Laporan Praktikum Algoritma dan Pemrograman

Semester Genap 2023/2024

NIM	**71230982**
Nama Lengkap	**Rainie Fanita Chrisabel Hadisantoso**
Minggu ke / Materi	**03 / Struktur Kontrol Percabangan**

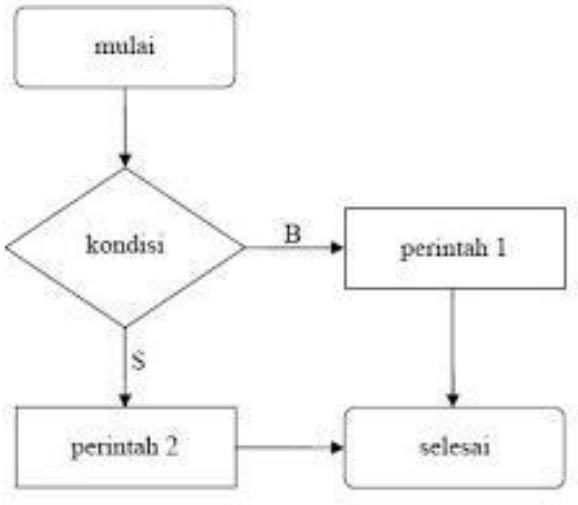
SAYA MENYATAKAN BAHWA LAPORAN PRAKTIKUM INI SAYA BUAT DENGAN USAHA SENDIRI TANPA MENGGUNAKAN BANTUAN ORANG LAIN. SEMUA MATERI YANG SAYA AMBIL DARI SUMBER LAIN SUDAH SAYA CANTUMKAN SUMBERNYA DAN TELAH SAYA TULIS ULANG DENGAN BAHASA SAYA SENDIRI.

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

Struktur Kontrol Percabangan

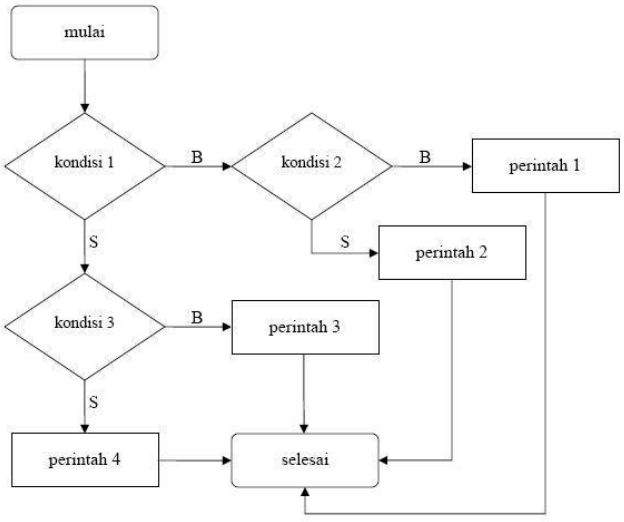
Tujuan dari pertemuan ke-3 ini adalah mahasiswa dapat merangkai boolean expression menggunakan operator logical, menyusun percabangan if else, menangani kesalahan input dengan try-except. Percabangan merupakan sebuah program yang menggunakan kondisi-kondisi tertentu untuk menghasilkan jawaban yang sesuai.



Gambar 1. Template Flowchart If dan Else

Seperti contoh template flowchart percabangan, dapat dilihat bahwa kondisi bercabang menjadi dua hasil B untuk benar dan S untuk salah. Ketika kondisi terpenuhi maka program mengalir ke panah benar untuk mengeksekusi perintah 1 dan program akan mengeksekusi perintah 2 ketika kondisi salah. Kedua jalur kondisi akan berhenti ketika perintah sudah dilaksanakan.

