Kelas Kolaborasi Pemrograman Python

Program Praktisi Mengajar Batch 4 2024 Universitas Amikom Yogyakarta

Muhammad Oriza Nurfajri S.Kom., M.IT.

Sesi 1 29 April 2024 07.30 - 10.30 Durasi 180 Menit Per Sesi

1. Tujuan Pembelajaran

Setelah menyelesaikan modul ini, mahasiswa diharapkan mampu:

- Memahami konsep percabangan (IF) dalam pemrograman Python.
- Menerapkan berbagai jenis percabangan (IF) dalam program Python.
- Menyusun program Python yang menggunakan percabangan untuk menyelesaikan berbagai kasus.

2. Konsep Percabangan (IF)

Percabangan (IF) adalah struktur kontrol program yang memungkinkan program untuk mengeksekusi kode yang berbeda berdasarkan kondisi tertentu. Percabangan IF memungkinkan program untuk membuat keputusan dan mengambil tindakan yang berbeda berdasarkan situasi yang berbeda.

3. Operators

Operator	Meaning	Sample Condition	Evaluates To
==	equal to	5 == 5	True
!=	not equal to	8 != 5	True
>	greater than	3 > 10	False
<	less than	5 < 8	True
>=	greater than or equal to	5 >= 10	False
<=	less than or equal to	5 <= 5	True

Statement	Description
if <condition>: <block>	if statement. If <condition> is true, <block> is executed; otherwise it's skipped.

if <condition>:
  <block 1>
else:
  <block 2>
	if statement with else clause. If <condition> is true, <block1> is executed; otherwise <block2> is executed.

if <condition 1>:
  <block 1>
elif <condition 2>:
  <block 2>
.
.
.
elif <condition N>:
  <block N>
else:
  <block N+1>
	if statement with elif clauses and optional final else clause. The block of the first true condition is executed. If no condition is true, the optional else clause's block is executed.

4. Jenis-jenis Percabangan (IF)

- **Percabangan IF Sederhana**

```
if kondisi:
    kode_jika_benar
else:
    kode_jika_salah
```

- **Percabangan IF Elif**

```
if kondisi1:
    kode_jika_benar1
elif kondisi2:
    kode_jika_benar2
else:
    kode_jika_salah
```

- **Percabangan IF Multi Elif**

```
if kondisi1:
    kode_jika_benar1
elif kondisi2:
    kode_jika_benar2
elif kondisi3:
    kode_jika_benar3
else:
    kode_jika_salah
```

- **Kasus 1**

```
nilai = int(input("Masukkan nilai ujian: "))

if nilai >= 75:
    print("Selamat, Anda lulus!")
else:
    print("Maaf, Anda tidak lulus.")
```

- **Kasus 2**

```
bil1 = float(input("Masukkan bilangan pertama: "))
bil2 = float(input("Masukkan bilangan kedua: "))

if bil1 > bil2:
    print(f"{bil1} adalah nilai terbesar")
else:
    print(f"{bil2} adalah nilai terbesar")
```

- **Kasus 3**

```
total_pembelian = float(input("Masukkan total pembelian: "))

if total_pembelian < 50000:
    diskon = 0
elif total_pembelian < 100000:
    diskon = total_pembelian * 0.1
else:
    diskon = total_pembelian * 0.15

total_bayar = total_pembelian - diskon

print(f"Total diskon: Rp{diskon:,.2f}")
print(f"Total bayar: Rp{total_bayar:,.2f}")
```

- ## Nested in Nested IF

```
if (kondisi_luar):
  if (kondisi_dalam):
    // Pernyataan yang dijalankan jika kedua kondisi benar
  elif:
    // Pernyataan yang dijalankan jika kondisi_dalam salah
else:
  // Pernyataan yang dijalankan jika kondisi_luar salah
```

- **Kasus**

```
total_pembelian = float(input("Masukkan total pembelian: "))

if total_pembelian > 200000:
    if(total_pembelian >= 1000000):
        diskon = 0.25 * total_pembelian
    elif(total_pembelian >= 500000):
        diskon = 0.20 * total_pembelian
    else:
        diskon = 0.15 * total_pembelian
elif total_pembelian > 100000:
    diskon = 0.1 * total_pembelian
else:
    diskon = 0

total_bayar = total_pembelian - diskon

# Menampilkan hasil
print(f"Total pembelian: {total_pembelian}")
print(f"Diskon: {diskon}")
print(f"Total bayar: {total_bayar}")
```

- ## LATIHAN Multi IF (Bukan IF - ELIF - ELSE)

Buatlah studi kasus program dengan Multi IF

PERULANGAN / LOOP

Tujuan Pembelajaran

Setelah menyelesaikan modul ini, mahasiswa diharapkan mampu:

- Memahami konsep perulangan (loop) dalam pemrograman Python.
- Menerapkan perulangan **for** untuk mengulangi urutan elemen.
- Menggunakan perulangan **while** untuk mengulangi blok kode berdasarkan kondisi tertentu.
- Menjelaskan perbedaan antara perulangan **for** dan **while**.
- Mengidentifikasi kasus penggunaan yang tepat untuk setiap jenis perulangan.
- Menulis kode Python yang efisien dan efektif dengan menggunakan perulangan.

Konsep Perulangan (Loop)

Perulangan (loop) adalah struktur kontrol yang memungkinkan blok kode dieksekusi berulang kali. Hal ini berguna untuk mengotomatiskan tugas yang berulang, seperti memproses elemen dalam urutan atau mengulangi operasi sampai kondisi tertentu terpenuhi.

Jenis-jenis Perulangan

Python menyediakan dua jenis perulangan utama:

- **Perulangan for:** Digunakan untuk mengulangi urutan elemen, seperti daftar, string, atau tuple.
- **Perulangan while:** Digunakan untuk mengulangi blok kode berdasarkan kondisi tertentu.

FOR Basic

```
for variabel_loop in urutan:
    blok_kode
```

Kasus

```
angka_list = [1, 2, 3, 4, 5]

for angka in angka_list:
    print(angka)
```

- ## WHILE Basic

```
while kondisi:
    blok_kode
```

- ## Kasus

```
angka = 1

while angka <= 5:
    print(angka)
    angka += 1
```

- ## Perbedaan Perulangan for dan while

Perbedaan utama antara perulangan for dan while adalah:

- **Perulangan for** digunakan untuk mengulangi urutan elemen yang diketahui, sedangkan **perulangan while** digunakan untuk mengulangi blok kode berdasarkan kondisi tertentu.
- **Perulangan for** memiliki jumlah iterasi yang pasti, sedangkan **perulangan while** memiliki jumlah iterasi yang tidak pasti dan tergantung pada kondisi.

- ## Kasus Penggunaan

- Gunakan perulangan for untuk mengulangi urutan elemen yang diketahui, seperti memproses daftar data, mengiterasi string, atau menghitung dari 1 sampai n.
- Gunakan perulangan while untuk mengulangi blok kode berdasarkan kondisi yang tidak pasti, seperti menunggu input pengguna, memeriksa status file, atau melakukan iterasi tanpa batas.

- ## Variasi Penggunaan For

- Mengiterasi Elemen dalam Urutan

```
angka_list = [1, 2, 3, 4, 5]

for angka in angka_list:
  print(angka)
```

- Mengiterasi Indeks Elemen dalam Urutan

```
angka_list = [1, 2, 3, 4, 5]

for i in range(len(angka_list)):
  print(f"Indeks: {i}, Nilai: {angka_list[i]}")
```

- Mengiterasi Nilai dan Kunci dalam Kamus

```
mahasiswa = {"nama": "Budi", "umur": 20, "jurusan": "Teknik Informatika"}

for key, value in mahasiswa.items():
  print(f"Kunci: {key}, Nilai: {value}")
```

- Mengiterasi String dengan Perulangan 'for'

```
kata = "Hello, World!"

for karakter in kata:
  print(karakter)
```

- Menggunakan 'in' untuk Memeriksa Keanggotaan

```
kata = "Hello, World!"
karakter_cari = "o"

for karakter in kata:
  if karakter == karakter_cari:
    print(f"Karakter '{karakter_cari}' ditemukan")
```

• Variasi Penggunaan While

- Mengulangi Blok Kode Sampai Kondisi Terpenuhi

```
input_pengguna = ""

while input_pengguna != "stop":
  input_pengguna = input("Masukkan kata: ")
  print(f"Anda memasukkan: {input_pengguna}")
```

- Memeriksa Status File

```
try:
  file = open("data.txt", "r")
  while not file.readable():
    pass  # Tunggu sampai file siap dibaca

  isi_file = file.read()
  print(isi_file)
except FileNotFoundError:
  print("File tidak ditemukan")
finally:
  file.close()
```

