Laporan Praktikum Algoritma dan Pemrograman

Semester Genap 2023/2024

NIM	**71200609**
Nama Lengkap	**Airell Aristo Subagia**
Minggu ke / Materi	**13 / 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

Rekursif

Apa Itu Rekursif

Dalam pemrograman, rekurif merupakan suatu fungsi yang memanggil dirinya sendiri secara langsung atau tidak. Fungsi rekursif harus memiliki kondisi dasar (base case) untuk menghitkan loopingnya agar tidak menjadi infinite looping.

Contoh :

```
def genap(n):
    if n < 0:
        return 0
    elif n % 2 == 0:
        print(n)
        genap(n - 2)
    else:
        genap(n - 1)

genap(10)
```

Outputnya :

```
10
8
6
4
2
0
```

Karakteristik Dari Rekursif

1. Base Case (Kondisi Dasar)

   Kondisi di mana fungsi tidak memanggil dirinya sendiri lagi, sehingga menghentikan rekursi.

2. Recursive Case (Kondisi Rekursif)

   Bagian dari fungsi di mana ia memanggil dirinya sendiri dengan argumen yang diubah, mendekati base case.

Penjelasannya

```python
def genap(n):
    if n < 0:
        return # Ini Base Casenya
    elif n % 2 == 0:
        print(n)
        genap(n - 2) # Ini Rekursif Casenya
    else:
        genap(n - 1) # Ini Rekursif Casenya

genap(10)
```

Kelebihan Dan Kekurangan Pada Rekursif

Kelebihan :

1. Tampak sederhana dan elegan.
2. Penggunaan yang alami pada Struktur Data Tertentu.
3. Dapat memecahkan masalah yang kompleks.

Kekurangan :

1. Overhead pada memori

   Penggunaan rekursif dapat menyebabkan kehabisan memory, hal ini dapat terjadi karena adanya call stack (tumpukkan panggilan).
2. Inefisiensi pada Kasus Tertentu

   Rekursif dapat menjadi tidak effisien jika sub-masalah dihitung berkali-kali tanpa adanya penyimpanan hasil. Hal ini dapat menyebabkan kenaikan pada time complexity.
3. Kesulitan dalam debugging

   Rekursif dapat membuat debugging menjadi sulit karena banyaknya lapisan panggilan fungsi yang harus dilacak.

Contoh Kasus

- Dapat digunakan seperti for loop, pada kasus ini digunakan untuk pengaksesan item yang ada dalam list

  Code :

```
def ambilItem(list):
    if len(list) == 0:
        return ""
    elif len(list) == 1:
        return list[0]
    else:
        return list[0] + " " + ambilItem(list[1:])

print(ambilItem(["apel", "mangga", "jeruk","nangka"]))
```

  Outputnya :

```
apel mangga jeruk nangka
```

  Proses rekursifnya :

1. Panggilan Pertama:

   list = ["apel", "mangga", "jeruk", "nangka"]

   len(list) = 4

   ambilItem(list[1:]) = ambilItem(["mangga", "jeruk", "nangka"])

2. Panggilan Kedua:

   list = ["mangga", "jeruk", "nangka"]

   len(list) = 3

   ambilItem(list[1:]) = ambilItem(["jeruk", "nangka"])

3. Panggilan Ketiga:

   list = ["jeruk", "nangka"]

   len(list) = 2

ambilItem(list[1:]) = ambilItem(["nangka"])

4. Panggilan Keempat:

   list = ["nangka"]

   len(list) = 1

   ambilItem(list[1:]) = ambilItem([])

5. Panggilan Kelima:

   list = []

   len(list) = 0

   Kondisi dasar terpenuhi, fungsi mengembalikan string kosong "".

- Dapat digunakan seperti for loop, pada kasus ini digunakan untuk membuat segitiga

  Code :

```
def buatSegitigaTerbalik(n):
    if n == 0:
        return ""
    else:
        segitiga = "*" * n + "\n"
        return segitiga + buatSegitigaTerbalik(n-1)

print(buatSegitigaTerbalik(6))
```

Outputnya :

```
******
*****
****
***
**
*
```

Penjelasan :

1. Panggilan Pertama :

- Fungsi menerima argumen n.
- Jika n sama dengan 0, fungsi mengembalikan string kosong.
- Jika tidak, fungsi menghasilkan string dengan spasi sebanyak n-1 diikuti dengan karakter '*' sebanyak n dan ditambahkan karakter newline '\n'.
- Selanjutnya, fungsi melakukan rekursi dengan n-1.

2. Panggilan Kedua hingga Terakhir:
    - Proses di atas berulang hingga n menjadi 0.
    - Setiap panggilan rekursif menghasilkan satu baris segitiga.
    - Baris-baris ini digabungkan bersama untuk membentuk segitiga terbalik secara keseluruhan.

