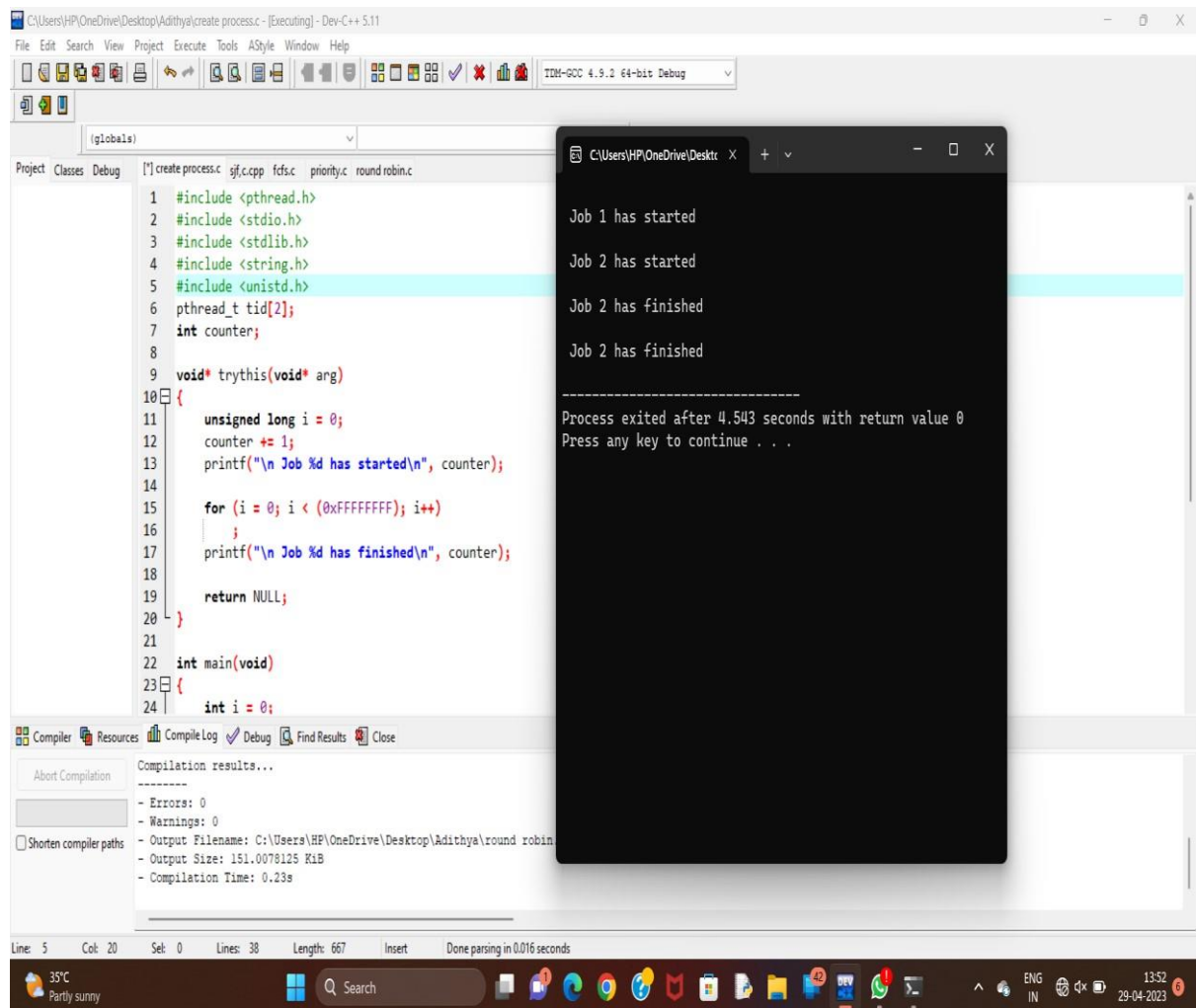


## 1.PROCESS CREATION



## 2.FCFS SCHEDULING

The screenshot displays a C++ IDE with the following components:

- Source Code (round robin.c):**

```
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 }
```
- Compiler Output:**

