Laporan Praktikum Algoritma dan Pemrograman

Semester Genap 2023/2024

NIM	**71230977**
Nama Lengkap	**MICHAEL HOSEA**
Minggu ke / Materi	**07/ PENGOLAHAN STRING**

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 1: Pengantar String

String adalah salah satu tipe data yang fundamental dalam pemrograman komputer. Dalam pemrograman, string adalah serangkaian karakter yang diurutkan, yang dapat berupa huruf, angka, atau karakter khusus lainnya. String dapat digunakan untuk merepresentasikan teks, baik dalam bentuk pesan yang ditampilkan kepada pengguna, input dari pengguna, atau data yang diproses oleh program.

String biasanya diapit oleh tanda kutip (baik tunggal maupun ganda), tergantung pada bahasa pemrograman yang digunakan. Misalnya, dalam bahasa pemrograman Python, sebuah string dapat dinyatakan dengan menggunakan tanda kutip tunggal atau ganda, seperti ini:

```
my_string = "Ini adalah contoh string.”
```

Atau kita juga dapat membuat menjadi seperti ini:

```
my_string = ‘Ini adalah contoh string.’
```

```
namasaya = "Antonius Rachmat C"
temansaya1 = "Yuan Lukito"
temansaya2 = 'Laurentius Kuncoro'
temansaya3 = "Matahari" + 'Bakti'

print(temansaya3)
print(namasaya[0]) #'A'
print(namasaya[9]) #'R'
print(temansaya1[1]) #'u'
huruf = temansaya2[0]
print(huruf) #'L'
```

Program Python yang Anda berikan adalah contoh sederhana penggunaan string dan pengindeksan (indexing) dalam Python. Berikut adalah hasil keluaran dari program tersebut:

```
MatahariBakti
A
R
u
```

```
L
```

Penjelasan:

- Pada baris 3, variabel `temansaya3` diinisialisasi dengan hasil penggabungan string "Matahari" dan "Bakti". Sehingga, ketika diprint, akan menghasilkan keluaran "MatahariBakti".
- Pada baris 4, menggunakan indeks 0 dari string `namasaya`, sehingga akan menghasilkan huruf pertama dari nama, yaitu "A".
- Pada baris 5, menggunakan indeks 9 dari string `namasaya`, sehingga akan menghasilkan huruf ke-10 dari nama, yaitu "R".
- Pada baris 6, menggunakan indeks 1 dari string `temansaya1`, sehingga akan menghasilkan huruf kedua dari nama teman, yaitu "u".
- Pada baris 7, nilai dari huruf pertama dari string `temansaya2` disimpan dalam variabel `huruf`, dan kemudian diprint. Ini akan menghasilkan huruf pertama dari nama teman, yaitu "L".

Materi 2: Operator dan Metode String

- OPERATOR “in”

  Test case ini menguji apakah karakter tertentu ada dalam string **string**. Hasilnya akan **True** jika karakter tersebut ditemukan dalam string, dan **False** jika tidak.

```python
string = "hello"
print("h" in string)  # True
print("z" in string)  # False
```

- FUNGSI “len”

  Test case ini menggunakan fungsi **len()** untuk menghitung panjang (jumlah karakter) dari string **string**. Hasilnya akan menunjukkan panjang string tersebut.

```python
string = "hello"
panjang = len(string)
print(panjang)  # 5
```

- TRANVERSING STRING

  Test case ini menggunakan loop **for** untuk mengiterasi melalui setiap karakter dalam string **string** dan mencetaknya satu per satu.

```python
string = "Melia"
for karakter in string:
    print(karakter)

# Output:
```

```
# M
# e
# l
# i
# a
```

- STRING SLICE
  Test case ini menggunakan string slice untuk mengambil bagian tertentu dari string **string**, yaitu dari indeks 1 hingga indeks 2 (indeks terakhir tidak termasuk).

```
string = "hello kawan"
awal = 0
ahkir = 6
slice = string[awal:ahkir] # "hello"
slice = string[7:len(string)] # "kawan"
slice = string[1:3] # "el"
slice = string[:3] # "hel"
slice = string[2:] # "llo kawan"
slice = string[-1] # "n"
slice = string[:] # "hello kawan"
```

- OPERATOR " * " DAN " + " PADA STRING
  Test case ini menggabungkan dua string menggunakan operator **+**, dan mengulangi string **string1** sebanyak tiga kali menggunakan operator *****.

```
string1 = "hello"
string2 = "world"
gabungan = string1 + " " + string2
ulang = string1 * 3
print(gabungan)  # "hello world"
print(ulang)  # "hellohellohello"
```

- PARSING STRING
  Test case ini menggunakan metode **split()** untuk memecah string **string** menjadi bagian-bagian yang dipisahkan oleh koma. Hasilnya adalah list yang berisi bagian-bagian tersebut.

```
string = "satu,dua,tiga,empat"
pecah = string.split(",")
print(pecah)  # ['satu', 'dua', 'tiga', 'empat']
```

MATERI 3: Metode String

1. Metode upper() dan lower()

Metode ini digunakan untuk mengubah string menjadi huruf kapital atau huruf kecil.

```
string = "Hello, World!"
kapital = string.upper()
kecil = string.lower()

print(kapital)  # "HELLO, WORLD!"
print(kecil)    # "hello, world!"
```

2. Metode strip()
   Metode ini menghapus karakter spasi (atau karakter lain yang disebut) dari awal dan akhir string.

```
string = "   Hello, World!   "
bersih = string.strip()

print(bersih)  # "Hello, World!"
```

3. Metode replace()
   Metode ini menggantikan setiap kemunculan string dengan string lain.

```
string = "Hello, World!"
ganti = string.replace("Hello", "Hi")
print(ganti)  # "Hi, World!"
```

MATERI 4: Format String

1. Format String dengan “%”
   Digunakan untuk menyematkan nilai ke dalam string dengan format tertentu.

```
nama = "John"
umur = 25
teks = "Nama: %s, Umur: %d" % (nama, umur)
print(teks)  # "Nama: John, Umur: 25"
```

2. Format String dengan format()
   Metode yang lebih modern untuk menyisipkan nilai ke dalam string.

```python
nama = "John"
umur = 25
teks = "Nama: {}, Umur: {}".format(nama, umur)

print(teks)  # "Nama: John, Umur: 25"
```

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

Buatlah sebuah program yang dapat mendeteksi apakah suatu kata adalah anagram dari kata lainnya atau bukan! Anagram adalah kata yang dibolak-balik susunan hurufnya sama. Misal: mata anagram dengan atma, maat, taam, tama, dsb.

- Source Code dan Output

```
def cek_anagram(word1, word2):
    # Mengubah kata menjadi lowercase dan menghapus spasi
    word1 = word1.lower().replace(" ", "")
    word2 = word2.lower().replace(" ", "")

    # Mengurutkan huruf-huruf dalam kata
    sorted_word1 = sorted(word1)
    sorted_word2 = sorted(word2)

    # Membandingkan kata yang sudah diurutkan
    if sorted_word1 == sorted_word2:
        return True
    else:
        return False

# Contoh penggunaan
kata1 = input("Masukkan kata pertama: ")
kata2 = input("Masukkan kata kedua: ")

if is_anagram(kata1, kata2):
    print("Kedua kata adalah anagram.")
else:
    print("Kedua kata bukan anagram.")
```

- Penjelasan

  Fungsi cek_anagram(word1, word2) yang telah Anda buat adalah untuk mengecek apakah dua kata yang dimasukkan pengguna merupakan anagram atau bukan dengan cara:

1. Mengubah ke lowercase dan menghapus spasi:
   - Kedua kata yang dimasukkan diubah menjadi lowercase dengan .lower() agar perbedaan huruf besar dan kecil tidak mempengaruhi hasil.
   - Selanjutnya, spasi dalam kata dihapus dengan .replace(" ", "") agar spasi tidak mempengaruhi perbandingan.
2. Mengurutkan huruf dalam kata:
   - Setelah itu, setiap karakter dalam kata diurutkan dengan sorted(). Ini membuat urutan huruf dalam kata tidak mempengaruhi hasil anagram.
3. Membandingkan kata yang sudah diurutkan:
   - Setelah kedua kata diubah dan diurutkan, fungsi membandingkan kedua kata yang sudah diurutkan. Jika kedua kata sama setelah diurutkan, maka fungsi mengembalikan True, jika tidak, maka mengembalikan False.

