Laporan Praktikum Algoritma dan Pemrograman

Semester Genap 2023/2024

NIM	**Josephine Marcelia**
Nama Lengkap	**71231005**
Minggu ke / Materi	**05 / Struktur Kontrol Perulangan**

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

Definisi Perulangan

Perulangan adalah struktur kontrol yang mengulangi suatu blok kode berdasarkan kondisi yang telah ditentukan. Hal ini memungkinkan kita untuk melakukan tugas yang sama berkali-kali tanpa harus menulis ulang kode secara manual. Dalam bahasa pemrograman python, perulangan biasanya digunakan untuk mengeksekusi tugas yang berulang atau berulang kali.

Ada 2 perulangan yang umum digunakan, yaitu for dan while.

- For loop : Kita gunakan Ketika kita sudah mengetahui berapa kali iterasi akan terjadi dan kapan program akan berhenti. Perulangan for akan melakukan iterasi sebanyak elemen dalam urutan atau range yang ditentukan.
- While Loop : Kita gunakan unakan ketika kondisi berhenti tidak diketahui sebelumnya dan bergantung pada evaluasi suatu kondisi. Dimana loop akan terus berjalan selama kondisi evaluasi benar.

Perulangan for

Perulangan for biasanya digunakan pada kondisi:
• Jumlah perulangan sudah diketahui sejak awal.
• Perulangan terjadi karena operasi yang sama pada suatu rentang data atau rentang nilai

Contoh :

1. Program untuk mencetak angka dari satu sampai sepuluh

```
for i in range(1, 11):
    print(f"Angka ke-{i} adalah {i}")
```

Pada program diatas, kita menggunakan range(), yang dimulai dari 1 (start) sampai 11 (stop). Kemudian, perulangan akan dilakukan untuk setiap angka dalam deret tersebut yang menghasilkan output angka 1 sampai 10 seperti berikut:

```
Angka ke-1 adalah 1
Angka ke-2 adalah 2
Angka ke-3 adalah 3
Angka ke-4 adalah 4
Angka ke-5 adalah 5
Angka ke-6 adalah 6
Angka ke-7 adalah 7
Angka ke-8 adalah 8
Angka ke-9 adalah 9
Angka ke-10 adalah 10
```

2. Program untuk menentukan kelipatan bilangan

```python
for i in range(1, 101):
    if i % 3 == 0 and i % 5 == 0 and i % 7 == 0:
        print("FizzBuzz")
    elif i % 3 == 0 and i % 5 == 0:
        print("Fizz")
    elif i % 3 == 0 and i % 7 == 0:
        print("FizzBuzz")
    elif i % 5 == 0 and i % 7 == 0:
        print("Buzz")
    elif i % 3 == 0:
        print("Fizz")
    elif i % 5 == 0:
        print("Buzz")
    elif i % 7 == 0:
        print("FizzBuzz")
```

Pada program diatas kita akan mencetak deret bilangan bulat dari 1 hingga 100, tetapi untuk bilangan yang merupakan kelipatan 3, mencetak "Fizz", bilangan yang merupakan kelipatan 5, mencetak "Buzz", dan untuk bilangan yang merupakan kelipatan 7, mencetak "FizzBuzz"

Output yang dihasilkan adalah :

```
FizzBuzz
Buzz
Fizz
Fizz
FizzBuzz
Fizz
Buzz
Fizz
FizzBuzz
Fizz
Buzz
```

Perulangan While

Bentuk umum perulangan while adalah :

<start>
While <stop> :
peration
operation
<step (optional)>

Contoh : Program untuk menentukan bilangan prima atau bukan

```python
def cek_prima(bilangan):
    prima = True
    if bilangan < 2:
        prima = False
    else:
        i = 2
        while i < bilangan:
            if bilangan % i == 0:
                prima = False
                break
            i += 1
    return prima

bilangan = int(input("Masukkan bilangan: "))
if cek_prima(bilangan):
    print(f"{bilangan} adalah bilangan prima.")
else:
    print(f"{bilangan} bukan bilangan prima.")
```

Pada fungsi cek_prima diatas, kita akan menerima sebuah bilangan bulat sebagai argumen. Kemudian fungsi akan melakukan perulangan sampai nilai variabel i kurang dari nilai argumen. Jika argumen dapat dibagi habis oleh nilai variabel i, maka variabel prima akan diubah menjadi False. Jika variabel prima masih True, maka argumen adalah bilangan prima.

Setelah itu, program akan meminta inputan bilangan dari pengguna. Bilangan yang dimasukkan oleh pengguna akan dicek oleh fungsi cek_prima. Jika hasil dari fungsi cek_prima adalah True, maka bilangan tersebut adalah bilangan prima. Jika hasil dari fungsi cek_prima adalah False, maka bilangan tersebut bukan bilangan prima.

