# Modul 3

Operator Kondisi,
Perulangan,
List dan Operasinya

youtube.com/isnaalfi


**github.com/isnaalfi**

## Luaran Pembelajaran

Module ini akan menjelaskan beberapa materi sebagai berikut :

- Nilai Boolean;
- Struktur pemilihan if-elif-else;
- Struktur perulangan while dan for;
- Operasi logika dan bitwise;
- Lists dan Array.

# Operator Relasional </p>

Operator relasional adalah operator yang digunakan untuk membandingkan dua buah nilai. Hasil yang akan diperoleh dari operasi perbandingan ini adalah logika (bertipe bool), yaitu True (benar) atau False (salah).

In [2]:
```
a = input("masukkan a = ")
b = input("masukkan b = ")

c= int(input("masukkan c= "))
d=a
print("a==b",a==b)# sama dengan
print("a!=b",a!=b)# tidak sama dengan

print("a>b",a>b)#
print("b>=a",b>=a)

#print()"a>c", a>c)
```

```
masukkan a = 5
masukkan b = 3
masukkan c= 7
a==b False
a!=b True
a>b True
b>=a False
```

In [ ]:
```
n = int(input())
print(n>=100)
```

# Struktur Pemilihan </p>

Berbeda dengan bahasa pemrograman lainnya, Python hanya memiliki satu statement untuk melakukan proses pemilihan, yaitu dengan menggunakan if.

In [4]:
```
a = 5
b = 5

if a==b:
    print("a dan b bernilai sama")
    print("ini contoh")
print("tidak sama")
```

```
a dan b bernilai sama
ini contoh
tidak sama
```

In [7]:
```
a = 5
b = 6

if a==b:
    print("a dan b bernilai sama")
else:
    print("a dan b bernilai berbeda")
```

```
a dan b bernilai berbeda
```

In [ ]:
```
a = 80

if a > 80:
    print("Lulus")
elif a > 60:
    print("Mengulang")
else:
    print("Tidak Lulus")
```

In [ ]:
```
x = 10

if x > 5: # True
    if x == 6: # False
        print("nested: x == 6")
    elif x == 10: # True
        print("nested: x == 10")
    else:
        print("nested: else")
else:
    print("else")
```

In [6]:
```
a = int(input("masukkan nilai anda= "))
b = 10

c = b if a < b else a
print(c)
```

```
masukkan nilai anda= 23
23
```

# Mencari Bilangan Terbesar </p>

In [8]:
```
# read two numbers
number1 = int(input("Enter the first number: "))
number2 = int(input("Enter the second number: "))

# choose the larger number
if number1 > number2:
    largerNumber = number1
else:
    largerNumber = number2

# print the result
print("The larger number is:", largerNumber)
```

```
Enter the first number: 12
Enter the second number: 45
The larger number is: 45
```

In [9]:
```
# read two numbers
number1 = int(input("Enter the first number: "))
number2 = int(input("Enter the second number: "))

# choose the larger number
largerNumber = number1 if number1>number2 else number2

# print the result
print("The larger number is:", largerNumber)
```

```
Enter the first number: 12
Enter the second number: 35
The larger number is: 35
```

In [ ]:
```
# read three numbers
number1 = int(input("Enter the first number: "))
number2 = int(input("Enter the second number: "))
number3 = int(input("Enter the third number: "))

#asumsikan number1 terbesar
largestNumber = number1


if number2 > largestNumber:
    largestNumber = number2

if number3 > largestNumber:
    largestNumber = number3

# print the result
print("The largest number is:", largestNumber)
```

## Lab

Abrakadabra!

- jika mantra yang diinput adalah `"Abrakadabra"` maka muncul tulisan `"Keajaiban terjadi"`
- jika mantra yang dituliskan menggunakan huruf kecil semua, maka muncul `"Bukan. Mantra harus dengan huruf Kapital"`
- jika mantra berbeda maka muncul tulisan `"Abrakadabra! Bukan [input]!"` (input merupakan input pengguna)

In [12]:
```
mantra = input("Tuliskan mantra= ")

if mantra=="Abrakadabra":
    print("Keajaiban terjadi")
elif mantra=="abrakadabra":
    print("Bukan. Mantra harus dengan huruf Kapital")
else:
    print("Abrakadabra! Bukan " + mantra + "!")
```

```
Tuliskan mantra= mantramantra
Abrakadabra! Bukan mantramantra!
```

## Lab

Di sebuah negara yang makmur dan bahagia, berlaku aturan pajak sebagai berikut:

- nilai `"kena_pajak"` adalah `"income"` dikurangi nilai sesuai `"PTKP"`.
- PTKP terdiri dari 2 jenis, `"Tipe A = 35 juta, Tipe B= 65 juta"`.
- jika `"income < PTKP"` maka nilai `"pajak = 0"`.
- Jika nilai `"kena_pajak"` kurang dari 50 juta maka dikenai pajak 10%
- Jika nilai `"kena_pajak"` lebih dari 50 juta namun kurang dari 250 juta maka dikenai pajak sebesar 15%.
- Jika nilai `"kena_pajak"` lebih dari 250 juta maka dikenai pajak 25%.

