

OPERATING SYSTEMS PROGRAMS

M Satheesh
192011140

1.PROCESS CREATION

The screenshot displays a C++ IDE with a project named 'create process.c'. The code defines two threads, 'Job 1' and 'Job 2', which perform a loop of operations. The output window shows the execution sequence: 'Job 1 has started', 'Job 2 has started', 'Job 2 has finished', and 'Job 1 has finished'. The process exits after 4.543 seconds with a return value of 0. The compilation results show no errors or warnings.

```
#include <pthread.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
pthread_t tid[2];
int counter;

void* trythis(void* arg)
{
    unsigned long i = 0;
    counter += 1;
    printf("\n Job %d has started\n", counter);

    for (i = 0; i < (0xFFFFFFFF); i++)
    {
    }
    printf("\n Job %d has finished\n", counter);

    return NULL;
}

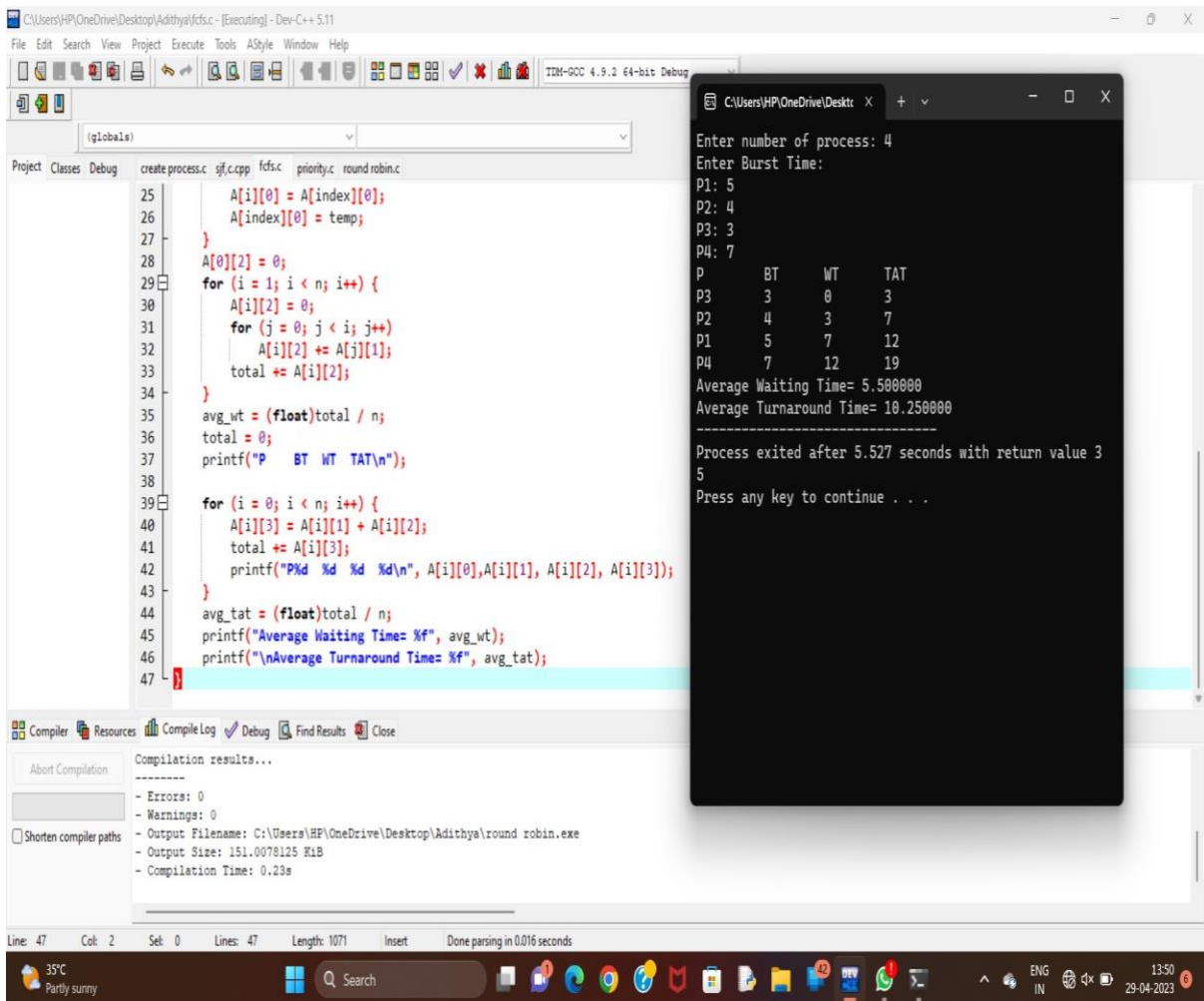
int main(void)
{
    int i = 0;
```

