b" sama dengan "menambahkan" a pada dirinya sendiri sebanyak b kali
- Algoritma:
- Initial state
  - **i ← b**
  - **Result ← 0**
- Tangkap state dengan cara iterasi sejumlah angka i yang dimulai saat b, dan decrement i sampai bernilai 0
- Pada komputasi di setiap iterasi, lakukan: **result ← result + a**

Realisasi

```
Fungsi kali(a:integer, b:integer) → integer

Kamus
i ← b
Result ← 0

Algoritma
while i > 0
   Result ← Result + a
   i ← i – 1
→ Result
```

Contoh Perkalian: solusi Rekursif

- **Langkah rekursif**
  - Pikirkan bagaimana kita mengurangi masalah dengan membuat penyederhanaan atau versi yang kecil dari permasalahan yang sama
- **Langkah basis**
  - Terus kurangi masalah sampai mencapai kasus yang paling sederhana yang dapat diselesaikan secara langsung
  - Misalnya: Ketika **b=1**, sehingga **a* b = a**

```
a * b  = a + a + a + a + … + a        (b kali)
       = a + a + a + … + a            (b-1 kali)
       = a + a * (b-1)                Recursive reduction
```

```
int kali(a,b){
   if(b==1){
      return a;
   }                                  Base case
   else{
      return a + kali(a, b-1)
   }                                  Recursive Step
}
```

Bentuk Kerangka Notasi Umum

```
Fungsi F(<berisi list-parameter>)
IF <kondisi-basis> then <ekspresi-1>
ELSE <kondisi-rekurens> then <ekspresi-2>
```

```
Fungsi Kali(a:int, b:int){
   if (b==1) {
      return a
   }
   else{
      return a + Kali(a,b-1)
   }
}
```

Analisis

- Diketahui a=3 dan b=4

```
b = 4   = 3 + Kali(3, 4-1)
        = 3 + 3 + Kali(3, 3-1)
        = 3 + 3 + 3 + Kali(3, 2-1) , basis dimana Kali(3,1) = 3 (jika
                                     b=1 maka return a)
        = 3 + 3 + 3 + 3
        = 12
```

Step by Step Analisis

- Diketahui a=3 dan b=4

```
b = 4 | = 3 + Kali(3, 4-1)
```

```
Fungsi Kali(a:int, b:int){
   if (b==1) {
      return a
   }
   else{
      return a + Kali(a,b-1)
   }
}
```

Step by Step Analisis

- Diketahui a=3 dan b=4

```
b = 3 | = 3 + Kali(3, 4-1)
      | = 3 + 3 + Kali(3, 3-1)
```

```
Fungsi Kali(a:int, b:int){
   if (b==1) {
      return a
   }
   else{
      return a + Kali(a,b-1)
   }
}
```

Step by Step Analisis

- Diketahui a=3 dan b=4

```
b = 2 | = 3 + Kali(3, 4-1)
      | = 3 + 3 + Kali(3, 3-1)
      | = 3 + 3 + 3 + Kali(3, 2-1)
```

```
Fungsi Kali(a:int, b:int){
   if (b==1) {
      return a
   }
   else{
      return a + Kali(a,b-1)
   }
}
```

Step by Step Analisis

- Diketahui a=3 dan b=4

```
b = 1 | = 3 + Kali(3, 4-1)
      | = 3 + 3 + Kali(3, 3-1)
      | = 3 + 3 + 3 + Kali(3, 2-1)
      | = 3 + 3 + 3 + 3
```

```
Fungsi Kali(a:int, b:int){
   if (b==1) {
      return a
   }
   else{
      return a + Kali(a,b-1)
   }
}
```

Bentuk Kerangka Notasi Umum

```
Fungsi F(<berisi list-parameter>)
IF <kondisi-basis> then <ekspresi-1>
ELSE <kondisi-rekurens> then <ekspresi-2>
```

```
Fungsi Faktorial(n:int){
   if (n==1) {
      return 1
   }
   else{
      return n + Faktorial(n-1)
   }
}
```

Iterasi vs Rekursif

- Rekursif mungkin lebih sederhana, lebih mudah dipahami secara intuitif
- Rekursif mungkin lebih efisien dilihat dari sudut pandang programmer (tidak banyak mengetik program)
- Rekursif mungkin tidak efisien dilihat dari sudut pandang mesin

Faktorial

- n! = n * (n-1) * (n-2) * (n-3) * … * 1
- Contoh: 4! = 4 * (4-1) * (4-2) * (4-3) = 4 * 3 * 2 * 1 = 24
- Dalam hal ini, sejauh yang kita tahu, jika n adalah 1 maka, 1! Bernilai 1. Hal ini dapat kita gunakan sebagai acuan basis
- Bagaimana cara mengurangi masalah? Tulis Kembali persamaan yang lebih sederhana untuk meraih basis
  - n * (n-1)

Referensi

Utama:

1. Bjarne Stroustrup, 2014, Programming: Principles and Practice Using C++ (Second Edition), Addison-Wesley Professional

Pendukung:

1. Introduction to Computer Science and Programming in Python, MIT https://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-0001-introduction-to-computer-science-and-programming-in-python-fall-2016

2. Introduction to Computer Science and Programming, MIT https://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-00sc-introduction-to-computer-science-and-programming-spring-2011/index.htm

TERIMA KASIH

ANY QUESTIONS?