UNIVERSITAS INDONESIA
Veritas, Probitas, Iustitia

FAKULTAS
ILMU
KOMPUTER

Topic 3.1: Conditionals/Selection

CSGE601020 - Dasar-Dasar Pemrograman 1

Lintang Matahari Hasani, S.Kom., M.Kom. | Dr.Eng. Lia Sadita, S.Kom., M.Eng.

Acknowledgement

This slide is an adapted version of Conditionals/Selection slides used in DDP1 Course (2020/2021)
by **Hafizh Rafizal Adnan, M.Kom.**

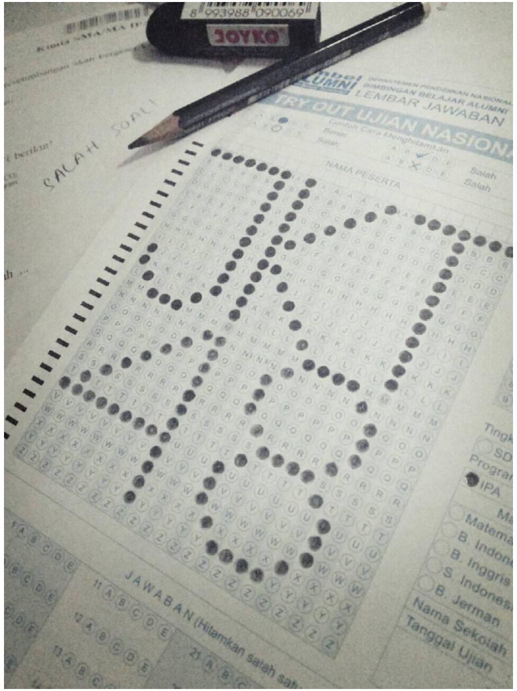
Several materials are reused from Conditionals/Percabangan slides used in Dasar-Dasar Pemrograman 1 dengan Python (CSGE601020/4 SKS) Course (https://ocw.ui.ac.id/course/view.php?id=142)
by **Fariz Darari, Ph.D.** and 'Control (Branching and Loops)' slides by **Raja O. P. Damanik, M.Sc.** and **Dr. Eng. Lia Sadita**

Some additional contents, illustrations and visual design elements are provided by
Lintang Matahari Hasani, M.Kom.

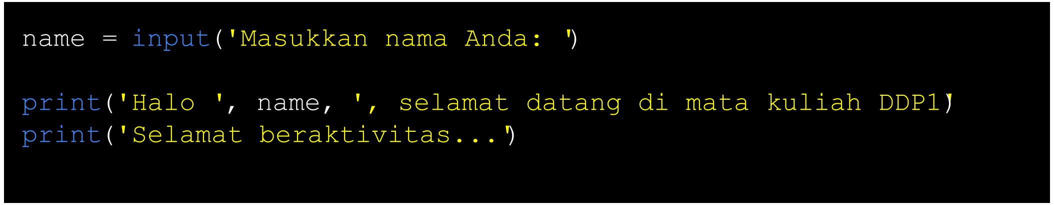
In this session, you will learn ...

- **Selection in Programming**
- **Boolean Expression**
- **Relational Expression**
- **Some selection examples in Python**

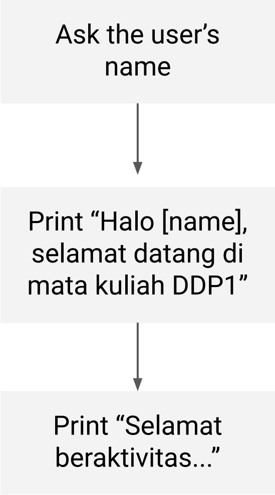
Everyday We Make Decisions



Sequential Program Flow in Python Programming



```
name = input('Masukkan nama Anda: ')

print('Halo ', name, ', selamat datang di mata kuliah DDP1')
print('Selamat beraktivitas...')
```



Ask the user's name

Print "Halo [name], selamat datang di mata kuliah DDP1"

Print "Selamat beraktivitas..."

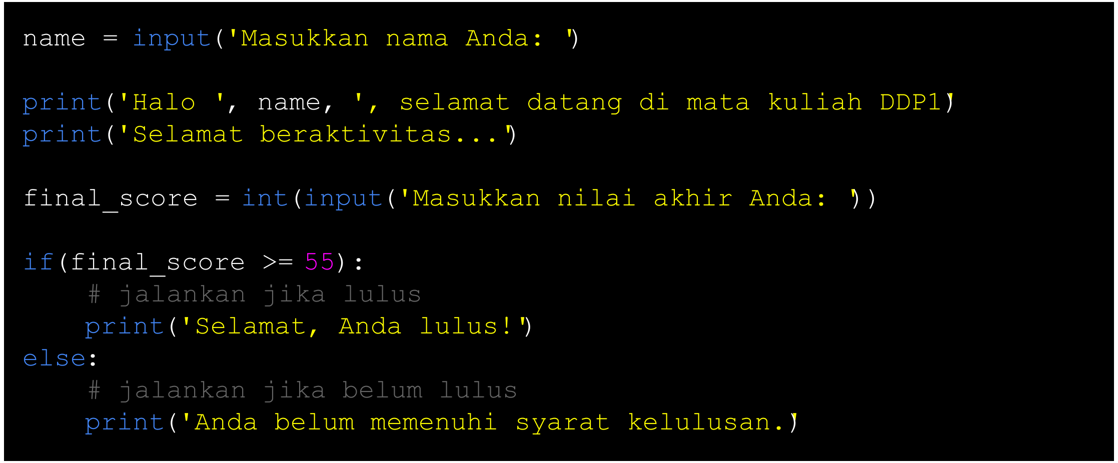
Selection/Decision in Python Programming (2)

Problem Example:

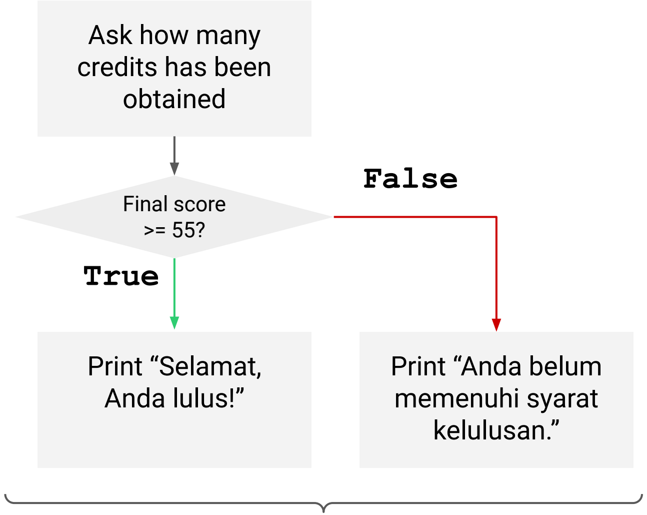
Suppose we want to make a simple program to check if a student passed a course

Criterion for passing the course: *Have final score >55*

If the student has met the criterion, the program prints “Selamat, Anda lulus!.” Otherwise, the program prints “Anda belum lulus.”



```
name = input('Masukkan nama Anda: ')

print('Halo ', name, ', selamat datang di mata kuliah DDP1')
print('Selamat beraktivitas...')

final_score = int(input('Masukkan nilai akhir Anda: '))

if(final_score >= 55):
    # jalankan jika lulus
    print('Selamat, Anda lulus!')
else:
    # jalankan jika belum lulus
    print('Anda belum memenuhi syarat kelulusan.')
```



Ask how many credits has been obtained

Final score >= 55?