Job 1 has started
Job 2 has started
Job 2 has finished
Job 1 has finished

Process exited after 4.543 seconds with return value 0
Press any key to continue . . .

Compilation results...
- Errors: 0
- Warnings: 0
- Output Filename: C:\Users\HP\OneDrive\Desktop\Adithya\round robin
- Output Size: 151.0078125 KiB
- Compilation Time: 0.23s

2.FCFS SCHEDULING



The screenshot displays a C++ IDE with a project named 'globals'. The code implements a First-Come-First-Served (FCFS) scheduling algorithm. It takes the number of processes and their burst times as input, calculates the waiting and turnaround times for each process, and outputs a table of results. The output window shows the following data:

```
Enter number of process: 4
Enter Burst Time:
P1: 5
P2: 4
P3: 3
P4: 7

P    BT    WT    TAT
P3   3     0     3
P2   4     3     7
P1   5     7    12
P4   7    12    19

Average Waiting Time= 5.500000
Average Turnaround Time= 10.250000

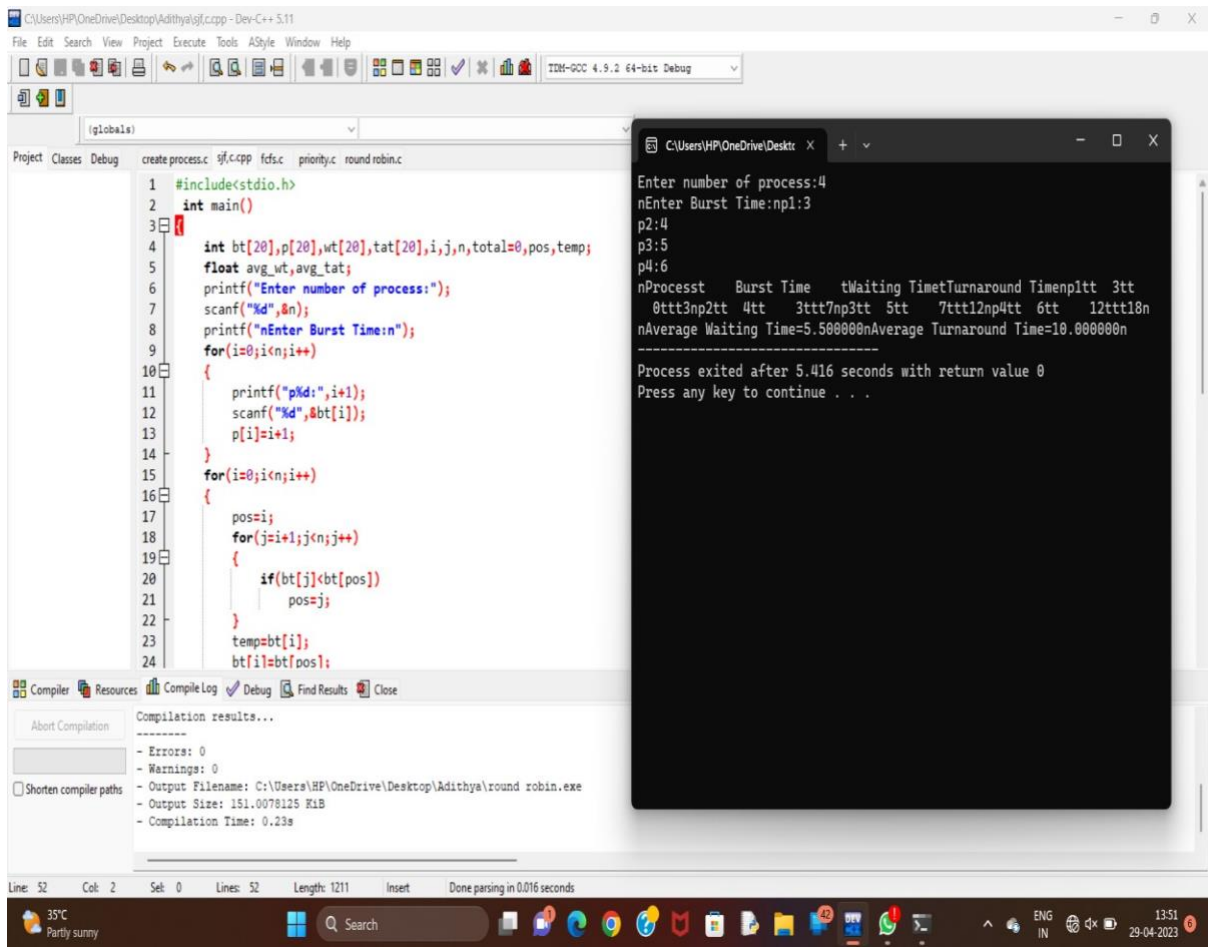
Process exited after 5.527 seconds with return value 3
Press any key to continue . . .
```

The code in the IDE is as follows:

```
25 | A[i][0] = A[index][0];
26 | A[index][0] = temp;
27 | }
28 | A[0][2] = 0;
29 | for (i = 1; i < n; i++) {
30 |     A[i][2] = 0;
31 |     for (j = 0; j < i; j++)
32 |         A[i][2] += A[j][1];
33 |     total += A[i][2];
34 | }
35 | avg_wt = (float)total / n;
36 | total = 0;
37 | printf("P    BT    WT    TAT\n");
38 |
39 | for (i = 0; i < n; i++) {
40 |     A[i][3] = A[i][1] + A[i][2];
41 |     total += A[i][3];
42 |     printf("P%d    %d    %d    %d\n", A[i][0], A[i][1], A[i][2], A[i][3]);
43 | }
44 | avg_tat = (float)total / n;
45 | printf("Average Waiting Time= %f", avg_wt);
46 | printf("\nAverage Turnaround Time= %f", avg_tat);
47 | }
```

The compilation results show 0 errors and 0 warnings. The output filename is 'C:\Users\HP\OneDrive\Desktop\Adithya\round robin.exe', the output size is 151.0078125 KiB, and the compilation time is 0.23s.

3.SJF SCHEDULING



The image shows a C++ IDE with a source code editor, a compiler window, and a terminal window. The source code implements a Shortest Job First (SJF) scheduling algorithm. It takes the number of processes and their burst times as input, sorts them by burst time, and then calculates the waiting and turnaround times for each process. The compiler window shows successful compilation with no errors or warnings. The terminal window displays the program's execution, showing the input, the sorted processes, and the calculated metrics.

```
1 #include<stdio.h>
2 int main()
3 {
4     int bt[20],p[20],wt[20],tat[20],i,j,n,total=0,pos,temp;
5     float avg_wt,avg_tat;
6     printf("Enter number of process:");
7     scanf("%d",&n);
8     printf("nEnter Burst Time:n");
9     for(i=0;i<n;i++)
10    {
11        printf("p%d:",i+1);
12        scanf("%d",&bt[i]);
13        p[i]=i+1;
14    }
15    for(i=0;i<n;i++)
16    {
17        pos=i;
18        for(j=i+1;j<n;j++)
19        {
20            if(bt[j]<bt[pos])
21                pos=j;
22        }
23        temp=bt[i];
24        bt[i]=bt[pos];
```