Tugas anda adalah membuat **kalkulator pajak**.

- input berupa bilangan desimal dengan nama variabel income
- input pilihan tipe PTKP berupa string.
- tampilkan hasil perhitungan pajak dengan pembulatan menggunakan fungsi round()

In [ ]:
```
income = float(input("Masukan pendapatan dalam juta: "))
PTKP = input("Masukkan tipe PTKP= ").upper()
pajak = 0

kena_pajak = (income-35) if PTKP=='A' else PTKP=='B' (income-65)
if kena_pajak <0:
    pajak=0;
else:
    if kena_pajak <50:
        pajak=0.1*kena_pajak
    elif kena_pajak < 250:
        pajak=0.15*kena_pajak
    else:
        pajak=0.25*kena_pajak

print("Pajak:", pajak, "juta")
```

## Lab

Since the introduction of the Gregorian calendar (in 1582), the following rule is used to determine the kind of year:

- if the year number isn't divisible by four, it's a common year;
- otherwise, if the year number isn't divisible by 100, it's a leap year;
- otherwise, if the year number isn't divisible by 400, it's a common year;
- otherwise, it's a leap year.

Look at the code in the editor - it only reads a year number, and needs to be completed with the instructions implementing the test we've just described.

```
year = int(input("Enter a year: "))

#
# Put your code here.
#
```

The code should output one of two possible messages, which are Leap year or Common year, depending on the value entered.

It would be good to verify if the entered year falls into the Gregorian era, and output a warning otherwise: Not within the Gregorian calendar period. Tip: use the != and % operators.

Test your code using the data we've provided.

Sample input: `2000`

Expected output: `Leap year`

Sample input: `2015`

Expected output: `Common year`

Sample input: `1999`

Expected output: `Common year`

Sample input: `1996`

Expected output: `Leap year`

Sample input: `1580`

Expected output: `Not within the Gregorian calendar period`

# Perulangan </p>

## Perulangan / Looping

Perulangan dengan while:

```
while conditional_expression:
    instruction_one
    instruction_two
    instruction_three
    :
    :
    instruction_n
```

Inifinite Loop

```
while True:
    print("I'm stuck inside a loop.")
```

## Perulangan While</p>

In [ ]:
```
pasien = 1
while pasien<10:
    print(pasien)#memeriksa
    pasien = pasien+1 #masukan daftar
```

## Contoh: Menghitung bilangan terbesar

In [ ]:
```
# we will store the current largest number here
largestNumber = -999999999

# input the first value
number = int(input("Enter a number or type -1 to stop: "))

# if the number is not equal to -1, we will continue
while number != -1:
    # is number larger than largestNumber?
    if number > largestNumber:
        # yes, update largestNumber
        largestNumber = number
    # input the next number
    number = int(input("Enter a number or type -1 to stop: "))

# print the largest number
print("The largest number is:", largestNumber)
```

## Contoh: Menghitung berapa ganjil dan genap dari rangkaian bilangan

In [13]:
```
# program membaca inputan sejumlah nilai
# program menghitung jumlah bilangan genap dan ganjil
# program diakhiri jika mendapat input 0

# 1,2,3,4,5
ganjil = 0
genap = 0

# membaca angka pertama
angka = int(input("Masukkan angka (masukkan 0 jika ingin berhenti): "))

# karena 0 berarti berhenti, maka dibuat kondisi != 0
while angka != 0:
    # cek jika angka ganjil
    if angka % 2 == 1: #jika angka modulus 2 hasilnya 1 (dibagi dua sisa 1), mak
a ganjil
        # tambah jumlah angka ganjil
        ganjil += 1  # ganjil=ganjil-1
    else:
        # increase the evenNumbers counter
        genap += 1
    # input lagi
    angka = int(input("Enter a number or type 0 to stop: "))

# print results
print("Jumlah angka ganjil:", ganjil)
print("Jumlah angka genap:", genap)
```

```
Masukkan angka (masukkan 0 jika ingin berhenti): 2
Enter a number or type 0 to stop: 3
Enter a number or type 0 to stop: 5
Enter a number or type 0 to stop: 0
Jumlah angka ganjil: 2
Jumlah angka genap: 1
```

## Lab

A junior magician has picked a secret number. He has hidden it in a variable named `secretNumber`. He wants everyone who run his program to play the Guess the secret number game, and guess what number he has picked for them. Those who don't guess the number will be stuck in an endless loop forever! Unfortunately, he does not know how to complete the code.

