**OS lab assignment 5**

**Agnim Gupta**

**2028082**

**A-23 CSSE**

**Question 1**

#include<stdio.h>

void findWaitingTime(int processes[], int n,int bt[], int wt[])

{

    wt[0] = 0;

    for (int i = 1; i < n ; i++ )

        wt[i] = bt[i-1] + wt[i-1] ;

}

void findTurnAroundTime( int processes[], int n,int bt[], int wt[], int tat[])

{

    for (int i = 0; i < n ; i++)

        tat[i] = bt[i] + wt[i];

}

void findavgTime( int processes[], int n, int bt[])

{

    int wt[n], tat[n], total\_wt = 0, total\_tat = 0;

    findWaitingTime(processes, n, bt, wt);

    findTurnAroundTime(processes, n, bt, wt, tat);

    //Display processes along with all details

    printf("Processes: ");

    for (int i=0; i<n; i++)

    {

        printf(" %d", (i+1));

    }

    printf("\nBurst Time: ");

    for (int i=0; i<n; i++)

    {

        printf(" %d", bt[i]);

    }

    printf("\nWaiting Time: ");

    for (int i=0; i<n; i++)

    {

        printf(" %d", wt[i]);

        total\_wt+=wt[i];

    }

    printf("\nTurn Around Time: ");

    for (int i=0; i<n; i++)

    {

        printf(" %d", tat[i]);

        total\_tat+=tat[i];

    }

    int s=(float)total\_wt / (float)n;

    int t=(float)total\_tat / (float)n;

    printf("\nAverage waiting time = %d",s);

    printf("\n");

    printf("Average turn around time = %d ",t);

}

int main()

{

    int num,val;

    printf("enter number of processes:");

    scanf("%d",&num);

    int processes[num];

    printf("enter processes:");

    for (int i=0; i<num; i++)

    {

        scanf("%d",&val);

        processes[i]=val;

    }

    int n = sizeof processes / sizeof processes[0];

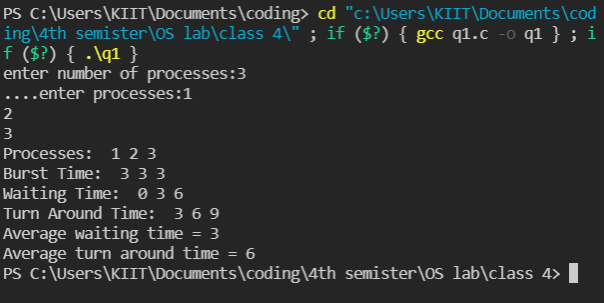
    int burst\_time[] = {};

    findavgTime(processes, n, burst\_time);

    return 0;

}

**Output**

****

**Question 2**

// C++ program to implement Shortest Job first with Arrival

// Time

#include <iostream>

using namespace std;

int mat[10][6];

void swap(int\* a, int\* b)

{

    int temp = \*a;

    \*a = \*b;

    \*b = temp;

}

void arrangeArrival(int num, int mat[][6])

{

    for (int i = 0; i < num; i++) {

        for (int j = 0; j < num - i - 1; j++) {

            if (mat[j][1] > mat[j + 1][1]) {

                for (int k = 0; k < 5; k++) {

                    swap(mat[j][k], mat[j + 1][k]);

                }

            }

        }

    }

}

void completionTime(int num, int mat[][6])

{

    int temp, val;

    mat[0][3] = mat[0][1] + mat[0][2];

    mat[0][5] = mat[0][3] - mat[0][1];

    mat[0][4] = mat[0][5] - mat[0][2];

    for (int i = 1; i < num; i++) {

        temp = mat[i - 1][3];

        int low = mat[i][2];

        for (int j = i; j < num; j++) {

            if (temp >= mat[j][1] && low >= mat[j][2]) {

                low = mat[j][2];

                val = j;

            }

        }

        mat[val][3] = temp + mat[val][2];

        mat[val][5] = mat[val][3] - mat[val][1];

        mat[val][4] = mat[val][5] - mat[val][2];

        for (int k = 0; k < 6; k++) {

            swap(mat[val][k], mat[i][k]);

        }

    }

}

int main()

{

    int num, temp;

    cout << "Enter number of Process: ";

    cin >> num;

    cout << "...Enter the process ID...\n";

    for (int i = 0; i < num; i++) {

        cout << "...Process " << i + 1 << "...\n";

        cout << "Enter Process Id: ";

        cin >> mat[i][0];

        cout << "Enter Arrival Time: ";

        cin >> mat[i][1];

        cout << "Enter Burst Time: ";

        cin >> mat[i][2];

    }

    cout << "Before Arrange...\n";

    cout << "Process ID\tArrival Time\tBurst Time\n";

    for (int i = 0; i < num; i++) {

        cout << mat[i][0] << "\t\t" << mat[i][1] << "\t\t"

            << mat[i][2] << "\n";

    }

    arrangeArrival(num, mat);

    completionTime(num, mat);

    cout << "Final Result...\n";

    cout << "Process ID\tArrival Time\tBurst Time\tWaiting "

            "Time\tTurnaround Time\n";

    for (int i = 0; i < num; i++) {

        cout << mat[i][0] << "\t\t" << mat[i][1] << "\t\t"

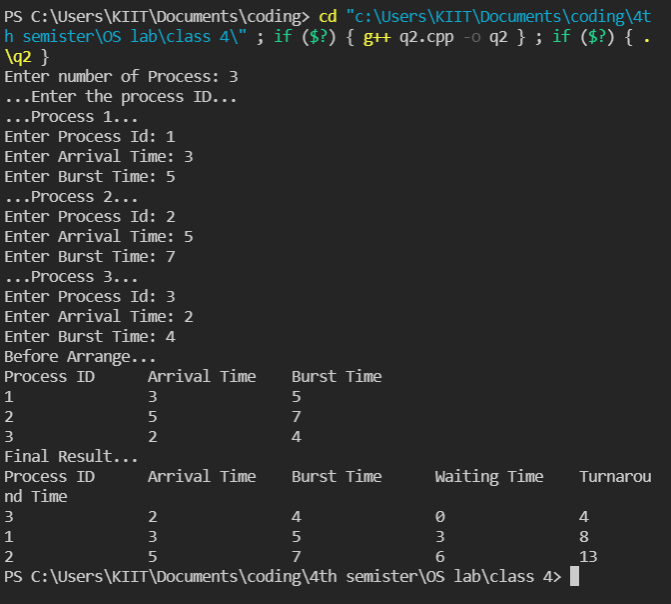
            << mat[i][2] << "\t\t" << mat[i][4] << "\t\t"

            << mat[i][5] << "\n";

    }

}

**Output**

****