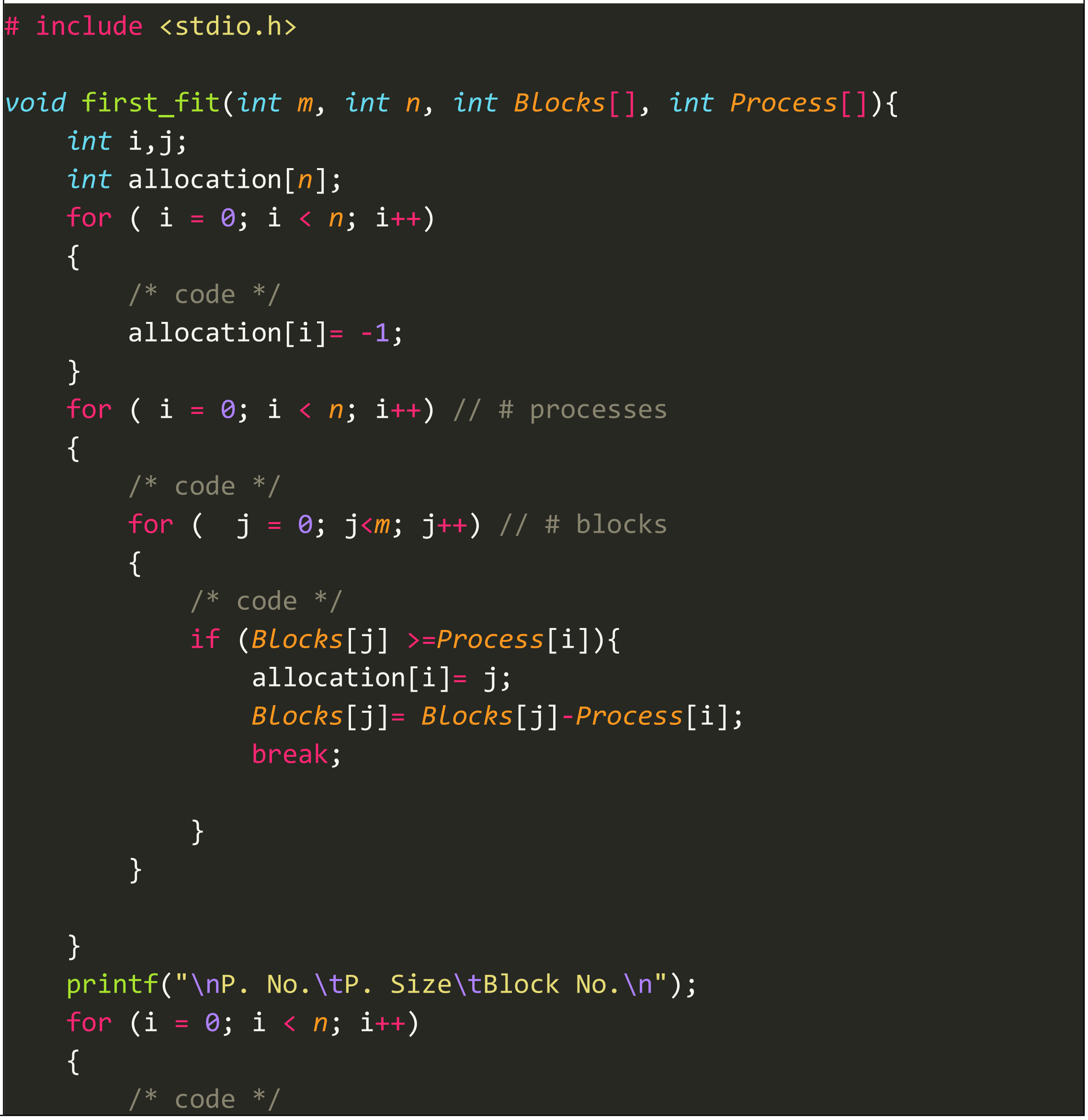
INTERNAL ASSIGNMENT

Course Code: OMC-109 **Last Date of Submission: 15/01/24**

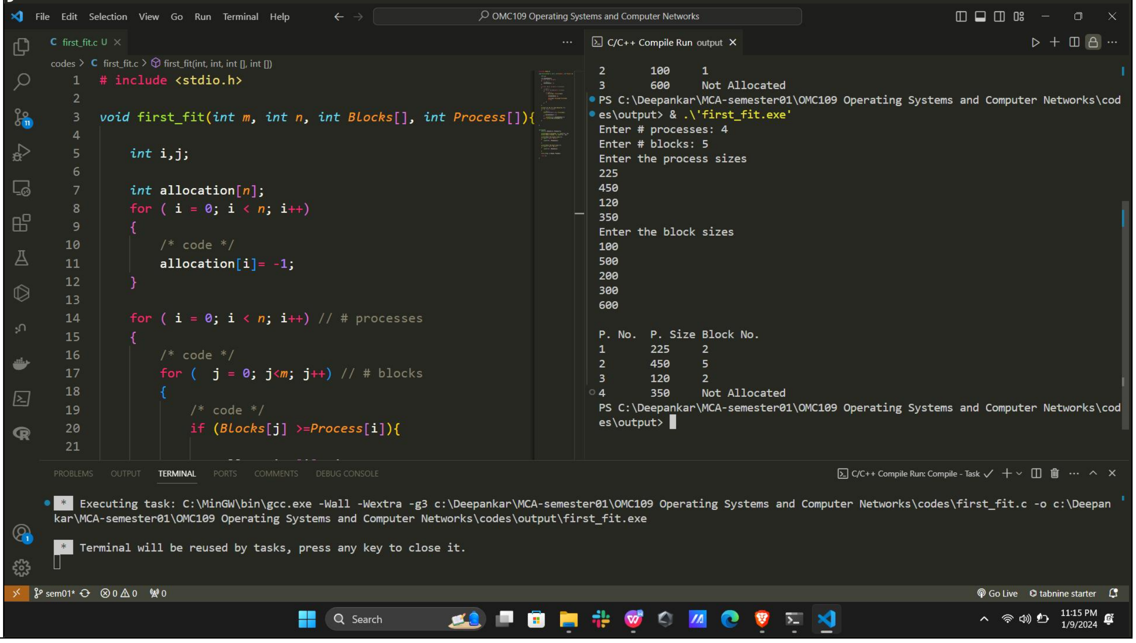
Course Title: Operating Systems & Computer Networks lab **Maximum Marks: 30**

Session: July 2023

Please submit answers to any four programs, accompanied by output snapshots. Ensure that at least two sets of outputs are provided for two different inputs. (6*4=24)

Q.No.	Question
1	Write a C program to Simulate the following Memory management algorithm-First fit 

