

Pendahuluan

Fungsi adalah salah satu konsep fundamental dalam pemrograman yang memungkinkan kita untuk mengelompokkan sekelompok pernyataan menjadi satu blok yang dapat dijalankan berulang kali. Dalam Python, fungsi memainkan peran penting dalam membantu pemrogram mengorganisir, mengelola, dan menulis kode yang lebih modular, terstruktur, dan mudah dipelihara.

Tujuan dan Manfaat

Tujuan

1. **Memecah Program Menjadi Bagian Kecil:** Fungsi memungkinkan pemrogram untuk membagi program besar menjadi modul-modul kecil yang lebih mudah dikelola. Setiap modul atau fungsi menangani tugas tertentu, yang membuat program lebih terorganisir.
2. **Reusabilitas Kode:** Dengan mendefinisikan fungsi sekali, kita dapat memanggilnya berulang kali dalam berbagai bagian program tanpa harus menulis ulang kode yang sama, sehingga mengurangi redundansi.
3. **Abstraksi dan Enkapsulasi:** Fungsi menyediakan mekanisme untuk menyembunyikan detail implementasi dan hanya menampilkan antarmuka penggunaannya. Ini membantu dalam mengelola kompleksitas dan menjaga fokus pada tugas utama.
4. **Meningkatkan Pemahaman dan Dokumentasi:** Dengan memberikan nama yang bermakna pada fungsi, program menjadi lebih mudah dipahami dan didokumentasikan. Setiap fungsi dapat dideskripsikan dengan jelas tujuan dan cara kerjanya.

Manfaat

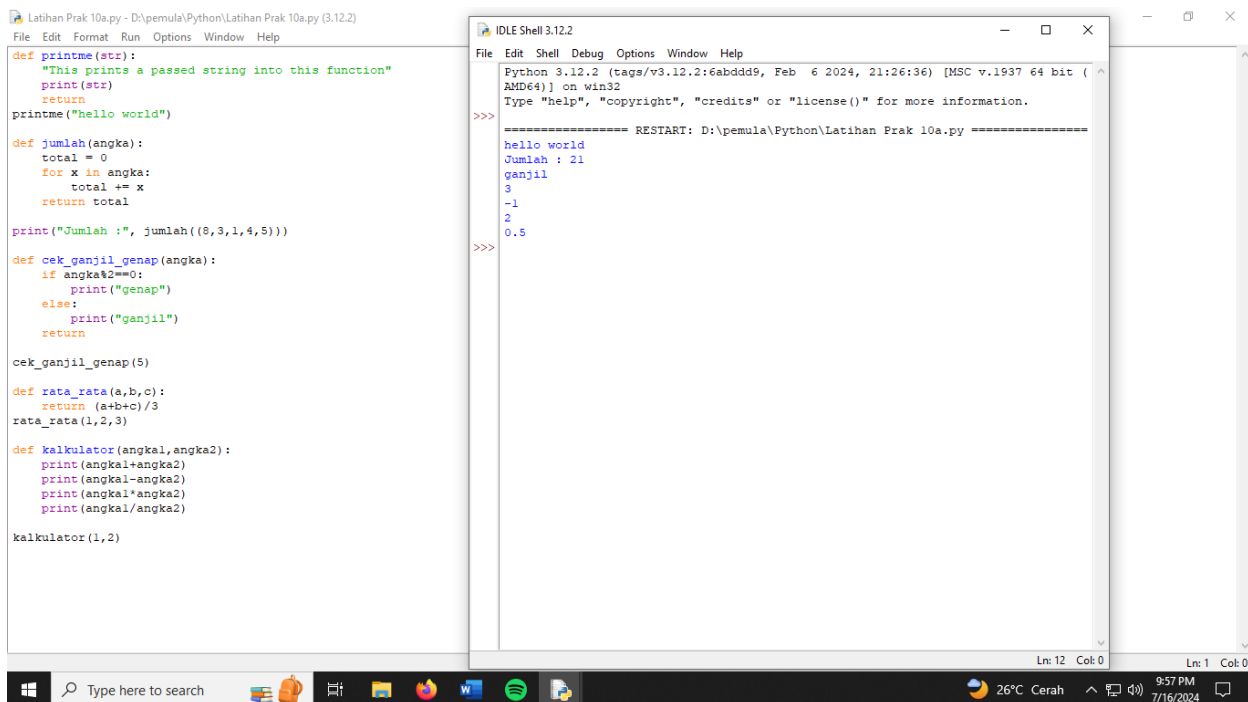
1. **Modularitas:** Fungsi memungkinkan pemrogram untuk mengorganisir kode ke dalam blok-blok yang dapat dikelola secara terpisah. Ini mempermudah pengembangan, pengujian, dan pemeliharaan program.
2. **Pemeliharaan Kode yang Lebih Mudah:** Dengan memecah program menjadi fungsi-fungsi kecil, perbaikan bug dan pembaruan fitur menjadi lebih mudah karena hanya perlu memodifikasi bagian-bagian tertentu tanpa mempengaruhi keseluruhan program.
3. **Mengurangi Duplikasi Kode:** Fungsi membantu dalam menghindari duplikasi kode dengan memungkinkan pemrogram menulis kode sekali dan