Your task is to help the magician complete the code in the editor in such a way so that the code:

- will ask the user to enter an integer number;
- will use a `while` loop;
- will check whether the number entered by the user is the same as the number picked by the magician. If the number chosen by the user is different than the magician's secret number, the user should see the message `"Ha ha! You're stuck in my loop!"` and be prompted to enter a number again. If the number entered by the user matches the number picked by the magician, the number should be printed to the screen, and the magician should say the following words: `"Well done, muggle! You are free now."`

```
secretNumber = 777

print(
"""
+================================+
| Welcome to my game, muggle!    |
| Enter an integer number        |
| and guess what number I've     |
| picked for you.                |
| So, what is the secret number? |
+================================+
""")
```

The magician is counting on you! Don't disappoint him.

In [ ]:
```
print(
"""
+================================+
| Welcome to my game, muggle!    |
| Enter an integer number        |
| and guess what number I've     |
| picked for you.                |
| So, what is the secret number? |
+================================+
""")

guessNumber = 0
secretNumber = 777

while guessNumber != secretNumber:
    guessNumber = int(input("So, what is the secret number? "))
    if guessNumber != secretNumber:
        print("Ha ha! You're stuck in my loop!")

print("Well done, muggle! You are free now.")
```

# Perulangan For</p>

Bentuk lain dari perulangan selain while

```
for i in range(100):
    # do_something()
```

In [14]:
```
for i in range(5): #0,1,2,3,4
    print("Nilai i = ", i)
```

```
Nilai i =  0
Nilai i =  1
Nilai i =  2
Nilai i =  3
Nilai i =  4
```

In [15]:
```
for i in range(2, 8):
    print("Nilai i = ", i)
```

```
Nilai i =  2
Nilai i =  3
Nilai i =  4
Nilai i =  5
Nilai i =  6
Nilai i =  7
```

In [16]:
```
for i in range(1,8,2):
    print("Nilai i = ", i)
```

```
Nilai i =  1
Nilai i =  3
Nilai i =  5
Nilai i =  7
```

## Lab

Do you know what Mississippi is? Well, it's the name of one of the states and rivers in the United States. The Mississippi River is about 2,340 miles long, which makes it the second longest river in the United States (the longest being the Missouri River). It's so long that a single drop of water needs 90 days to travel its entire length!

The word Mississippi is also used for a slightly different purpose: to count mississippily.

If you're not familiar with the phrase, we're here to explain to you what it means: it's used to count seconds.

The idea behind it is that adding the word Mississippi to a number when counting seconds aloud makes them sound closer to clock-time, and therefore "one Mississippi, two Mississippi, three Mississippi" will take approximately an actual three seconds of time! It's often used by children playing hide-and-seek to make sure the seeker does an honest count.

Your task is very simple here: write a program that uses a for loop to "count mississippily" to five. Having counted to five, the program should print to the screen the final message "Ready or not, here I come!"

Use the skeleton we've provided in the editor.

```
import time

# Write a for loop that counts to five.
    # Body of the loop - print the loop iteration number and the word "Mississipp
i".
    # Body of the loop - use: time.sleep(1)

# Write a print function with the final message.
```

**EXTRA INFO**

Note that the code in the editor contains two elements which may not be fully clear to you at this moment: the import time statement, and the sleep() method. We're going to talk about them soon.

For the time being, we'd just like you to know that we've imported the time module and used the sleep() method to suspend the execution of each subsequent print() function inside the for loop for one second, so that the message outputted to the console resembles an actual counting. Don't worry - you'll soon learn more about modules and methods.

In [18]:
```
import time
for i in range(1,6):
    print(i," Missisipi")
    time.sleep(1)
```

```
1  Missisipi
2  Missisipi
3  Missisipi
4  Missisipi
5  Missisipi
```

# Break dan Continue</p>

In [19]:
```
# break - example
print("The break instruction:")
for i in range(1, 6):
    print(i) #1, 2, 3
    if i == 3:
        break
# continue - example
print("\nThe continue instruction:")
for i in range(1, 6): #2, 3, 4
    if i == 3: #1,2,4
        continue
    print(i)  # 1, 2, 4
```

```
The break instruction:
1
2
3

The continue instruction:
1
2
4
5
```

In [ ]:
```
largestNumber = -99999999
counter = 0

while True:
    number = int(input("Enter a number or type -1 to end program: "))
    if number == -1:
        break
    counter += 1
    if number > largestNumber:
        largestNumber = number

if counter != 0:
    print("The largest number is", largestNumber)
else:
    print("You haven't entered any number.")
```

In [ ]:
```
largestNumber = -99999999
counter = 0

number = int(input("Enter a number or type -1 to end program: "))

while number != -1:
    if number == -1:
        continue
    counter += 1

    if number > largestNumber:
        largestNumber = number
    number = int(input("Enter a number or type -1 to end program: "))

if counter:
    print("The largest number is", largestNumber)
else:
    print("You haven't entered any number.")
```

# Lab

The `break` statement is used to exit/terminate a loop.

Design a program that uses a `while` loop and continuously asks the user to enter a word unless the user enters `"chupacabra"` as the secret exit word, in which case the message `"You've successfully left the loop."` should be printed to the screen, and the loop should terminate.