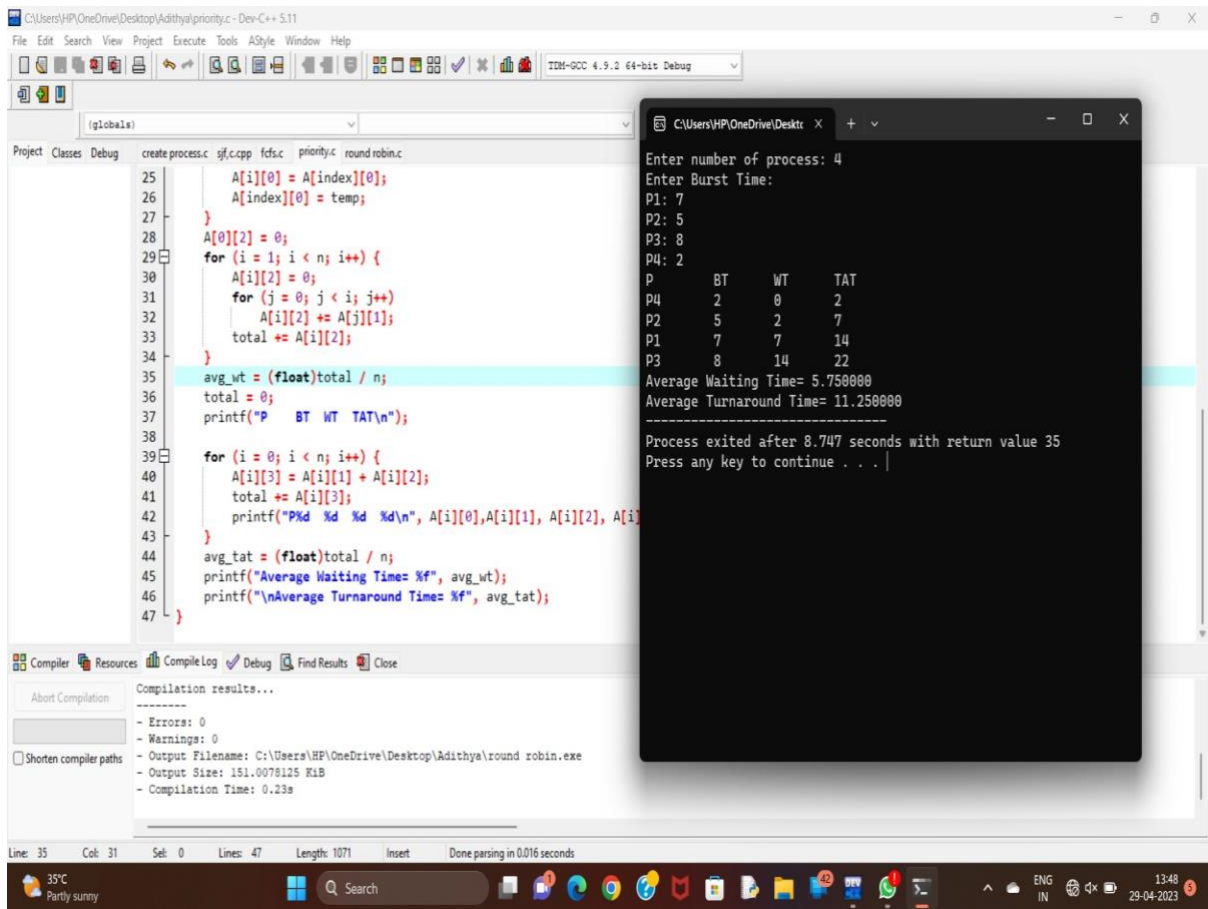
Compilation results...

- Errors: 0
- Warnings: 0
- Output Filename: C:\Users\HP\OneDrive\Desktop\Adithya\round robin.exe
- Output Size: 151.0078125 KiB
- Compilation Time: 0.23s

Enter number of process:4
nEnter Burst Time:np1:3
p2:4
p3:5
p4:6
nProcesst Burst Time tWaiting TimetTurnaround Timenpltt 3tt
0ttt3np2tt 4tt 3ttt7np3tt 5tt 7ttt12np4tt 6tt 12ttt18n
nAverage Waiting Time=5.500000nAverage Turnaround Time=10.000000n

Process exited after 5.416 seconds with return value 0
Press any key to continue . . .