SOAL 2

Buatlah suatu program yang dapat menghitung frekuensi kemunculan suatu kata yang ada pada String. Misal terdapat kalimat "Saya mau makan. Makan itu wajib. Mau siang atau malam saya wajib makan". Ditanyakan kata "makan". Output: makan ada 3 buah

- Source Code dan Output

```
def hitung_frekuensi_kata(kalimat, kata):
    # Mengubah kalimat menjadi lowercase untuk menghindari
perbedaan case
    kalimat = kalimat.lower()
    # Menghitung frekuensi kemunculan kata
    frekuensi = kalimat.count(kata.lower())
    return frekuensi

kalimat = input("Masukkan kalimat: ")
kata = input("Masukkan kata yang ingin dihitung: ")

frekuensi = hitung_frekuensi_kata(kalimat, kata)
print(f"{kata} ada {frekuensi} buah")
```

- Penjelasan

  Fungsi hitung_frekuensi_kata(kalimat, kata) yang Anda buat berfungsi untuk menghitung berapa kali sebuah kata tertentu muncul dalam sebuah kalimat dengan cara:

  1. Mengubah kalimat menjadi lowercase:
     - Sebelum menghitung frekuensi, kalimat diubah menjadi lowercase dengan .lower() untuk menghindari perbedaan huruf besar dan kecil saat menghitung.
  2. Menghitung frekuensi kemunculan kata:

- Fungsi count() digunakan untuk menghitung berapa kali kata yang dicari muncul dalam kalimat. kata.lower() digunakan untuk memastikan perbedaan huruf besar dan kecil tidak mempengaruhi hasil.

3. Mengembalikan nilai frekuensi:
   - Fungsi kemudian mengembalikan jumlah frekuensi kemunculan kata dalam kalimat.

SOAL 3

Buatlah suatu program yang dapat menghapus semua spasi yang berlebih pada sebuah string, dan menjadikannya satu spasi normal! Misal: "saya tidak suka memancing ikan " Output: "saya tidak suka memancing ikan".

- Source Code dan Output

```
def hapus_spasi_berlebih(string):
    string = ' '.join(string.split())
    return string

# Contoh penggunaan
kalimat = input("Masukkan kalimat: ")
kalimat_terhapus_spasi = hapus_spasi_berlebih(kalimat)
print(kalimat_terhapus_spasi)
```

- Penjelasan

  Fungsi hapus_spasi_berlebih(string) yang Anda buat berfungsi untuk menghapus spasi berlebih dalam sebuah string dengan cara:

1. Mengubah string menjadi list kata:
   - Fungsi split() digunakan untuk memecah string menjadi list kata-kata. Secara default, pemisah adalah spasi.
2. Menggabungkan kembali kata-kata tanpa spasi berlebih:
   - List kata-kata yang dihasilkan dari split() kemudian digabungkan kembali menjadi string dengan menggunakan ' '.join() untuk menggabungkan kata-kata dengan satu spasi di antara setiap kata.

SOAL 4

Buatlah suatu program mengetahui kata terpendek dan terpanjang dari suatu kalimat yang diinputkan! Misal: "red snakes and a black frog in the pool" Output: terpendek: a, terpanjang: snakes

- Source Code dan Output

```
kalimat = input("Masukkan kalimat: ")
```

```
# Memisahkan kata-kata dalam kalimat
kata = kalimat.split()

# Menginisialisasi kata terpendek dan terpanjang dengan kata
pertama dalam kalimat
kata_terpendek = kata[0]
kata_terpanjang = kata[0]

# Mencari kata terpendek dan terpanjang
for k in kata:
    if len(k) < len(kata_terpendek):
        kata_terpendek = k
    if len(k) > len(kata_terpanjang):
        kata_terpanjang = k

# Menampilkan hasil
print("Kata terpendek:", kata_terpendek)
print("Kata terpanjang:", kata_terpanjang)
```

- Penjelasan

Program ini bertujuan untuk mencari kata terpendek dan terpanjang dalam sebuah kalimat.

1. Memisahkan kata-kata dalam kalimat:
- Kalimat yang dimasukkan oleh pengguna dipecah menjadi sebuah list kata menggunakan fungsi split(). Ini memisahkan kalimat menjadi kata-kata berdasarkan spasi.
2. Menginisialisasi kata terpendek dan terpanjang:
- Kata terpendek dan terpanjang diinisialisasi dengan kata pertama dalam kalimat (kata[0]) karena pada awalnya belum ada kata lain untuk dibandingkan.
3. Mencari kata terpendek dan terpanjang:
- Program kemudian melakukan iterasi melalui setiap kata dalam list kata.
- Untuk setiap kata, program membandingkan panjangnya dengan panjang kata_terpendek dan kata_terpanjang saat ini.
- Jika panjang kata saat ini lebih pendek dari kata_terpendek, maka kata_terpendek diubah menjadi kata tersebut.
- Jika panjang kata saat ini lebih panjang dari kata_terpanjang, maka kata_terpanjang diubah menjadi kata tersebut.
4. Menampilkan hasil:
- Setelah selesai mencari, program akan menampilkan kata terpendek dan terpanjang yang telah ditemukan.