Don't print any of the words entered by the user. Use the concept of conditional execution and the break statement.

In [ ]:
```
while True:
    word = input("Masukkan Sebuah Kata untuk keluar dari perulangan = ")
    if(word=="chupacabra"):
        break

print("You've successfully left the loop")
```

In [ ]:
```
userWord = input("Masukkan sebuah kata = ")

for letter in userWord:
    print(letter.upper())
```

## Lab

The `continue` statement is used to skip the current block and move ahead to the next iteration, without executing the statements inside the loop.

It can be used with both the while and for loops.

Your task here is very special: you must design a vowel eater! Write a program that uses:

- a for loop;
- the concept of conditional execution (if-elif-else)
- the continue statement.

Your program must:

- ask the user to enter a word;
- use `userWord = userWord.upper()` to convert the word entered by the user to upper case; we'll talk about the so-called string methods and the `upper()` method very soon - don't worry;
- use conditional execution and the `continue` statement to "eat" the following vowels A, E, I, O, U from the inputted word;
- print the uneaten letters to the screen, each one of them on a separate line.

```
# Prompt the user to enter a word
# and assign it to the userWord variable.

for letter in userWord:
    # Complete the body of the for loop.
```

Test your program with the data we've provided for you.

Test data

Sample input: `Gregory`

Expected output:

```
G
R
G
R
Y
```

Sample input: `abstemious`

Expected output:

```
B
S
T
M
S
```

## Vowel Eater

In [ ]:
```
# Prompt the user to enter a word
userWord = input("Masukkan sebuah kata = ")

for letter in userWord:
    if letter.upper() in ['A','I','U','E','O']:
        continue

print(letter.upper())
```

In [20]:
```
wordWithoutVovels = ""

# Prompt the user to enter a word
# and assign it to the userWord variable.
userWord = input("Masukkan sebuah kata = ")

for letter in userWord: #isna alfi bustoni
    if letter.upper() in ['A','I','U','E','O']: #ISNA ALFI BUSTONI
        continue

    wordWithoutVovels += letter.upper() #SNLF

print(wordWithoutVovels)
```

```
Masukkan sebuah kata = isna alfi bustoni
SN LF BSTN
```

# Perulangan dengan Else</p>

In [21]:
```
i = 1
while i < 5:
    print(i)
    i += 1
else:
    print("else:", i)
```

```
1
2
3
4
else: 5
```

In [22]:
```
for i in range(5):
    print(i)
else:
    print("else:", i)
```

```
0
1
2
3
4
else: 4
```

### Lab

Listen to this story: a boy and his father, a computer programmer, are playing with wooden blocks. They are building a pyramid.

Their pyramid is a bit weird, as it is actually a pyramid-shaped wall - it's flat. The pyramid is stacked according to one simple principle: each lower layer contains one block more than the layer above.

The figure illustrates the rule used by the builders:

```
=
==
===
blocks : 6
height : 3
```

Your task is to write a program which reads the number of blocks the builders have, and outputs the height of the pyramid that can be built using these blocks.

Note: the height is measured by the number of fully completed layers - if the builders don't have a sufficient number of blocks and cannot complete the next layer, they finish their work immediately.

Test your code using the data we've provided.

Sample input: `6`

Expected output: The height of the pyramid: `3`

Sample input: `20`

Expected output: The height of the pyramid: `5`

Sample input: `1000`

Expected output: The height of the pyramid: `44`

Sample input: `2`

Expected output: The height of the pyramid: `1`

In [ ]:
```
balok = int(input("Masukan jumlah balok: "))
terpakai = 0
base = 0
while True:
    base+=1
    terpakai+=base
    if terpakai > balok:
        break

print("Tinggi piramid:",base-1)
```

## Lab

In 1937, a German mathematician named Lothar Collatz formulated an intriguing hypothesis (it still remains unproven) which can be described in the following way:

1. take any non-negative and non-zero integer number and name it `c0`;
2. if it's even, evaluate a new `c0` as `c0 ÷ 2`;
3. otherwise, if it's odd, evaluate a new `c0` as `3 × c0 + 1`;
4. if `c0 ≠ 1`, skip to point 2.

The hypothesis says that regardless of the initial value of `c0`, it will always go to 1.

Of course, it's an extremely complex task to use a computer in order to prove the hypothesis for any natural number (it may even require artificial intelligence), but you can use Python to check some individual numbers. Maybe you'll even find the one which would disprove the hypothesis.

Write a program which reads one natural number and executes the above steps as long as `c0` remains different from 1. We also want you to count the steps needed to achieve the goal. Your code should output all the intermediate values of `c0`, too.

Hint: the most important part of the problem is how to transform Collatz's idea into a while loop - this is the key to success.

Test your code using the data we've provided.