Output :

```
Masukkan bilangan: 5
5 adalah bilangan prima.
```

```
Masukkan bilangan: 12
12 bukan bilangan prima.
```

Break dan Continue

Break
Perintah ini kita gunakan untuk menghentikan perulangan secara terpaksa. Jika perintah break dipanggil, perulangan akan segera dihentikan dan program akan melanjutkan eksekusi setelah perulangan.

Contoh :

```
for i in range(1, 11):
    if i == 5:
        break
    else:
        print(i)
print('Selesai')
```

Dalam codingan di atas, perintah break kita gunakan untuk melewati iterasi ketika nilai variabel i sama dengan 5 dan menghentikan perulangan. Setelah perulangan dihentikan, perintah print('Selesai') akan dicetak ke layar.

Output :

```
1
2
3
4
Selesai
```

Continue
Perintah ini kita gunakan untuk melewati satu iterasi perulangan dan melanjutkan ke iterasi berikutnya. Jika perintah continue dipanggil, semua perintah yang ada setelah perintah continue tidak akan dieksekusi dan perulangan akan segera dimulai ulang dengan nilai variabel yang baru.

Contoh :

```
for i in range(1, 11):
    if i == 5:
        continue
    else:
        print(i)
print('Selesai')
```

Dalam codingan di atas, perintah continue kita gunakan untuk melewati iterasi ketika nilai variabel i sama dengan 5 dan melanjutkan ke iterasi berikutnya. Namun, perintah else akan

dicetak setelah setiap iterasi, kecuali jika perintah continue dipanggil. Setelah perulangan selesai, perintah print('Selesai') akan dicetak ke layar.

Output:

```
1
2
3
4
6
7
8
9
10
Selesai
```

Mengubah While Menjadi For

Untuk mengubah perulangan for menjadi while, kita perlu menginisialisasi variabel kontrol, mengatur kondisi perulangan, dan mengupdate variabel kontrol setiap iterasi.

Contoh perulangan for:

```
for i in range(5):
    print(i)
```

Untuk mengubahnya menjadi perulangan while, kita dapat menggunakan kode berikut:

```
i = 0
while i < 5:
    print(i)
    i += 1
```

Pada contoh diatas, dapat diperhatikan bahwa perubahan tersebut melibatkan beberapa Langkah, yaitu:

- Inisialisasi variabel kontrol iterasi di luar loop while (i = 0).
- Gantilah for I in range(5) menjadi while i < 5
- Pindahkan seluruh kode blok loop ke dalam loop while
- Tambahkan Langkah iterasi di dalam loop while (i +=1)

LATIHAN MANDIRI

Latihan 5.1

```
def perkalian(a, b):
    hasil = 0
    for i in range(b):
        hasil += a
    return hasil

a1 = int(input("Masukkan bilangan pertama: "))
a2 = int(input("Masukkan bilangan kedua: "))
hasil_perkalian = perkalian(a1,a2)

print(f"{a1} x {a2} = ", end='')
for i in range(a2):
    print(a1, end='')
    if i < a2 - 1:
        print(" + ", end='')
    else:
        print(f" = {hasil_perkalian}")
```

```
masukkan bilangan pertama: 6
Masukkan bilangan kedua: 5
6 x 5 = 6 + 6 + 6 + 6 + 6 = 30
```

Penjelasan

Pada program def perkalian diatas saya menggunakan 2 parameter yaitu a dan b

hasil = 0 merupakan inisialisasi variabel hasil dengan nilai 0 yang akan digunakan untuk menyimpan hasil perkalian.

for i in range(b) digunakan untuk melakukan iterasi sebanyak b kali yang artinya loop akan berjalan sebanyak nilai b.

hasil += a artinya pada setiap iterasi, nilai a ditambahkan ke variabel hasil. Yang berarti program akan melakukan penjumlahan berulang sebanyak b kali.

Fungsi return hasil saya gunakan untuk mengembalikan nilai hasil setelah proses perkalian selesai.

print(f"{a1} x {a2} = ", end='') saya gunakan untuk mencetak bilangan pertama, operator perkalian, dan bilangan kedua tanpa berpindah baris.

for i in range(a2) saya gunakan untuk mencetak bilangan pertama sebanyak bilangan kedua.

print(a1, end='') saya gunakan untuk mencetak bilangan pertama.

if i < a2 – 1 saya gunakan utnuk mengecek apakah iterasi belum mencapai iterasi terakhir.

print(" + ", end='') artinya jika belum mencapai iterasi terakhir, maka program akan mencetak tanda tambah.