Dengan prinsip percabangan ini, kita dapat menciptakan program dengan berbagai kondisi yang lebih rumit dengan operator dan struktur percabangan yang lebih kompleks juga. Seperti digambarkan flowchart percabangan berikut. Terdapat 3 kondisi percabangan yang menghasilkan perintah yang berbeda-beda. Kondisi pertama bercabang menjadi kondisi kedua dan ketiga yang masing-masing memiliki cabang benar dan salah dengan perintah untuk dieksekusi sendiri-sendiri.



Gambar 2. Template Flowchart If dan Else 2

Dalam pertemuan ini akan dibahas 3 materi utama mengenai percabangan dalam bahasa Python: Boolean Expression dan Logical Operator, Bentuk-bentuk Percabangan, dan Penanganan Kesalahan Input Menggunakan

Exception Handling. Dalam materi-materi tersebut akan dipelajari mengenai operator comparison, And dan Or, Try dan Except, ternary operator, dan If Else.

Boolean Expression dan Logical Operator

Boolean expression adalah operator yang berbasis logika dalam komputer. Operator ini hanya memiliki dua jawaban yaitu: True atau False, juga bisa dituliskan sebagai True = 1 dan False = 0. Sehingga operator ini dapat digunakan ketika hanya ada dua kemungkinan jawaban: benar dan salah.

Untuk membentuk kondisi-kondisi logika berbentuk bilangan, dapat digunakan operator comparison. Contoh operator comparison seperti lambang perbandingan dalam matematika: sama dengan (=), lebih kecil (<), lebih besar (>) dan lain-lain. Namun dalam bentuk bahasa Python, lambangnya sedikit berbeda dengan lambang matematika. Berikut tabel daftar operator comparison beserta lambang dan contoh penggunaan operator tersebut menggunakan angka.

Operators	Meaning	Example	Result
<	Less than	5<2	False
>	Greater than	5>2	True
<=	Less than or equal to	5<=2	False
>=	Greater than or equal to	5>=2	True
==	Equal to	5==2	False
!=	Not equal to	5!=2	True

Gambar 3. Tabel Operator Comparison

Salah satu contoh penerapan boolean expression dalam sebuah program membandingkan nilai variabel dan nilai perbandingan dengan nilai variabel. Dapat digunakan operator comparison sama dengan ==, lebih kecil sama dengan <=, dan lebih besar sama dengan >=. Ketika kedua nilai yang dimasukkan tidak sama, maka program akan menampilkan True, dan ketika tidak memenuhi program akan menampilkan False.

```
C:\Users\asus>python
Python 3.11.8 (tags/v3.11.8:db85d51, Feb  6 2024, 22:03:32) [MSC v.1937 64 bit (AMD6
4)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> umur = 18
>>> umur == 18
True
>>> umur = 18
>>> umur < 20
True
>>> umur >= 10
True
>>> umur > 16
True
>>> umur <= 3
False
>>> umur == umur
True
>>> umur !=19
True
>>> umur != 18
False
>>>
```

Gambar 4. Contoh Operasi Boolean pada CMD

Beberapa hal yang perlu diperhatikan dalam penggunaan Boolean untuk sebuah program adalah:

- Boolean expression hanya dapat memberikan dua hasil: True atau False.
- Kata-kata minimum, maksimum, tidak lebih dari, tidak kurang dari, tidak sama, tidak berbeda.
- Variabel-variabel tang hendak dibandingkan.

Ada juga persyaratan menggunakan logika dengan operator and atau or. And berarti kedua kondisi atau lebih harus dipenuhi sehingga dapat dikatakan sebagai True, apabila ada satu saja kondisi yang tidak terpenuhi maka akan False. Or digunakan ketika hanya salah satu kondisi yang perlu dipenuhi dari daftar probabilitas kondisi yang ada untuk menghasilkan jawaban True, or akan dikatakan False ketika sama sekali tidak ada kondisi yang terpenuhi.

Contoh penggunaan And dalam persyaratan menjadi Asisten Dosen:

Ipk >= 3.00 ***And*** *Tekkom = Lulus* ***And*** *Semester <= 8*

Contoh penggunaan Or dalam pendaftaran UKDW:

Jalur prestasi ***Or*** *Jalur tes*

Bentuk-bentuk Percabangan

Terdapat tiga jenis percabangan dalam Python yaitu conditional, alternative dan chained conditional.

