Laporan Praktikum Algoritma dan Pemrograman

Semester Genap 2023/2024

NIM	**71231023**
Nama Lengkap	**Yehezkiel Darren Putra Wardoyo**
Minggu ke / Materi	**03 / 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

BAGIAN 1: MATERI MINGGU INI (40%)

Boolean Expression dan Logical Operator

Boolean expression adalah sebuah permasalahan yang hanya memiliki dua jawaban, antara TRUE (1) atau FALSE (0). Salah satu ciri dari Boolean expression adalah menggunakan operator logika (logical operator), seperti : kurang dari (<), lebih dari (>), sama dengan (==), dll. Selain operator logika, ciri sebuah boolean expression, adanya gerbang logika (AND, OR, NOT, NAND,NOR,dll). Akan tetapi, di dalam Python sendiri hanya mengenal operator ”and”, ”or”, dan ”!=” , untuk yang lainnya bisa dideklarasikan dalam sebuah function. Misal, saya ingin membuat sebuah fungsi untuk gerbang logika XOR :

```
def XOR(a,b):
  if a!=b:
    return True
  else:
    return False

print(XOR(1,0))
```

Output:

True

Fungsi diatas merupakan sebuah fungsi untuk XOR(exclusive or). XOR bernilai satu(benar) ketika kedua proposisi saling memiliki nilai yang berbeda. Sehingga kita dapat membuat dua kondisi, yaitu ketika dua proposisi bernilai tidak sama atau dua proposisi bernilai sama. Ketika kedua proposisi bernilai sama menghasilkan nilai salah(false), begitupun sebaliknya.

Selanjutnya yaitu logical operator. Operator logika (logical operator), antara lain : kurang dari (<), lebih dari (>), sama dengan (==), dll. Operator tersebut digunakan untuk membandingkan dua nilai, apakah sama, kurang dari, atau lebih dari. Operator ini umumnya digunakan untuk membangun sebuah kemungkinan, bisa satu kemungkinan, dua kemungkinan, atau bahkan lebih dari dua kemungkinan. Output dari logical operator hanya berdasarkan benar(true) atau salah(false). Berikut salah satu contoh penggunaan dari operator logika :

```
#Program Diskon
pembelian = 100000
```

```
if pembelian >120000:
   pembelian-=(pembelian*(1-0,2))
else:
   pembelian =100000
print("Rp.",pembelian)
```
Output:
Rp.100000

Program diatas bekerja dengan cara membandingkan nilai dengan persyaratannya. Diskon 20% didapatkan jika harga pembelian lebih dari 120.000. Jika harga pembelian lebih dari 120.000 maka akan mendapat potongan harga sebesar 20%. Akan tetapi, jika harga pembelian tidak lebih dari 120.000 makan tidak mendapat potongan harga.

Namun, jika persyaratan tersebut diubah, yaitu minimal pembelian Rp. 100.000, maka persyaratannya pun akan berubah :

```
 if pembelian >=100000:
   pembelian-=(pembelian*(1-0,2))
else:
   ……..
```

Selanjutnya, apabila sebuah program menggunakan gabungan gerbang logika dan juga operator logika.

```
#Program Penerimaan Mahasiswa Baru
status = "Lulus SMA"
nilai_akhir = "C"
pembayaran = "lunas"
#persyaratan
if status=="Lulus SMA" and nilai_akhir >="C" and pembayaran =="lunas" :
   print("Anda Diterima sebagai Mahasiswa Baru")
else :
   print("Anda belum lulus penerimaan")
```
Output :
Anda Diterima sebagai Mahasiswa Baru

Berdasarkan code diatas, persyaratan agar dapat terima sebagai seorang mahasiswa baru adalah Lulus SMA dan nilai akhir minimal C dan pembayaran telah lunas. Perlu di garis bawahi kata ”dan” memiliki makna bahwa kondisi tersebut dapat diterima ketika semua proposisi atau persyaratan(dalam kasus ini) terpenuhi. Jika hanya salah satu saja yang tidak memenuhi persyaratan maka kondisi tersebut tidak dapat diterima.

Bentuk Percabangan

Percabangan digunakan untuk membuat keadaan lebih dari satu. Percabangan di dalam Python sendiri dibagi menjadi 3 : **conditional, alternative, chained conditional**.