menggunakan kembali di berbagai tempat. Ini menghemat waktu dan usaha serta mengurangi kemungkinan kesalahan.

4. **Meningkatkan Kolaborasi Tim:** Dalam tim pengembangan, fungsi memungkinkan pembagian tugas yang lebih efisien. Setiap anggota tim dapat bekerja pada fungsi yang berbeda tanpa saling mengganggu, sehingga meningkatkan produktivitas.
5. **Pengujian dan Debugging yang Lebih Mudah:** Fungsi memudahkan pengujian unit karena setiap fungsi dapat diuji secara terpisah. Ini membuat proses debugging lebih cepat dan lebih terfokus pada area tertentu dari kode.
6. **Abstraksi Logika Kompleks:** Fungsi membantu dalam menyembunyikan logika kompleks di balik antarmuka yang sederhana. Pengguna fungsi tidak perlu memahami detail implementasi, cukup mengetahui cara memanggil dan menggunakan fungsi tersebut.
7. **Efisiensi Waktu Eksekusi:** Penggunaan fungsi yang terstruktur dapat meningkatkan efisiensi waktu eksekusi program dengan mengurangi overhead dan meningkatkan performa keseluruhan.

Program

1.



The image shows a screenshot of a Python IDE with two windows. The left window, titled 'Latihan Prak 10a.py', contains the following Python code:

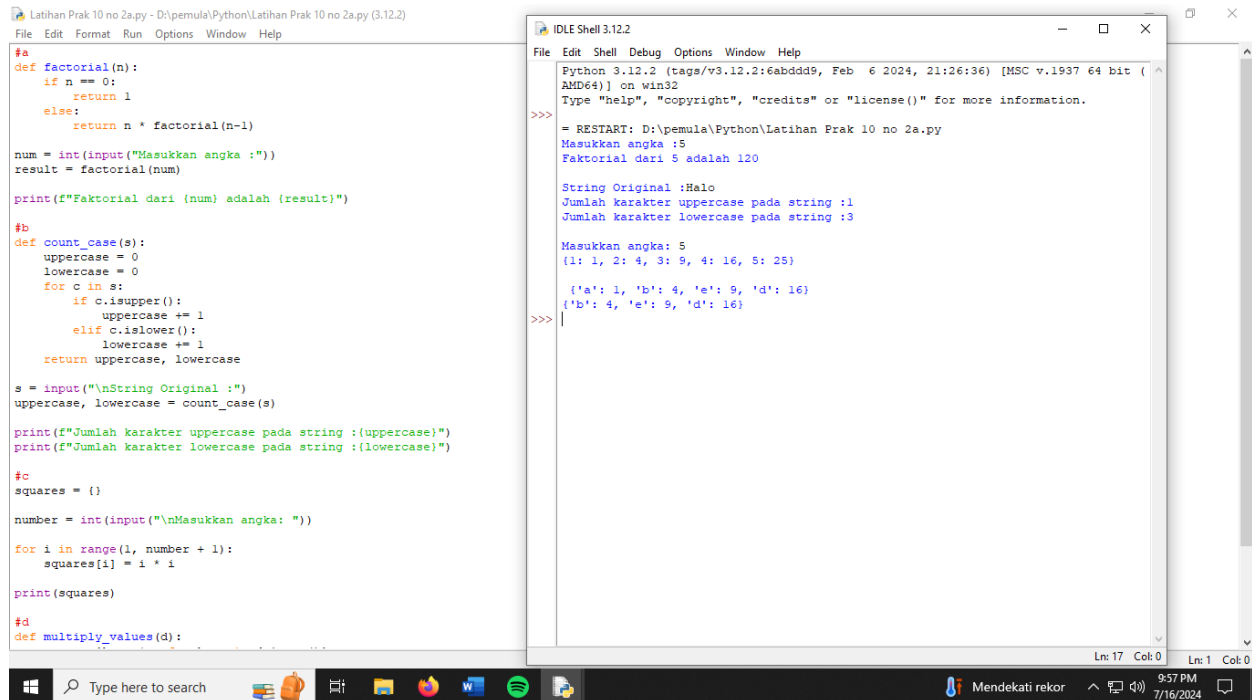
```
def printme(str):  
    "This prints a passed string into this function"  
    print(str)  
    return  
printme("Hello world")  
  
def jumlah(angka):  
    total = 0  
    for x in angka:  
        total += x  
    return total  
print("Jumlah :", jumlah((0,3,1,4,5)))  
  
def cek_ganjil_genap(angka):  
    if angka%2==0:  
        print("genap")  
    else:  
        print("ganjil")  
    return  
cek_ganjil_genap(5)  
  
def rata_rata(a,b,c):  
    return (a+b+c)/3  
rata_rata(1,2,3)  
  
def kalkulator(angka1,angka2):  
    print(angka1+angka2)  
    print(angka1-angka2)  
    print(angka1*angka2)  
    print(angka1/angka2)  
kalkulator(1,2)
```

