# Ex No. 5

# Date :

# IMPLEMENTATION OF FCFS CPU SCHEDULING ALGORITHM

# Problem Statement :

# Create a python program to implement FCFS CPU scheduling algorithm.

# Problem Description:

# FCFS is one of the simplest scheduling algorithms used by operating systems to manage the execution of processes. The primary goal is to develop a Python program that simulates the FCFS algorithm to manage a queue of processes and allocate CPU time to each process in the order they arrive.

# Algorithm:

# Create a list or data structure to hold information about each process. Each entry should include:

# Process ID (an identifier for the process).

# Arrival Time (the time at which the process arrives).

# Burst Time (the time required to complete the process).

# Sort Processes by Arrival Time

# Process Execution Loop:

# Iterate through the sorted list of processes.

# For each process:

# If the process has not yet arrived (its arrival time is greater than the current time), wait until it arrives.

# Update the current time to be the maximum of the process's arrival time and the current time (ensuring the current time moves forward).

# Execute the process for its burst time.

# Calculate the turnaround time for the process (turnaround time = completion time - arrival time).

# Calculate the waiting time for the process (waiting time = turnaround time - burst time).

# Add the waiting time to the total waiting time.

# Calculate and Display Metrics:

# Calculate the average waiting time (average waiting time = total waiting time / number of processes).

# Display the turnaround time, waiting time, and average waiting time for each process.

# CODE:

n=int(input("Enter the no.of proceses:"))

at=[]

bt=[]

pid=[]

for i in range(n):

  at.append(int(input(f"Enter the arrival time of processor {i+1}: ")))

  bt.append(int(input(f"Enter the burst time of processor {i+1}: ")))

  pid.append(f"P{i+1}")

print()

print("PID  AT  BT")

for i in range(n):

   print(f"P{i+1}   ", at[i], " ",bt[i])

d={}

for j in range(n):

  d[f"P{j+1}"]=[at[j],bt[j]]

print()

overhead=int(input("Enter the no.of overhead unit: "))

print()

d = sorted(d.items(), key=lambda item: item[1][0])

CT=[]

idle=0

st=""

for i in range(len(d)):

    if(i==0):

       v=d[i][1][1]

       CT.append(v)

       st+=("|"+"\_"\*v+str(d[i][0])+"|")

    elif CT[i-1]<d[i][1][0]:

       v1=CT[i-1] + d[i][1][1]

       idle+=((d[i][1][0]-CT[i-1])+overhead)

       CT.append(idle+ v1)

       st+=("\*"\*idle+"|")

       st+=("\_"\*(d[i][1][1])+str(d[i][0])+"|")

    else:

       v2=(CT[i-1] + d[i][1][1])

       CT.append(v2)

       st+=("\*"\*overhead+"|")

       st+=("\_"\*(d[i][1][1])+str(d[i][0])+"|")

TT = []

for i in range(len(d)):

    TT.append(CT[i] - d[i][1][0])

WT = []

for i in range(len(d)):

    WT.append(TT[i] - d[i][1][1])

AWT = 0

for i in WT:

    AWT +=i

AWT = (AWT/n)

ATT = 0

for i in TT:

    ATT +=i

ATT = (ATT/n)

print("GANTT CHART"+"\n")

print(st+"\n")

print("PID    AT        BT      CT       TT          WT   ")

print("---------------------------------------------------")

for p in pid:

 for i in range(len(d)):

   if p==d[i][0]:

      print(d[i][0],"      ",d[i][1][0],"     ",d[i][1][1],"     ",CT[i],"      ",TT[i],"        ",WT[i],"     ")

print("Average Waiting Time: ",AWT)

print("Average Turnaround Time: ",ATT)

# Output:

A screenshot of a computer

Description automatically generated

A black and white screen with numbers and text

Description automatically generated

A screenshot of a computer program

Description automatically generated

A black screen with white text

Description automatically generated

**Result :**

The FCFS (First-Come-First-Serve) CPU scheduling algorithm has been successfully executed for the provided set of processes. The turnaround times and waiting times for each process have been calculated, and the average waiting time has been determined.