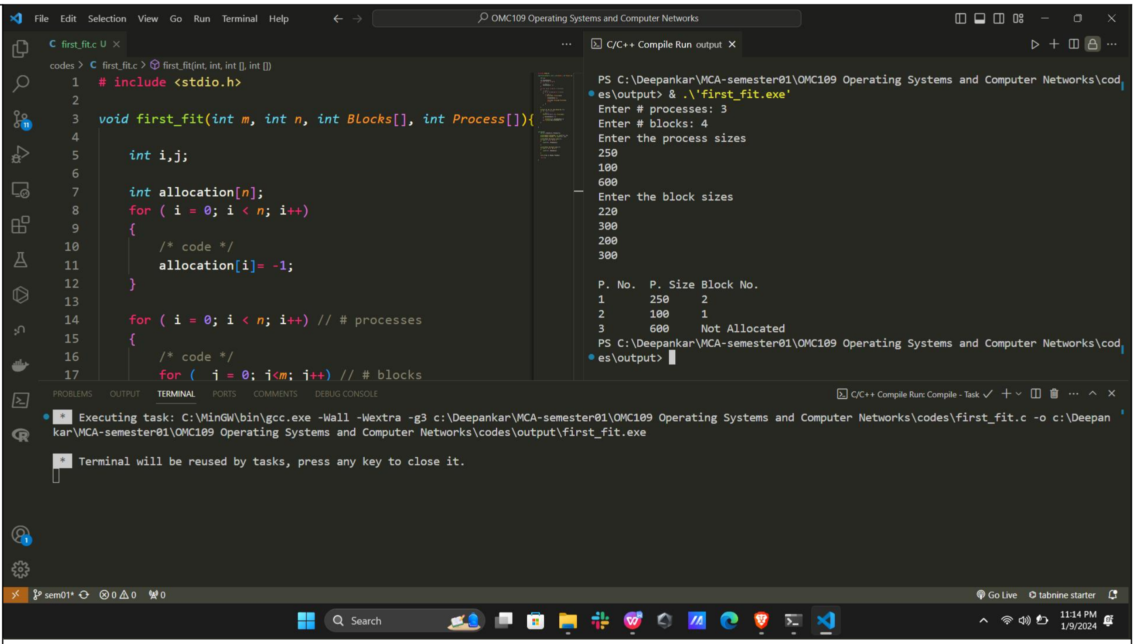
```c
# include <stdio.h>

void first_fit(int m, int n, int Blocks[], int Process[]){
    int i,j;
    int allocation[n];
    for ( i = 0; i < n; i++)
    {
        /* code */
        allocation[i]= -1;
    }
    for ( i = 0; i < n; i++) // # processes
    {
        /* code */
        for (  j = 0; j<m; j++) // # blocks
        {
            /* code */
            if (Blocks[j] >=Process[i]){
                allocation[i]= j;
                Blocks[j]= Blocks[j]-Process[i];
                break;

            }
        }

    }
    printf("\nP. No.\tP. Size\tBlock No.\n");
    for (i = 0; i < n; i++)
    {
        /* code */
        printf("%d\t%d\t", i+1, Process[i]);
        if (allocation[i]!=-1)
        {
            printf("%i\n", allocation[i]+1);
        }else printf("Not Allocated\n");

    }



}
int main(){
    int m, n, Blocks[10], Process[10];
    printf("Enter # processes: "); scanf("%d", &n);
    printf("Enter # blocks: "); scanf("%d", &m);
    printf("Enter the process sizes\n");
    for (int i = 0; i < n; i++)
    {
        scanf("%d", &Process[i]);
    }
    printf("Enter the block sizes\n");
    for (int i = 0; i < m; i++)
    {
        scanf("%d", &Blocks[i]);
    }
    first_fit(m, n, Blocks, Process);
    return 0;

}
```


File Edit Selection View Go Run Terminal Help
OMC109 Operating Systems and Computer Networks
first_fit.c U
codes > first_fit.c > first_fit(int, int, int [], int [])
C/C++ Compile Run output
PROBLEMS OUTPUT TERMINAL PORTS COMMENTS DEBUG CONSOLE
C/C++ Compile Run: Compile - Task
sem01* 0 0 0
Go Live tabnine starter
Search
11:15 PM
1/9/2024

```
# include <stdio.h>

void first_fit(int m, int n, int Blocks[], int Process[]){

    int i,j;

    int allocation[n];
    for ( i = 0; i < n; i++)
    {
        /* code */
        allocation[i]= -1;
    }

    for ( i = 0; i < n; i++) // # processes
    {
        /* code */
        for (  j = 0; j<m; j++) // # blocks
        {
            /* code */
            if (Blocks[j] >=Process[i]){

```

```
2       100     1
3       600     Not Allocated
PS C:\Deepankar\MCA-semester01\OMC109 Operating Systems and Computer Networks\cod
es\output> & .\'first_fit.exe'
Enter # processes: 4
Enter # blocks: 5
Enter the process sizes
225
450
120
350
Enter the block sizes
100
500
200
300
600

P. No.  P. Size Block No.
1       225     2
2       450     5
3       120     2
4       350     Not Allocated
PS C:\Deepankar\MCA-semester01\OMC109 Operating Systems and Computer Networks\cod
es\output> 
```

```
 *  Executing task: C:\MinGW\bin\gcc.exe -Wall -Wextra -g3 c:\Deepankar\MCA-semester01\OMC109 Operating Systems and Computer Networks\codes\first_fit.c -o c:\Deepan
kar\MCA-semester01\OMC109 Operating Systems and Computer Networks\codes\output\first_fit.exe

 *  Terminal will be reused by tasks, press any key to close it.
```



