Code for Priority Scheduling :-

```
# Python3 implementation for Priority Scheduling with  different Arrival Time priority scheduling
"""1. sort the processes according to arrival time
   2. if arrival time is same the acc to priority
   3. apply fcfs """

totalprocess = 5
proc = []
for i in range(5):
    l = []
    for j in range(4):
        l.append(0)
    proc.append(l)

# Using FCFS Algorithm to find Waiting time
def get_wt_time( wt):

    # declaring service array that stores cumulative burst time
    service = [0] * 5

    # Initialising initial elements of the arrays
    service[0] = 0
    wt[0] = 0

    for i in range(1, totalprocess):
        service[i] = proc[i - 1][1] + service[i - 1]
        wt[i] = service[i] - proc[i][0] + 1

        # If waiting time is negative, change it to zero
        if(wt[i] < 0) :
            wt[i] = 0

def get_tat_time(tat, wt):

    # Filling turnaroundtime array
    for i in range(totalprocess):
        tat[i] = proc[i][1] + wt[i]

def findgc():

    # Declare waiting time and turnaround time array
    wt = [0] * 5
    tat = [0] * 5

    wavg = 0
    tavg = 0

    # Function call to find waiting time array
    get_wt_time(wt)
```

```
    # Function call to find turnaround time
    get_tat_time(tat, wt)

    stime = [0] * 5
    ctime = [0] * 5
    stime[0] = 1
    ctime[0] = stime[0] + tat[0]

    # calculating starting and ending time
    for i in range(1, totalprocess):
        stime[i] = ctime[i - 1]
        ctime[i] = stime[i] + tat[i] - wt[i]

    print("Process_no\tStart_time\tComplete_time","\tTurn_Around_Time\tWaiting_Time")

    # display the process details
    for i in range(totalprocess):
        wavg += wt[i]
        tavg += tat[i]

        print(proc[i][3], "\t\t", stime[i], "\t\t", end = " ")
        print(ctime[i], "\t\t", tat[i], "\t\t\t", wt[i])


    # display the average waiting time and average turn around time
    print("Average waiting time is : ", end = " ")
    print(wavg / totalprocess)
    print("average turnaround time : " , end = " ")
    print(tavg / totalprocess)

# Driver code
if __name__ =="__main__":
    arrivaltime = [1, 2, 3, 4, 5]
    bursttime = [3, 5, 1, 7, 4]
    priority = [3, 4, 1, 7, 8]

    for i in range(totalprocess):

        proc[i][0] = arrivaltime[i]
        proc[i][1] = bursttime[i]
        proc[i][2] = priority[i]
        proc[i][3] = i + 1

    # Using inbuilt sort function
    proc = sorted (proc, key = lambda x:x[2])
    proc = sorted (proc)

    # Calling function findgc for finding Gantt Chart
    findgc()
```