Bentuk dari **conditional** dalam program python adalah;

```
if <kondisi> :
        #code
        #code
        ....
```

Secara pengertian, satu **conditional** mewakili satu keadaan/kemungkinan. Sebagai contoh, untuk lulus kuliah harus memiliki total sks sebanyak 144. Kode programnya sebagai berikut:

```
if sks == 144 :
        print("Anda telah dinyatakan lulus")
```

Selanjutnya, **alternative conditional** yaitu bentuk percabangan dengan dua kemungkinan/alternatif. Bentuk umum dari **alternative conditional**, sebagai berikut :

```
if <kondisi> :
        #code
        ....
else :
        #code
        ....
```

Terakhir, **chained conditional** yaitu percabangan jika kemungkinan langkah yang diperlukan lebih dari 2. Bentuk umum dari **chained conditional** sebagai berikut :

```
if <kondisi> :
        #code
        ....
elif <kondisi 2> :
        #code
        ....
```

…

```
else :
        #code
        ....
```

Untuk mempermudah memahaminya, saya akan membuat contoh untuk implementasi dari **chained conditional**. Pada umumnya **chained conditional** banyak diimplementasikan pada penentuan grade nilai. Beruikut contoh program sebagai berikut:

```
def grade(grade):
   if grade <= 100 and grade >= 0:
      if grade>= 81:
         IPK = "A"
      elif grade >= 78:
         IPK="A-"
      elif grade >= 75 :
         IPK ="B+"
      elif grade >= 70:
         IPK="B"
      elif grade >= 65:
         IPK="B-"
      elif grade>= 60:
         IPK="C+"
      elif grade>=55:
         IPK="C"
      elif grade>= 40:
         IPK="D"
      else :
         IPK="E"
   return IPK


inputan=float(input("Masukkan Nilai Anda : "))
print(grade(inputan))
```

Ketika kita memasukkan input yaitu nilai 80.99 maka program akan menampilkan grade ”A-”.

```
Masukkan Nilai Anda : 80.99
A-
```

Cara kerja dari program diatas yaitu nilai yang telah kita inputkan ke program, program akan membandingkan nilai dengan percabangan yang terdapat di program. Perlu kita lihat di bagian :

```
grade <= 100 and grade >= 0
```

Pernyataan diatas sama saja dengan nilai grade diantara 0 sampai dengan 100. Contoh diatas merupakan contoh implementasi dari penggunaan boolean expression.

Selain itu, terdapat juga bentuk syntax yang lebih mudah untuk merepresentasikan percabangan yaitu menggunakan **ternary operator**. Bentuk syntax dari **ternary operator** sebagai berikut :

```
Variable = Expression if Condition1 else Condition2
```

Penanganan Kesalahan Input Menggunakan Exception Handling

Dalam membuat suatu program, kita memerlukan user untuk memasukkan suatu nilai tertentu agar program yang kita buat dapat berjalan. Akan tetapi, tidak dapat dipungkiri bahwa user bisa saja mengalami *miss-connection* sehingga memasukkan jenis data yang tidak sesuai dengan yang developer inginkan. Agar user mengetahui bahwa nilai yang mereka input tidak sesuai, kita dapat menggunakan *try except*. Sebagai contoh, kita akan membuat program yang akan menampilkan bilangan terbesar diantara 2 bilangan integer. Jika user salah memasukkan nilai maka interpreter akan menampilkan bahwa nilai yang diinput tidak sesuai.

```
bil1=input("masukkan bilangan pertama :")
bil2=input("masukkan bilangan kedua :")
try :
   a=int(bil1)
   b=int(bil2)
   if a>b:
      print(a)
   else:
      print(b)
except:
   print("anda salah memasukkan input")
```

Jika kita memasukkan suatu nilai yang bukan integer maka interpreter akan mengeksekusi di bagian except.

```
masukkan bilangan pertama :a
masukkan bilangan kedua :3
anda salah memasukkan input
```

Dapat dilihat di atas bahwa jika kita hanya salah menginput pada satu variable saja maka interpreter tetap akan menganggap error.

