



Laporan Praktikum Algoritma dan Pemrograman

Semester Genap 2023/2024

| | |
|---------------------------|--------------------------------|
| NIM | 71230987 |
| Nama Lengkap | Prastha Pradipta Purusa |
| Minggu ke / Materi | 14 / Fungsi Rekrusif |

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.

Penguasaan Rekrusif

Fungsi ekursif adalah fungsi yang memanggil fungsi sendiri. Fungsi rekursif merupakan fungsi matematis yang berulang dan memiliki pola yang terstruktur. Fungsi rekrusif harus digunakan secara hati-hati karena dapat bersifat unlimited loop. Fungsi rekrusif mirip dengan penggunaan perulangan while. Fungsi rekrusif memiliki 2 blok yaitu titik berhenti dan yang memanggil dirinya sendiri. 2 bagian dalam rekrusif:

- a. Base case:
 1. bagian untuk memberhentikan fungsi rekrusif
 2. Biasanya berupa solusi terkecil dari masalah
 3. Sudah langsung diketahui jawabannya
- b. Rekrusif case:
 1. bagian perulangan terus menerus hingga ke base case
 2. Solusi belum didapatkan, sehingga perlu perhitungan lebih lanjut
 3. Proses menuju base case

Contoh penempatan base case dan rekrusif case:

```
def pangkat(m, n):  
    if n == 0:  
        return 1  
    elif n == 1:  
        return m  
    else:  
        return m * pangkat(m, n-1)
```

Base case

Recursive case

```
hasil = pangkat(4, 3)
```

3 jenis rekrusif:

- a. Head recursion
Recursive case diletakkan di bagian pertama dari suatu fungsi rekursif
- b. Tail recursion
Recursive case diletakkan di bagian akhir dari suatu fungsi rekursif
- c. Indirect recursion
Melibatkan lebih dari 1 fungsi seperti contoh fungsi a() memanggil fungsi b() kemudian didalam fungsi b() memanggil fungsi c() kemudian didalam fungsi c() memanggil fungsi a(). Penggunaan fungsi indirect recursion harus dihindari

Contoh penempatan Head Recursion:

Head Recursion

```
def cetak_angka(n):  
    if n > 0:  
        cetak_angka(n-1)  
        print(n)
```

`cetak_angka(4)`

Operasi pertama langsung rekursif

`print(n)` harus menunggu sampai pemanggilan rekursif selesai

Contoh penempatan Tail Recursion:

Tail Recursion

```
def cetak_angka(n):  
    if n > 0:  
        print(n)  
        cetak_angka(n-1)
```

`cetak_angka(4)`

Pemanggilan rekursif di bagian akhir

Saat terjadi pemanggilan rekursif, tidak ada perintah di bawahnya yang menunggu

Jika hanya memanggil satu fungsi rekursif maka disebut linear recursion sedangkan jika memanggil lebih dari satu fungsi rekursif maka disebut tree recursion. Contoh tree recursion yang memanggil 2 fungsi rekursif:

```
def pasanganHuruf(s):  
    pasangan = 0  
    if len(s) < 3:  
        return 0  
    elif s[0] == s[2]:  
        pasangan += 1  
    return pasangan + pasanganHuruf(s[1:])
```

Kelebihan dan Kekurangan

Kelebihan menggunakan fungsi rekursif:

1. Kode lebih singkat dan bagus
2. Masalah kompleks dapat di breakdown menjadi masalah yang kecil

Kekurangan menggunakan fungsi rekursif

1. Menggunakan memori yang lebih besar

2. Mengurangi efisiensi dan kecepatan memproses
3. Sulit melakukan debugging dan sulit dimengerti

Bentuk Umum dan Studi Kasus

Bentuk umum dalam rekursif:

```
tabnine: test | explain | document | ask
11 def nama_fungsi(paramter):
12     return nama_fungsi
```

Contoh program rekursif:

```
tabnine: test | explain | document | ask
18 def cek_pythagoras(a, b, c):
19     return a**2 + b**2 == c**2
20 a = 3
21 b = 4
22 c = 5
23 if cek_pythagoras(a, b, c):
24     print(f"({a}, {b}, {c}) Pythagoras.")
25 else:
26     print(f"({a}, {b}, {c}) bukan Pythagoras.")
27
```

(3, 4, 5) Pythagoras.

Program tersebut adalah contoh rekursif Pythagoras dimana fungsi dipanggil sendiri di line 2 yang berarti $a^2 + b^2 = c^2$ dengan test case $a = 3, b = 4, c = 5$.

