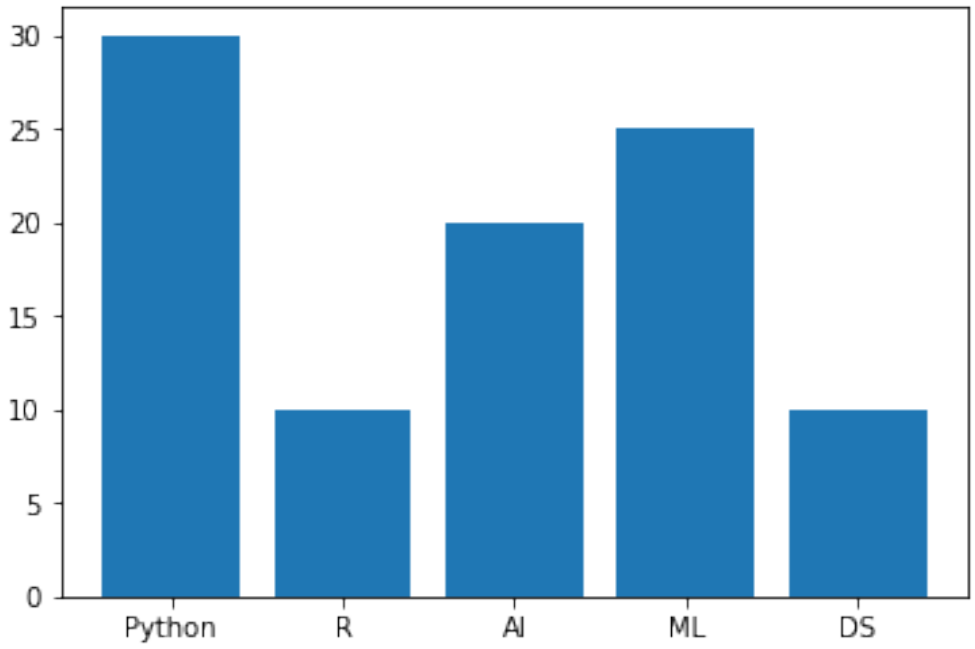
Python Matplotlib Tutorial Part - 5 & part -6

Ploting Bar Chart

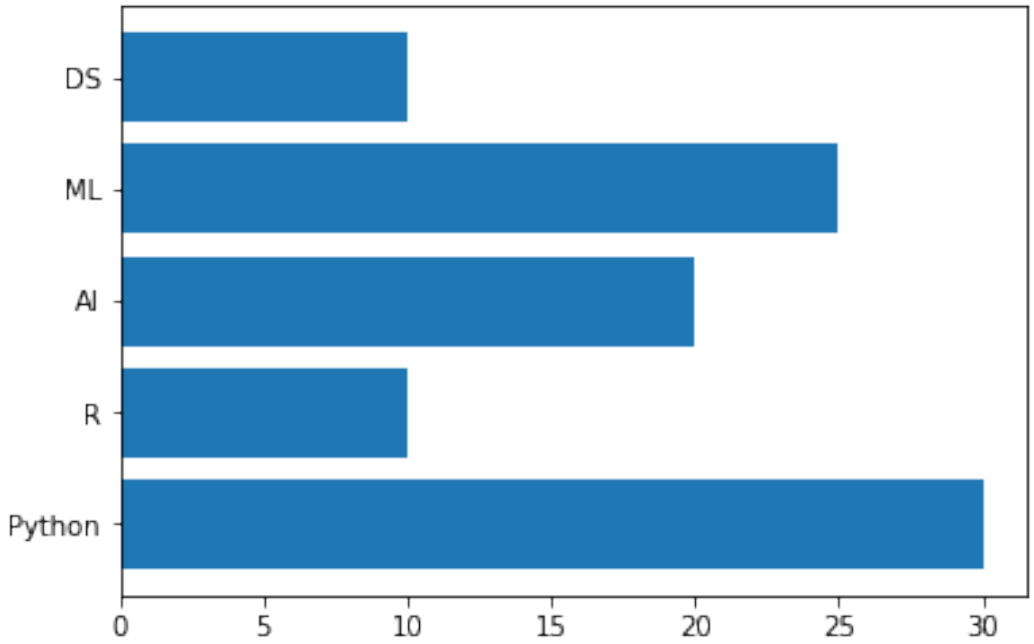
```
import matplotlib.pyplot as plt
import numpy as np
from matplotlib import style

# Dataset of 'Indian Artificial Intelligence Production (IAIP) Class"

#classes = ["Python", "R", "Artificial Intelligence", "Machine
Learning", "Data Science"]

classes = ["Python", "R", "AI", "ML", "DS"]
class1_students = [30, 10, 20, 25, 10] # out of 100 student in each
class
class2_students = [40, 5, 20, 20, 10]
class3_students = [35, 5, 30, 15, 15]

plt.bar(classes, class1_students)
```

```
<BarContainer object of 5 artists>
```




```
plt.barh(classes, class1_students)
```

```
<BarContainer object of 5 artists>
```




```
"""
plt.bar(
    x,
    height,
    width=0.8,
    bottom=None,
    *,
    align='center',
    data=None,
    **kwargs,
)

Parameters
----------
x : sequence of scalars
height : scalar or sequence of scalars
width : scalar or array-like, optional .......(default: 0.8).