- Bentuk conditional secara umum menggunakan If. Dengan conditional if, program dapat memberikan beberapa kondisi dan perintah yang dijalankan ketika kondisi tersebut dipenuhi. Namun output yang dihasilkan conditional If hanyalah satu.

  Contoh aplikasi If:

```
2   #If
3   if belanja >= 100000:
4       print(Anda mendapatkan diskon 5%!”)
```

Gambar 5. Contoh Aplikasi If

- Bentuk alternatif menggunakan If dan Else. Dengan If dan Else program dapat memberikan alternatif jawaban dari bagian Else apabila kondisi-kondisi If tidak terpenuhi.

  Contoh aplikasi If dan Else:

```
6   #If Else
7   if belanja >= 100000:
8       print(“Anda mendapatkan diskon 5%!”)
9   else belanja < 100000:
10      print(“Anda tidak mendapat diskon.”)
```

Gambar 6. Contoh Aplikasi If dan Else

- Bentuk chained conditional If, Elif, dan Else. Chained conditional memiliki urutan If - Elif- Else dengan jumlah Elif yang dapat berjumlah banyak. Dengan bentuk percabangan ini, program dapat dibangun lebih kompleks dengan banyaknya jumlah kemungkinan kondisi yang dapat digunakan dan output yang beragam pula.

Contoh aplikasi If, Elif, Else:

```
12   #If else Else
13   if  belanja >= 100000:
14       print(“Anda mendapatkan diskon 5%!”)
15   elif belanja >= 500000:
16       print(“Anda mendapatkan diskon 10%!”)
17   ...
18   else belanja < 100000:
19       print(“Anda tidak mendapat diskon.”)
```

Gambar 7. Contoh Aplikasi If, Elif, dan Else

Percabangan juga memiliki bentuk lain yaitu: Ternary operator. Ternary operator hanya menggunakan percabangan If dan Else dan menggunakan satu baris untuk dituliskan. Dengan bentuk ternary, dapat dicapai chained conditional hanya menggunakan operator If dan Else. Sayangnya bentuk ini rawan error sebab conditional yang hanya ditulis dalam satu baris dapat menjadi sangat rumit dan tidak terlalu enak dilihat.

```
21   #Ternary operator
22   bilangan = int(input("Masukkan suatu bilangan: "))
23
24   print("Positif" if bilangan > 0 else "Negatif" if bilangan < 0 else "Nol")
```

Gambar 8. Contoh Ternary Operator

Dari kode di atas, dapat dilihat ternary operator secara garis besar memiliki pola:

Output kondisi 1 -> If -> Kondisi 1 -> Else -> Output 2 -> If -> Kondisi 2 -> Else -> Output 3.

Penanganan Kesalahan Input Menggunakan Exception Handling

Dalam sebuah program perlu diperhatikan jawaban dari pengguna yang tidak sesuai dengan tipe jawaban program. Diperlukan adanya perintah yang mendeteksi ketika ada input jawaban yang salah atau invalid. Dengan begitu program dapat memberikan output terhadap input yang salah tersebut. Salah satu contoh apa yang dimaksud dan penerapannya dengan sebuah program untuk mendeteksi segitiga pythagoras dengan input sisi-sisi dari user.

belajar python 2 > alpro3.py > ...

```
1   a = int(input("a ="))
2   b = int(input("b ="))
3   c = int(input("c ="))
4
5   def pythagoras(a, b, c):
6
7       if a**2 + b**2 == c**2 or b**2 + c**2 == a**2 or c**2 + a**2 == b**2:
8           return True
9       else:
10          return False
11
12  print(pythagoras(a,b,c))
```