True

False

Print “Selamat, Anda lulus!”

Print “Anda belum memenuhi syarat kelulusan.”

Branching

Selection/Decision in Python Programming (3)

Selection is how programs make **choices** (decisions)

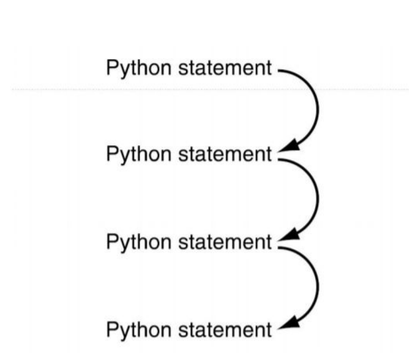


FIGURE 2.1 Sequential program flow.

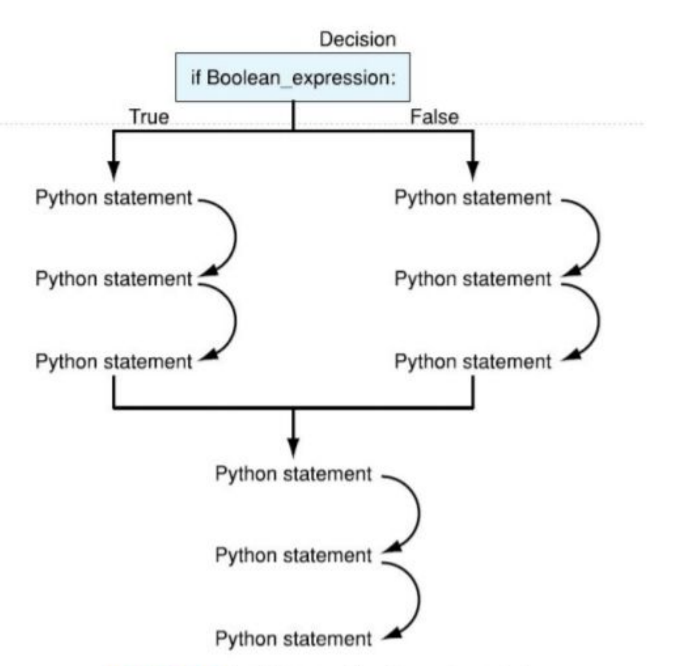
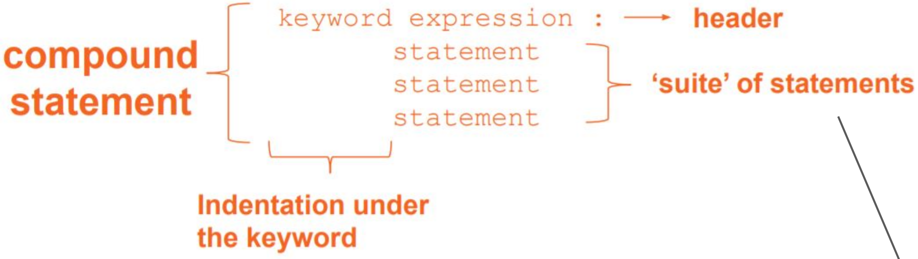


FIGURE 2.2 Decision making flow of control.

Indentation (important! ^^)



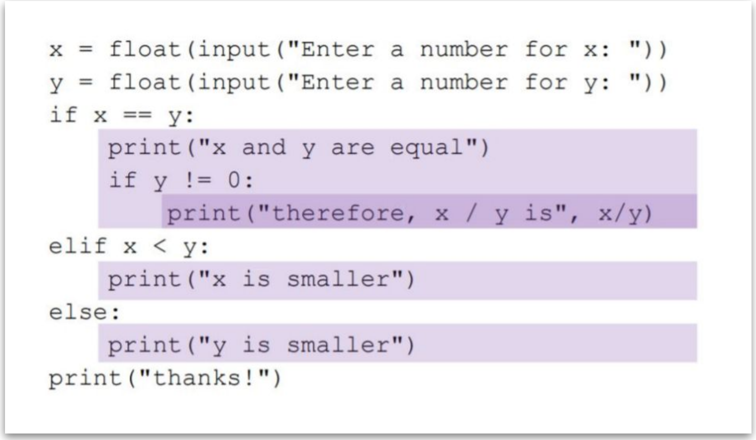
compound statement

keyword expression : → header

statement
statement
statement

'suite' of statements

Indentation under the keyword



```
x = float(input("Enter a number for x: "))
y = float(input("Enter a number for y: "))
if x == y:
    print("x and y are equal")
    if y != 0:
        print("therefore, x / y is", x/y)
elif x < y:
    print("x is smaller")
else:
    print("y is smaller")
print("thanks!")
```

➔ Indentation matters in Python.

➔ How you denote blocks of code (suite) in Python.

Indentation has dual purposes:

1. To indicate **compound statements** (indicate the structure of the code)
2. To make compound statements easier to read

A 'Suite' is a **block** of code, that is a collection of valid python statements

Warning About Indentation

a

```
print("duarr")
     print("wadaw")
print("wadidaw")
```

b

```
a = 10
if(a == 10):
          print('test')
else:
          print('duarr')
```

c

```
a = 10
if(a == 10):
     print('test')
     else:
          print('duarr')
```

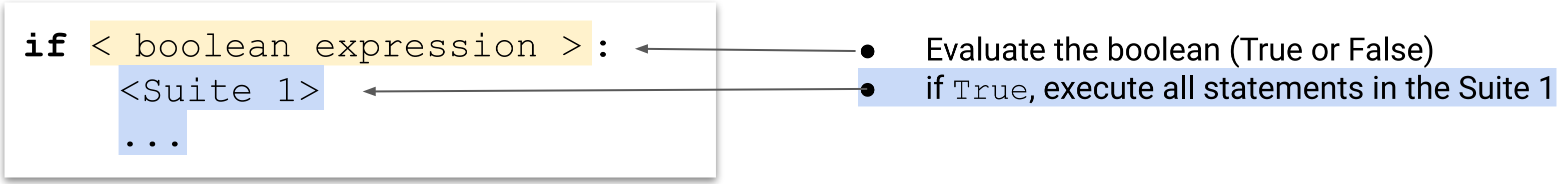
which codes will produce an error?