**Test Data**

Sample input: `15`

Expected output:

```
46
23
70
35
106
53
160
80
40
20
10
5
16
8
4
2
1
steps = 17
```

Sample input: `16`

Expected output:

```
8
4
2
1
steps = 4
```

Sample input: `1023`

Expected output:

# Exercise

**Exercise 1**

Create a for loop that counts from 0 to 10, and prints odd numbers to the screen. Use the skeleton below:

```
for i in range(1, 11):
    # line of code
        # line of code
```

**Exercise 2**

Create a while loop that counts from 0 to 10, and prints odd numbers to the screen. Use the skeleton below:

```
x = 1
while x < 11:
    # line of code
        # line of code
    # line of code
```

**Exercise 3**

Create a program with a for loop and a break statement. The program should iterate over characters in an email address, exit the loop when it reaches the @ symbol, and print the part before @ on one line. Use the skeleton below:

```
for ch in "john.smith@pythoninstitute.org":
    if ch == "@":
        # line of code
    # line of code
```

**Exercise 4**

Create a program with a for loop and a continue statement. The program should iterate over a string of digits, replace each 0 with x, and print the modified string to the screen. Use the skeleton below:

```
for digit in "0165031806510":
    if digit == "0":
        # line of code
        # line of code
    # line of code
```

**Exercise 5**

What is the output of the following code?

```
n = 3

while n > 0:
    print(n + 1)
    n -= 1
else:
    print(n)
```

**Exercise 6**

What is the output of the following code?

```
n = range(4)

for num in n:
    print(num - 1)
else:
    print(num)
```

# Operasi Logika</p>

Operasi Logika `and`

| Argument A | Argument B | A AND B |
|---|---|---|
| False | False | False |
| False | True | False |
| True | False | False |
| True | True | True |

Operasi Logika `or`

| Argument A | Argument B | A OR B |
|---|---|---|
| False | False | False |
| False | True | True |
| True | False | True |
| True | True | True |

Operasi Logika `not`

| Argument | not Argument |
|---|---|
| False | True |
| True | False |

> Jika tidak ada wabah, **and** saya punya uang jajan, saya akan main ke mall
>
> Jika saya di mall **atau** di toko buku, saya akan selfi.

In [23]:
```
x = 1
y = 0

z = ((x == y) and (x == y)) or not(x == y) # ((false) OR True
print(not(z))
```

```
False
```

# Bitwise</p>

| OPERATOR | DESCRIPTION | SYNTAX |
|---|---|---|
| & | Bitwise AND | x & y |
| \| | Bitwise OR | x \| y |
| ~ | Bitwise NOT | ~x |
| ^ | Bitwise XOR | x ^ y |
| >> | Bitwise right shift | x>> |
| << | Bitwise left shift | x<< |

Bitwise: akan melakukan operasi Integer dilevel bit. Jadi, Integer diubah ke binary, dioperasi, kemudian dikembalikan dalam bentuk desimal.

## Bitwise AND

```
a = 10 = 1010 (Binary)
b = 4 =  0100 (Binary

a & b = 1010
         &
        0100
      = 0000
      = 0 (Decimal)
```

## Bitwise OR

```
a = 10 = 1010 (Binary)
b = 4 =  0100 (Binary

a & b = 1010
         |
        0100
      = 1110
      = 14 (Decimal)
```

## Bitwise NOT

Menambah nilai komplement 1 digit

```
a = 10 = 1010 (Binary)

~a = ~1010
   = -(1010 + 1)
   = -(1011)
   = -11 (Decimal)
```

## Bitwise SHIFT Right >>

Menggeser 1 bit ke kanan, mengisi ruang kosong di kiri dengan nol. Dapat digunakan untuk membagi sebuah nilai dengan pangkat 2 (dari nilai bit yang digeser)

Misal:

- 10 = 0000 1010
- 10 >> 1 = 0000 0101 = 5
- 10/2^1 = 5

```
Example 1:
a = 10
a >> 1 = 5

Example 2:
a = -10
a >> 1 = -5
```

## Bitwise SHIFT Left <<

Menggeser 1 bit ke kiri, mengisi ruang kosong di kanan dengan nol. Dapat digunakan untuk mengkalikan nilai.

- 5 << 1 = 5 * 2^1 = 10
- 5 << 1 = 5 * 2^2 = 20

```
a = 5 = 0000 0101
b = -10 = 1111 0110

a << 1 = 0000 1010 = 10
a << 2 = 0001 0100 = 20

b << 1 = 0000 1010 = -20
b << 2 = 0001 0100 = -40
```

In [24]:
```
a = 10
b = -10

# print bitwise right shift operator
print("a >> 1 =", a >> 1)
print("b >> 1 =", b >> 1)

a = 5
b = -10

# print bitwise left shift operator
print("a << 1 =", a << 1)
print("b << 1 =", b << 1)
```

```
a >> 1 = 5
b >> 1 = -5
a << 1 = 10
b << 1 = -20
```