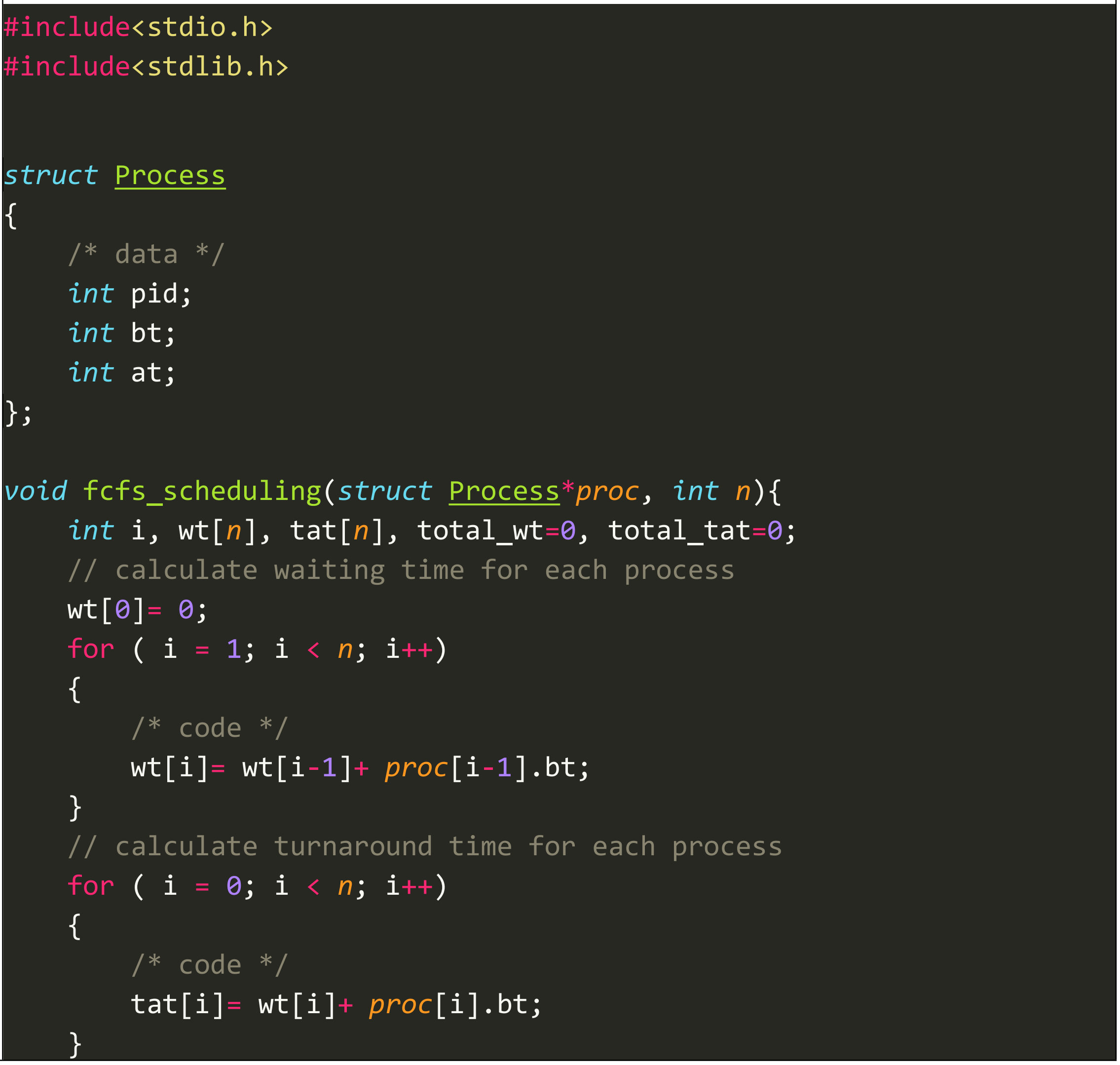
File Edit Selection View Go Run Terminal Help
OMC109 Operating Systems and Computer Networks
first_fit.c U
codes > first_fit.c > first_fit(int, int, int [], int [])
C/C++ Compile Run output
PROBLEMS OUTPUT TERMINAL PORTS COMMENTS DEBUG CONSOLE
C/C++ Compile Run: Compile - Task
sem01* 0 0 0
Go Live tabnine starter
Search
11:14 PM
1/9/2024

```c
# include <stdio.h>

void first_fit(int m, int n, int Blocks[], int Process[]){

    int i,j;

    int allocation[n];
    for ( i = 0; i < n; i++)
    {
        /* code */
        allocation[i]= -1;
    }

    for ( i = 0; i < n; i++) // # processes
    {
        /* code */
        for (  i = 0; i<m; i++) // # blocks
```

