Laporan Praktikum Algoritma dan Pemrograman

Semester Genap 2023/2024

NIM	**71210695**
Nama Lengkap	**Cahaya Sampebua**
Minggu ke / Materi	**13 / Fungsi Rekursif**

SAYA MENYATAKAN BAHWA LAPORAN PRAKTIKUM INI SAYA BUAT DENGAN USAHA SENDIRI TANPA MENGGUNAKAN BANTUAN ORANG LAIN. SEMUA MATERI YANG SAYA AMBIL DARI SUMBER LAIN SUDAH SAYA CANTUMKAN SUMBERNYA DAN TELAH SAYA TULIS ULANG DENGAN BAHASA SAYA SENDIRI.

SAYA SANGGUP MENERIMA SANKSI JIKA MELAKUKAN KEGIATAN PLAGIASI, TERMASUK SANKSI TIDAK LULUS MATA KULIAH INI.

PROGRAM STUDI INFORMATIKA
FAKULTAS TEKNOLOGI INFORMASI
UNIVERSITAS KRISTEN DUTA WACANA
YOGYAKARTA
2024

BAGIAN 1: MATERI MINGGU INI (40%)

Pada bagian ini, tuliskan kembali semua materi yang telah anda pelajari minggu ini. Sesuaikan penjelasan anda dengan urutan materi yang telah diberikan di saat praktikum. Penjelasan anda harus dilengkapi dengan contoh, gambar/ilustrasi, contoh program (source code) dan outputnya. Idealnya sekitar 5-6 halaman.

MATERI

Fungsi rekursif adalah bentuk lain dari perulangan. Fungsi rekursif dilakukan dengan cara memanggil fungsi di dalam fungsi itu sendiri. Jika salah menggunakan, fungsi ini dapat menjadi infinite loop.

Untuk membuat rekursif diperlukan dua hal:

1. Base case : case dimana kita tidak perlu menghitung hasilnya, biasanya sebagai penentu untuk kapan rekursif itu berhenti
2. Recursive case : bagian dimana ada suatu statement yang diulang terus menerus sampai mencapai base case.

Recursive case harus dapat diulang terus menerus sampai dapat mereturn nilai parameter yang memenuhi base case.

Kelebihan dan Kekurangan

Kelebihan

1. Bentuk lebih singkat dan elegan
2. Masalah rumit bisa dibagi menjadi masalah yang lebih kecil

Kekurangan

1. Efisiensi kurang
2. Banyak makan memori
3. Sulit dimengerti dan sulit untuk melakukan debugging

Bentuk Umum dan Studi Kasus

Bentuk yang sering dijumpai pada fungsi rekursif adalah

```
soal.py > ...
1   def rekursif(parameter):
2       if parameter == ... :   # Base case
3           ...
4       else:                   # Recursive case
5           ...                 # Statement diulang sampai mencapai base case
```

1. Kita coba menyelesaikan kasus deret dengan menggunakan fungsi rekursif

Buatlah program yang dapat menampilkan jumlah deret bilangan dengan parameter yang fleksibel!

Program

```
soal.py > ...
1   def deret(n):
2       if n == 0:
3           print("\n=",end="")
4           return 0
5       else:
6           if n != 1:
7               print(n,end="+")
8           else:
9               print(n,end="")
10          return n + deret(n-1)
11
12  print(deret(10))
```

```
10+9+8+7+6+5+4+3+2+1
=55
```

Dengan melihat output program di atas, kita dapat menyimpulkan pola yang dihasilkan, yaitu

$$n + (n - 1) + (n - 2) + (n - 3) + \cdots + 1$$

Jadi, bentuk rekursif di atas hampir sama dengan bentuk factorial. Coba

bandingkan dengan program factorial

soal.py > ...

```python
def faktorial(n):
    if n == 0:
        return 1
    else:
        print(n,end="*") if n != 1 else print(n,end="\n= ")
        return n * faktorial(n-1)

print(faktorial(5))
```

```
5*4*3*2*1
= 120
```

Dalam program factorial, kita mendapatkan pola yang hampir sama, yaitu

$n! = n * (n - 1)!$

$n! = n * (n - 1) * (n - 2)!$

$n! = n * (n - 1) * (n - 2) * (n - 3)!$

Dst.

Sehingga kesimpulan yang bisa diambil adalah rekursif case akan terus menerus melakukan operasi sampai mencapai titik akhir.

Dalam Python, rekursi memiliki batas yaitu sebesar 1000 rekursi. Jika sudah mencapai angka 1000, rekursi akan berhenti dan program akan crash.

2. Menyelesaikan fungsi f(x) dengan rekursi

Misalkan kita memiliki fungsi $f(x) = 2x^2 + 4x$. Berapakah jumlah $f(x)$ jika x adalah semua bilangan bulat dari 1 sampai 5?

soal.py > ...

```python
def fungsi(x):
    if x == 0:
        return 0
    else:
        print(f"((2*{x})**2 + 4*{x})")
        return ((2*x)**2 + 4*x) + fungsi(x-1)

print(fungsi(5))
```

```
((2*5)**2 + 4*5)
((2*4)**2 + 4*4)
((2*3)**2 + 4*3)
((2*2)**2 + 4*2)
((2*1)**2 + 4*1)
280
```

Percayalah bahwa hasil diatas adalah benar.

3. Menyelesaikan kasus bakteri yang berlipat ganda setiap menit.

Di dalam toples ada berisi bakteri 7 biji. Mereka selalu berkembang biak dan setiap menit akan bertambah menjadi 2 kali lipat. Berapakah jumlah bakteri jika sudah 1 jam?

bakteri.py > ...

```python
def bakteri(n,menit):
    if menit == 0:
        return n
    else:
        return 2 * bakteri(n,menit-1)

print(bakteri(7,60))
```

Output

```
8070450532247928832
```

Ternyata dapat berkembang biak sebanyak 8 Quintilion.

4. Menyelesaikan masalah menulis angka terbalik urutannya

rekursif.py > ...

```python
def angkaTerbalik(n):
    if n == 0:
        return 0
    else:
        return str(n) +", "+ str(angkaTerbalik(n-1))

print(angkaTerbalik(120))
```

Output

```
120, 119, 118, 117, 116, 115, 114, 113, 112, 111, 110, 109, 108, 107, 106, 105, 104,
 103, 102, 101, 100, 99, 98, 97, 96, 95, 94, 93, 92, 91, 90, 89, 88, 87, 86, 85, 84,
 83, 82, 81, 80, 79, 78, 77, 76, 75, 74, 73, 72, 71, 70, 69, 68, 67, 66, 65, 64, 63,
 62, 61, 60, 59, 58, 57, 56, 55, 54, 53, 52, 51, 50, 49, 48, 47, 46, 45, 44, 43, 42,
 41, 40, 39, 38, 37, 36, 35, 34, 33, 32, 31, 30, 29, 28, 27, 26, 25, 24, 23, 22, 21,
 20, 19, 18, 17, 16, 15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5, 4, 3, 2, 1, 0
```

Semua bentuk rekursif dapat diubah menjadi iterative (for, while). Namun tidak berlaku sebaliknya. Bentuk iterative tidak mesti bisa diubah menjadi rekursif.

BAGIAN 2: LATIHAN MANDIRI (60%)

Pada bagian ini anda menuliskan jawaban dari soal-soal Latihan Mandiri yang ada di modul praktikum. Jawaban anda harus disertai dengan source code, penjelasan dan screenshot output.