# List - collection of data</p>

In [25]:
```
var1 = 10
var2 = 5
var3 = 7
var4 = 2
var5 = 1

# VERSUS #

var = [10, 5, 7, 2, 1]
print(var)
```

```
[10, 5, 7, 2, 1]
```

In [26]:
```
#bisa menampung berbagai tipe data
var = ["Digital", 90 , 1000.50 , "Talent"]
print(var)
```

```
['Digital', 90, 1000.5, 'Talent']
```

### Indexing

In [27]:
```
numbers = [10, 5, 7, 2, 1] #list dimulai dari index ke 0
print(numbers)
print(numbers[2])
print(numbers[0])
```

```
[10, 5, 7, 2, 1]
7
10
```

In [28]:
```
numbers = [10, 5, 7, 2, 'a']
print("Original list content:", numbers) # printing original list content

numbers[0] = 111
print("\nPrevious list content:", numbers) # printing previous list content

numbers[1] = numbers[4] # copying value of the fifth element to the second
print("Previous list content:", numbers) # printing previous list content

print("\nList length:", len(numbers)) # printing the list's length
```

```
Original list content: [10, 5, 7, 2, 'a']

Previous list content: [111, 5, 7, 2, 'a']
Previous list content: [111, 'a', 7, 2, 'a']

List length: 5
```

In [ ]:
```
# swap
numbers[0] , numbers[1] = numbers[1] , numbers[0]
print(numbers)
```

## Menghapus Elemen

In [29]:
```
# MENGHAPUS ELEMEN

del numbers[1] # removing the second element from the list
print("New list's length:", len(numbers)) # printing new list length
print("\nNew list content:", numbers) # printing current list content
```

```
New list's length: 4

New list content: [111, 7, 2, 'a']
```

## Menghapus List

In [30]:
```
print(numbers)
del numbers
print(numbers)
```

```
[111, 7, 2, 'a']

---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-30-b5dd281bb807> in <module>
      1 print(numbers)
      2 del numbers
----> 3 print(numbers)

NameError: name 'numbers' is not defined
```

## Indeks Negatif

In [31]:
```
numbers = [111, 7, 2, 1]
print(numbers[-1])
print(numbers[-2])
```

```
1
2
```

## Lab

There once was a hat. The hat contained no rabbit, but a list of five numbers: 1, 2, 3, 4, and 5.

Your task is to:

- write a line of code that prompts the user to replace the middle number in the list with an integer number entered by the user (step 1)
- write a line of code that removes the last element from the list (step 2)
- write a line of code that prints the length of the existing list (step 3.) Ready for this challenge?

In [ ]:
```
hatList = [1, 2, 3, 4, 5]  # This is an existing list of numbers hidden in the h
at.

# Step 1: write a line of code that prompts the user
numb=int(input("Masukkan angka="))

# to replace the middle number with an integer number entered by the user.
hatList[2]=numb

# Step 2: write a line of code here that removes the last element from the list.
del hatList[-1]

# Step 3: write a line of code here that prints the length of the existing list.
print("Isi topi=",len(hatList))

print(hatList)
```

# List - method</p>

| Method | Description |
|---|---|
| append() | Menambah elemen di akhir list |
| clear() | Mengosongkan isi list |
| count() | Menghitung jumlah element tertentu |
| extend() | Mnemabahkan element dari list,ke akhir list yang seleksi |
| index() | Mengembalikan index pertama dari value yang dicari |
| insert() | Menambah elemen di posisi tertentu |
| pop() | Menghapus elemen pada posiis tertentu |
| remove() | Menghapus element pertama yang sesuai dengan value yang dicari |
| reverese() | Membalik order/ susunan list |
| sort() | Melakukan pengurutan list |

In [32]:
```
numbers = [111, 7, 2, 1]
print("Jumlah awal= ", len(numbers))
print("number awal=", numbers)

###

numbers.append(4) #menambahkan 1 elemen di ujung list

print("\nPanjang setelah di-append=", len(numbers))
print("Numbers saat ini=", numbers)

###

numbers.insert(0, 222)#insert "222" ke index 0
print("\nPanjang setelah di-insert=",len(numbers))
print("Numbers saat ini=",numbers)

#
```

```
Jumlah awal=  4
number awal= [111, 7, 2, 1]

Panjang setelah di-append= 5
Numbers saat ini= [111, 7, 2, 1, 4]

Panjang setelah di-insert= 6
Numbers saat ini= [222, 111, 7, 2, 1, 4]
```

In [33]:
```
nama=["Isna","Alfi"]
nama.append("Bustoni")
print(nama)

nama.extend("Bustoni") #list of character
print(nama)

nama.extend(["Dosen","Ilmu","Komputer","UGM"])
print(nama)
```

```
['Isna', 'Alfi', 'Bustoni']
['Isna', 'Alfi', 'Bustoni', 'B', 'u', 's', 't', 'o', 'n', 'i']
['Isna', 'Alfi', 'Bustoni', 'B', 'u', 's', 't', 'o', 'n', 'i', 'Dosen', 'Ilmu
', 'Komputer', 'UGM']
```

In [34]:
```
myList = [] # creating an empty list

for i in range(5):
    myList.append(i + 1)

print(myList)
```

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

In [ ]:
```
myList = [10, 1, 8, 3, 5]
total = 0

for i in range(len(myList)):
    total += myList[i]

print(total)
```

### Lab

The Beatles were one of the most popular music group of the 1960s, and the best-selling band in history. Some people consider them to be the most influential act of the rock era. Indeed, they were included in Time magazine's compilation of the 20th Century's 100 most influential people.