The right window, titled 'IDLE Shell 3.12.2', shows the output of the script after execution:

```
Python 3.12.2 (tags/v3.12.2:6abddd9, Feb  6 2024, 21:26:36) [MSC v.1937 64 bit (AMD64)] on win32  
Type "help", "copyright", "credits" or "license()" for more information.  
>>>  
===== RESTART: D:\pemula\Python\Latihan Prak 10a.py =====  
hello world  
Jumlah : 21  
ganjil  
3  
-1  
2  
0.5  
>>>
```

The Windows taskbar at the bottom shows the system clock as 9:57 PM on 7/16/2024, with a temperature of 26°C and the word 'Cerah' (Clear).

2.



```
#a
def factorial(n):
    if n == 0:
        return 1
    else:
        return n * factorial(n-1)

num = int(input("Masukkan angka :"))
result = factorial(num)

print(f"Faktorial dari (num) adalah (result)")

#b
def count_case(s):
    uppercase = 0
    lowercase = 0
    for c in s:
        if c.isupper():
            uppercase += 1
        elif c.islower():
            lowercase += 1
    return uppercase, lowercase

s = input("\nString Original :")
uppercase, lowercase = count_case(s)

print(f"Jumlah karakter uppercase pada string :{uppercase}")
print(f"Jumlah karakter lowercase pada string :{lowercase}")

#c
squares = {}

number = int(input("\nMasukkan angka: "))

for i in range(1, number + 1):
    squares[i] = i * i

print(squares)

#d
def multiply_values(d):
```

```
Python 3.12.2 (tags/v3.12.2:6abddd9, Feb  6 2024, 21:26:36) [MSC v.1937 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.

>>>
= RESTART: D:\pemula\Python\Latihan Prak 10 no 2a.py
Masukkan angka :5
Faktorial dari 5 adalah 120

String Original :Halo
Jumlah karakter uppercase pada string :1
Jumlah karakter lowercase pada string :3

Masukkan angka: 5
{1: 1, 2: 4, 3: 9, 4: 16, 5: 25}

{'a': 1, 'b': 4, 'e': 9, 'd': 16}
{'b': 4, 'e': 9, 'd': 16}
>>>
```

Analisis program

Pendekatan berbasis fungsi dalam Python sangat bermanfaat untuk menulis kode yang modular, dapat digunakan kembali, mudah diuji, dan efisien. Dengan mendefinisikan fungsi untuk tugas-tugas spesifik, kita dapat membuat program yang lebih terstruktur dan lebih mudah dipelihara. Penggunaan fungsi-fungsi ini tidak hanya menyederhanakan pengembangan dan pemeliharaan kode tetapi juga membantu dalam memastikan performa optimal dan pengelolaan kompleksitas yang lebih baik.

Referensi

<https://www.w3schools.com/python/>