Chapter No 4. Control Flow

Conditional Statements:

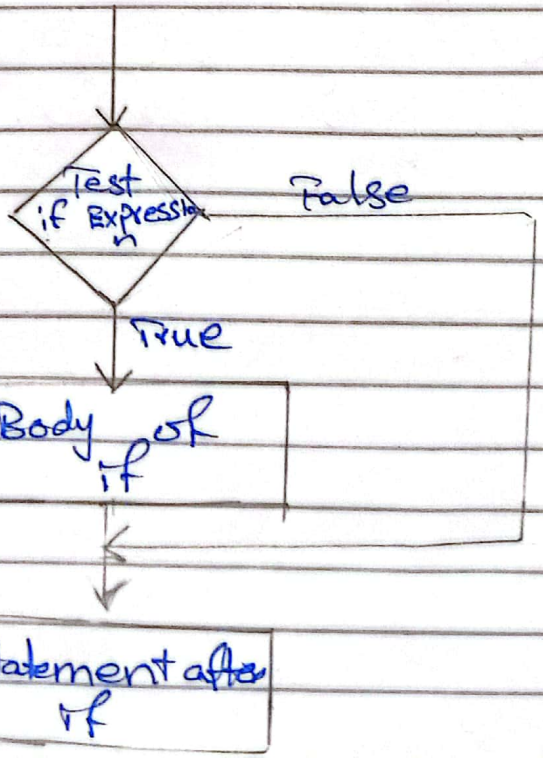
IF Statement:

If statement is the most simple decision-making statement, for deciding whether a certain statement should be executed or not.

Syntax:

```
if condition :
Indent    # Statements to execut if
          # condition is true.
        # Statement after if block
```

Flowchart:



Example:

See 01_if Statement.ipynb

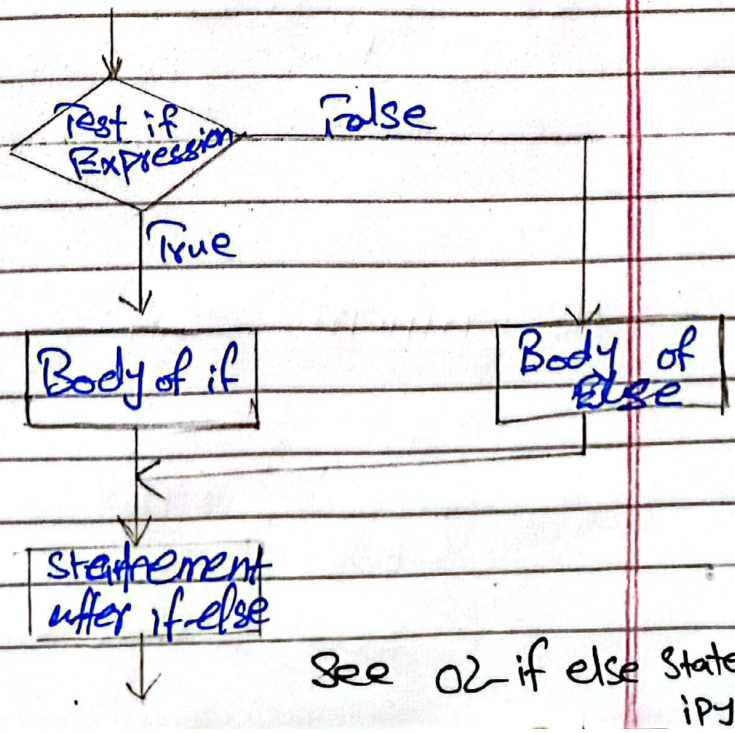
IF - Else Statement:

If we want to execute one statement for if block is True and another for false, then we use if-else statements.

Syntax:

```
if (condition):
Indent    # Executes this block if
          # condition is True
else:
Indent    # Execute this block if
          # condition is False
```

Flow chart:



See 02_if else Statement.ipynb

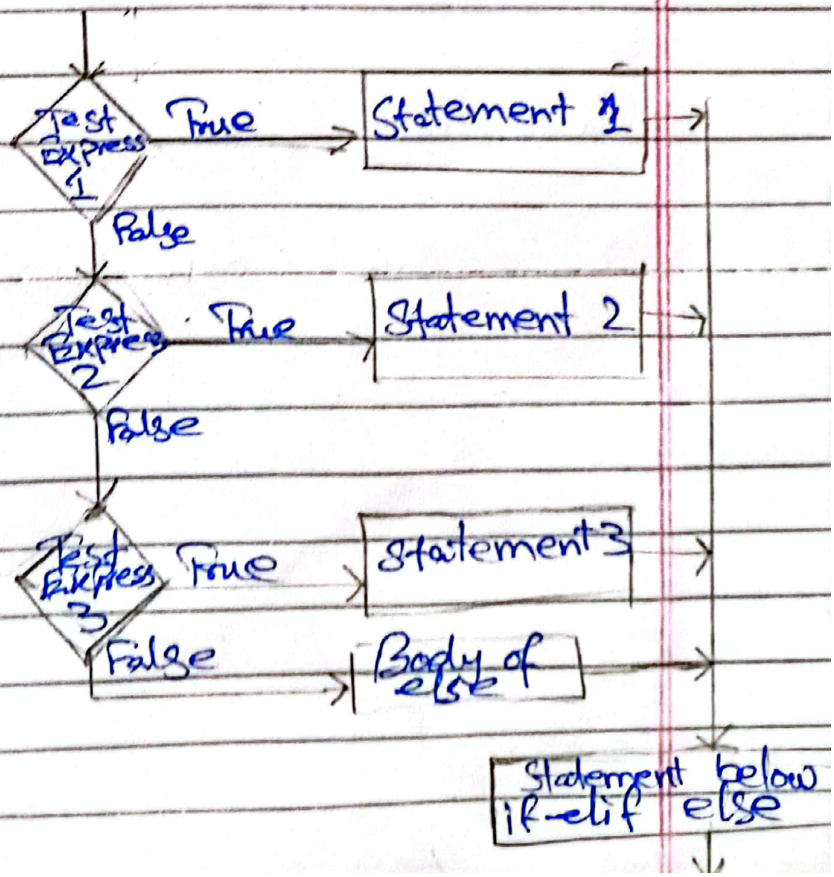
IF-Elif Statement:

Statements In the if-elif else ~~todoter~~, multiple conditions are given. As soon as the one of the if condition is evaluated is True, all other conditions are bypassed.