Elements of the suite must all be indented **the same number of spaces/tabs**

Python only recognizes suites when they are indented the **same distance** (standard is 4 spaces)

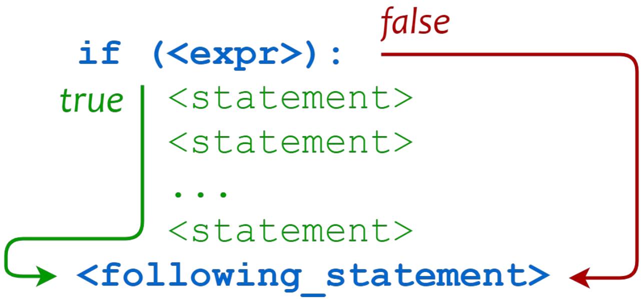
You must be careful to get the indentation right to get suites right.

Selection in Python - Alternative 1

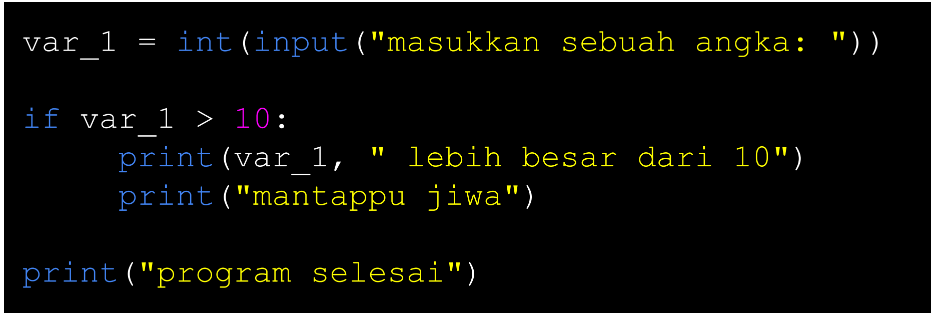


```
if < boolean expression > :
    <Suite 1>
    ...
```

- Evaluate the boolean (True or False)
- if `True`, execute all statements in the Suite 1



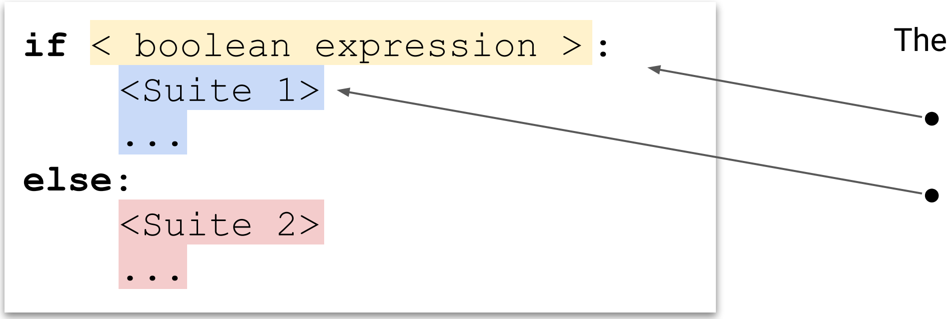
```
if (<expr>):
    <statement>
    <statement>
    ...
    <statement>
<following_statement>
```



```
var_1 = int(input("masukkan sebuah angka: "))

if var_1 > 10:
      print(var_1, " lebih besar dari 10")
      print("mantappu jiwa")

print("program selesai")
```

Illustration: https://realpython.com/python-conditional-statements/

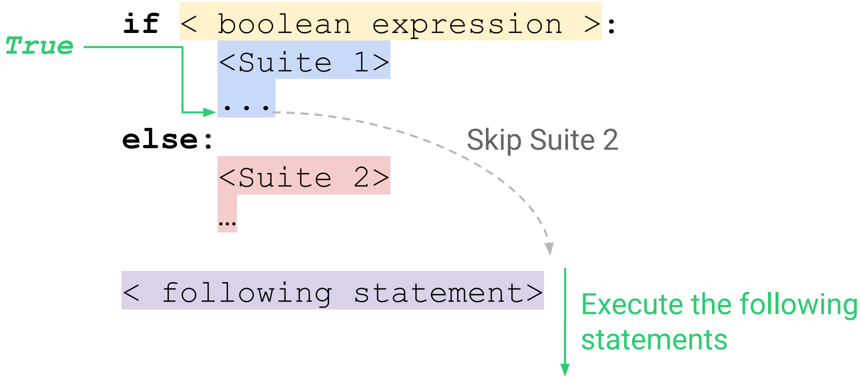
Selection in Python - Alternative 2



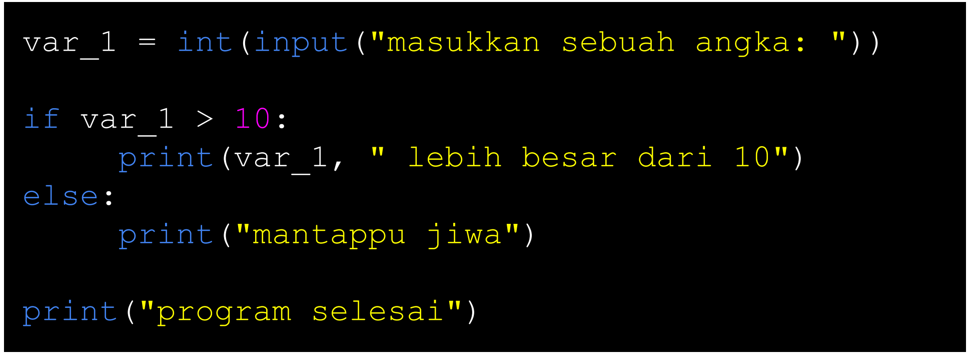
```
if < boolean expression >:
    <Suite 1>
    ...
else:
    <Suite 2>
    ...
```

The process is:

- evaluate the Boolean expression
- if True, run <Suite1>



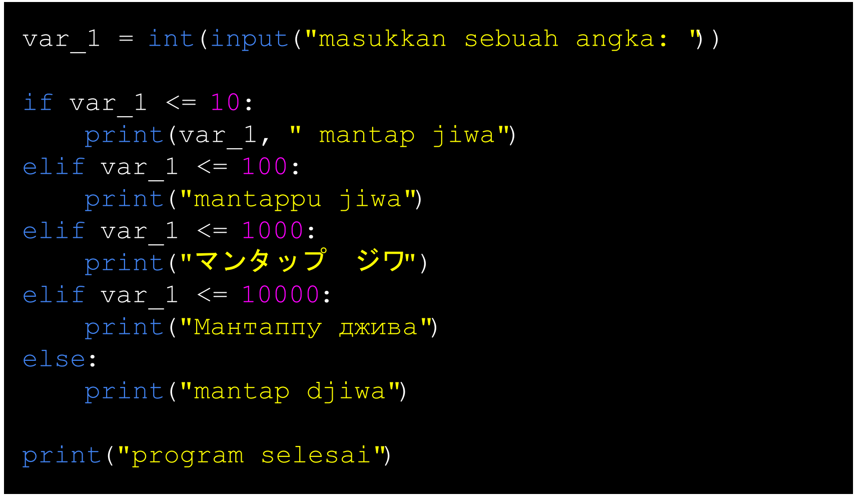
```
if < boolean expression >:
    <Suite 1>
    ...
else:
    <Suite 2>
    ...

< following statement>
```



```
var_1 = int(input("masukkan sebuah angka: "))

if var_1 > 10:
    print(var_1, " lebih besar dari 10")
else:
    print("mantappu jiwa")

print("program selesai")
```

