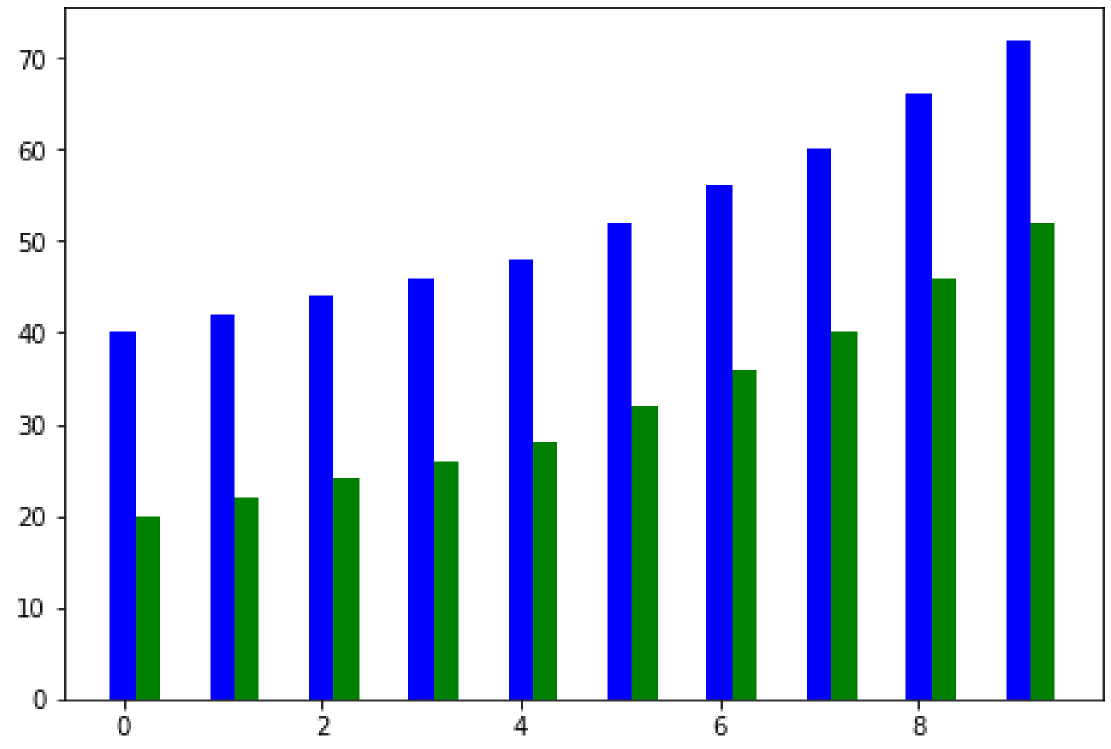
```
import random
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import statistics
import math

class Cpuschedule:

    def __init__(self):
        # The queue of process burst time
        self.data = []
        self.n = 0  # No of processes

        #avg
        self.Twt_m_ls = []
        self.Tat_m_ls = []

    # Getting the No of processes & burst time
    def getData(self):
        self.n = input("Enter the no of processes:\n")
        self.n = 10
        for i in range(int(self.n)):
            temp = input("Enter The BurstTime for Process p" + str(i)
+"\n")
            self.data.append(temp)

    def Rnd(self, n):

      ls = []

      for i in range(1, n+1):

        ls.append(i)

      random.shuffle(ls)

      return(ls[n-1])

    # Priority Algorithm
    def Priority(self):

        Twt = 0.0
        Bst = 0.0
        Tat = 0.0
        w=0.0
        B = []
        P = []
        self.n=4
        #self.n = int(input("Enter the no of processes:\n"))
        #self.n = 10
        #print("the no of processes:"+str(self.n))
        pMax=6
        Wt = [0]*self.n
        #b, p = input("Enter The Range of BurstTime and Priority for 
Process: " +"\n").split()
        b, p = 100, 50
        #print("The Range of BurstTime and Priority for 
Process:"+str(b)+","+str(p))
        b = int(b)
        p = int(p)

        for i in range(int(self.n)):

            if b > 1:
              b1 = self.Rnd(b)
              #print("Chosen BurstTime:"+str(b1))
              B.append(b1)

            else:
              B.append(b)

            if p > 1:
              p1 = self.Rnd(p)
              #print("Chosen Priority:"+str(p))
              P.append(p1)

            else:
              P.append(p)


            if pMax<int(p1):
                pMax=int(p1)
                p=0
            elif pMax<int(p):
                pMax=int(p)

        # print(B)
        # print(P)

        for j in range(pMax):
            for i in range(0,self.n):
                if P[i]==j:
                    Wt[i]=w
                    w=w+B[i]

        T = []
        for i in range(0, int(self.n)):
            Twt = Twt + Wt[i]
        for i in range(0, int(self.n)):
            Bst = Bst + Wt[i]
        for i in range(0, int(self.n)):
            Tat = Wt[i]+int(B[i])
            T.append(Tat)

        mean_Twt = Twt/int(self.n)
        mean_Tat = Tat/int(self.n)


        # print("Total Waiting Time:"+str(Twt))
        # print("Average Waiting Time:"+str(mean_Twt))
        self.Twt_m_ls.append(mean_Twt)
        # print("Total Turnaround Time:"+str(Tat))