BAGIAN 2: LATIHAN MANDIRI (60%)

SOAL 1

#Program Demam atau Tidak

```
#Demam atau Tidak
fever= input("Your Temperature : ")
print("%d Celcius" %(int(fever)))
try :
    fever_ = int(fever)
    if fever_ >=38:
        print("Fever")
    else:
        print("Not Fever")
except:
    print("ERROR: INVALID INPUT!!!")
```

Output:

```
28 Celcius
Not Fever
```

Penjelasan :

Program tersebut kerja awalnya ada memastikan value yang dimasukkan ke inputan sesuai dengan tipe data yang diinginkan. Jika tipe data value tidak sesuai maka program akan hanya memproses di bagian *except*. Akan tetapi, jika tipe data value sesuai maka program akan menjalankan yang berada di dalam *try*. Setelah itu, di bagian *try* terdapat dua kondisi (statement) yaitu ketika suhu minimal 38 derajat celcius dan ketika suhu di bawah 38 derajat celcius. Jika suhu tidak kurang dari 38 derajat celcius maka program akan menampilkan bahwa kita demam. Jika suhu kurang dari 38 derajat celcius maka program akan menampilkan bahwa kita tidak demam. Terakhir, ketika program mengeksekusi di bagian *except*, maka program akan menampilkan bahwa kita salah memasukkan input.

#Program Negatif-Positif

```
#Positif/Negatif
Z = input("Input an Integer : ")
try:
    z =int(Z)
    print("Number : %d " %z)
    print("Positive Integer") if z>0 else (print("Negative Integer") if z<0 else print("NOL"))
except:
    print("Not an Integer!!!")
```

Output:

```
Number : -50
Negative Integer
```

Penjelasan :

Program tersebut berfungsi untuk menentukan apakah value yang kita masukkan termasuk bilangan positif atau bilangan negatif atau nol. Oleh karena itu, program tersebut terdapat 3 kemungkinan yang ketika value lebih dari 0 atau kurang dari 0 atau sama dengan 0. Namun sebelumnya program akan mengecek inputan yang masuk apakah sesuai dengan tipe data yang diinginkan, jika tidak sesuai maka interpreter akan langsung melompat dan memulai eksekusi dari *except*.

#Program segitiga

```
#Nilai Terbesar
var1 =input("Input first number : ")
var2 = input("Input second number : ")
var3 = input("Input third number : ")
try:
    a= int(var1)
    b=int(var2)
    c=int(var3)
    print(a,b,c)
    if a >(b and c):
        print("Nilai terbesar : %d" %a)
    elif b>(a and c):
        print("Nilai terbesar : %d" %b)
    else:
        print("Nilai Terbesar : %d"%c)
except:
    print("ERROR : INVALID INPUT !!!")
```

Output:

```
101 98 100
Nilai terbesar : 101
```

Penjelasan:

Program tersebut bekerja dengan cara membandingkan 3 angka dengan nilai yang berbeda dan menampilkan nilai yang terbesar diantara 3 angka tersebut. Dengan demikian, dalam program ini memiliki 3 keadaan. Keadaan pertama, jika bilangan a lebih besar dari b dan c maka bilangan a yang terbesar. Keadaan kedua, jika bilangan b lebih besar dari a dan c maka bilangan b yang terbesar. Keadaan ketiga, jika bilangan c lebih besar dari b dan a maka bilangan c yang terbesar.

SOAL 2

```
print("Determine Integers")

Z = int(input("Input an Integer : "))
print("Number : %d " %Z)
print("Positive Integer") if Z>0 else (print("Negative Integer") if Z<0 else print("NOL"))
```

Output:

```
Determine Integers
Number : 0
NOL
```

Penjelasan:

Program diatas digunakan untuk menentukan bilangan tersebut apakah positif atau negatif atau nol. Program tersebut dibagi menjadi 3 keadaan. Pertama, jika nilai dari bilangan lebih dari nol maka termasuk bilangan positif. Kedua, jika nilai bilangan kurang dari nol maka termasuk bilangan negatif. Ketiga, jika nilai bilangan sama dengan nol maka bilangan tersebut adalah nol.