Latihan 5.2

```
def ganjil(bawah, atas):
    if bawah < atas:
        for i in range(bawah, atas + 1):
            if i % 2 != 0:
                print(i, end=" ")
    else:
        for i in range(bawah, atas - 1, -1):
            if i % 2 != 0:
                print(i, end=" ")
bawah = int(input("Masukkan batas bawah: "))
atas = int(input("Masukkan batas atas: "))
print("Deret bilangan ganjil:")
ganjil(bawah, atas)
```

```
Masukkan batas bawah: 10
Masukkan batas atas: 30
Deret bilangan ganjil:
11 13 15 17 19 21 23 25 27 29
```

```
Masukkan batas bawah: 97
Masukkan batas atas: 82
Deret bilangan ganjil:
97 95 93 91 89 87 85 83
```

Penjelasan :

Pada program diatas fungsi ganjil, kondisi if bawah < atas saya gunakan untuk memeriksa apakah batas bawah (bawah) lebih kecil dari batas atas (atas). Jika benar, maka program akan melakukan iterasi dari batas bawah hingga batas atas dan setiap bilangan yang diperoleh akan dicek apakah bilangan tersebut ganjil atau tidak.
Jika kondisi di atas tidak terpenuhi (artinya batas bawah lebih besar dari batas atas), maka akan dilakukan iterasi dari batas bawah hingga batas atas (termasuk) dengan langkah yang bernilai negatif. Setiap bilangan yang diperoleh akan dicek apakah bilangan tersebut ganjil atau tidak.

Latihan 5.3

```python
def hitung_ips(jumlah_matkul):
    total_bobot = 0
    for i in range(jumlah_matkul):
        nilai = input(f"Masukkan nilai mata kuliah ke-{i+1} (A/B/C/D): ").upper()
        if nilai == "A":
            total_bobot += 4
        elif nilai == "B":
            total_bobot += 3
        elif nilai == "C":
            total_bobot += 2
        elif nilai == "D":
            total_bobot += 1
        else:
            print("Input tidak valid. Masukkan nilai A/B/C/D.")
    if jumlah_matkul > 0:
        ips = total_bobot / jumlah_matkul
        print(f"Indeks Prestasi Semester (IPS) Anda adalah: {ips:.2f}")
    else:
        print("Tidak ada mata kuliah yang diinput.")
jumlah_matkul = int(input("Masukkan jumlah mata kuliah: "))
hitung_ips(jumlah_matkul)
```

```
Masukkan jumlah mata kuliah: 6
Masukkan nilai mata kuliah ke-1 (A/B/C/D): A
Masukkan nilai mata kuliah ke-2 (A/B/C/D): B
Masukkan nilai mata kuliah ke-3 (A/B/C/D): C
Masukkan nilai mata kuliah ke-4 (A/B/C/D): A
Masukkan nilai mata kuliah ke-5 (A/B/C/D): D
Masukkan nilai mata kuliah ke-6 (A/B/C/D): C
Indeks Prestasi Semester (IPS) Anda adalah: 2.67
```

Penjelasan :

Pada program diatas, saya menggunakan fungsi hitung_ips(jumlah_matkul) dengan mengambil satu argumen, jumlah_matkul, yang merepresentasikan jumlah mata kuliah yang akan diinput oleh pengguna.

Kemudian fungsi hitung_ips akan melakukan iterasi sebanyak jumlah_matkul dan setiap iterasi akan meminta pengguna untuk memasukkan nilai mata kuliah ke-i+1. Nilai yang diinput akan diubah menjadi huruf kapital menggunakan upper().

Setiap nilai yang diinput akan dicek apakah sama dengan "A", "B", "C", atau "D". Jika sama, maka nilai tersebut akan dijumlahkan ke total_bobot dengan bobot yang sesuai (4 untuk "A", 3 untuk "B", 2 untuk "C", dan 1 untuk "D").

Jika nilai yang diinput tidak valid, maka akan dicetak pesan “Input tidak valid. Masukkan nilai A/B/C/D.”

Jika jumlah mata kuliah lebih dari 0, maka total_bobot akan dibagi dengan jumlah_matkul untuk menghitung IPS. Hasil perhitungan akan dicetak dengan dua digit di belakang koma.

Pada akhir fungsi hitung_ips, program diatas akan meminta pengguna untuk memasukkan jumlah mata kuliah. Nilai yang dimasukkan oleh pengguna akan diubah menjadi tipe data integer menggunakan fungsi int().

Setelah pengguna memasukkan jumlah mata kuliah, fungsi hitung_ips(jumlah_matkul) akan dipanggil dengan argumen yang telah diberikan oleh pengguna.

Link Github :

https://github.com/Josephinemrc/Tugas5-PrakAlpro.git