BAGIAN 2: LATIHAN MANDIRI (60%)

SOAL 1

```python
#Latihan 13.1

def is_prime_recursive(n, divisor=None):
    if n <= 1:
        return False
    if n == 2:
        return True
    if divisor is None:
        divisor = int(n ** 0.5)
    if divisor == 1:
        return True
    if n % divisor == 0:
        return False
    return is_prime_recursive(n, divisor - 1)

# Contoh penggunaan
print(is_prime_recursive(97))
print(is_prime_recursive(30))
```

Outputnya :

```
True
False
```

Penjelasannya :

1. Pertama-tama, dibuat dulu **function** bernama **is_prime_recursive** yang menggunakan 2 parameter, yang dimana **n** digunakan untuk nilai yang ingin di cek, dan **divisor** untuk pembagiannya.
2. Kemudian, didalamnya dibuat 2 base case yang dimana yang pertama jika n <= 1 maka sudah pasti bukan prima, dan Jika n == 2 maka pasti prima.
3. Seteleh itu, dilakukan operasi **if statement** ini digunakan untuk check apakah divisor masih none. Jika iya maka akan diisi dengan **akar kuadrat dari n**. Nilai **n** berupa integer.
4. Setelah itu dibuat 2 **if statement** terakhir, yang pertama digunakan untuk melihat apakah sampai **divisor = 1** tidak ditemukan adanya pembagi, maka itu merupakan bilangan prima. Jika ternyata **ditemukan adanya pembagi** maka itu bukan bilangan prima dan return false.

5. Terakhir, tinggal dilakukan pengurangan divisor secara rekursif. Hingga stop jika tersisa 1 atau ditemukan adanya pembagi.

SOAL 2

```
#Latihan 13.2

def cekPalindrom (kata) :
    kata1 = kata.replace(" ","")
    if len(kata1) == 0 :
        return True
    else :
        if kata1[0] == kata1[-1] :
            return cekPalindrom(kata1[1:len(kata1)-1])
        else :
            return False


print(cekPalindrom("rotator"))
print(cekPalindrom("makanan"))
```

Outputnya :

```
True
False
```

Penjelasannya :

1. Pertama-tama, dibuat fungsi bernama **cekPalindrom** dengan 1 parameter yang digunakan untuk menyimpan katanya.
2. Setelah itu, dihapus dulu untuk whitespacenya.
3. Kemudian dibuat base casenya dimana jika panjang dari **kata** yang dimasukkan itu 0 maka return true dan stop dari fungsi rekursifnya.
4. Setelah itu dilakukan **if statement** untuk pengecekan abjad dari indeks pertama dan terakhir. Jika sama maka akun dilakukan slicing dengan penggunaan rekursif. Akan tetapi jika tidak maka akan langsung return false.

SOAL 3

```
#Latihan 13.3

def sumGanjil (n) :
    if n == 0 :
        return 0
    if n % 2 != 0 :
        return n + sumGanjil(n -1)
    else :
        return sumGanjil(n-1)


print(sumGanjil(10))
```

Outputnya

```
30
```

Penjelasannya :

1. Pertama-tama, dibuat fungsi bernama **sumGanjil** dengan 1 parameter yang digunakan untuk pengurangan angka secara rekursif dan untuk insiasi awal pengecekan.
2. Kemudian, untuk base casenya jika **n sama dengan 0** maka akan langsung return 0 dan digunakan juga untuk stop rekursifnya.
3. Setelah itu, dilakukan **if statement** lagi untuk menentukan bilangan ganjil atau bukan. Jika **n** bilangan ganjil maka n akan **ditambahkan dengan hasil rekursif dari n-1.** Akan tetapi jika **n merupakan bilangan genap** maka akan dilakukan **rekursif untuk pengurangan angka saja.**

SOAL 4

```
#Latihan 13.4

def sumDigit(n) :
    if len(n) == 0 :
        return 0
    else :
        if n.isnumeric() :
            return int(n[0]) + sumDigit(n[1:])
        else :
            return False

print(sumDigit("234"))
print(sumDigit("1a23"))
```

Outputnya

```
9
False
```

Penjelasannya :

1. Pertama-tama, dibuat fungsi bernama **sumDigit** dengan 1 parameter yang digunakan untuk pengurangan angka secara rekursif dan untuk insiasi awal pengecekan.
2. Kemudian, untuk base casenya jika **n sama dengan 0** maka akan langsung return 0 dan digunakan juga untuk stop rekursifnya.
3. Setelah itu, dilakukan **if statement** lagi untuk menentukan apakah string tersebut merupakan angka semua atau bukan. Jika **n** angka semua maka **n** akan **diganti menjadi integer** lalu **akan ditambahkan dengan indeks awal dari hasil slicing secara rekursif**.
4. Jika terdapat sebuah huruf dalam string maka akan langsung return false.

SOAL 5

```python
#Latihan 13.5

def kombinasi(n, x):
    if n == 0 or x == 0 or n == x:
        return 1
    else:
        return kombinasi(n - 1, x - 1) * n // x

print(kombinasi(5, 2))  # Output: 10
```

Outputnya

```
9
False
```

Penjelasannya :

1. Pertama-tama, dibuat fungsi bernama **kombinasi** dengan 2 parameter yang digunakan untuk pengurangan angka secara rekursif dan untuk insiasi awal pengecekan.
2. Kemudian, untuk base casenya jika **n atau x sama dengan 0 atau n sama dengan x** maka akan return 1.
3. Setelah itu, dimasukkan rumus kombinasi dimana kombinasi(n - 1, x - 1) * n // x, yang dimana karena kombinasi itu adalah misalnya dari angka 5 maka 5 x 4 x 3 x 2 x1. Maka letak rekursifnya berada disana.

Link Github :

https://github.com/AirellAristo/TugasPrakAlpro/tree/main/71200609_Pertemuan13(Rekursif)