The band underwent many line-up changes, culminating in 1962 with the line-up of John Lennon, Paul McCartney, George Harrison, and Richard Starkey (better known as Ringo Starr).

Write a program that reflects these changes and lets you practice with the concept of lists. Your task is to:

- step 1: create an empty list named `beatles`;
- step 2: use the `append()` method to add the following members of the band to the list: `John Lennon`, `Paul McCartney`, and `George Harrison`;
- step 3: use the `for` loop and the `append()` method to prompt the user to add the following members of the band to the list: `Stu Sutcliffe`, and `Pete Best`;
- step 4: use the `del` instruction to remove `Stu Sutcliffe` and `Pete Best` from the list;
- step 5: use the `insert()` method to add `Ringo Starr` to the beginning of the list. By the way, are you a Beatles fan?

In [ ]:
```
# step 1
print("Step 1:", beatles)

# step 2
print("Step 2:", beatles)

# step 3
print("Step 3:", beatles)

# step 4
print("Step 4:", beatles)

# step 5
print("Step 5:", beatles)


# testing list legth
print("The Fab", len(beatles))
```

### Reverse()

In [35]:
```
tabel=[1,2,6,7,9]
tabel.reverse()
print(tabel)
```

```
[9, 7, 6, 2, 1]
```

# 3.1.5 Bubble Sort

In [ ]:
```
myList = [8, 10, 6, 2, 4] # list to sort
swapped = True # it's a little fake - we need it to enter the while loop

while swapped:
    swapped = False # no swaps so far
    for i in range(len(myList) - 1):
        if myList[i] > myList[i + 1]:
            swapped = True # swap occured!
            myList[i], myList[i + 1] = myList[i + 1], myList[i]

print(myList)
```

In [ ]:
```
myList = []
swapped = True
num = int(input("How many elements do you want to sort: "))

for i in range(num):
    val = float(input("Enter a list element: "))
    myList.append(val)

while swapped:
    swapped = False
    for i in range(len(myList) - 1):
        if myList[i] > myList[i + 1]:
            swapped = True
            myList[i], myList[i + 1] = myList[i + 1], myList[i]

print("\nSorted:")
print(myList)
```

## Operation on List

In [ ]:
```
list1 = [1]  # ini input ke list = 1
list2 = list1[:]  # ini meng-COPY [:] seluruh isi list
list1[0] = 2  # mengubah nilai list1 index 0 menjadi 2
print(list1)
print(list2)#list2 tidak berubah
```

In [ ]:
```
list3=list1.copy()
print(list1)
print(list2)
print(list3)

list1[0]=8
print("\nList 1 dan 3 sekarang")
print(list1)
print(list3)
```

In [38]:
```
# Copying part of the list
myList = [10, 8, 6, 4, 2]
newList = myList[:3] #dari element ke 1 sampai sebelum ke 3, tiga tidka disertak
an
print(newList)
```

```
[10, 8, 6]
```

In [ ]:
```
myList = [10, 8, 6, 4, 2]
newList = myList[1:-1] #dari element ke 1 sampai sebelum terakhir
print(newList)
```

In [ ]:
```
myList = [10, 8, 6, 4, 2]
newList = myList[:3]
print(newList)
```

In [ ]:
```
myList[2:len(myList)]
print(myList)
```

In [39]:
```
siswa=["Ani","Budi","Citra","Deni","Eka","Fajar","Hanung","Isna","Joko","Lukma
n","Mita"]
print(siswa)
print(siswa[1:7:2]) #ambil dari index ke 1, sampai ke 7, per dua

nim=[1,2,3,4,5,6,7,8,9,10]
print("")
print(nim)
print(nim[1:7:2])
```

```
['Ani', 'Budi', 'Citra', 'Deni', 'Eka', 'Fajar', 'Hanung', 'Isna', 'Joko', 'Lu
kman', 'Mita']
['Budi', 'Deni', 'Fajar']

[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
[2, 4, 6]
```

In [40]:
```
siswa=["Ani","Budi","Citra","Deni","Eka","Fajar","Hanung","Isna","Joko","Lukma
n","Mita"]
print("")
print(siswa)
print(siswa[7:2:-1]) #ambil dari index ke 7, sampai ke 2, per satu
```

```
['Ani', 'Budi', 'Citra', 'Deni', 'Eka', 'Fajar', 'Hanung', 'Isna', 'Joko', 'Lu
kman', 'Mita']
['Isna', 'Hanung', 'Fajar', 'Eka', 'Deni']
```

