COLLEGE OF COMPUTING AND INFORMATION SCIENCES

Task # 04

Class Id	110084	**Course Title**	Operating System LAB
Student Id	**64091**	**Student Name**	**Hassaan Raheem**
Total Marks	05	**Obtained Marks**	

QUESTION:

Write a C program for SJF algorithm (Non-preemptive).

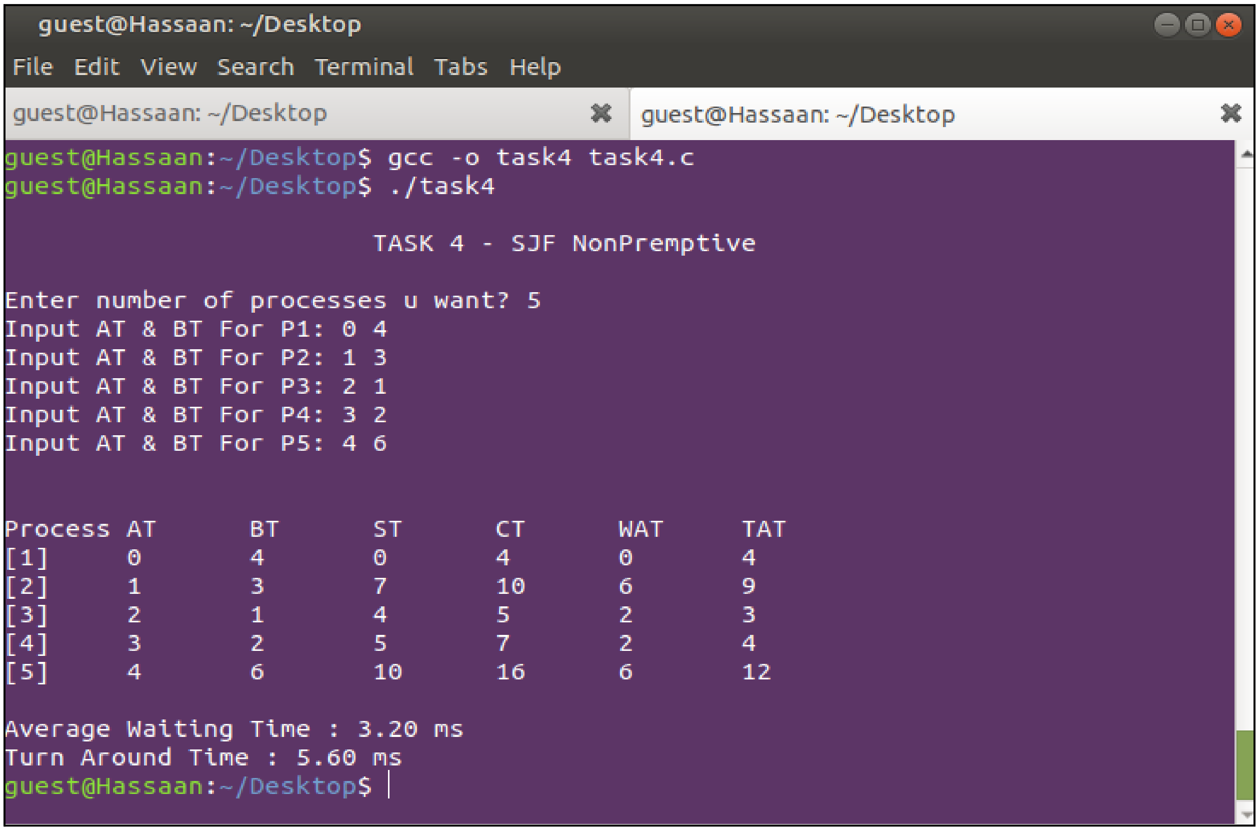
You have to take process name, arrival time and burst time as input from user. On the basis of given input calculate Starting time, completion time, waiting time, turnaround time, average waiting time and average turnaround time.

CODE:

```
#include <stdio.h>

#include <math.h>


int main(){


      printf("\n\t\t\tTASK 4 - SJF NonPremptive\n\n");

// input no. of processes

      int nop;

      printf("Enter number of processes u want? ");

      scanf("%i",&nop);


// declare array of process , arrival & burst time with the size of nop &
declaring t array & temp variable that stores copy of bursttime & index of
minimum value from arrival time.

      int at[nop] , bt[nop] , t[nop] , temp = 2147483647 ,index;


// 2147483647 --> it is last largest value of int type.
// getting input for arrival & burst time for each process.
    for(int i=0; i<nop; i++){
        printf("Input AT & BT For P%i: ",i+1);
        scanf("%i %i",&at[i],&bt[i]);
        // storing the copy of bursttime in another array
        t[i] = bt[i];
        // calculating the index minimum of arrival time
        if (temp > at[i]){
            temp = at[i];
            index = i;
            t[index] = 2147483647;
        }
    }

// declare 2d array to store staring & ending point of each process.
    int process[nop][6] , starting = bt[index];

    process[index][0] = at[index];
    process[index][1] = bt[index];
    process[index][2] = at[index];
    process[index][3] = bt[index];
    process[index][4] = process[index][0] - at[index];
    process[index][5] = process[index][1] - at[index];

    printf("\n");
// implementing the logic of SJF(non-preemptive) of each individual processes.
    int swap , sumwt = 0 , sumtat = 0;
    for(int i=0;i<nop-1;i++){
        //sorting the array t[nop]
        for(int j=i+1 ; j<nop;j++)
        {
            if (t[i] > t[j]){
                swap = t[i];
                t[i] = t[j];
                t[j] = swap;
            }
        }

        //comparing each sorted value with the array of burst time to return it index.
        if (t[i] != 2147483647){
            for(int j=0; j<nop; j++){
                if (t[i] == bt[j]){
                    process[j][0] = at[j];
                    process[j][1] = bt[j];
                    process[j][2] = starting;
                    process[j][3] = starting + bt[j];
                    process[j][4] = process[j][2] - at[j];
                    process[j][5] = process[j][3]-at[j];
                    sumwt += process[j][4];
                    sumtat += process[j][5];
                    starting = process[j][3];
                    t[i] = 2147483647;
                    break;
                }
            }

        }
    }



    //printing the table
    printf("\n");
    printf("Process\tAT\tBT\tST\tCT\tWAT\tTAT\n");
```

```c
        for(int i=0 ; i<nop; i++){
                printf("[%i]",i+1);
                for(int j=0; j<6;j++){
                        printf("\t%i",process[i][j]);
                }
                printf("\n");
        }


        printf("\nAverage Waiting Time : %0.2f ms\n",(sumwt/(float)nop));
        printf("Turn Around Time : %0.2f ms\n",(sumtat/(float)nop) );
return 0;
}
```

OUTPUT:



```
guest@Hassaan:~/Desktop$ gcc -o task4 task4.c
guest@Hassaan:~/Desktop$ ./task4

                        TASK 4 - SJF NonPremptive

Enter number of processes u want? 5
Input AT & BT For P1: 0 4
Input AT & BT For P2: 1 3
Input AT & BT For P3: 2 1
Input AT & BT For P4: 3 2
Input AT & BT For P5: 4 6


Process AT      BT      ST      CT      WAT     TAT
[1]     0       4       0       4       0       4
[2]     1       3       7       10      6       9
[3]     2       1       4       5       2       3
[4]     3       2       5       7       2       4
[5]     4       6       10      16      6       12

Average Waiting Time : 3.20 ms
Turn Around Time : 5.60 ms
guest@Hassaan:~/Desktop$ 
```