Selection in Python - Alternative 3

```
if < boolean expression >:
    suite1
elif:
    suite2
-- as many elif you want
else:
    last_suite
```

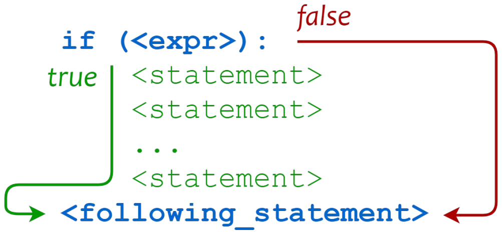


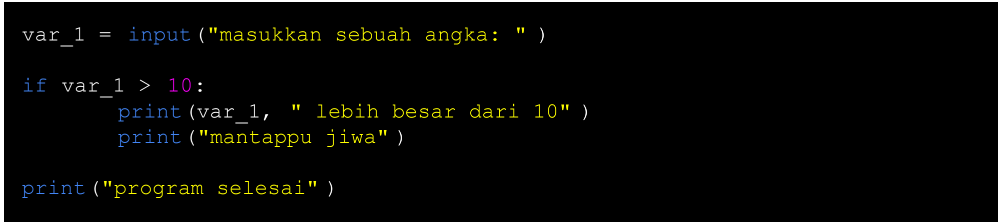
```python
var_1 = int(input("masukkan sebuah angka: "))

if var_1 <= 10:
    print(var_1, " mantap jiwa")
elif var_1 <= 100:
    print("mantappu jiwa")
elif var_1 <= 1000:
    print("マンタップ　ジワ")
elif var_1 <= 10000:
    print("Мантаппу джива")
else:
    print("mantap djiwa")

print("program selesai")
```

- ➔ Evaluate Boolean expressions until:
    - ◆ the Boolean expression returns `True`
    - ◆ none of the Boolean expressions return `True`

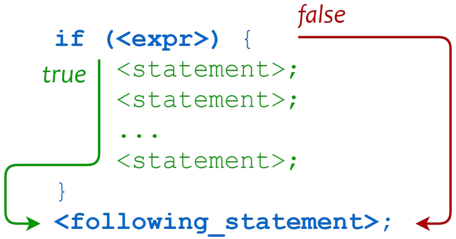
- ➔ if a boolean returns `True`, run the corresponding suite. Skip the rest of the if
- ➔ if no boolean returns `True`, run the `else` suite, the default suite

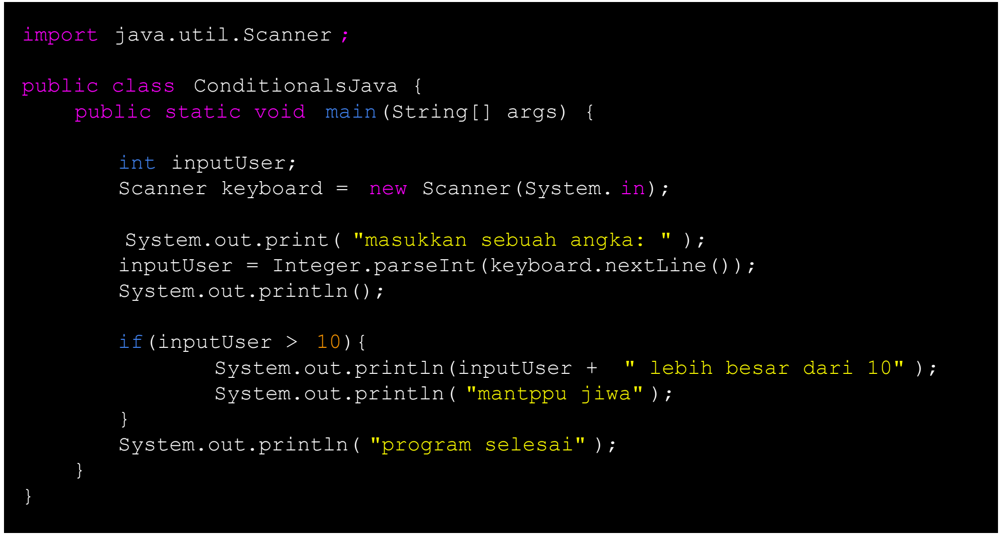
Python vs Java





```
var_1 = input("masukkan sebuah angka: " )

if var_1 > 10:
        print(var_1, " lebih besar dari 10" )
        print("mantappu jiwa" )

print("program selesai" )
```



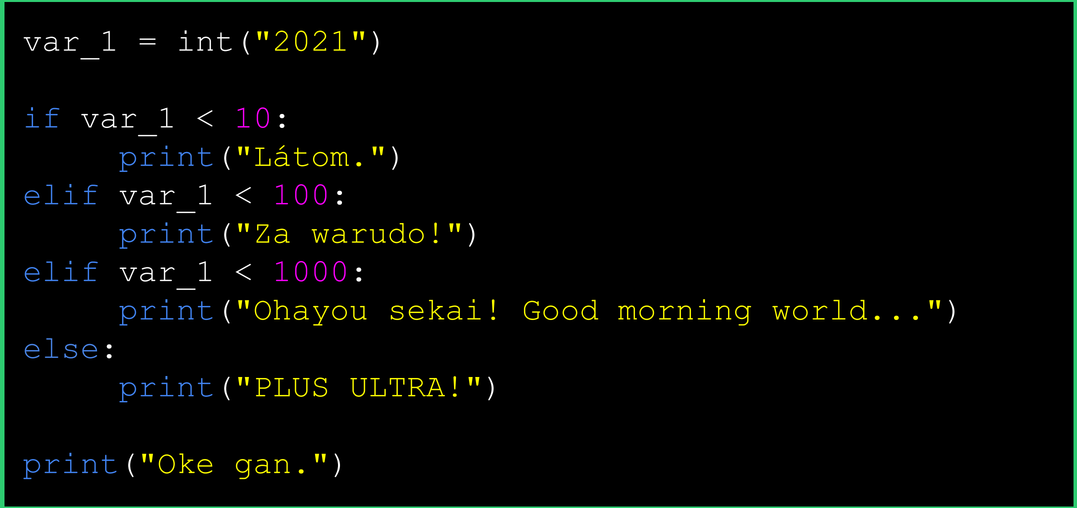


```
import java.util.Scanner ;

public class ConditionalsJava {
    public static void main(String[] args) {

        int inputUser;
        Scanner keyboard = new Scanner(System. in);

         System.out.print( "masukkan sebuah angka: " );
        inputUser = Integer.parseInt(keyboard.nextLine());
        System.out.println();

        if(inputUser > 10){
                System.out.println(inputUser + " lebih besar dari 10" );
                System.out.println( "mantppu jiwa" );
        }
        System.out.println( "program selesai" );
    }
}
```

