အားလုံးပဲ မင်္ဂလာပါ။ Data Analytics with Python မှာ Python Programming Language ကို အသုံးပြုပြီး ဒေတာတွေကို ဘယ်လိုမျိုး လွယ်လွယ်ကူကူနဲ့ Analysis ပြုလုပ်မလဲ? မိမိရဲ့ရလဒ်တွေကို ပြန်လည်တင်ပြဖို့အတွက် ဘယ်လိုမျိုး ကားချပ်တွေဆွဲပြီး ဖော်ပြကြမလဲဆိုတာကို သင်ကြားပေးသွားမှာ ဖြစ်ပါတယ်။

Python Language ကိုအသုံးပြုပြီး Data Analysis နဲ့ Data Visualization ကို ပြုလုပ်တဲ့နေရာမှာ အဓိကအားဖြင့်အသုံးများတဲ့ Python Library သုံးခု ရှိပါတယ်။

1. Numpy
2. Pandas
3. Matplotlib တို့ ဖြစ်ပါတယ်။

Getting Started with Numpy

Numpy ဆိုတာက Python Programming Language ကိုအသုံးပြုပြီး သိပ္ပံဆိုင်ရာတွက်ချက်မှုတွေ လုပ်ဆောင်ဖို့ ဖန်တီးထားတဲ့ Python Library တစ်ခု ဖြစ်ပါတယ်။ Numpy ရဲ့ နာမည်အရှည်က Numerical Python ဖြစ်ပါတယ်။ Numpy အသုံးပြုပုံကို စတင်လေ့လာလိုက်ရအောင်။

ကျွန်တော်တို့ဆီမှာ အမေရိကန်သမ္မတ ၄၅ ယောက်ရဲ့ အသက်တွေကို စုစည်းထားတဲ့ List တစ်ခု ရှိပါတယ်။ ကျွန်တော်တို့ဆီမှာ အမေရိကန်သမ္မတ ၄၅ ယောက်ရဲ့ အရပ်အမောင်းတွေကို စုစည်းထားတဲ့ List တစ်ခု ရှိပါတယ်။ စင်တီမီတာနဲ့ တိုင်းတာထားတာ ဖြစ်ပါတယ်။

```
heights = [189, 170, 189, 163, 183, 171, 185, 168, 173, 183, 173,
           173, 175, 178, 183, 193, 178, 173, 174, 183, 183, 180,
           168, 180, 170, 178, 182, 180, 183, 178, 182, 188, 175,
           179, 183, 193, 182, 183, 177, 185, 188, 188, 182, 185, 191]
```

ဒီ ဥပမာကိုကြည့်မယ်ဆိုရင် ပထမဆုံးအမေရိကန်သမ္မတက ဂျော့ဝါရှင်တန်ဖြစ်ပြီးတော့ သူ့ရဲ့ အရပ်အမောင်းက 189 cm ဖြစ်ပါတယ်။ ဒီတော့ ကျွန်တော်တို့က သူ့လိုမျိုး အရပ် 188 cm ထက်ရှည်တဲ့ သမ္မတဘယ်နှစ်ဦးရှိသလဲသိချင်ရင် အောက်ပါအတိုင်း Python နဲ့ ရေးသားရမှာ ဖြစ်ပါတယ်။

```
count = 0
for height in heights:
    if height > 188:
        count = count + 1
print(count)
```

အဖြေက 5 ဖြစ်ပါတယ်။

Numpy ကို အသုံးပြုပြီးတော့ ခုနက မေးခွန်းကို အောက်ပါအတိုင်း အလွယ်တကူဖြေရှင်းနိုင်ပါတယ်။

```
import numpy as np
heights_arr = np.arrany(heights)
taller = (heights_arr > 188).sum()
print(taller)
```

import က **numpy** library ကို import ပြုလုပ်တာဖြစ်ပြီး ဒုတိယစာကြောင်းက numpy ရဲ့ object တစ်ခုဖြစ်တဲ့ array ကို ဖန်တီးလိုက်တာဖြစ်ပါတယ်။ Python Beginner ပြီးထားပြီးသားသူတွေဖြစ်လို့ object အကြောင်းကိုသိပြီးသားလို့ မျှော်လင့်ပါတယ်။ array ပြုလုပ်တဲ့အခါမှာ အပေါ်မှာကျွန်တော်တို့ရဲ့ ဒေတာတွေကို သိမ်းဆည်းထားတဲ့ **heights** ဆိုတဲ့ list ကို ထည့်သွင်းပေးလိုက်ပါတယ်။ တတိယစာကြောင်းကတော့ 188 cm ထက်ရှည်တဲ့ ဒေတာတွေကို ရှာပေးတာဖြစ်ပြီး **.sum()** က ရှာလို့ရလာတဲ့ရလဒ်တွေကို ပေါင်းပြီး အဖြေထုတ်ပေးတာဖြစ်ပါတယ်။

လောလောဆယ်မှာ ကနဦးရေးပြခဲ့တဲ့ for loop နဲ့ သိပ်မကွာပေမဲ့ ဒေတာတွေတဖြည်းဖြည်းများလာတဲ့အခါ အခုလို loop ပတ်စရာမလိုဘဲ ရေးလို့ရတဲ့နည်းလမ်းတွေက ပိုပြီးလွယ်ကူမြန်ဆန်စေမှာပါ။

Size and Shape

array object ရဲ့ မိခင် class က ndarray ဖြစ်ပါတယ်။ (အရှည်က n-dimensional array ဖြစ်ပါတယ်။) ndarray မှာပါတဲ့ attribute တစ်ခုက size ဖြစ်ပါတယ်။ python object oriented programming ကို နားလည်ပြီးသားဆိုရင် attribute ကို သိပြီးသားဖြစ်မှာပါ။ ndarray မှာပါတဲ့ size attribute ကိုအသုံးပြုပြီး array တစ်ခုမှာပါတဲ့ ဒေတာတန်ဖိုးအရေအတွက်ကို သိရှိနိုင်ပါတယ်။

```
heights_arr.size
```

```
45
```

ndarray ရဲ့ အသုံးဝင်တဲ့နောက်ထပ် attribute တစ်ခုက shape ဖြစ်ပါတယ်။ dimension အရေအတွက်ကို ပြောတာဖြစ်ပါတယ်။ .shape ကိုအသုံးပြုမယ်ဆိုရင် ပြန်ရလာမဲ့ ဒေတာက tuple အမျိုးအစား ဖြစ်ပါတယ်။

```
heights_arr.shape
```

```
(45,)
```

ကျွန်တော်တို့ရဲ့ array မှာ dimension တစ်ခုပဲ ရှိတဲ့အတွက် (45,) ဆိုပြီး ပြန်ထွက်လာပါတယ်။ list တစ်ခုထဲကို အသုံးပြုပြီး array ပြုလုပ်ထားတာဖြစ်တဲ့အတွက် dimension တစ်ခုပဲ ရှိတာဖြစ်ပါတယ်။ ပိုပြီးရှင်းလင်းသွားအောင် 2 Dimension array တစ်ခု ဆွဲပြပါမယ်။

```
listA = [ [1,2,3,4],[6,7,8,9] ]
myarr = np.array(listA)
myarr.shape
```

```
(2, 4)
```

ပထမဆုံးမှာ ကျွန်တော်တို့က list နှစ်ခုပါဝင်တဲ့ list တစ်ခု ဖန်တီးလိုက်ပါတယ်။ .shape နဲ့ ကြည့်လိုက်တဲ့အခါမှာ (2, 4) ဆိုပြီး ပြန်ထွက်လာပါတယ်။

ကျွန်တော်တို့က array အဖြစ်ဖန်တီးလိုက်တဲ့အခါမှာ အောက်ပါပုံစံအတိုင်း array က တည်ရှိသွားပါတယ်။

1	2	3	4
6	7	8	9

ဒီတော့ (2, 4) မှာ အရှေ့က 2 က row ကို ကိုယ်စားပြုပါတယ်။

1	1	2	3	4
2	6	7	8	9

(2, 4) မှာ အနောက်က 4 က column ကို ကိုယ်စားပြုပါတယ်။

1	**2**	**3**	**4**
1	2	3	4
6	7	8	9

အထက်မှာပြောခဲ့တဲ့ (45,) ကတော့ one dimension array ဖြစ်တဲ့အတွက် row တန်ဖိုးမပါတော့ဘဲ ထွက်လာတာ ဖြစ်ပါတယ်။ , ရဲ့ ရှေ့မှာရေးထားပေမဲ့ 45 ကို row လို့ ယူဆလို့မရပါဘူး။ one dimensional array မှာ ဒီအချက်ကို သတိပြုရပါမယ်။

Reshape

ကျွန်တော်တို့ရဲ့ list (အရပ်အမောင်း)ဒေတာထဲကို သမ္မတတွေရဲ့ အသက်ကို ထပ်မံထည့်သွင်းချင်ပါတယ်။ သမ္မတအရေအတွက်က အတူတူပဲဖြစ်တဲ့အတွက် ဒီ list နှစ်ခုကို အလွယ်တကူ ပေါင်းနိုင်ပါတယ်။

```
ages = [57, 61, 57, 57, 58, 57, 61, 54, 68, 51, 49,
        64, 50, 48, 65, 52, 56, 46, 54, 49, 51, 47,
        55, 55, 54, 42, 51, 56, 55, 51, 54, 51, 60,
        62, 43, 55, 56, 61, 52, 69, 64, 46, 54, 47, 70]
```

```
heights_and_ages = heights + ages
# convert the list to a numpy array
heights_and_ages_arr = np.array(heights_and_ages)
```