Syntax:

```
if (condition):
    statement
elif (condition):
    statement
    :
    :
else:
    statement
```

Flowchart:



Example: See 03-if elif and else.ipynb

Nested if Statement:

A nested if is an if statement inside the body of another if statement. We can do this with if-else statement and if-elif as well.

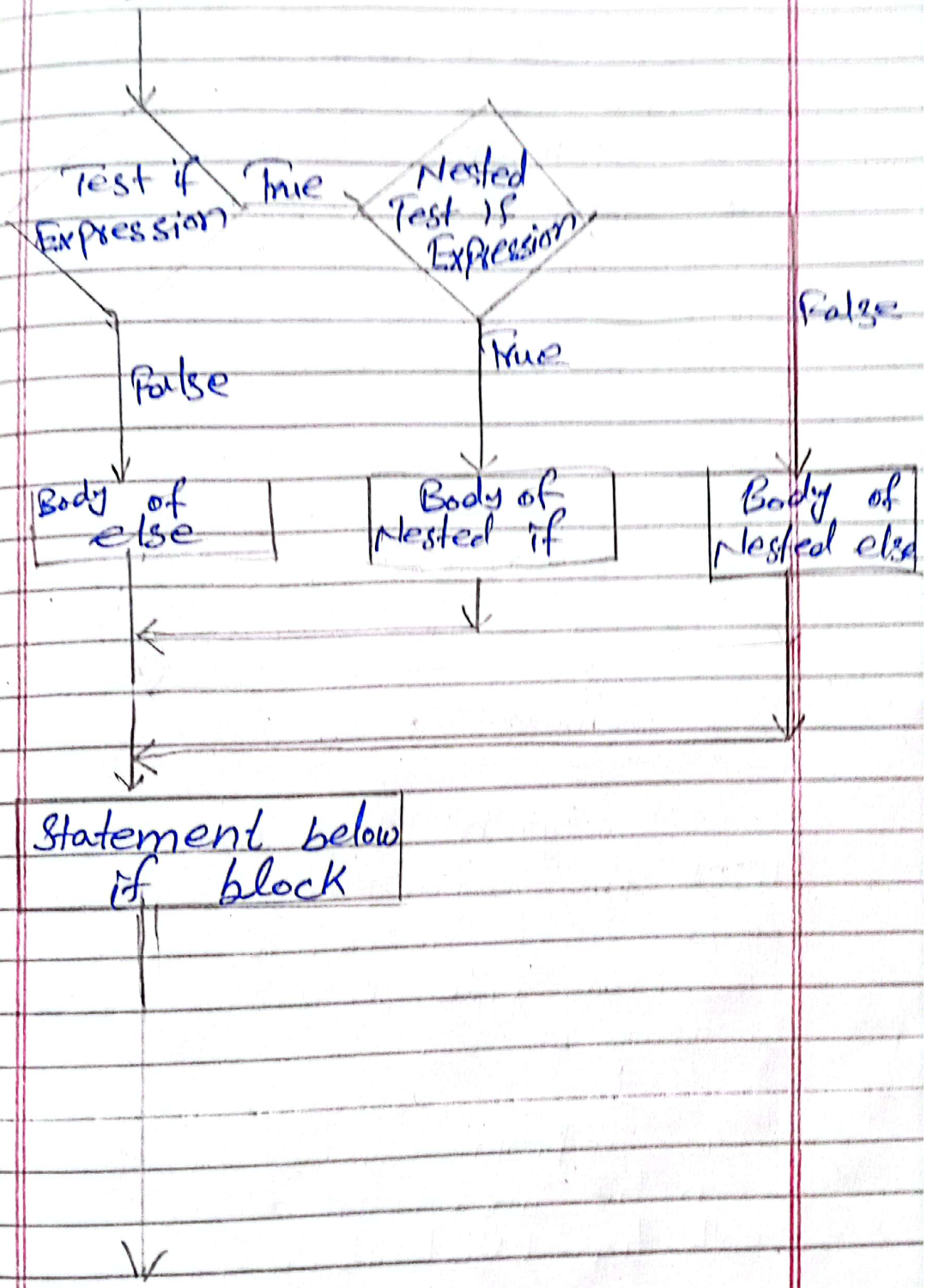
Syntax:

```
if (condition 1):
Indent  # Executes when condition
        #1 is True
Indent  if (condition 2):
        Indent # Executes when
        Indent # condition 2 is True
        #if block end
#if block end
```

Example:

See 04-Nested if.ipynb

Flowchart:



Shorthand if and if-else :

Short hand if and if-else are simple if else statements written in a single line

Example:
See 05-shorthand if else.ipynb

: Loops :

1 For Loop:

Python for loop is used for sequential traversal. It is used to iterate over iterables like lists, strings, Tuple, Dictionary, or set.

Note: In Python for loop only can be implemented on collection-based iterabals

Syntax:

```
for var in iterable:
    #statements
```

Flowchart: