# Continuous Assessment – 1

## Aromal S Kunnel – 22BAI10288

FCFS

CODE :

def fcfs(n):

  processes = []

  for i in range(n):

    process = {"pid": i + 1, "bt": int(input(f"Enter burst time for process {i + 1}: "))}

    processes.append(process)

  processes[0]["wt"] = 0

  processes[0]["tt"] = processes[0]["bt"] + processes[0]["wt"]

  for i in range(1, n):

    processes[i]["wt"] = processes[i - 1]["tt"]

    processes[i]["tt"] = processes[i]["bt"] + processes[i]["wt"]

  total\_wt = sum(process["wt"] for process in processes)

  total\_tt = sum(process["tt"] for process in processes)

  avg\_wt = total\_wt / n

  avg\_tt = total\_tt / n

  print("\nProcess ID\tBurst Time\tWaiting Time\tTurnaround Time")

  for process in processes:

    print(f"{process['pid']}\t\t{process['bt']}\t\t{process['wt']}\t\t{process['tt']}")

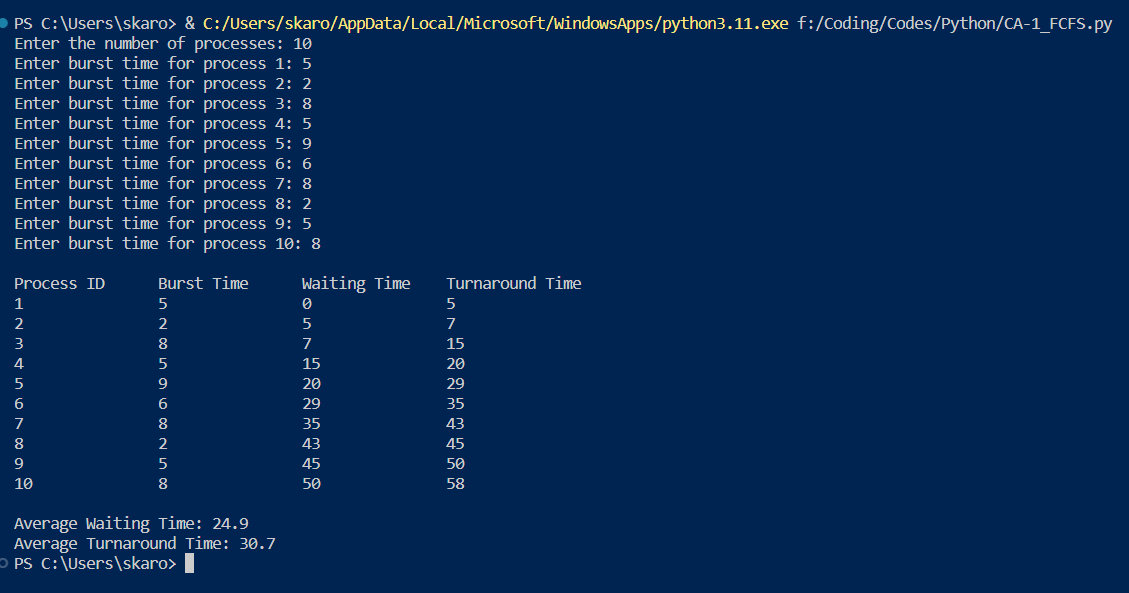
  print(f"\nAverage Waiting Time: {avg\_wt}")

  print(f"Average Turnaround Time: {avg\_tt}")

n = int(input("Enter the number of processes: "))

fcfs(n)

OUTPUT :



SJF

CODE :

def sjf(processes):

  processes.sort(key=lambda process: process["bt"])

  for i in range(len(processes)):

    if i == 0:

      processes[i]["wt"] = 0

    else:

      processes[i]["wt"] = processes[i - 1]["tt"]

    processes[i]["tt"] = processes[i]["bt"] + processes[i]["wt"]

  total\_wt = sum(process["wt"] for process in processes)

  total\_tt = sum(process["tt"] for process in processes)

  avg\_wt = total\_wt / len(processes)

  avg\_tt = total\_tt / len(processes)

  print("\nProcess ID\tBurst Time\tWaiting Time\tTurnaround Time")

  for process in processes:

    print(f"{process['pid']}\t\t{process['bt']}\t\t{process['wt']}\t\t{process['tt']}")

  print(f"\nAverage Waiting Time: {avg\_wt}")

  print(f"Average Turnaround Time: {avg\_tt}")