bottom : scalar or array-like, optional
align : {'center', 'edge'}, optional, ..........default: 'center'


Other Parameters
----------------

color : scalar or array-like, optional
edgecolor : scalar or array-like, optional
linewidth : scalar or array-like, optional
tick_label : string or array-like, optional ....... name of Bar
xerr, yerr : scalar or array-like of shape(N,) or shape(2,N), optional
```

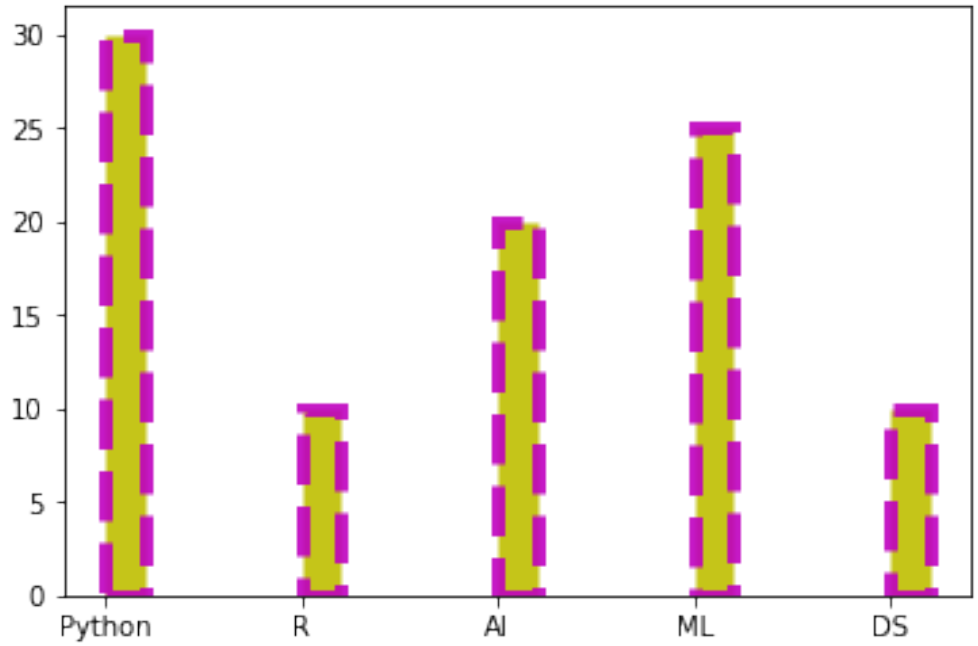
```
ecolor : scalar or array-like, optional, default: 'black'
capsize : scalar, optional
error_kw : dict, optional
log : bool, optional, default: False
orientation : {'vertical',  'horizontal'}, optional


See also
--------
barh: Plot a horizontal bar plot.