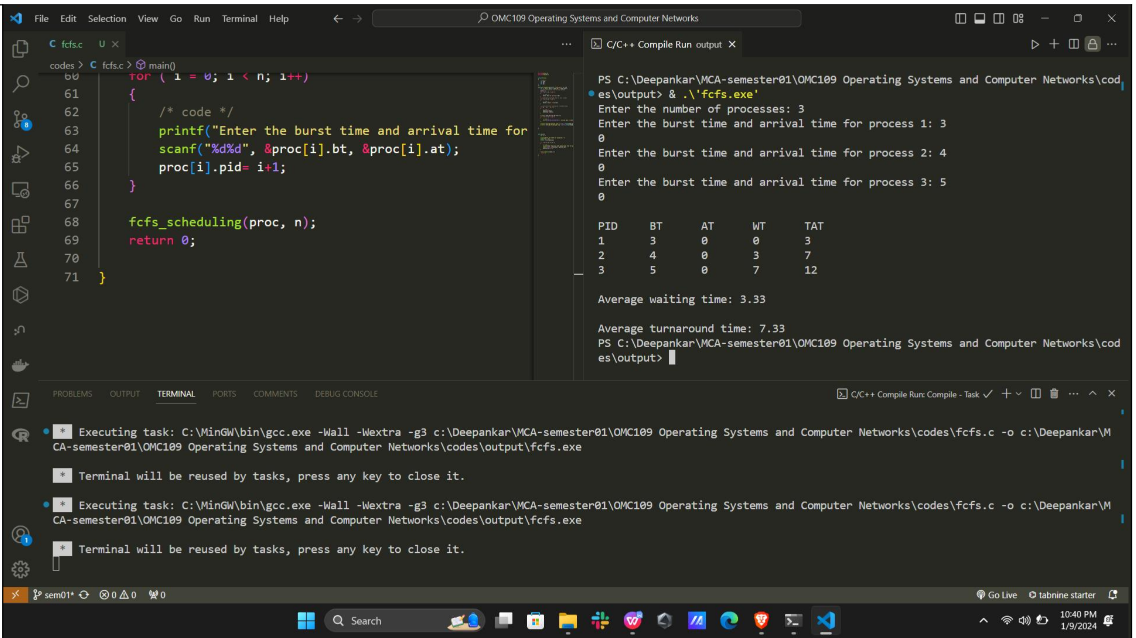
```
PS C:\Deepankar\MCA-semester01\OMC109 Operating Systems and Computer Networks\cod
es\output> & .\'first_fit.exe'
Enter # processes: 3
Enter # blocks: 4
Enter the process sizes
250
100
600
Enter the block sizes
220
300
200
300

P. No.  P. Size Block No.
1       250     2
2       100     1
3       600     Not Allocated
PS C:\Deepankar\MCA-semester01\OMC109 Operating Systems and Computer Networks\cod
es\output> 
```

```
 *  Executing task: C:\MinGW\bin\gcc.exe -Wall -Wextra -g3 c:\Deepankar\MCA-semester01\OMC109 Operating Systems and Computer Networks\codes\first_fit.c -o c:\Deepan
kar\MCA-semester01\OMC109 Operating Systems and Computer Networks\codes\output\first_fit.exe

 *  Terminal will be reused by tasks, press any key to close it.
```

2	Write a C program to Implement the optimal page replacement algorithm
3	Implement a program in C to extract process ID (PID) and parent process ID (PPID)
4	Simulate the following CPU scheduling algorithms-FCFS



```c
#include<stdio.h>
#include<stdlib.h>


struct Process
{
    /* data */
    int pid;
    int bt;
    int at;
};

void fcfs_scheduling(struct Process*proc, int n){
    int i, wt[n], tat[n], total_wt=0, total_tat=0;
    // calculate waiting time for each process
    wt[0]= 0;
    for ( i = 1; i < n; i++)
    {
        /* code */
        wt[i]= wt[i-1]+ proc[i-1].bt;
    }
    // calculate turnaround time for each process
    for ( i = 0; i < n; i++)
    {
        /* code */
        tat[i]= wt[i]+ proc[i].bt;
    }
    // calculate total waiting and turnaround time
    for (i = 0; i < n; i++)
    {
        /* code */
        total_wt+=wt[i];
        total_tat+=tat[i];
    }
    printf("\nPID\tBT\tAT\tWT\tTAT\n");
    for ( i = 0; i < n; i++)
    {
        /* code */
        printf("%d\t%d\t%d\t%d\t%d\n", proc[i].pid, proc[i].bt,
proc[i].at, wt[i], tat[i]);
    }
    printf("\nAverage waiting time: %.2f\n", (float)total_wt/n);
    printf("\nAverage turnaround time: %.2f\n",
(float)total_tat/n);


}


int main(){
    int n, i;
    printf("Enter the number of processes: ");
    scanf("%d", &n);
    struct Process proc[n];
    for ( i = 0; i < n; i++)
    {
        /* code */
        printf("Enter the burst time and arrival time for process %d:
", i+1);
        scanf("%d%d", &proc[i].bt, &proc[i].at);
        proc[i].pid= i+1;
    }
    fcfs_scheduling(proc, n);
    return 0;

}
```

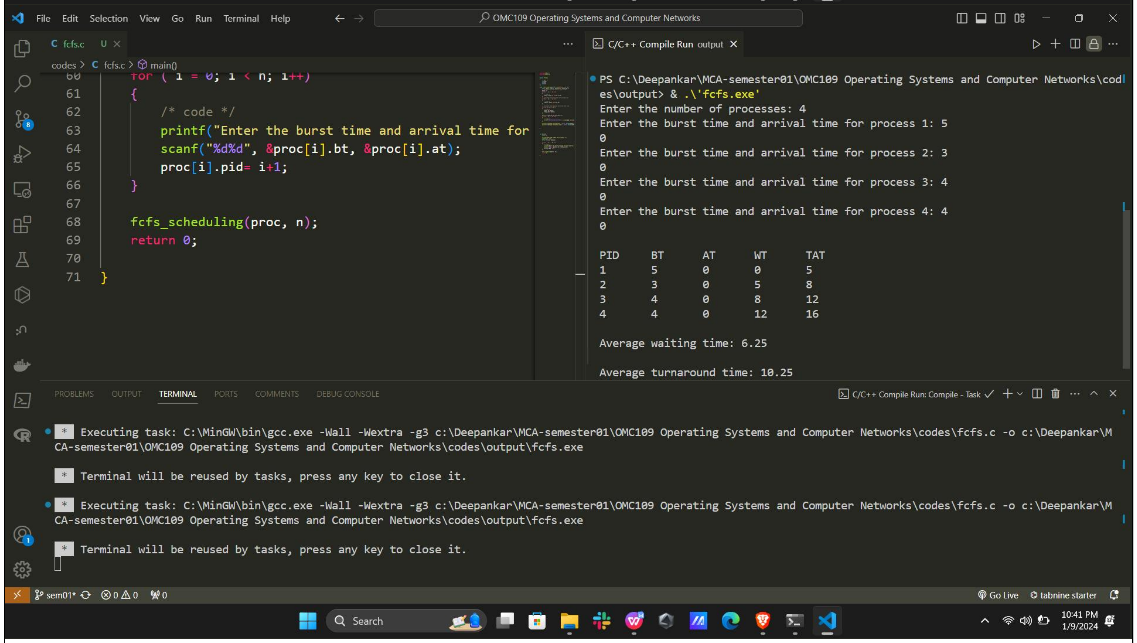


File Edit Selection View Go Run Terminal Help
OMC109 Operating Systems and Computer Networks
fcfs.c U
codes > fcfs.c > main()
C/C++ Compile Run output
PROBLEMS OUTPUT TERMINAL PORTS COMMENTS DEBUG CONSOLE
C/C++ Compile Run: Compile - Task
sem01* 0 0 0
Go Live
tabnine starter
Search
10:40 PM
1/9/2024

```c
    for ( i = 0; i < n; i++)
    {
        /* code */
        printf("Enter the burst time and arrival time for
        scanf("%d%d", &proc[i].bt, &proc[i].at);
        proc[i].pid= i+1;
    }

    fcfs_scheduling(proc, n);
    return 0;

}
```

```
PS C:\Deepankar\MCA-semester01\OMC109 Operating Systems and Computer Networks\cod
es\output> & .\'fcfs.exe'
Enter the number of processes: 3
Enter the burst time and arrival time for process 1: 3
0
Enter the burst time and arrival time for process 2: 4
0
Enter the burst time and arrival time for process 3: 5
0

PID     BT      AT      WT      TAT
1       3       0       0       3
2       4       0       3       7
3       5       0       7       12

Average waiting time: 3.33

Average turnaround time: 7.33
PS C:\Deepankar\MCA-semester01\OMC109 Operating Systems and Computer Networks\cod
es\output>
```

```
 *  Executing task: C:\MinGW\bin\gcc.exe -Wall -Wextra -g3 c:\Deepankar\MCA-semester01\OMC109 Operating Systems and Computer Networks\codes\fcfs.c -o c:\Deepankar\M
CA-semester01\OMC109 Operating Systems and Computer Networks\codes\output\fcfs.exe

 *  Terminal will be reused by tasks, press any key to close it.

 *  Executing task: C:\MinGW\bin\gcc.exe -Wall -Wextra -g3 c:\Deepankar\MCA-semester01\OMC109 Operating Systems and Computer Networks\codes\fcfs.c -o c:\Deepankar\M
CA-semester01\OMC109 Operating Systems and Computer Networks\codes\output\fcfs.exe

 *  Terminal will be reused by tasks, press any key to close it.
```



File Edit Selection View Go Run Terminal Help
OMC109 Operating Systems and Computer Networks
fcfs.c U
codes > fcfs.c > main()
C/C++ Compile Run output
PROBLEMS OUTPUT TERMINAL PORTS COMMENTS DEBUG CONSOLE
C/C++ Compile Run: Compile - Task
sem01* 0 0 0
Go Live
tabnine starter
Search
10:41 PM
1/9/2024

```c
    for ( i = 0; i < n; i++)
    {
        /* code */
        printf("Enter the burst time and arrival time for
        scanf("%d%d", &proc[i].bt, &proc[i].at);
        proc[i].pid= i+1;
    }

    fcfs_scheduling(proc, n);
    return 0;

}
```

```
PS C:\Deepankar\MCA-semester01\OMC109 Operating Systems and Computer Networks\cod
es\output> & .\'fcfs.exe'
Enter the number of processes: 4
Enter the burst time and arrival time for process 1: 5
0
Enter the burst time and arrival time for process 2: 3
0
Enter the burst time and arrival time for process 3: 4
0
Enter the burst time and arrival time for process 4: 4
0

PID     BT      AT      WT      TAT
1       5       0       0       5
2       3       0       5       8
3       4       0       8       12
4       4       0       12      16

Average waiting time: 6.25

Average turnaround time: 10.25
```

```
 *  Executing task: C:\MinGW\bin\gcc.exe -Wall -Wextra -g3 c:\Deepankar\MCA-semester01\OMC109 Operating Systems and Computer Networks\codes\fcfs.c -o c:\Deepankar\M
CA-semester01\OMC109 Operating Systems and Computer Networks\codes\output\fcfs.exe

 *  Terminal will be reused by tasks, press any key to close it.

 *  Executing task: C:\MinGW\bin\gcc.exe -Wall -Wextra -g3 c:\Deepankar\MCA-semester01\OMC109 Operating Systems and Computer Networks\codes\fcfs.c -o c:\Deepankar\M
CA-semester01\OMC109 Operating Systems and Computer Networks\codes\output\fcfs.exe

 *  Terminal will be reused by tasks, press any key to close it.
```

5	Write a C program to Implement the SSTF Disk scheduling
6	Implement the producer consumer problem with solution using semaphore

Compulsory question: Explain the installation steps for Cisco Packet Tracer, and include snapshots for clarification. (6*1=6)