Illustration: *https://realpython.com/python-conditional-statements/*

Triggering Question 1

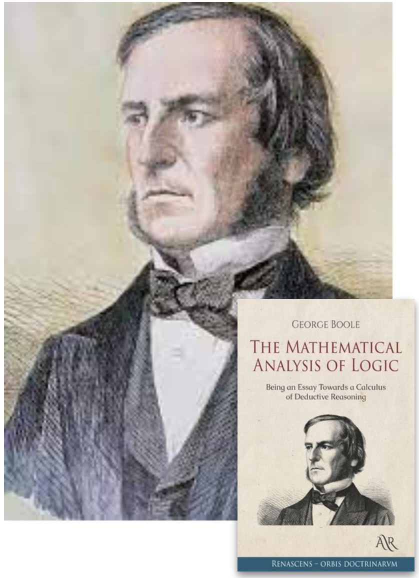
Guess the output.



```python
var_1 = int("2021")

if var_1 < 10:
     print("Látom.")
elif var_1 < 100:
     print("Za warudo!")
elif var_1 < 1000:
     print("Ohayou sekai! Good morning world...")
else:
     print("PLUS ULTRA!")

print("Oke gan.")
```

Write your answer in the **comment section**

Boolean Expression


GEORGE BOOLE
THE MATHEMATICAL ANALYSIS OF LOGIC
Being an Essay Towards a Calculus of Deductive Reasoning
AR
RENASCENS - ORBIS DOCTRINARVM

➔ George Boole's (mid-1800's) mathematics of logical expressions

➔ Boolean expressions (conditions) have a value of `True` or `False`

➔ Conditions are the basis of choices in a computer, and, hence, are the basis of the appearance of intelligence in them.

What is True and what is False

in Python;
`True`: **any nonzero number or non-empty object**.
1, 100, "hello", [a,b]

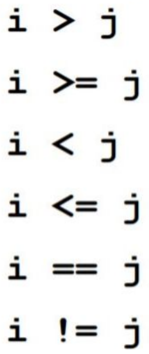
`False`: **a zero number or empty object**.

- `False`,
- `None`,
- numeric zero of all types (i.e., `0`, `0.0`), and
- empty strings and
- empty containers (including strings, tuples, lists, dictionaries, sets and frozensets)

Special values called `True` and `False`, which are just substitutions for 1 and 0.

Comparison on `int`, `float`, `string`

- ➔ `i` and `j` are variable names
- ➔ Comparisons below evaluates to a Boolean (`True` or `False`):



```
i > j
i >= j
i < j
i <= j
i == j
i != j
```

equality test
`True` if i is the same as j

inequality test
`True` if i is **not** the same as j

Relational Operators

Subset of Boolean operator. Relational operators can be used to **compare the relation between two values.**

- `3 > 2` → `True`

Relational Operators have **lower precedence** than Arithmetic Operators

- `5 + 3 < 3 - 2`
- `8 < 1` → `False`

`'1'< 2` → TypeError

- We cannot compare between String and integer using operators: order comparison (<, >, <=, and >=)

  More info:
  https://docs.python.org/3/reference/expressions.html#value-comparisons

`int('1') < 2` → `True`

- same types, regular comparison

Chained Expression

In Python, chained comparisons work just like you would expect in a mathematical expression:

Given `myInt` has the value 5

- ➔ `0 <= myInt<= 5 → True`
- ➔ `0 < myInt<= 5 < 1 → False`

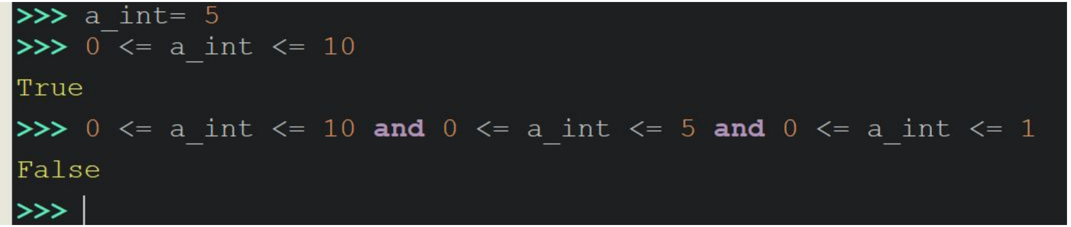
`x < y <= z`
is equivalent to
`x < y and y <= z,`

Compound Expressions

Python allows bracketing of a value between two Booleans, as in math

```
a_int= 5

0 <= a_int <= 10 → True

0 <= a_int <= 10 and 0 <= a_int <= 1 5 and 0 <= a_int <= 1 → False
```

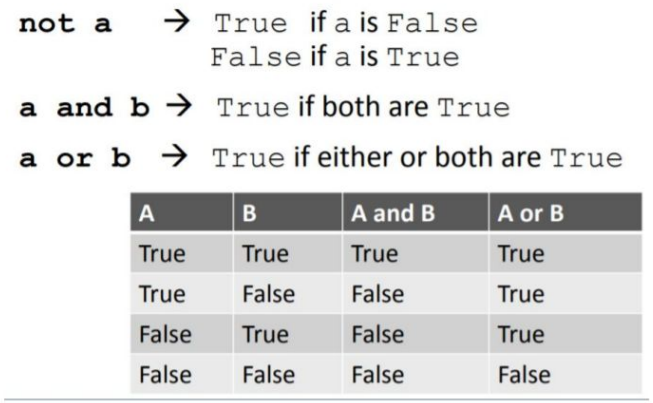
➔ **and, or, not** are the three **Boolean operators** in Python



```
>>> a_int= 5
>>> 0 <= a_int <= 10
True
>>> 0 <= a_int <= 10 and 0 <= a_int <= 5 and 0 <= a_int <= 1
False
>>> |
```

Boolean operators (and, or) vs. relational operators

- ➔ Relational operations always return `True` or `False`
- ➔ Boolean operators (`and`, `or`) are different in that:
  - ◆ They can return values (that represent `True` or `False`)
  - ◆ They have **short circuiting**



`not a` → `True` if a is `False`
`False` if a is `True`

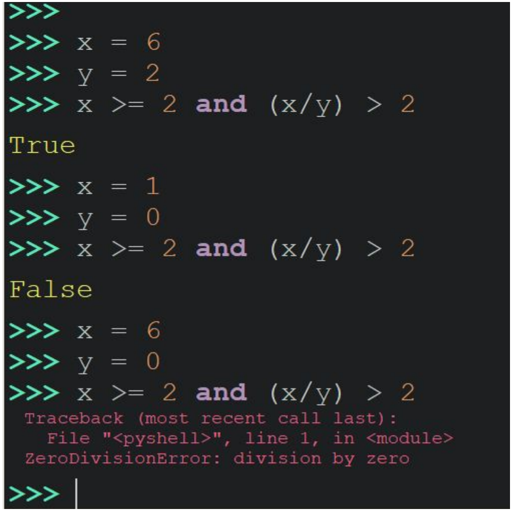
`a and b` → `True` if both are `True`

`a or b` → `True` if either or both are `True`

A	B	A and B	A or B
True	True	True	True
True	False	False	True
False	True	False	True
False	False	False	False

Short Circuit in in Boolean Operators

Operation	Result	Notes
X or Y	If X is false, then Y, else X	Y is executed only if X is false. Else if X is true, X is result
X and Y	If X is false, then X else Y	Y is executed only if X is true, else if X is false, X is result
not X	If X is true, then false, else true	Not has lower priority than non boolean operators. E.g. not a == b → not (a==b)



```
>>>
>>> x = 6
>>> y = 2
>>> x >= 2 and (x/y) > 2
True
>>> x = 1
>>> y = 0
>>> x >= 2 and (x/y) > 2
False
>>> x = 6
>>> y = 0
>>> x >= 2 and (x/y) > 2
 Traceback (most recent call last):
   File "<pyshell>", line 1, in <module>
 ZeroDivisionError: division by zero
>>> 
```

Short circuit: **Stopping the execution of boolean operation** if the truth value of the expression has already been determined. The expression is evaluated **from left to right.**

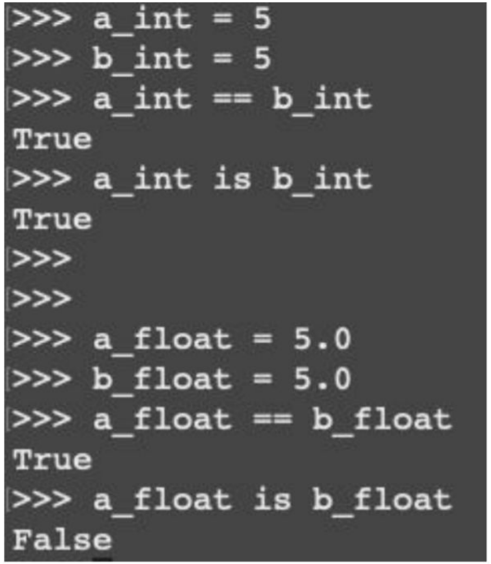
Truth Tables

A	B	A and B	A or B	not A
False	False	False	False	True
False	True	False	True	True
True	False	False	True	False
True	True	True	True	False

Equal vs Same

`==` compares **values** between the objects of two variables

`is` operator determines if two variables are associated with the **same object**



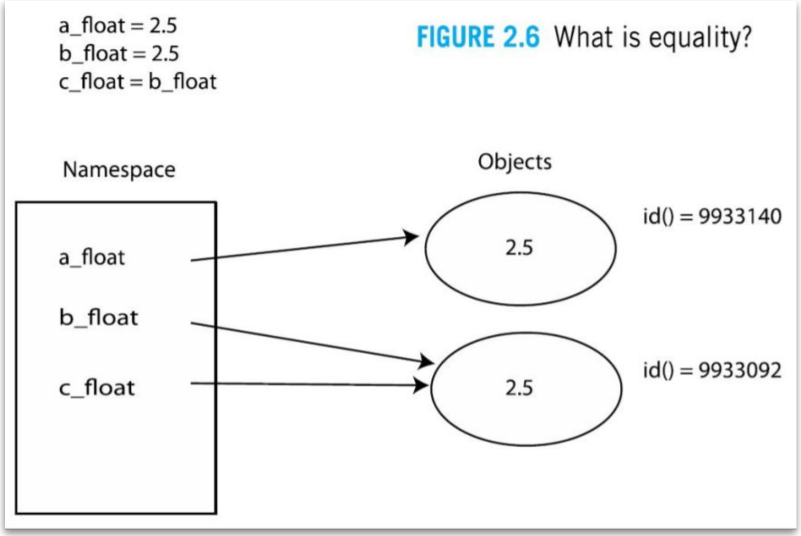
```
>>> a_int = 5
>>> b_int = 5
>>> a_int == b_int
True
>>> a_int is b_int
True
>>>
>>>
>>> a_float = 5.0
>>> b_float = 5.0
>>> a_float == b_float
True
>>> a_float is b_float
False
```

What Does Equality Mean?

Let’s recall our previous discussion on the namespace in Python

Two senses of equality:

- ➔ two variables refer to different objects, each object representing the **same value**
- ➔ two variables refer to the **same object**.
- ➔ The `id()` function is used to check the object id.

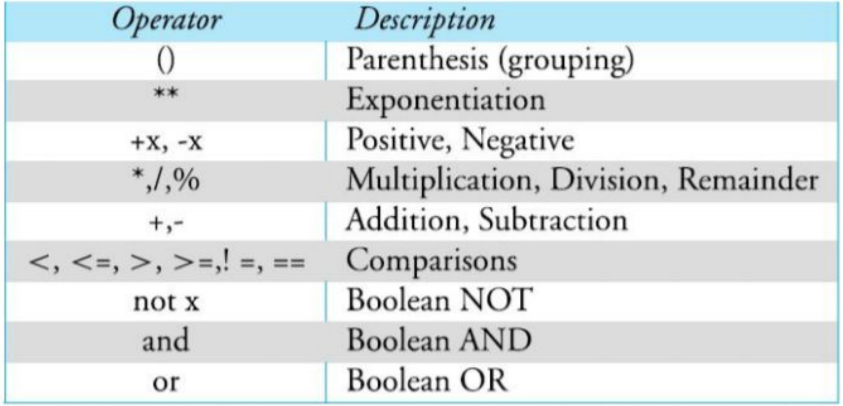


```
a_float = 2.5
b_float = 2.5
c_float = b_float
```

FIGURE 2.6 What is equality?