Gambar 9. Program Pythagoras

Pada baris pertama, program memberikan fungsi input kepada pengguna. Apabila pengguna memasukkan input yang sesuai dengan yang diminta oleh program, program akan mengeluarkan output yang sesuai pula. Namun apabila user memasukkan input yang tidak sesuai, maka program tidak akan memberikan output apa pun (hanya pesan error pada editor Python). Padahal semua program yang baik memerlukan output untuk semua kemungkinan jawaban.

```
PROBLEMS 3   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS

PS C:\Users\asus\belajar-git> python -u "c:\Users\asus\belajar-git\belajar python 2\alpro3.py"
a =5
b =4
c =3
True
PS C:\Users\asus\belajar-git> python -u "c:\Users\asus\belajar-git\belajar python 2\alpro3.py"
a =3
b =4
c =lima
Traceback (most recent call last):
  File "c:\Users\asus\belajar-git\belajar python 2\alpro3.py", line 3, in <module>
    c = int(input("c ="))
        ^^^^^^^^^^^^^^^^
ValueError: invalid literal for int() with base 10: 'lima'
```

Gambar 10. Output Program Pythagoras

Untuk hal tersebut, dapat digunakan Try dan Except pada program. Program dapat ditampung dalam bagian Try, sedangkan output terhadap jawaban yang salah dapat diberikan pada bagian Except setelah program utama.

belajar python 2 > alpro3.py > ...

```python
try:
    a = int(input("a ="))
    b = int(input("b ="))
    c = int(input("c ="))

    def pythagoras(a, b, c):

        if a**2 + b**2 == c**2 or b**2 + c**2 == a**2 or c**2 + a**2 == b**2:
            return True
        else:
            return False

    print(pythagoras(a,b,c))

except:
    print("Input invalid")
```

Gambar 11. Program Pythagoras dengan Try dan Except

Dengan demikian program akan tetap mengeluarkan output apabila pengguna memasukkan input yang tidak sesuai.

Gambar 12. Output Program Pythagoras dengan Try dan Except

LATIHAN MANDIRI 3

SOAL 3.1

Mengimplementasikan penanganan kesalahan input pengguna menggunakan TRY EXCEPT dari program-program contoh 3.1, 3.2 dan 3.3.

1. CONTOH 3.1

```
suhu = int(input("Masukkan suhu tubuh: "))
if suhu >= 38:
    print("Anda demam")
else:
    print("Anda tidak demam")
```

a. Source Code

```
[64] try:
       suhu = int(input("Masukkan suhu tubuh: "))

       if suhu >= 38:
         print("Anda demam")
       else: print("Anda tidak demam")

     except:
       print("Input tidak valid")
```

b. Output

Suhu >= 38:

```
Masukkan suhu tubuh: 40
Anda demam
```

Invalid:

```
Masukkan suhu tubuh: empat puluh delapan
Input tidak valid
```

c. Penjelasan

Contoh 3.1 merupakan program kalkulasi demam. Ketika program di atas atau sama dengan 38, program akan menampilkan hasil “Anda demam” sedangkan di bawah 38 menampilkan “Anda tidak demam”. Untuk mengaplikasikan TRY EXCEPT, ditambahkan Try: di awal program dan Except: di akhir program dengan perintah print “Input tidak valid.”.

2. CONTOH 3.2

```
bilangan = int(input("Masukkan suatu bilangan: "))
if bilangan > 0:
    print("Positif")
elif bilangan < 0:
    print("Negatif")
elif bilangan == 0:
    print("Nol")
```

a. Source Code

```
[65] try:
       bilangan = int(input("Masukkan suatu bilangan: "))

       if bilangan > 0:
         print("Positif")
       elif bilangan < 0:
         print("Negatif")
       elif bilangan == 0:
         print("Nol")

     except:
       print("Input tidak valid")
```

b. Output

Bilangan > 0:

```
Masukkan suatu bilangan: 3
Positif
```

Invalid:

```
Masukkan suatu bilangan: nol
Input tidak valid
```

c. Penjelasan

Program contoh 3.2 adalah program untuk mengklasifikasi apakah bilangan integer yang dimasukkan pengguna merupakan bilangan positif, negatif atau nol. Kemudian ditambahkan Try dan Except untuk menambahkan output “Input tidak valid” ketika pengguna tidak memasukkan input interger.

