#1

Buatlah suatu program untuk mengurutkan array mahasiswa berdasarkan NIM, yang elemennya terbuat dari class MhsTIF, yang telah kamu buat sebelumnya.

In [1]:
```
# L200220194
class Manusia(object):
    """ Class 'Manusia' dengan inisiasi 'nama' """
    keadaan = 'lapar'

    def __init__(self, nama):
        self.nama = nama

    def ucapkanSalam(self):
        print("Salaam, namaku", self.nama)

    def makan(self, s):
        print("Saya baru saja makan", s)
        self.keadaan = 'kenyang'

    def olahraga(self, k):
        print("Saya baru saja latihan", k)
        self.keadaan = 'lapar'

    def mengalikanDenganDua(self, n):
        return n * 2

# Kelas Mahasiswa
class Mahasiswa(Manusia):
    """Class Mahasiswa yang dibangun dari class Manusia."""
    def __init__(self, nama, NIM, kota, us):
        """Metode inisiasi ini menutupi metode inisiasi di class Manusia.
        self.nama = nama
        self.NIM = NIM
        self.kotaTinggal = kota
        self.uangSaku = us
        self.listKuliah = []

    def __getitem__(self, key=None):
        if key is None:
            return vars(self)
        else:
            return getattr(self, key)


    def __str__(self):
        s = self.nama + ', NIM ' + str(self.NIM) \
            + '. Tinggal di ' + self.kotaTinggal \
            + '. Uang saku Rp ' + str(self.uangSaku) \
            + ' tiap bulannya.'
        return s

    def ambilNama(self):
        return self.nama
```

```
    def ambilNIM(self):
        return self.NIM

    def ambilUangSaku(self):
        return self.uangSaku

    def makan(self, s):
        """Metode ini menutupi metode 'makan' dari class Manusia.
        Mahasiswa makan sambil belajar."""
        print("Saya baru saja makan", s, "sambil belajar.")
        self.keadaan = 'kenyang'

    def ambilKotaTinggal(self):
        return self.kotaTinggal

    def perbaruiKotaTinggal(self, kota):
        self.kotaTinggal = kota

    def tambahUangSaku(self, uang):
        self.uangSaku += uang

    def ambilKuliah(self, mk):
        mk = str(mk)
        self.listKuliah = self.listKuliah + [mk]

    def hapusKuliah(self, mk):
        mk = str(mk)
        self.listKuliah.remove(mk)

class MhsTIF(Mahasiswa):  # Perhatikan class induknya: Mahasiswa
    """Class MhsTIF yang dibangun dari class Mahasiswa"""
    def katakanPy(self):
        print('Python is cool.')

# Array Mahasiswa
class daftarMhs:
    def __init__(self, src=None, nama=None, nim=None, kota=None, us=None)
        self.data = []
        self.count = 0
        if all(arg is not None for arg in (src, nama, nim, kota)):
            us = kota
            kota = nim
            nim = nama
            nama = src
            self.data.append(MhsTIF(nama, nim, kota, us))
            self.count += 1
        elif all(arg is not None for arg in (nama, nim, kota, us)):
            self.data.append(MhsTIF(nama, nim, kota, us))
            self.count += 1
        elif src is not None:
            if isinstance(src, MhsTIF):
                self.data.append(src)
                self.count += 1
            elif isinstance(src, list):
                if isinstance(src[0], MhsTIF):
                    self.data.extend(src)
                    self.count += len(src)
                elif isinstance(src[0], tuple):
                    for i in src:
```

```
                        self.data.append(MhsTIF(*i))
                        self.count += 1
                else:
                    print("Data tidak bisa dimasukkan")
            else:
                print("Data tidak bisa dimasukkan")

    def __setitem__(self, index, value):
        if 0 <= index < len(self.data):
            self.data[index] = value
        else:
           raise IndexError("Indeks diluar rentang array mahasiswa")

    def __getitem__(self, key=None):
        results = []
        index = val = None
        if(isinstance(key, tuple)):
            index, val = key
        if(isinstance(key, int)):
            return self.data[key]
        elif(isinstance(key, str)):
            for mhs in self.data:
                try:
                    value = getattr(mhs, key)
                    results.append(value)
                except AttributeError:
                    pass
            return results
        elif(index != None and isinstance(index, int) and isinstance(val,
            rest = None
            try:
                value = getattr(self.data[index], val)
                rest = value
            except AttributeError:
                pass
            return rest
        else:
            return [vars(mhs) for mhs in self.data]

    def __getattr__(self, key="None"):
        if all(hasattr(mhs, key) for mhs in self.data) :
            return [getattr(mhs, key, None) for mhs in self.data]
        else:
            raise AttributeError(f"'{self.__class__.__name__}' object has

    def __repr__(self):
        return repr([vars(mhs) for mhs in self.data])

    def __sizeof__(self):
        return len(self.data)
```

In [2]:

```
def insertionSortNIM(A):
    n = len(A.data)
    for i in range(1, n):
        nim = int(A[i, 'NIM'])  # Menggunakan __getitem__ untuk mengakses
        tmp = A[i]
        pos = i
        while pos > 0 and nim < int(A[pos - 1, 'NIM']):  # Perbandingan n
            A[pos] = A[pos - 1]
            pos = pos - 1
```

```
        A[pos] = tmp
    # return A
```

In [3]:
```
!bash sortNIM.sh
```

```
Memulai skrip isi shell interaktif Python...

>>> from sortNIM import *
>>> c0 = MhsTIF('Ika', 10, 'Sukoharjo', 240000)
>>> c1 = MhsTIF('Budi', 51, 'Sragen', 230000)
>>> c2 = MhsTIF('Ahmad', 2, 'Surakarta', 250000)
>>> c3 = MhsTIF('Chandra', 18, 'Surakarta', 235000)
>>> c4 = MhsTIF('Eka', 4, 'Boyolali', 240000)
>>> c5 = MhsTIF('Fandi', 31, 'Salatiga', 250000)
>>> c6 = MhsTIF('Deni', 13, 'Klaten', 245000)
>>> c7 = MhsTIF('Galuh', 5, 'Wonogiri', 245000)
>>> c8 = MhsTIF('Janto', 23, 'Klaten', 245000)
>>> c9 = MhsTIF('Hasan', 64, 'Karanganyar', 270000)
>>> c10 = MhsTIF('Khalid', 29, 'Purwodadi', 265000)
>>>
>>> n = [c0, c1, c2, c3, c4, c5, c6, c7, c8, c9, c10]
>>> daftar = daftarMhs(n)
>>>
>>> for i, mhs in enumerate(daftar, 0): print(f'[{i}] {mhs['nama']}, NIM
{mhs['NIM']}')
[0] Ika, NIM 10
[1] Budi, NIM 51
[2] Ahmad, NIM 2
[3] Chandra, NIM 18
[4] Eka, NIM 4
[5] Fandi, NIM 31
[6] Deni, NIM 13
[7] Galuh, NIM 5
[8] Janto, NIM 23
[9] Hasan, NIM 64
[10] Khalid, NIM 29
>>>
>>> insertionSortNIM(daftar)
>>>
>>> for i, mhs in enumerate(daftar, 0): print(f'[{i}] {mhs['nama']}, NIM
{mhs['NIM']}')
[0] Ahmad, NIM 2
[1] Eka, NIM 4
[2] Galuh, NIM 5
[3] Ika, NIM 10
[4] Deni, NIM 13
[5] Chandra, NIM 18
[6] Janto, NIM 23
[7] Khalid, NIM 29
[8] Fandi, NIM 31
[9] Budi, NIM 51
[10] Hasan, NIM 64

Keluar dari shell interaktif Python.
```

2

Misal terdapat dua buah array yang sudah urut A dan B.

Buatlah suatu program untuk menggabungkan, secara efisien, kedua array itu menjadi suatu array C yang urut.

In [4]:
```
def gabungkanDuaArray(A, B):
    la = len(A)
    lb = len(B)
    C = list()  # C adalah list baru
    i = 0
    j = 0

    # Gabungkan keduanya sampai salah satu kosong
    while i < la and j < lb:
        if A[i] < B[j]:
            C.append(A[i])
            i += 1
        else:
            C.append(B[j])
            j += 1

    # Sisipkan sisa dari A jika ada
    while i < la:
        C.append(A[i])
        i += 1

    # Sisipkan sisa dari B jika ada
    while j < lb:
        C.append(B[j])
        j += 1

    print("Array setelah di gabungkan\n", C)
```

In [5]:
```
arr1 = [1, 3, 5, 7, 7]
arr2 = [2, 4, 6, 8]
gabungkanDuaArray(arr1, arr2);
```

```
Array setelah di gabungkan
 [1, 2, 3, 4, 5, 6, 7, 8]
```

3

Kamu mungkin sudah menduga, bubble sort lebih lambat dari selection sort dan juga insertion sort. Tapi manakah yang lebih cepat antara selection sort dan insertion sort?

Untuk memulai menyelidikinya, kamu bisa membandingkan waktu yang diperlukan untuk mengurutkan sebuah array yang besar, misal sepanjang 6000 (enam ribu) elemen.

```
from time import time as detak
from random import shuffle as kocok
k = range(1,6001)
kocok(k)
u_bub = k[:] ## \\
u_sel = k[:] ## -- Jangan lupa simbol [:]-nya!.
u_ins = k[:] ## //
```

```
aw=detak();bubbleSort(u_bub);ak=detak();print('bubble: %g detik'
%(ak-aw) );
aw=detak();selectionSort(u_sel);ak=detak();print('selection: %g
detik' %(ak-aw) );
aw=detak();insertionSort(u_ins);ak=detak();print('insertion: %g
detik' % (ak-aw) );
```

Bandingkan hasil percobaan kamu dengan hasil teman-temanmu. Jika waktu untuk pengurutan dirasa terlalu cepat, kamu bisa memperbesar ukuran array itu.

In [6]:
```
def swap(A, p, q):
    tmp = A[p]
    A[p] = A[q]
    A[q] = tmp

def bubbleSort(A):
    n = len(A)
    for i in range(n-1):        #-> Lakukan operasi gelembung sebanyak n-
        for j in range(n-i-1):  #-> Dorong elemen terbesar ke ujung kanan
            if A[j] > A[j+1]:   #-> Jika di kiri lebih besar dari di kana
                swap(A,j,j+1)   #-> tukar posisi elemen ke j dengan ke j+

def cariPosisiYangTerkecil(A, dariSini, sampaiSini):
    posisiYangTerkecil = dariSini           #-> anggap ini yang terkecil
    for i in range(dariSini, sampaiSini):   #-> cari di sisa list
        if A[i] < A[posisiYangTerkecil]:    #-> kalau menemukan yang lebi
            posisiYangTerkecil = i          #-> anggapan dirubah
    return posisiYangTerkecil

def selectionSort(A):
    n = len(A)
    for i in range(n-1):
        indexKecil = cariPosisiYangTerkecil(A, i, n)
        if indexKecil != i:
            swap(A, i, indexKecil)

def insertionSort(A):
    n = len(A)
    for i in range(1, n):
        nilai = A[i]
        pos = i
        while pos > 0 and nilai < A[pos - 1]:   # -> Cari posisi yang tep
            A[pos] = A[pos - 1] # dan geser ke kanan terus
            pos = pos - 1       # nilai-nilai yang lebih besar
        A[pos] = nilai # -> Pada posisi ini tempatkan nilai elemen ke i.
```

In [7]:
```
from time import time as detak
from random import shuffle as kocok
k = list(range(1,6001))
kocok(k)
u_bub = k[:] ## \\
u_sel = k[:] ## -- Jangan lupa simbol [:]-nya!.
u_ins = k[:] ## //

aw=detak();bubbleSort(u_bub);ak=detak();print('bubble: %g detik' %(ak-aw)
aw=detak();selectionSort(u_sel);ak=detak();print('selection: %g detik' %(
```

```
aw=detak();insertionSort(u_ins);ak=detak();print('insertion: %g detik' %
```

```
bubble: 4.27772 detik
selection: 1.63022 detik
insertion: 1.38678 detik
```