Contoh program rekursif ke 2 yaitu menampilkan tulisan sebanyak n:

```
tabnine: test | explain | document | ask
26 def print_print(n):
27     if n <= 0:
28         return
29     else:
30         print("print")
31         print_print(n-1)
32 print_print(3)
```

```
print
print
print
```

Program tersebut mencetak print sebanyak 3x dimana jika n kurang dari sama dengan maka tidak ada output muncul. Jika lebih dari satu maka akan mencetak kata print dan `print_print(n-1)` adalah pemanggilan rekursif di mana fungsi memanggil dirinya sendiri dengan n yang dikurangi 1

Contoh program rekursif ke 3 menghitung nilai $1 + 2 + 3 + \dots + x$ secara rekursif:

```
34 def total(x):
35     if x == 1:
36         return 1
37     else:
38         return x + total(x - 1)
39 print(total(5))
40 print(total(10))
```

```
15
55
```

Program tersebut menghitung nilai $1 + 2 + 3 + \dots + x$ secara rekursif dengan test case yaitu 5 dan 10. Jika $x = 1$ maka kembalikan dengan nilai 1 sebagai base case dan asus else: $\text{return } x + \text{total}(x - 1)$ fungsi total memanggil dirinya sendiri dengan argumen $x - 1$ dan menjumlahkannya dengan x dan sebagai recursive case.

Contoh program rekursif ke 4 yaitu fungsi pangkat(int x, int n) untuk menghitung x^n dengan cara rekursif:

```
42 def pangkat(x, n):
43     if n == 0:
44         return 1
45     elif n > 0:
46         return x * pangkat(x, n - 1)
47     else:
48         return 1 / pangkat(x, -n)
49
50 print(pangkat(2, 3))
51 print(pangkat(5, 0))
52 print(pangkat(2, -2))
```

```
8
1
0.25
```

Menghitung pangkat dengan parameter (x,n) secara rekursif. Jika $n = 0$ maka dikembalikan nilai 1. Jika $n > 0$ maka fungsi memanggil dirinya sendiri dengan n dikurangi 1 dan mengalikan hasilnya dengan x . Jika $n < 0$ fungsi memanggil dirinya sendiri dengan $-n$ untuk menghitung pangkat positifnya, kemudian mengembalikan kebalikan dari hasilnya (karena $x^{-n} = 1/x^n$). Fungsi ini dapat menghitung pangkat dengan bilangan positif maupun negatif.

Contoh Program rekursif ke 5 adalah mencari huruf besar pertama dari sebuah string dengan menggunakan cara rekursif:

```

56 def cari_hurufbesar_pertama(string):
57     if string[0].isupper():
58         return string[0]
59     return cari_hurufbesar_pertama(string[1:])
60 string = "haihA10"
61 result = cari_hurufbesar_pertama(string)
62 print(f"Huruf besar pertama: {result}")

```

```

[Running] python -u C:\Users\...
Huruf besar pertama: A

```

Jika string karakter pertama dari string adalah huruf besar (string[0].isupper()) maka kembalikan karakter tersebut kemudian panggil fungsi secara rekursif mulai dari indeks ke-1. Test case adakag haihA10 dan output adalah A karena huruf besar pertama di string.

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.

SOAL 1

The screenshot shows a Jupyter Notebook window titled "Prastha Pradipta Purusa(71230987) bab 14 Las". The interface includes a menu bar (File, Edit, View, Insert, Cell, Kernel, Widgets, Help) and a toolbar with icons for file operations, running, and code execution. The code cell contains the following Python code:

```

In [2]: def bilangan_prima(n, pembagi):
        if pembagi == 1:
            return True
        if n % pembagi == 0:
            return False
        return bilangan_prima(n, pembagi - 1)

        n = int(input("Masukkan bilangan: "))
        if bilangan_prima(n,n-1):
            print(f"{n} bilangan prima.")
        else:
            print(f"{n} bukan bilangan prima.")

```

Below the code, the input and output are shown:

```

Masukkan bilangan: 13
13 bilangan prima.

```

Line1: membuat fungsi bilangan_prima dengan parameter(n,pembagi)

Line2: jika pembagi == 1

Line3: maka hasilnya benar/True

Line4: jika n habis dibagi pembagi

Line5: bukan bilangan prima dan menghasilkan False

Line6: Langkah rekursif jika kedua kondisi tidak terpenuhi dan fungsi akan memanggil dirinya sendiri dengan argumen pembagi dikurangi 1.