SOAL 3

```
print("Calender of 2020")
month=int(input("Input Number of Month (1-12) : "))
try:
    bulan=int(month)
    if bulan >0 and bulan <= 12:
        if bulan%2!=0 or bulan==8:
            print("This Month(%d) has 31 days" %bulan)
        elif bulan!=2 and bulan%2==0:
            print("This Month(%d) has 30 days" %bulan)
        elif bulan==2 :
            print("This Month(%d) has 29 days" %bulan)
    else:
        print("Not include in calender")
except:
    print("INVALID INPUT!!!")
```

Output:

```
Calender of 2020
This Month(2) has 29 days
```

Penjelasan:

Program diatas digunakan untuk menampilkan jumlah hari pada suatu bulan di tahun 2020. Perlu diketahui bahwa tahun 2020 adalah tahun kabisat, tahun yang memiliki jumlah hari sebanyak 366. Pada tahun kabisat, bulan 2 (Februari) memiliki 29 hari. Setiap bulan ganjil dan juga Agustus memiliki 31 hari, sedangkan bulan genap kecuali Februari memiliki 30 hari. Algoritma dari program ini dibagi menjadi 4 keadaan; bulan ganjil dan bulan ke-8 memiliki jumlah 31 hari, bulan genap kecuali Februari memiliki 30 hari, bulan Februari memiliki 29 hari, angka bulan harus berupa integer positif antara 1 sampai 12.

Pertama, bulan harus berupa integer positif. Jika angka bulan kurang dari satu dan lebih dari 12 maka interpreter akan menampilkan bahwa tidak ada bulan tersebut. Jika angka bulan lebih dari nol dan

kurang dari 13 maka interpreter akan melanjutkan ke keadaan selanjutnya. Selanjutnya interpreter akan mengeksekusi ke keadaan berikutnya. Interpreter akan membandingkan value yang diinput dengan melakukan modulus 2 untuk menentukan apakah bulan tersebut termasuk ganjil atau genap. Jika bulan termasuk ganjil maka interpreter akan menampilkan jumlah hari sebanyak 31 hari. Jika bulan genap kecuali bulan ke-2 dan bulan ke-8 maka interpreter akan menampilkan jumlah hari sebanyak 30 hari. Selanjutnya, jika bulan termasuk bulan ke-8 maka interpreter akan menampilkan jumlah hari sebanyak 31 hari. Jika bulan termasuk bulan ke-2 maka interpreter akan menampilkan jumlah hari sebanyak 29 hari.

SOAL 4

```
#Program menentukan apakah 3 buah angka ada yang sama
var1=input("Masukkan sisi pertama : ")
var2=input("Masukkan sisi kedua : ")
var3=input("Masukkan sisi ketiga : ")

#code executed
try:
    a=int(var1)
    b=int(var2)
    c=int(var3)
    print("%d, %d, %d"%(a,b,c))
    if (a and b and c) != 0:
        if a==b==c:
            print("SEMUA SAMA")
        elif a==b or a==c or b==c :
            print("2 SISI SAMA")
        else:
            print("TIDAK ADA YANG SAMA")
    else:
        print("Sisi segitiga harus lebih dari nol")
#kecuali
except:
    print("INVALID INPUT!!!")
```

Output:

```
0, 1, 2
Sisi segitiga harus lebih dari nol
```

Penjelasan:

Program diatas adalah menentukan apakah dalam suatu segitiga, ketiga sisinya ada yang sama atau tidak. Perlu diketahui bahwa suatu sisi harus memiliki panjang dan berupa suatu bilangan, artinya sisi harus lebih dari 0 dan berupa suatu integer. Jika user menginputkan nilai yang bukan suatu integer maka interpreter akan menampilkan bahwa kita salah dalam menginput suatu nilai. Selanjutnya, jika semua nilai yang diinput sudah berupa integer maka program bisa berjalan ke code selanjutnya. Nilai-nilai yang user sudah input akan di cek apakah bukan 0, jika semua nilai 0 maka interpreter akan menampilkan bahwa sisi yang diinput harus lebih dari 0. Setelah itu, interpreter akan mengcek dari ketiga nilai. Pertama, jika ketiga nilai tersebut sama maka interpreter akan menampilkan semua sisi sama besar. Kedua, jika 2 dari 3 nilai tersebut ada yang sama maka interpreter akan menampilkan 2 sisi sama besar. Terakhir, jika tidak ada sisi yang sama maka intepreter akan menampilkan bahwa semua sisi tidak akan yang sama.

Link Repository :

https://github.com/YehezkielDarren/PrAlPro_Minggu3.git