Other optional kwargs:

  agg_filter: a filter function, which takes a (m, n, 3) float array
and a dpi value, and returns a (m, n, 3) array
  ===== alpha: float or None
  animated: bool
  antialiased: unknown
  capstyle: {'butt', 'round', 'projecting'}
  clip_box: `.Bbox`
  clip_on: bool
  clip_path: [(`~matplotlib.path.Path`, `.Transform`) | `.Patch` |
None]
   ===== color: color
  contains: callable
   ===== edgecolor: color or None or 'auto'
   ===== facecolor: color or None
   ===== figure: `.Figure`
  fill: bool
  gid: str
  hatch: {'/', '\\', '|', '-', '+', 'x', 'o', 'O', '.', '*'}
  in_layout: bool
  joinstyle: {'miter', 'round', 'bevel'}
   ===== label: object
   ===== linestyle: {'-', '--', '-.', ':', '', (offset, on-off-
seq), ...}
   ===== linewidth: float or None for default
  path_effects: `.AbstractPathEffect`
  picker: None or bool or float or callable
  rasterized: bool or None
  sketch_params: (scale: float, length: float, randomness: float)
  snap: bool or None
  transform: `.Transform`
  url: str
   ===== visible: bool
  zorder: float
```

```
"""
```

```
"\nplt.bar(\n    x,\n    height,\n    width=0.8,\n    bottom=None,\n
*,\n    align='center',\n    data=None,\n    **kwargs,\n)\n\
nParameters\n----------\nx : sequence of scalars\nheight : scalar or