3. CONTOH 3.3

```python
# input a, b dan c
a = int(input("Masukkan bilangan pertama: "))
b = int(input("Masukkan bilangan kedua: "))
c = int(input("Masukkan bilangan ketiga: "))

# secara berurutan tulis kriteria untuk a, b, dan c
if a > b and a > c:
    print("Terbesar: ", a)
elif b > a and b > c:
    print("Terbesar: ", b)
elif c > a and c > b:
    print("Terbesar: ", c)
```

a. Source Code

```python
[66] try:
       a = int(input("Masukkan bilangan pertama: "))
       b = int(input("Masukkan bilangan kedua: "))
       c = int(input("Masukkan bilangan ketiga: "))

       if a > b and a > c:
         print("Terbesar: ", a)
       elif b > a and b > c:
         print("Terbesar: ", b)
       elif c > a and c > b:
         print("Terbesar: ", c)

     except:
       print("Input tidak valid")
```

b. Output

Valid:

```
Masukkan bilangan pertama: 21
Masukkan bilangan kedua: 2
Masukkan bilangan ketiga: 2024
Terbesar:  2024
```

Invalid:

```
Masukkan bilangan pertama: besar
Input tidak valid
```

c. Penjelasan

Source code contoh 3.3 digunakan untuk mencari bilangan terbesar dari 3 input dari user. Program menggunakan percabangan If Else untuk membandingkan bilangan 1, bilangan 2, dan bilangan 3. Apabila user tidak memasukkan interger, program akan mengeluarkan output “Input tidak valid”.

SOAL 3.2

Mengubah percabangan pada Contoh 3.2 (Positif-Negatif) menjadi percabangan ternary operator.

```
bilangan = int(input("Masukkan suatu bilangan: "))
if bilangan > 0:
    print("Positif")
elif bilangan < 0:
    print("Negatif")
elif bilangan == 0:
    print("Nol")
```

a. Source Code

```
[67] try:
       bilangan = int(input("Masukkan suatu bilangan: "))

       print("Positif" if bilangan > 0 else "Negatif" if bilangan < 0 else "Nol")

     except:
       print("Input tidak valid")
```

```
[68] try:
       bilangan = int(input("Masukkan suatu bilangan: "))

       print("Positif") if bilangan > 0 else print("Negatif") if bilangan < 0 else print("Nol")

     except:
       print("Input tidak valid")
```

b. Output

Bilangan > 0:

```
Masukkan suatu bilangan: 3
Positif
```

Bilangan < 0:

```
Masukkan suatu bilangan: -3
Negatif
```

Bilangan = 0:

```
Masukkan suatu bilangan: 0
Nol
```

Invalid:

```
Masukkan bilangan pertama: besar
Input tidak valid
```

c. Penjelasan

Dengan ternary operator, kondisi if else hanya perlu ditulis dalam 1 baris kode. Dengan pola if else:

Output kondisi 1 -> If -> Kondisi 1 -> Else -> Output kondisi 2 -> If -> Kondisi 2 -> Else -> Output kondisi terakhir.

Ada 2 hasil mengubah contoh 3.2 menjadi menggunakan ternary operator: Yang pertama semua kode if else dimasukkan ke dalam perintah print(), sehingga tidak perlu menuliskan banyak perintah print(). Yang kedua, setiap kalimat yang hendak di print, dituliskan perintah print(), sehingga perintah if else berada di luar perintah print().