processes = [

    {"pid": 1, "bt": 6},

    {"pid": 2, "bt": 3},

    {"pid": 3, "bt": 8},

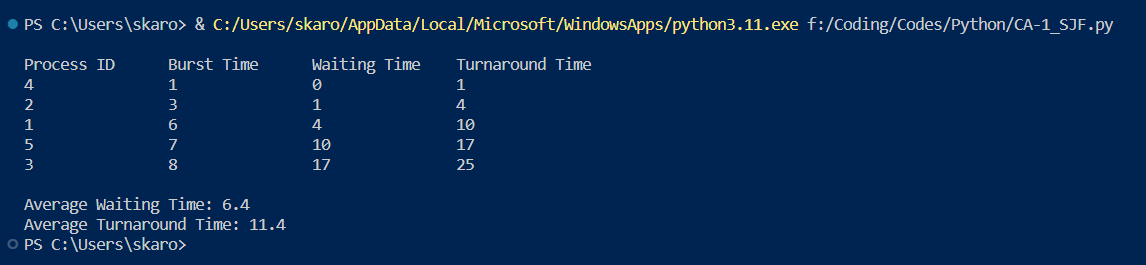
    {"pid": 4, "bt": 1},

    {"pid": 5, "bt": 7},

]

sjf(processes)

OUTPUT :



PRIORITY SCHEDULING

CODE :

def priority(processes):

  processes.sort(key=lambda process: process["prior"])

  for i in range(len(processes)):

    if i == 0:

      processes[i]["wt"] = 0

    else:

      processes[i]["wt"] = processes[i - 1]["tt"]

    processes[i]["tt"] = processes[i]["bt"] + processes[i]["wt"]

  total\_wt = sum(process["wt"] for process in processes)

  total\_tt = sum(process["tt"] for process in processes)

  avg\_wt = total\_wt / len(processes)

  avg\_tt = total\_tt / len(processes)

  print("\nProcess ID\tBurst Time\tPriority\tWaiting Time\tTurnaround Time")

  for process in processes:

    print(f"{process['pid']}\t\t{process['bt']}\t\t{process['prior']}\t\t{process['wt']}\t\t{process['tt']}")

  print(f"\nAverage Waiting Time: {avg\_wt}")

  print(f"Average Turnaround Time: {avg\_tt}")

processes = [

    {"pid": 1, "bt": 6, "prior": 3},

    {"pid": 2, "bt": 3, "prior": 5},

    {"pid": 3, "bt": 8, "prior": 2},

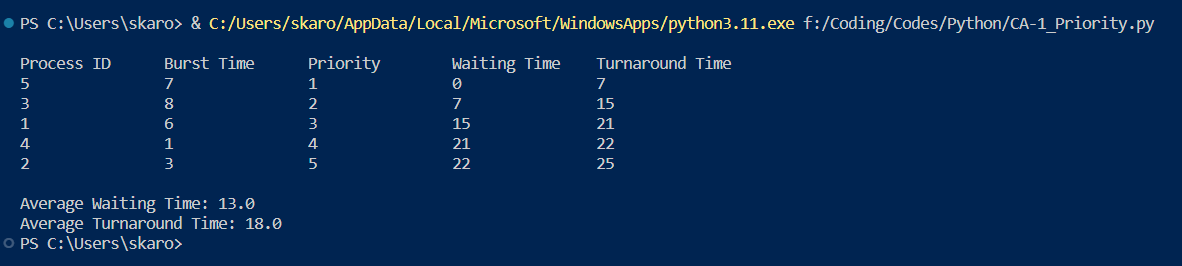
    {"pid": 4, "bt": 1, "prior": 4},

    {"pid": 5, "bt": 7, "prior": 1},

]

priority(processes)

OUTPUT :



RR

CODE :

def round\_robin(processes, time\_quantum):

  n = len(processes)

  completed = [False] \* n

  waiting\_time = [0] \* n

  turnaround\_time = [0] \* n

  current\_time = 0

  max\_burst\_time = max(process["bt"] for process in processes)

  while not all(completed):

    for i in range(n):

      if not completed[i] and processes[i]["bt"] > 0:

        executed = min(processes[i]["bt"], time\_quantum)

        # Update process data

        processes[i]["bt"] -= executed

        current\_time += executed

        if processes[i]["bt"] == 0:

          completed[i] = True

          turnaround\_time[i] = current\_time

          waiting\_time[i] = turnaround\_time[i] - processes[i]["rt"]

      if processes[i]["rt"] <= current\_time and not completed[i]:

        processes[i]["rt"] = current\_time

  average\_waiting\_time = sum(waiting\_time) / n

  average\_turnaround\_time = sum(turnaround\_time) / n

  print("\nProcess\tBurst Time\tWaiting Time\tTurnaround Time")

  for i in range(n):

    print(f"{i + 1}\t\t{processes[i]['bt']}\t\t{waiting\_time[i]}\t\t{turnaround\_time[i]}")

  print(f"\nAverage Waiting Time: {average\_waiting\_time}")

  print(f"Average Turnaround Time: {average\_turnaround\_time}")

processes = [

    {"bt": 6, "rt": 2},

    {"bt": 3, "rt": 0},

    {"bt": 8, "rt": 1},

    {"bt": 1, "rt": 0},

    {"bt": 7, "rt": 2},

]

time\_quantum = 3

round\_robin(processes, time\_quantum)

OUTPUT :