Precedence and Associativity

Boolean operators have **precedence** and **associativity** just like numerical operators

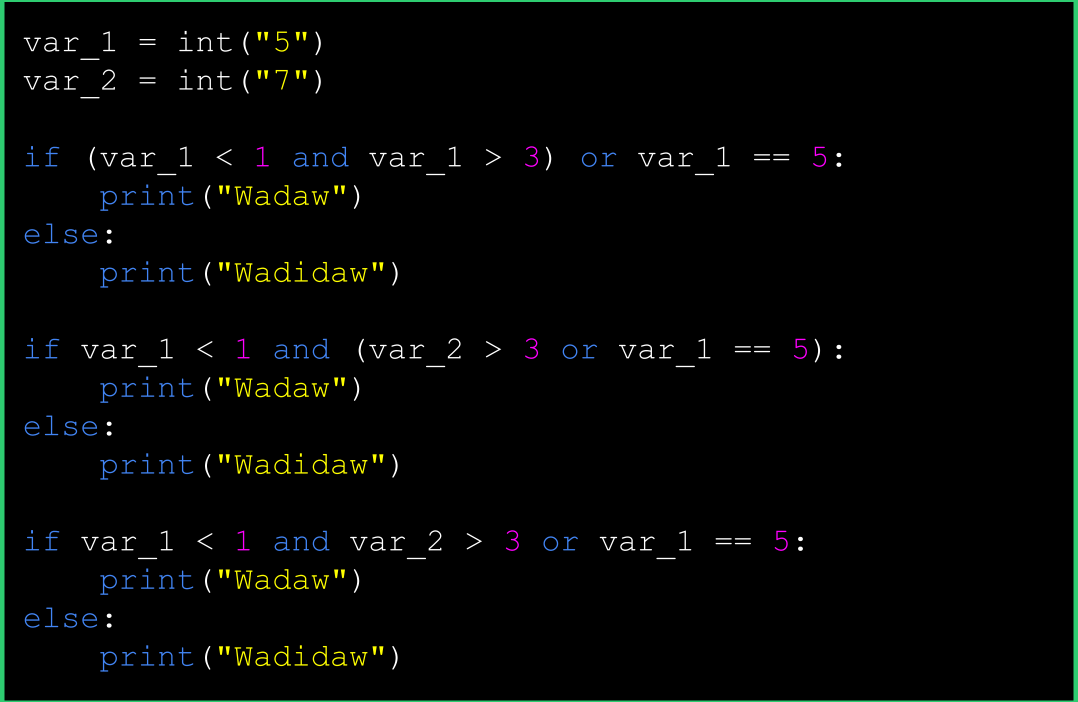


Operator	*Description*
()	Parenthesis (grouping)
**	Exponentiation
+x, -x	Positive, Negative
*,/,%	Multiplication, Division, Remainder
+,-	Addition, Subtraction
<, <=, >, >=,! =, ==	Comparisons
not x	Boolean NOT
and	Boolean AND
or	Boolean OR

TABLE 2.2 Precedence of Relational and Arithmetic Operators: Highest to Lowest

Triggering Question 2

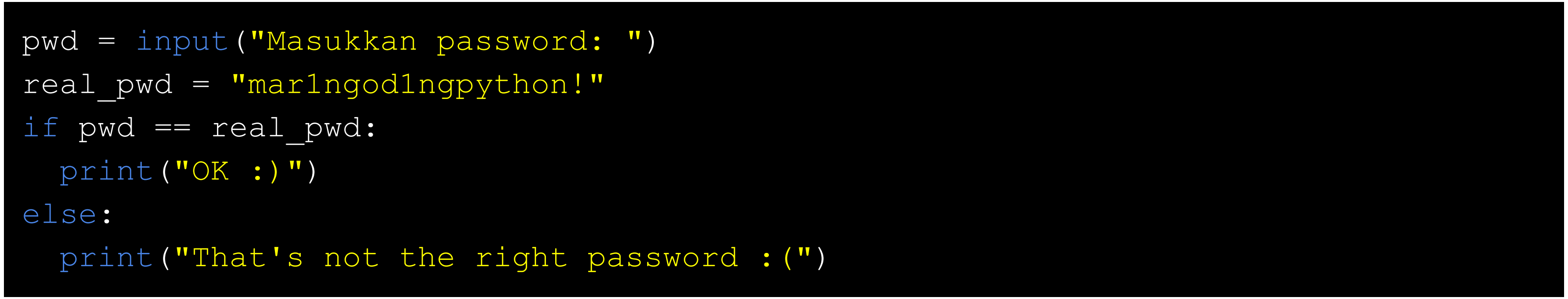
Guess the output.



```
var_1 = int("5")
var_2 = int("7")

if (var_1 < 1 and var_1 > 3) or var_1 == 5:
    print("Wadaw")
else:
    print("Wadidaw")

if var_1 < 1 and (var_2 > 3 or var_1 == 5):
    print("Wadaw")
else:
    print("Wadidaw")

if var_1 < 1 and var_2 > 3 or var_1 == 5:
    print("Wadaw")
else:
    print("Wadidaw")
```

Write your answer in the **comment section**

Code Example: Simple Password Checker



```
pwd = input("Masukkan password: ")
real_pwd = "mar1ngod1ngpython!"
if pwd == real_pwd:
  print("OK :)")
else:
  print("That's not the right password :(")
```

Code Example: Grade Converter

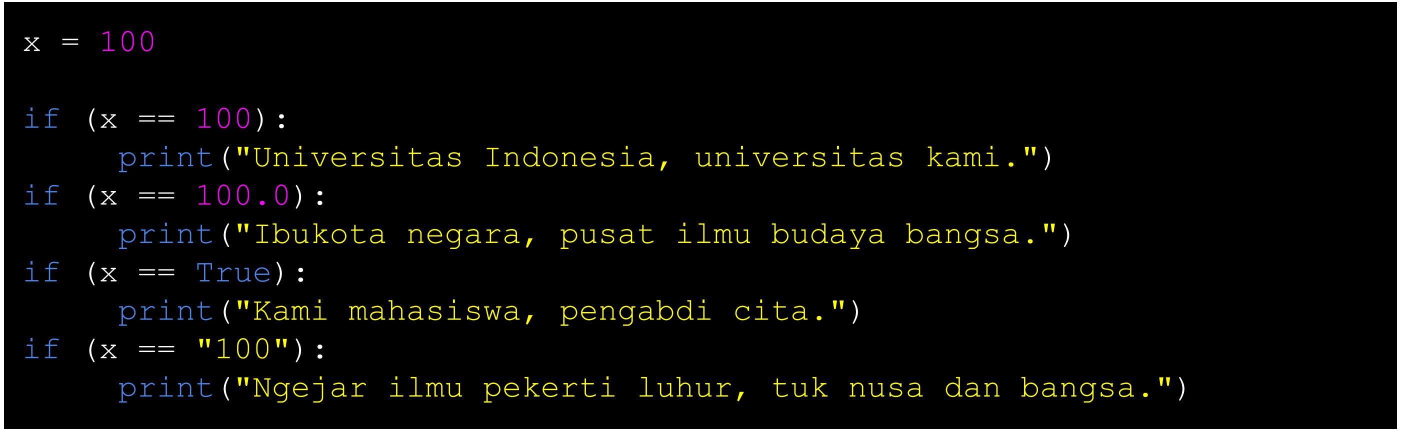
```
nilai = int(input("Masukkan nilai:"))

if  85 <= nilai <= 100:
     print("A")
elif 70 <= nilai < 85:
     print("B")
else:
     print("E")
```

What happens if the `elif` is replaced by `if` ?