- Melakukan Iterasi Tanpa Batas / Infinite ~~Tsukuyomi~~ Loop

```
angka = 1

while True:
  print(angka)
  angka += 1
```

- Menggunakan 'break' untuk Menghentikan Loop

```
angka = 1

while True:
  print(angka)
  if angka == 10:
    break  # Hentikan loop ketika angka mencapai 10
  angka += 1
```

- Menggunakan 'continue' untuk Melewati Iterasi

```
angka = 1

while True:
  if angka % 2 == 0:
    continue  # Lewati iterasi ini jika angka genap
  print(angka)
  angka += 1
```

- ## LATIHAN Perulangan

  Kode diatas dibuat agar perulangan hanya memberikan hasil nilai ganjil saja karena setiap angka genap akan kena continue. Akan tetapi, kode tersebut hanya akan menghasilkan output “1”. Perbaiki kode tersebut dengan caramu sendiri.

FUNCTION FIRST PART

Tujuan Pembelajaran

Setelah menyelesaikan modul ini, diharapkan peserta didik dapat:

- Memahami konsep dan definisi function dalam Python
- Menjelaskan jenis-jenis function dalam Python
- Menulis dan menggunakan function dalam program Python
- Menerapkan function untuk menyelesaikan permasalahan pemrograman
- Memahami manfaat dan keuntungan menggunakan function

Materi Pembelajaran

1. Konsep dan Definisi Function

Function adalah blok kode yang dirancang untuk melakukan tugas tertentu. Function dapat dipanggil berulang kali dengan parameter yang berbeda untuk menghasilkan output yang berbeda. Function membantu programmer untuk menulis kode yang lebih modular, terorganisir, dan mudah dipahami.

2. Jenis-jenis Function

- **Function tanpa parameter:** Function ini tidak memerlukan input dari pengguna.
- **Function dengan parameter:** Function ini memerlukan input dari pengguna, yang disebut parameter.
- **Function dengan nilai kembalian:** Function ini menghasilkan nilai yang dapat digunakan oleh program lain.
- **Function rekursif:** Function ini memanggil dirinya sendiri untuk menyelesaikan suatu tugas.

3. Basic Function

```
def nama_function(parameter):
  # Isi function

# Memanggil function
nama_function(argumen)
```

4. Contoh Fungsi

```
def luas_persegi_panjang(panjang, lebar):
  """
  Menghitung luas persegi panjang

  Args:
      panjang: Sisi panjang
      lebar: Sisi lebar
```

```
  Returns:
      Luas persegi panjang
  """
  luas = panjang * lebar
  return luas

# Memanggil function
panjang = 10
lebar = 5
luas_persegi = luas_persegi_panjang(panjang, lebar)
print(f"Luas persegi panjang: {luas_persegi}")
```

5. Variasi Fungsi Part 1

Fungsi Tanpa Parameter

```
def menyapa():
  """Menampilkan pesan sapaan"""
  print("Halo, dunia!")

menyapa()
```

Fungsi dengan Parameter

```
def menghitung_luas_persegi(sisi):
  """Menghitung luas persegi"""
  luas = sisi**2
  return luas

luas_persegi = menghitung_luas_persegi(5)
print(f"Luas persegi: {luas_persegi}")
```

Fungsi dengan Nilai Kembalian

```
def is_ganjil(bilangan):
  """Memeriksa apakah bilangan ganjil atau genap"""
  if bilangan % 2 == 0:
    return False
  else:
    return True

ganjil = is_ganjil(10)
print(f"Apakah 10 ganjil?: {ganjil}")
```

- ## TUGAS

1. Menghitung Nilai Akhir Mahasiswa

Sebuah universitas memiliki sistem penilaian untuk menghitung nilai akhir mahasiswa dengan ketentuan sebagai berikut:

- Nilai tugas (30%)
- Nilai UTS (30%)
- Nilai UAS (40%)

Buatlah sebuah function bernama hitung_nilai_akhir yang menerima parameter berupa nilai tugas, nilai UTS, dan nilai UAS. Function ini harus menghitung dan mengembalikan nilai akhir mahasiswa berdasarkan rumus di atas.

2. Buatlah program Python yang menghitung rata-rata nilai tiga mata pelajaran dan menentukan kelulusan berdasarkan nilai rata-rata tersebut.

3. Buatlah program Python yang menghitung biaya taksi berdasarkan jarak tempuh dan tarif per kilometer.

4. Buatlah program Python yang membuat menu sederhana untuk memesan makanan.