SOAL 3.3

Membuat program yang menampilkan jumlah hari dalam suatu bulan di tahun 2020. Program menggunakan input (1-12) untuk nomor bulan. Selain itu, program juga perlu menampilkan hasil input yang tidak valid dari pengguna.

a. Source Code

```
[69] try:
       bulan= int(input("Masukkan nomor bulan (1-12) = "))

       if bulan > 12 or bulan == 0:
         print("Salah input")
       elif bulan == 2:
         print("29")
       elif bulan <= 7 and bulan % 2 == 1:
         print("31")
       elif bulan <= 7 and bulan % 2 == 0:
         print("30")
       elif bulan > 7 and bulan % 2 == 1:
         print("30")
       elif bulan > 7 and bulan % 2 == 0:
         print("31")

     except:
       print("Input tidak valid")
```

b. Output

```
Masukkan nomor bulan (1-12) = 7
31
```

c. Penjelasan

Dalam program bulan hari ini, diperlukan 6 percabangan untuk memberikan output yang sesuai dengan bulan yang dimasukkan user dan jumlah hari, serta try except untuk input yang tidak sesuai.

1. If pertama memberikan hasil “Salah input” untuk input interger > 12 dan <= 0. Karena nomor bulan hanya memiliki rentang dari 1 sampai 12 bulan.

2. Kondisi percabangan ke-2 untuk menghasilkan output bulan ke-2 yang hanya berjumlah 29 hari. Bulan Februari berjumlah 29 karena soal meminta daftar bulan di tahun 2020. Kondisi ini diberikan di bagian percabangan awal agar input 2 tidak memasuki kondisi bulan genap berjumlah 30 hari.

3. Percabangan ke-3 dengan kondisi: bulan % 2 ==1 dan bulan <= 7, karena hanya bulan ganjil sebelum bulan Agustus yang berjumlah 31 hari. Juli dan Agustus sama-sama berjumlah 31 hari.
4. Percabangan ke-4 dengan kondisi: bulan % 2 == 0 dan bulan <= 7, karena bulan genap sebelum bulan Agustus berjumlah 30 hari.
5. Percabangan ke-5 kondisi: bulan % 2 == 1 dan bulan > 7, karena pola jumlah hari berubah setelah Juli. Pola hari menjadi genap = 31 hari diawali oleh Agustus dan ganjil = 30 hari.
6. Percabangan ke-6 kondisi: bulan % 2 == 0 dan bulan > 7, agar bulan genap setelah Juli berjumlah 31 hari.

SOAL 3.4

Membuat program untuk mendeteksi kesamaan panjang ketiga panjang sisi segitiga yang sisi-sisinya dimasukkan oleh pengguna. Program harus dapat mengeluarkan output: “3 sisi sama”, “2 sisi sama”, “Tidak ada yang sama” dan “Input tidak valid”.

a. Source Code

```
try:
  a= int(input("Masukkan sisi a = "))
  b= int(input("Masukkan sisi b = "))
  c= int(input("Masukkan sisi c = "))

  if a == 0 or b == 0 or c == 0:
    print("Input tidak valid")
  elif a == b == c:
    print("3 sisi sama")
  elif a == b or b == c or a == c:
    print("2 sisi sama")
  else:
     print("Tidak ada yang sama")

except:
  print("Input tidak valid")
```

b. Output

```
Masukkan sisi a = 28
Masukkan sisi b = 28
Masukkan sisi c = 28
3 sisi sama
```

c. Penjelasan

Variabel a b, dan c digunakan untuk memasukkan ketiga sisi sari pengguna. Digunakan Try dan except untuk memberikan output “Input tidak valid” kepada jawaban dengan tipe non-interger dari user. Terdapat 4 percabangan if then else:

1. Ketika salah satu input adalah 0 = “Input tidak valid”
2. Ketiga sisi sama menggunakan kondisi perbandingan sisi semua sama dengan ==
3. Menggunakan kondisi perbandingan // untuk sepasang sisi, dengan gabungan or untuk mengidentifikasi semua kombinasi pasangan sisi.
4. Selain kondisi sebelum, segitiga tidak sama sisi.

Link Github Repository:

https://github.com/rainiefch/Praktikum-Algoritma-Pemrograman_71230982.git