Code Example: Simple Recommender System

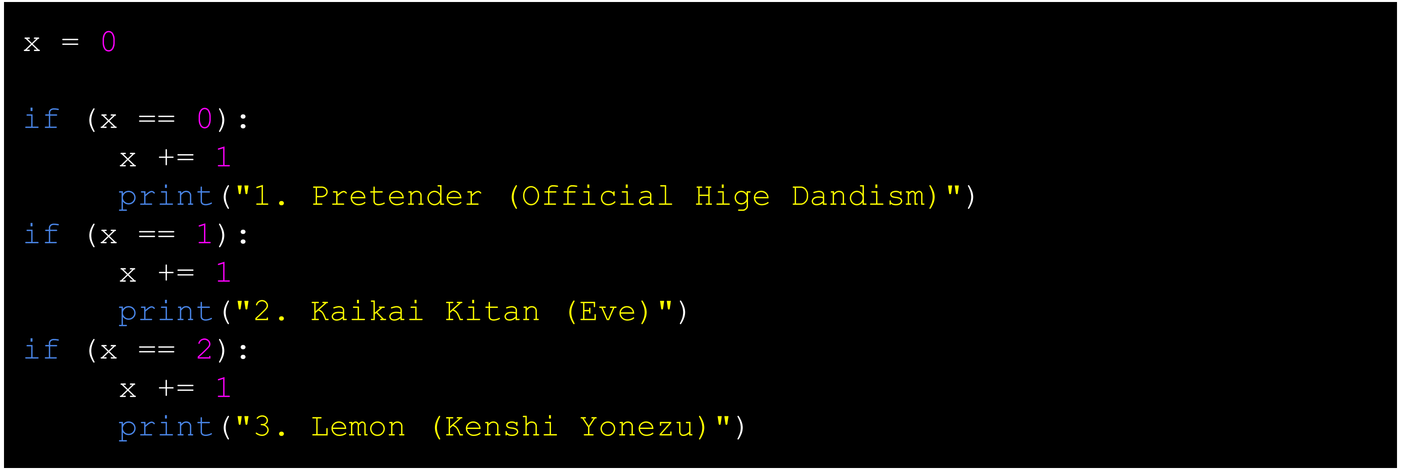
```
suka_pedas = input("Suka pedas (Y/T)? ")
tanggal_tua = input("Tanggal tua (Y/T)? ")

if suka_pedas == "Y":
  if tanggal_tua == "Y":
    print("Rekomendasi menu: Nasi sambal")
  else:
    print("Rekomendasi menu: Nasi rica-rica iga sapi")
else:
  if tanggal_tua == "Y":
    print("Rekomendasi menu: Nasi kecap")
  else:
    print("Rekomendasi menu: Nasi ayam kecap")
```

Review Question 1: Guess the Output!



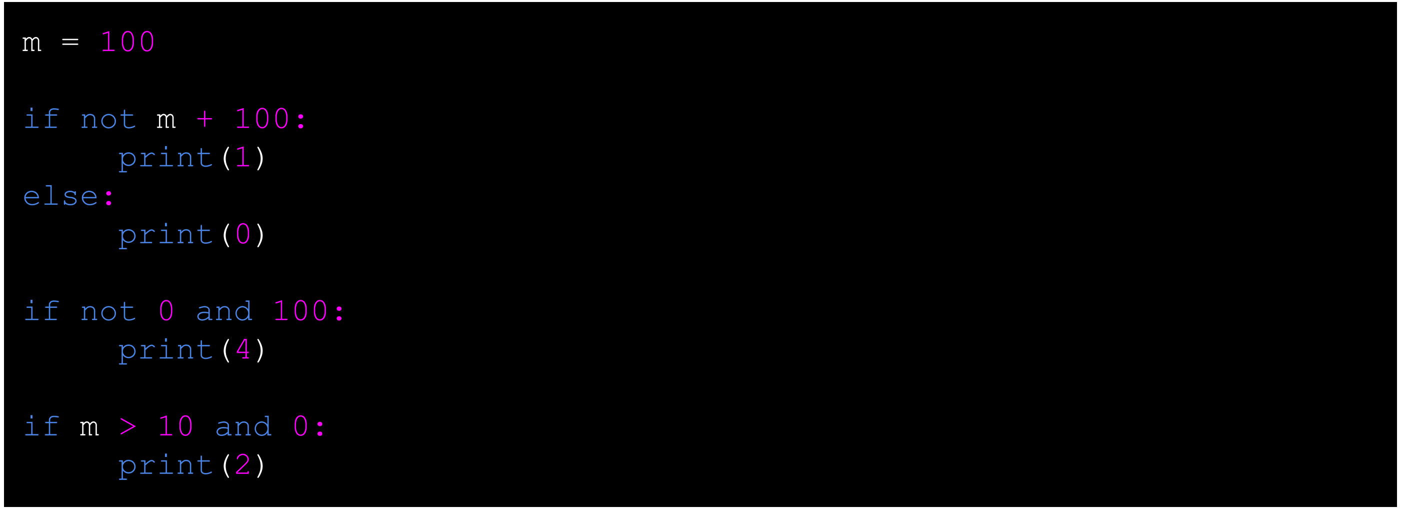
```python
x = 100

if (x == 100):
     print("Universitas Indonesia, universitas kami.")
if (x == 100.0):
     print("Ibukota negara, pusat ilmu budaya bangsa.")
if (x == True):
     print("Kami mahasiswa, pengabdi cita.")
if (x == "100"):
     print("Ngejar ilmu pekerti luhur, tuk nusa dan bangsa.")
```

Review Question 2: Guess the Output!



```python
x = 0

if (x == 0):
    x += 1
    print("1. Pretender (Official Hige Dandism)")
if (x == 1):
    x += 1
    print("2. Kaikai Kitan (Eve)")
if (x == 2):
    x += 1
    print("3. Lemon (Kenshi Yonezu)")
```

Review Question 3: Guess the Output!



```
m = 100

if not m + 100:
     print(1)
else:
     print(0)

if not 0 and 100:
     print(4)

if m > 10 and 0:
     print(2)
```

Review Question 4: Syntax Check

Which one of the following if statements will not execute successfully? Why?

A

```
if (1, 2, 3):
print('foo')
```

B

```
if (1, 2, 3): print('foo')
```

C

```
if (0):


    print('foo')
```

D

```
if (1):
    print('foo')
```

Q&A Session