4. PRIORITY SCHEDULING



The screenshot shows a C++ IDE with a project named "priority.c" and a file named "round robin.c". The code implements a priority scheduling algorithm. It takes the number of processes (4) and their burst times (P1: 7, P2: 5, P3: 8, P4: 2) as input. It then calculates the average waiting time (5.750000) and the average turnaround time (11.250000). The output is displayed in a console window.

```
25 | A[i][0] = A[index][0];
26 | A[index][0] = temp;
27 | }
28 | A[0][2] = 0;
29 | for (i = 1; i < n; i++) {
30 |     A[i][2] = 0;
31 |     for (j = 0; j < i; j++)
32 |         A[i][2] += A[j][1];
33 |     total += A[i][2];
34 | }
35 | avg_wt = (float)total / n;
36 | total = 0;
37 | printf("P\tBT\tWT\tTAT\n");
38 |
39 | for (i = 0; i < n; i++) {
40 |     A[i][3] = A[i][1] + A[i][2];
41 |     total += A[i][3];
42 |     printf("P%d\t%d\t%d\t%d\n", A[i][0], A[i][1], A[i][2], A[i][3]);
43 | }
44 | avg_tat = (float)total / n;
45 | printf("Average Waiting Time= %f", avg_wt);
46 | printf("\nAverage Turnaround Time= %f", avg_tat);
47 | }
```

Compilation results...

- Errors: 0
- Warnings: 0
- Output Filename: C:\Users\HP\OneDrive\Desktop\Adithya\round robin.exe
- Output Size: 151.0078125 KiB
- Compilation Time: 0.23s

Enter number of process: 4
Enter Burst Time:
P1: 7
P2: 5
P3: 8
P4: 2

P	BT	WT	TAT
P4	2	0	2
P2	5	2	7
P1	7	7	14
P3	8	14	22

Average Waiting Time= 5.750000
Average Turnaround Time= 11.250000

Process exited after 8.747 seconds with return value 35
Press any key to continue . . .

5.ROUND ROBIN SCHEDULING

The screenshot shows a C++ IDE with a source file named `round robin.c` and a terminal window displaying the program's output.

Source Code:

```
1 #include<stdio.h>
2 #include<conio.h>
3 int main()
4 {
5     int i, NOP, sum=0, count=0, y, quant, wt=0, tat=0;
6     float avg_wt, avg_tat;
7     printf(" Total number of process in the system: ");
8     scanf("%d", &NOP);
9     y = NOP;
10    for(i=0; i<NOP; i++)
11    {
12        printf("\n Enter the Arrival and Burst time of the Process: ");
13        printf(" Arrival time is: \t");
14        scanf("%d", &at[i]);
15        printf(" \nBurst time is: \t");
16        scanf("%d", &bt[i]);
17        temp[i] = bt[i];
18    }
19    printf("Enter the Time Quantum for the process: \t");
20    scanf("%d", &quant);
21    printf("\n Process No \t\t Burst Time \t\t TAT \t\t Waiting Time");
22    for(sum=0, i = 0; y!=0; )
23    {
24        if(temp[i] <= quant && temp[i] > 0)
```

Output:

```
Total number of process in the system: 4
Enter the Arrival and Burst time of the Process[1]
Arrival time is: 5
Burst time is: 6
Enter the Arrival and Burst time of the Process[2]
Arrival time is: 3
Burst time is: 6
Enter the Arrival and Burst time of the Process[3]
Arrival time is: 7
Burst time is: 8
Enter the Arrival and Burst time of the Process[4]
Arrival time is: 2
Burst time is: 3
Enter the Time Quantum for the process: 4
```

Process No	Burst Time	TAT	Waiting Time
Process No[4]	3	13	10
Process No[1]	6	12	6
Process No[2]	6	16	10
Process No[3]	8	16	8

Average Turn Around Time: 8.500000
Average Waiting Time: 14.250000

Compilation Results:

```
- Errors: 0
- Warnings: 0
- Output Filename: C:\Users\HP\OneDrive\Desktop\Adithya\round robin.c
- Output Size: 151.0078125 KiB
- Compilation Time: 0.23s
```