sequence of scalars\nwidth : scalar or array-like, optional .......
(default: 0.8).\nbottom : scalar or array-like, optional\nalign :
{'center', 'edge'}, optional, ..........default: 'center'\n\n\nOther
Parameters\n----------------\n\ncolor : scalar or array-like,
optional\nedgecolor : scalar or array-like, optional\nlinewidth :
scalar or array-like, optional\ntick_label : string or array-like,
optional ....... name of Bar\nxerr, yerr : scalar or array-like of
shape(N,) or shape(2,N), optional\necolor : scalar or array-like,
optional, default: 'black'\ncapsize : scalar, optional\nerror_kw :
dict, optional\nlog : bool, optional, default: False\norientation :
{'vertical',  'horizontal'}, optional\n\n\nSee also\n--------\nbarh:
Plot a horizontal bar plot.\n\nOther optional kwargs:\n\n  agg_filter:
a filter function, which takes a (m, n, 3) float array and a dpi
value, and returns a (m, n, 3) array \n  ===== alpha: float or None\n
animated: bool\n  antialiased: unknown\n  capstyle: {'butt', 'round',
'projecting'}\n  clip_box: `.Bbox`\n  clip_on: bool\n  clip_path:
[(`~matplotlib.path.Path`, `.Transform`) | `.Patch` | None] \n   =====
color: color\n  contains: callable\n   ===== edgecolor: color or None
or 'auto'\n   ===== facecolor: color or None\n   ===== figure:
`.Figure`\n  fill: bool\n  gid: str\n  hatch: {'/', '\\', '|', '-',
'+', 'x', 'o', 'O', '.', '*'}\n  in_layout: bool\n  joinstyle:
{'miter', 'round', 'bevel'}\n   ===== label: object\n   =====
linestyle: {'-', '--', '-.', ':', '', (offset, on-off-seq), ...}\n
===== linewidth: float or None for default \n  path_effects:
`.AbstractPathEffect`\n  picker: None or bool or float or callable\n
rasterized: bool or None\n  sketch_params: (scale: float, length:
float, randomness: float) \n  snap: bool or None\n  transform:
`.Transform`\n  url: str\n   ===== visible: bool\n  zorder: float\n\n\
n\n"
```

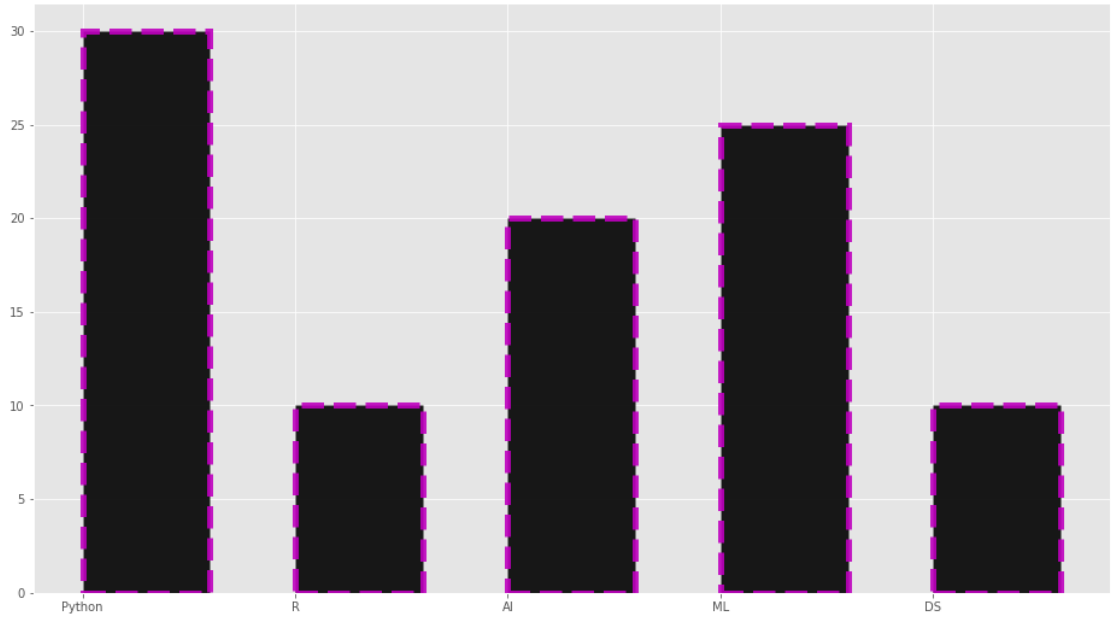
```
plt.bar(classes, class1_students, width = 0.2, align = "edge", color =
"y",
       edgecolor = "m", linewidth = 5, alpha = 0.9, linestyle = "--",
       label =" Class 1 Students") #visible=False
```

```
<BarContainer object of 5 artists>
```




```
style.use("ggplot")
plt.figure(figsize=(16,9))
plt.bar(classes, class1_students, width = 0.6, align = "edge", color =
"k",
       edgecolor = "m", linewidth = 5, alpha = 0.9, linestyle = "--",
       label =" Class 1 Students") #visible=False
```

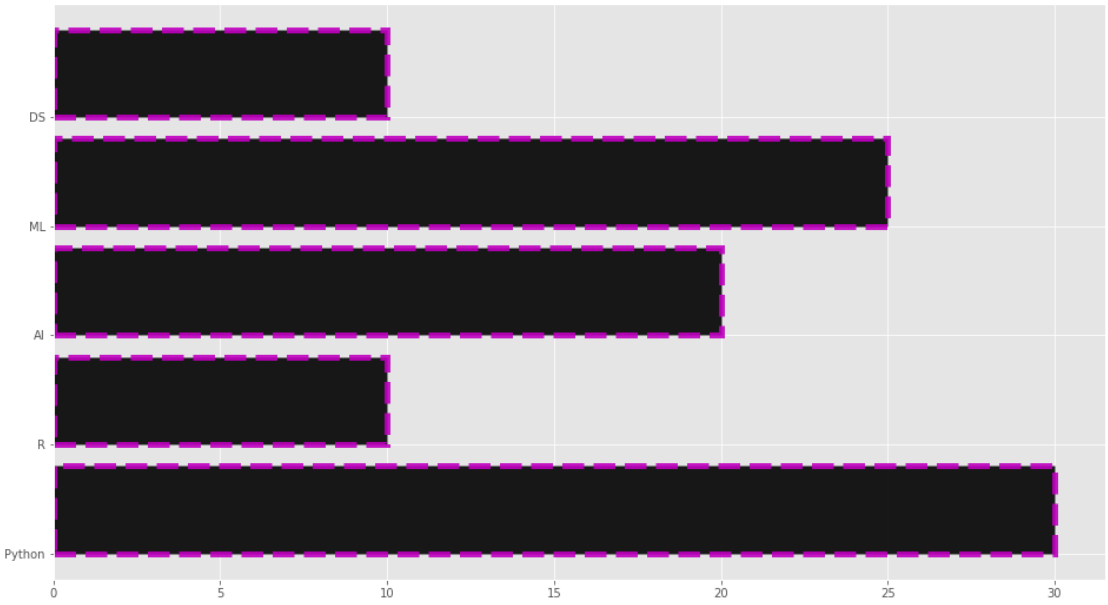
```
<BarContainer object of 5 artists>
```