        # print("Average Turnaround Time:"+str(mean_Tat))
        self.Tat_m_ls.append(mean_Tat)



class Runme:
    def __init__(self):
        self.schedular= Cpuschedule()

    def run(self):
        while True:
            print(" Menu:")
            print(" 1. Priority Algorithm")
            print(" 2. Quit")
            ch = int(input(" Select : "))

            if ch == 1:

              AWT = []
              SWT = []

              ATT = []
              STT = []

              for i in range(10):
                  #rep = int(input(" Repetition : "))
                  rep = 200
                  self.schedular.n = (input("Enter the no of
processes:\n"))
                  print("no of processes:"+str(self.schedular.n))
                  for i in range(rep):
                    self.schedular.Priority()
                  # print("Average Waiting
Time:"+str(self.schedular.Twt_m_ls))
                  # print("STD Waiting
Time:"+str(self.schedular.Twt_std_ls))
                  # print("Average Turnaround
Time:"+str(self.schedular.Tat_m_ls))
                  # print("STD Turnaround
Time:"+str(self.schedular.Tat_std_ls))

                  AWT.append(np.mean(self.schedular.Twt_m_ls))
                  SWT.append(np.std(self.schedular.Twt_m_ls))

                  ATT.append(np.mean(self.schedular.Tat_m_ls))
                  STT.append(np.std(self.schedular.Tat_m_ls))

                  # df = pd.DataFrame({'Average wating time': AWT})
                  # df.plot()
                  # plt.title("Wating Time")

                  # df = pd.DataFrame({'STD wating time': SWT})
                  # df.plot()
                  # plt.title("Wating Time")

                  # df = pd.DataFrame({'Average Turnaround time':
ATT})

                  # df.plot()
                  # plt.title("Turnaround Time")

                  # df = pd.DataFrame({'STD Turnaround time': STT})
                  # df.plot()
                  # plt.title("Turnaround Time")


              # data = [AWT, ATT]
              # X = np.arange(10)
              # fig = plt.figure()
              # ax = fig.add_axes([0,0,1,1])
              # ax.bar(X + 0.00, data[0], color = 'b', width = 0.25)
              # ax.bar(X + 0.25, data[1], color = 'g', width = 0.25)

            elif ch == 2:
                print("Bye-bye!")
                break
            else:
                print("1s and 2s only!")

if __name__ == "__main__":
    Runme().run()
```

```
 Menu:
 1. Priority Algorithm
 2. Quit
 Select : 1
Enter the no of processes:
1
no of processes:1
 Menu:
 1. Priority Algorithm
 2. Quit
 Select : 2
Bye-bye!
```


0
10
20
30
40
50
60
70
0
2
4
6
8