အရပ်အမောင်းနဲ့ အသက်ကို ပေါင်းထားတဲ့ array ရဲ့ shape ကို ကြည့်မယ်ဆိုရင် one dimensional array ဖြစ်ပြီး သူမှာ ပါဝင်တဲ့အရေအတွက်က 90 အထိရှိနေတာ တွေ့ရမှာပါ။ array က အရမ်းရှည်လျားသလို ကျွန်တော်တို့က height တန်ဖိုးနဲ့ age တန်ဖိုးကို သီးခြားစီလိုချင်တာဖြစ်တဲ့အတွက် array ရဲ့ shape ကို two-dimensional array အဖြစ် ပြောင်းလဲပါမယ်။

```
heights_and_ages_arr.reshape((2,45))
```

```
array([[189, 170, 189, 163, 183, 171, 185, 168, 173, 183, 173, 173, 175,
        178, 183, 193, 178, 173, 174, 183, 183, 180, 168, 180, 170, 178,
        182, 180, 183, 178, 182, 188, 175, 179, 183, 193, 182, 183, 177,
        185, 188, 188, 182, 185, 191],
       [ 57,  61,  57,  57,  58,  57,  61,  54,  68,  51,  49,  64,  50,
         48,  65,  52,  56,  46,  54,  49,  51,  47,  55,  55,  54,  42,
         51,  56,  55,  51,  54,  51,  60,  62,  43,  55,  56,  61,  52,
         69,  64,  46,  54,  47,  70]])
```

reshape က array ကို ပုံစံပြောင်းလိုက်တာဖြစ်ပြီး မူလ array ကတော့ စတင်ဖန်တီးထားတဲ့ ပုံစံအတိုင်း ရှိနေမှာပါ။

```
heights_and_ages_arr
```

```
array([189, 170, 189, 163, 183, 171, 185, 168, 173, 183, 173, 173, 175,
       178, 183, 193, 178, 173, 174, 183, 183, 180, 168, 180, 170, 178,
       182, 180, 183, 178, 182, 188, 175, 179, 183, 193, 182, 183, 177,
       185, 188, 188, 182, 185, 191,  57,  61,  57,  57,  58,  57,  61,
        54,  68,  51,  49,  64,  50,  48,  65,  52,  56,  46,  54,  49,
        51,  47,  55,  55,  54,  42,  51,  56,  55,  51,  54,  51,  60,
        62,  43,  55,  56,  61,  52,  69,  64,  46,  54,  47,  70])
```

Data Type

array မှာ မှတ်သားရမဲ့အချက်တစ်ခုက array ထဲမှာပါဝင်တဲ့ ဒေတာတန်ဖိုးတွေဟာ data type တစ်မျိုးတည်းဖြစ်ရပါမယ်။ integer တွေထဲမှာ float ပါဝင်လို့ မရပါဘူး။ အဲ့ဒီလိုပါဝင်ခဲ့ရင် integer တန်ဖိုးတွေဟာ float အဖြစ် ပြောင်းလဲသွားမှာပါ။ ဥပမာ - **heights** list ထဲမှာ ကျွန်တော်တို့က integer

တွေ့ချည်းပဲ ထည့်ထားတာပါ။ **heights** list ကို array ပြောင်းထားတဲ့ **heights_arr** ရဲ့ data type ကို **dtype** အသုံးပြုပြီး သိရှိနိုင်ပါတယ်။

```
heights_arr.dtype
```

```
dtype('int32')
```

အကယ်၍ ကျွန်တော်တို့ရဲ့ ဒေတာ List ထဲမှာ float တန်ဖိုးတစ်ခုပါဝင်လိုက်မယ်ဆိုရင် float အဖြစ်ပြောင်းသွားတာကို တွေ့ရပါလိမ့်မယ်။

```
heights_float = [189.0, 170, 189, 163, 183, 171, 185, 168, 173, 183, 173,
                 173, 175, 178, 183, 193, 178, 173, 174, 183, 183, 180,
                 168, 180, 170, 178, 182, 180, 183, 178, 182, 188, 175,
                 179, 183, 193, 182, 183, 177, 185, 188, 188, 182, 185, 191]
```

```
heights_float_arr = np.array(heights_float)
heights_float_arr
```

```
array([189., 170., 189., 163., 183., 171., 185., 168., 173., 183., 173.,
       173., 175., 178., 183., 193., 178., 173., 174., 183., 183., 180.,
       168., 180., 170., 178., 182., 180., 183., 178., 182., 188., 175.,
       179., 183., 193., 182., 183., 177., 185., 188., 188., 182., 185.,
       191.])
```

ဒေတာအမျိုးအစားကို ကြည့်လိုက်ရင်လည်း float အဖြစ် ရှိနေပါလိမ့်မယ်။

```
heights_float_arr.dtype
```

```
dtype('float64')
```

array တွေကို slice လုပ်တာ၊ index အသုံးပြုတာ စတာတွေကို နောက်တစ်ပိုင်းမှာ ပြောပြသွားပါမယ်။

ဇော်မျိုးထက်