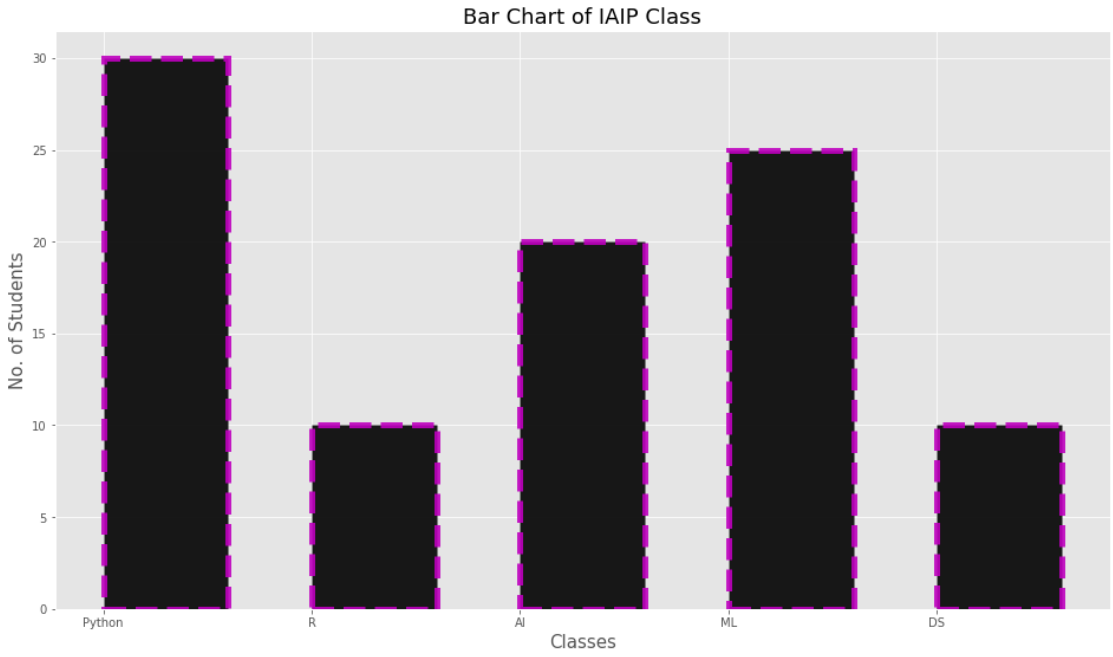

```
plt.figure(figsize=(16,9))
plt.barh(classes, class1_students,  align = "edge", color = "k",
```

```
        edgecolor = "m", linewidth = 5, alpha = 0.9, linestyle = "--",
        label =" Class 1 Students") #visible=False
```

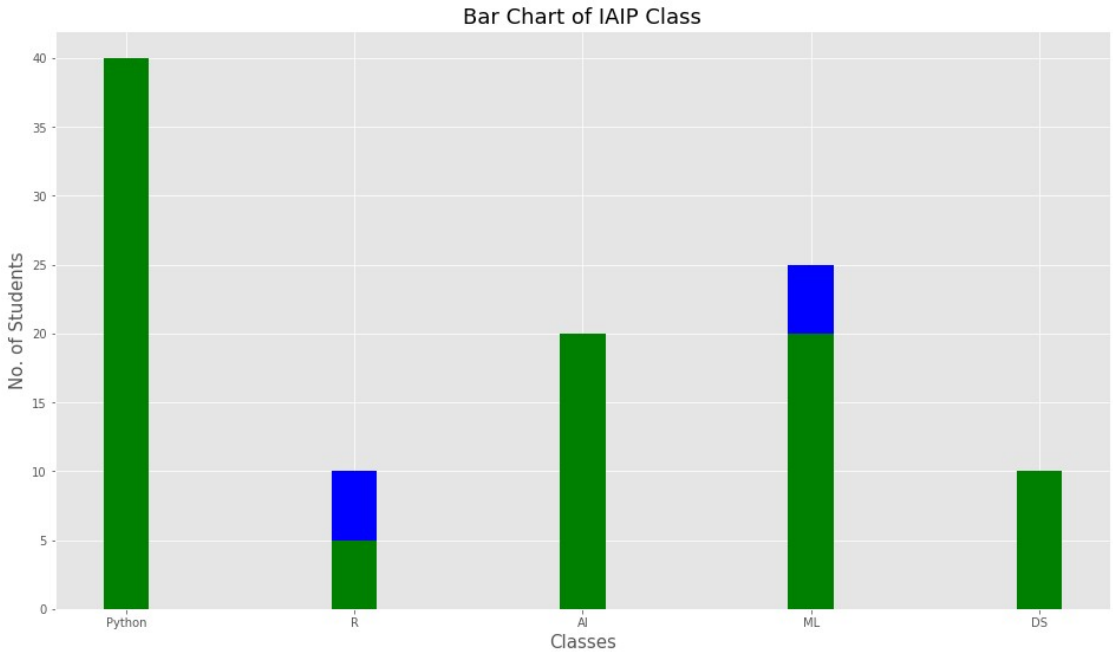
```
<BarContainer object of 5 artists>
```



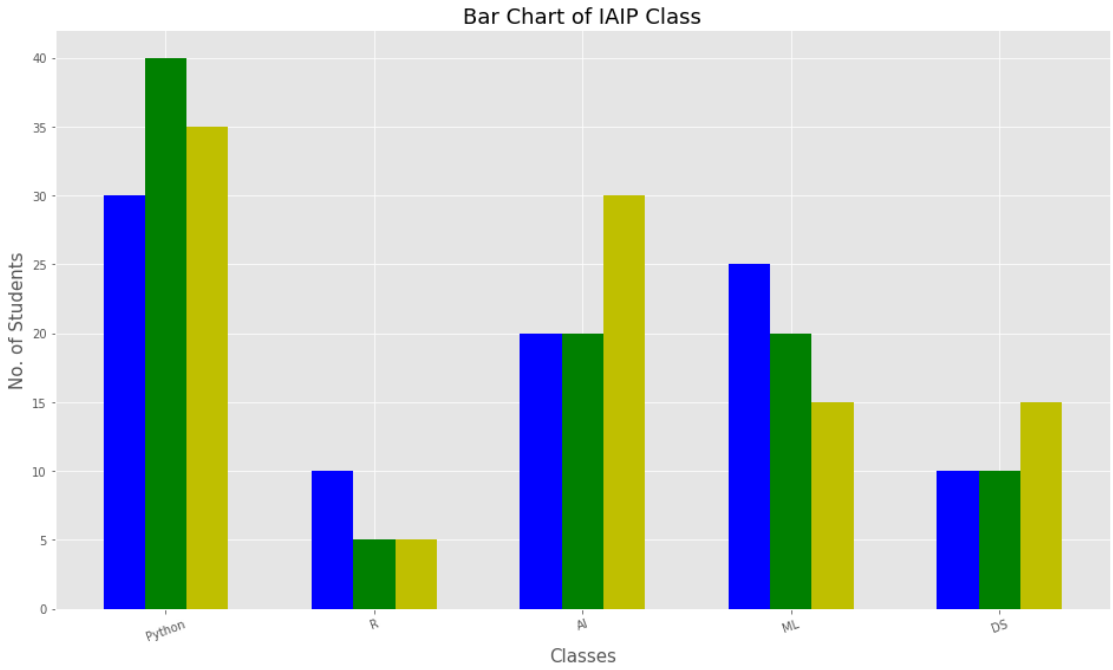

```
plt.figure(figsize=(16,9))
plt.bar(classes, class1_students, width = 0.6, align = "edge", color =
"k",
        edgecolor = "m", linewidth = 5, alpha = 0.9, linestyle = "--",
        label =" Class 1 Students") #visible=False

plt.title("Bar Chart of IAIP Class", fontsize = 18)
plt.xlabel("Classes",fontsize = 15)
plt.ylabel("No. of Students", fontsize = 15)
plt.show()
```



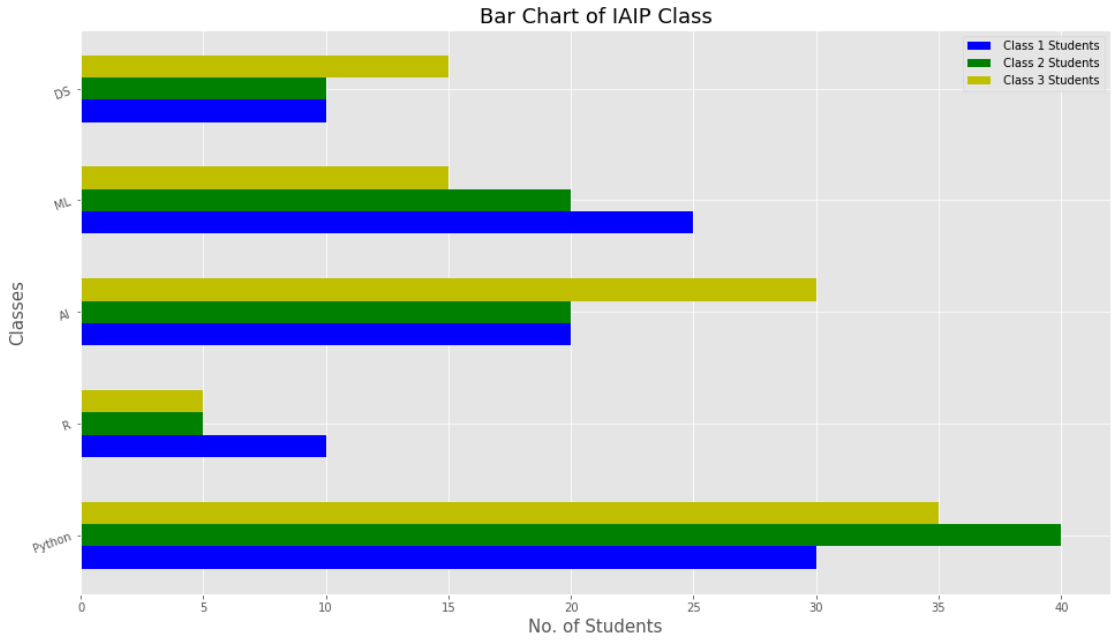

```
plt.figure(figsize=(16,9))

plt.bar(classes, class1_students, width = 0.2, color = "b",
        label =" Class 1 Students") #visible=False

plt.bar(classes, class2_students, width = 0.2, color = "g",
        label =" Class 2 Students")

plt.title("Bar Chart of IAIP Class", fontsize = 18)
plt.xlabel("Classes",fontsize = 15)
plt.ylabel("No. of Students", fontsize = 15)
plt.show()
```




```
plt.figure(figsize=(16,9))