Link GitHub ➔ https://github.com/cahaya-sampebua/Latihan-Mandiri-PrakALPRo-13.git

SOAL 1

```
#Soal 1

def cekPrima(n,default=2):
    if n == 0 or n == 1:
        return False
    elif n == 2:
        return True

    if n % default == 0:
        return False

    if default * default > n:
        return True

    return cekPrima(n,default + 1)

print(cekPrima(12))
```

```
False
```

Program di atas akan mengecek apakah n adalah 1 atau 0 ataupun 2. Jika n adalah 1 atau 0, akan mengeluarkan hasil False. Jika n adalah 2, akan menghasilkan True. Baris 2 sampai 5 merupakan base case.

Baris 7, jika angka % default = 0, maka sudah pasti bukan bilangan prima. Maka akan langsung mengeluarkan False.

Jika tidak, dicek sampai default * default melebihi n sebagai angka agar tidak terlalu lama menghitung.

SOAL 2

```
#Soal 2

def palindrom(kata, start, end):
    if start == end:
        return True

    if kata[start] != kata[end-1]:
        return False

    if start < end + 1:
        return palindrom(kata,start+1,end-1)

    return True

kata = "kasur ini rusak"
start = 0
end = len(kata)

print(palindrom(kata,start,end))
```

```
True
```

Fungsi akan mengecek jika start adalah sama dengan end atau bisa dibilang kata adalah string kosong maupun hanya berisi 1 huruf saja. Jika tidak, fungsi akan lanjut mengecek jika huruf pertama sama dengan huruf terakhir. Jika tidak, otomatis kata tersebut bukanlah palindrom.

Selanjutnya, fungsi akan mengecek satu persatu huruf dari awal sampai akhir. Awal akan ditambah 1 dan akhir dikurangi 1 sampai start == end.

SOAL 3

```python
#Soal 3

def jumlah_deret(n):
    # Base Case
    if n == 1:
        return 1

    # Jika input ganjil
    elif n % 2 != 0 and n > 0:
        return n + jumlah_deret(n-2)

    # Jika input genap
    elif n % 2 == 0 and n > 0:
        n = n - 1
        return jumlah_deret(n)

print(jumlah_deret(101))
```

```
2601
```

Base case di atas adalah saat n bernilai 1, karena kita tidak perlu menjumlahkan apa-apa jika n nya 1. Jika input yang dimasukkan ganjil, fungsi akan langsung menghitung n + fungsi jumlah deret berikutnya sampai n mencapai 1. Jika input genap, n akan dikurangi dulu dengan 1.

SOAL 4

```
#Soal 4

def jumlahDigit(n):
    if n < 10:
        return n
    else:
        kali = n // 10
        n = n % (10*(kali))
        return n + jumlahDigit(kali)

print(jumlahDigit(234))
```

```
9
```

Base case fungsi ini adalah saat n kurang dari 10. Kenapa? Karena kita tidak perlu menghitung apa-apa saat hanya ada 1 digit di angka tersebut.

Jika tidak, n akan dibagi tanpa koma untuk mengurangi angka di belakangnya dan dimasukkan ke variabel baru “kali”. Setelah itu, n akan diganti nilai menjadi n % 10 dikali dengan nilai dari variabel “kali” sehingga didapatkan angka terakhir dari n. Lalu return n dijumlahkan dengan fungsi berikutnya sebagai rekursif case.

SOAL 5

```python
#Soal 5

def kombinasi(n,r):
    if r == 0:
        return 1
    elif n < r:
        return 0
    else:
        print(r)
        return round(((n + 1 - r)/r) * kombinasi(n, r-1))

print(kombinasi(14,5))
```

```
5
4
3
2
1
2002
```

Fungsi di atas akan mengecek apakah r (sampel) adalah 0. Jika r adalah 0, fungsi akan mereturn 1 karena jika semua n dikombinasi dengan 0, hasilnya pasti 1. Dan jika n lebih kecil daripada sampel, fungsi akan mereturn 0.

Jika tidak, fungsi akan menghitung kombinasi dari n dan r sampai r mencapai 1.