Line7: untuk n masukan bilangan

Line8: fungsi bilangan_prima dipanggil dengan n sebagai bilangan yang akan dicek dan n-1 sebagai pembagi awal dan jika bilangan prima, maka bilangan tersebut adalah bilangan prima.

Line9: mencetak output bahwa n adalah bilangan prima

Line10: kondisi jika n tidak prima

Line11: mencetak output bahwa n bukan bilangan prima

SOAL 2

```
In [3]: def palindrom(kata):
        kata = ''.join(kata.split()).lower()
        if len(kata) <= 1:
            return True
        if kata[0] != kata[-1]:
            return False
        return palindrom(kata[1:-1])

        kata = "Rusak kasur"
        if palindrom(kata):
            print("Palindrom")
        else:
            print("Bukan palindrom")
```

Palindrom

Line1: membuat fungsi palindrom dengan parameter angka

Line2: Menghilangkan spasi dalam string kata dan mengubah semua karakternya menjadi huruf kecil. Metode split() memisahkan string menjadi potongan-potongan berdasarkan spasi, kemudian join() menggabungkan kembali potongan-potongan tersebut tanpa spasi.

Line3: Jika panjang string kata kurang dari atau sama dengan 1

Line4: merupakan palindrom

Line5: Jika (kata[0]) tidak sama dengan (kata[-1])

Line6: bukan palindrom

Line7: fungsi palindrom dipanggil kembali untuk menguji sisa string yang ada di antara karakter pertama dan terakhir dan proses terus berlanjut secara rekursif sampai string habis atau dinyatakan bukan palindrom.

Line8: test case

Line9: kondisi if yang memeriksa apakah hasil dari pemanggilan fungsi palindrom(kata) adalah True. Jika hasilnya True, artinya kata adalah palindrom.

Line10: mencetak output palindrom

Line11: Jika tidak palindrom

Line12: mencetak output bukan palindrom

SOAL 3

```
In [4]: def jumlah_deret_ganjil(n):  
        if n == 1:  
            return 1  
        elif n % 2 == 0:  
            return jumlah_deret_ganjil(n - 1)  
        else:  
            return n + jumlah_deret_ganjil(n - 2)  
  
        n = 7  
        print(jumlah_deret_ganjil(n))
```

16

Line1: membuat fungsi jumlah_deret_ganjil

Line2: jika n = 1

Line3: maka n adalah 1

Line4: jika n habis dibagi 2

Line5: Jika n genap, fungsi akan memanggil dirinya sendiri dengan argumen n - 1.

Line6: jika n bilangan ganjil

Line7: dilakukan untuk menjumlahkan bilangan ganjil yang lebih kecil dari n secara berulang-ulang melalui pemanggilan rekursif.

Line8: jika n adalah 7

Line9: memunculkan output hasil dari fungsi jumlah_deret_ganjil

SOAL 4

```
In [5]: def jumlah_digit(n):  
        if n == 0:  
            return 0  
        else:  
            return n % 10 + jumlah_digit(n // 10)  
  
        bilangan = 234  
        hasil = jumlah_digit(bilangan)  
        print("Jumlah digit dari", bilangan, "adalah", hasil)
```

Jumlah digit dari 234 adalah 9

Line1: membuat fungsi jumlah_digit dengan parameter n

Line2: Jika n = 0

Line3: maka nilai 0

Line4: jika nilai n tidak 0

Line5: Mengambil digit terakhir dari bilangan dengan $n \% 10$ kemudian menambahkannya dengan jumlah digit dari sisa bilangan setelah digit terakhir dihapus ($n // 10$).

Line6: contoh jika bilangan adalah 234

Line7: memanggil fungsi jumlah_digit dengan variabel hasil dengan bilangan

Line8: mencetak output jumlah digit dari 234 adalah 9

SOAL 5

```
In [6]: def kombinasi(r,n):  
        if r == 0 or r == n:  
            return 1  
        else:  
            return kombinasi(r-1,n-1) + kombinasi(r,n-1)  
        r = 12  
        n = 5  
        print(kombinasi(n,r))  
792
```

Line1: membuat fungsi kombinasi dengan parameter (r,n)

Line2: jika $r = 0$ atau $r = n$

Line3: maka nilai berupa 1

Line4: Jika kondisi tidak terpenuhi

Line5: Langkah rekursif menghitung kombinasi dengan menggunakan $C(n,r)$

Line6: test case jika r adalah 12

Line7: test case jika n adalah 5

Line8: menghasilkan output dari r adalah 12 dan n adalah 5 yaitu 792

Linkgithub:

<https://github.com/prasss020904/Laprak14-71230987.git>