classes_index = np.arange(len(classes))

width = 0.2

plt.bar(classes_index, class1_students, width , color = "b",
        label =" Class 1 Students") #visible=False

plt.bar(classes_index + width, class2_students, width , color = "g",
        label =" Class 2 Students")

plt.bar(classes_index + width + width, class3_students, width , color
= "y",
        label =" Class 2 Students")

plt.xticks(classes_index + width, classes, rotation = 20)
plt.title("Bar Chart of IAIP Class", fontsize = 18)
plt.xlabel("Classes",fontsize = 15)
plt.ylabel("No. of Students", fontsize = 15)
plt.show()
```




```
plt.figure(figsize=(16,9))

classes_index = np.arange(len(classes))

width = 0.2

plt.barh(classes_index, class1_students, width , color = "b",
        label =" Class 1 Students") #visible=False

plt.barh(classes_index + width, class2_students, width , color = "g",
        label =" Class 2 Students")

plt.barh(classes_index + width + width, class3_students, width , color
= "y",
        label =" Class 3 Students")

plt.yticks(classes_index + width, classes, rotation = 20)
plt.title("Bar Chart of IAIP Class", fontsize = 18)
plt.ylabel("Classes",fontsize = 15)
plt.xlabel("No. of Students", fontsize = 15)
plt.legend()
plt.show()
```




```
print("Thank you")
```

```
Thank you
```