In [41]:
```
nama="ISNA ALFI BUSTONI"
print("")
print(nama)
print(nama[1:-2:6])#ambil dari index ke 1, sampai ke sebelum terakhir ke -2, per
6
```

```
ISNA ALFI BUSTONI
SFT
```

# Menghapus List</p>

## Menggunakan del dan slicing

In [ ]:
```
myList = [10, 8, 6, 4, 2]
del myList[1:3]
print(myList)

del myList[:]
print(myList)

#del myList
#print(myList)
```

## Menggunakan Clear

In [42]:
```
myList = [10, 8, 6, 4, 2]
print(myList)
myList.clear()
print(myList)
```

```
[10, 8, 6, 4, 2]
[]
```

## Menggunakan POP

In [43]:
```
buah = ['apel', 'jeruk', 'mangga']

buah.pop(1) #menghapus berdasar index
print(buah)
```

```
['apel', 'mangga']
```

## Menggunakan Remove

In [44]:
```
myList = [2, 8, 6, 4, 2]
print(myList)
myList.remove(2)#menghapus berdasar value
print(myList)

buah = ['apel', 'jeruk', 'apel','mangga']
print(buah)
buah.remove('apel')
print(buah)
```

```
[2, 8, 6, 4, 2]
[8, 6, 4, 2]
['apel', 'jeruk', 'apel', 'mangga']
['jeruk', 'apel', 'mangga']
```

## Menggunakan Count

In [45]:
```
buah = ['apel', 'jeruk', 'apel','mangga','Apel']

print(buah.count('apel'))
```

```
2
```

## Menggunakan Reverse

In [47]:
```
myList = [2, 8, 6, 4, 1]
print(myList)
myList.reverse()
print(myList)

buah = ['jeruk', 'apel','mangga']
print(buah)
buah.reverse()
print(buah)
```

```
[2, 8, 6, 4, 1]
[1, 4, 6, 8, 2]
['jeruk', 'apel', 'mangga']
['mangga', 'apel', 'jeruk']
```

In [48]:
```
nama="ISNA ALFI"
print(nama)
print(nama[::-1])

myList = [1, 2, 3, 4, 5]
print(myList)
print(myList[::-1])

buah = ['jeruk', 'apel','mangga', 'pisang']
print(buah)
print(buah[::-1])
```

```
ISNA ALFI
IFLA ANSI
[1, 2, 3, 4, 5]
[5, 4, 3, 2, 1]
['jeruk', 'apel', 'mangga', 'pisang']
['pisang', 'mangga', 'apel', 'jeruk']
```

# keyword in & not in</p>

In [49]:
```
myList = [0, 3, 12, 8, 2]
initeks="aiueo"

print("a" in initeks)
print(5 in myList)
print(5 not in myList)
print(12 in myList)
```

```
True
False
True
True
```

In [50]:
```
daftarCekal = ["ani","budi","citra","deni","edi"]

print("budi" in daftarCekal)
print("budi" not in daftarCekal)
print("gerandong" in daftarCekal)
```

```
True
False
False
```

In [ ]:
```
myList = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
toFind = 5
found = False

for i in range(len(myList)):
    found = myList[i] == toFind
    if found:
        break

if found:
    print("Element found at index", i)
else:
    print("absent")
```

# List dalam List </p>

In [51]:
```
kuadrat = [(x ** 2) for x in range(10)] # 0,1,2,-9
print(kuadrat)
```

```
[0, 1, 4, 9, 16, 25, 36, 49, 64, 81]
```

In [52]:
```
ganjil = [z for z in kuadrat if z % 2 != 0 ]
print(ganjil)
```

```
[1, 9, 25, 49, 81]
```

## Membuat matriks

In [53]:
```
board = [] #[*,*,*]
for i in range(3):
    row = ["*" for i in range(3)] #ini isi per baris (3*)
    board.append(row) #menambah baris
print(board)
```

```
[['*', '*', '*'], ['*', '*', '*'], ['*', '*', '*']]
```

In [54]:
```
table = [["a", "b", "c"],
         ["d","e", "f"],
         ["g", "h","i"]]

#print(table)
print(table[0][0])
print(table[2][1])
```

```
a
h
```

In [55]:
```
# Cube - a three-dimensional array (3x3x3)

cube = [[[':(', 'x', 'z'],
         [':)', 'x', 'za'],
         [':(', 'x', 'z']],

        [[':)', 'x', 'z'],
         [':(', 'x', 'z'],
         [':)', 'x', 'z']],

        [[':(', 'x', 'z'],
         [':)', 'x', 'z'],
         [':)', 'hore :)', 'z']]]

#print(cube)
print(cube[0][1][2])
print(cube[2][2][1])
```

```
za
hore :)
```

In [ ]:
```

```