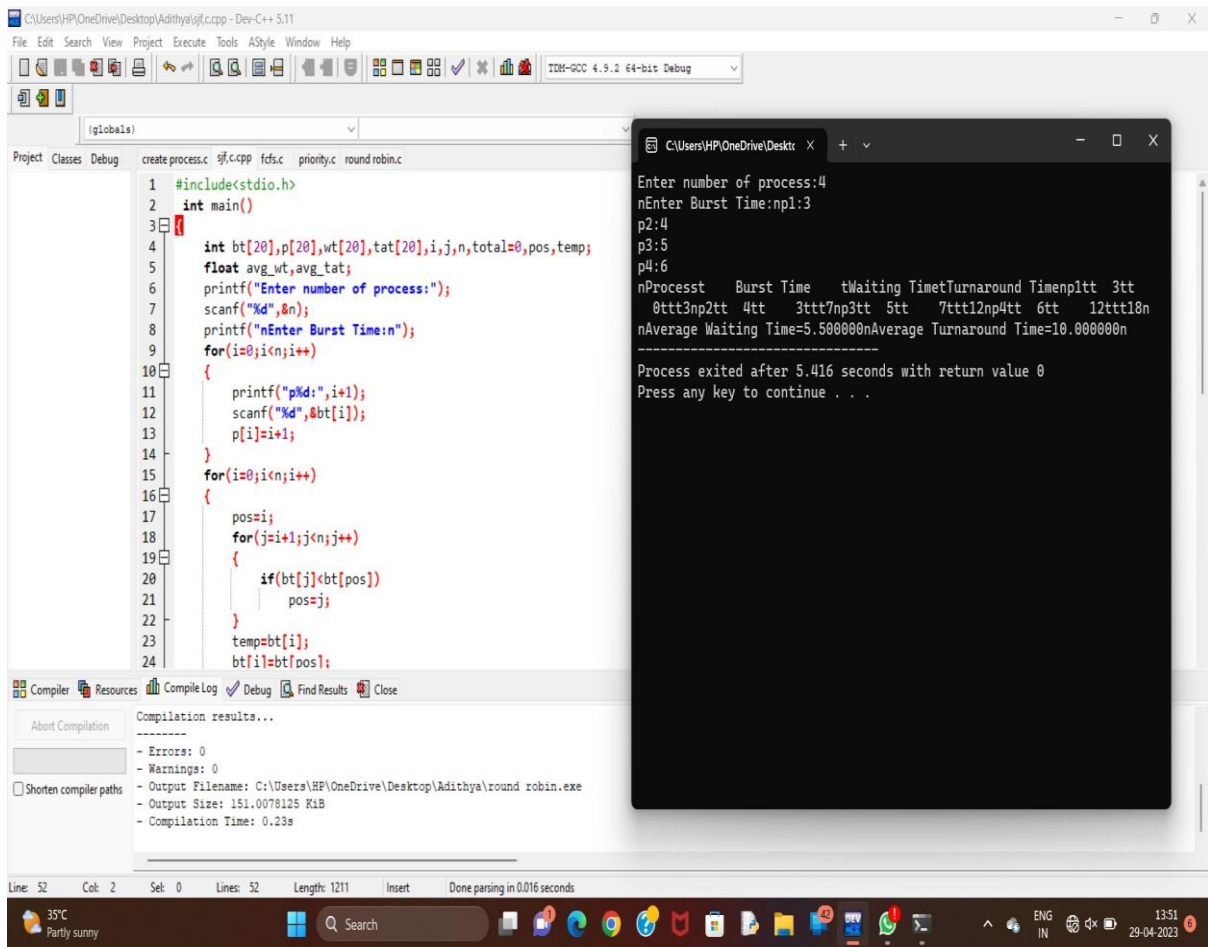
```
Compilation results...  
- Errors: 0  
- Warnings: 0  
- Output Filename: C:\Users\HP\OneDrive\Desktop\Adithya\round robin.exe  
- Output Size: 151.0078125 KiB  
- Compilation Time: 0.23s
```
- Terminal Output:**

```
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  
5  
Press any key to continue . . .
```

### 3.SJF SCHEDULING



The screenshot displays a C++ IDE with a project named 'round robin.c'. The code implements the Shortest Job First (SJF) scheduling algorithm. It prompts the user to enter the number of processes (4) and their burst times (3, 5, 4, 6). The program then calculates the waiting and turnaround times for each process based on the shortest burst time priority.

```
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];
```

The output window shows the following results:

```
Enter number of process:4
nEnter Burst Time:np1:3
p2:4
p3:5
p4:6
nProcesst   Burst Time   tWaiting TimetTurnaround Timenp1tt 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 . . .
```

The IDE also shows the compilation results, indicating 0 errors and 0 warnings. The output filename is 'round robin.exe' and the compilation time is 0.23s.

## 4. PRIORITY SCHEDULING

The screenshot shows a C++ IDE with a project named 'priority.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). It calculates the average waiting time (5.750000) and the average turnaround time (11.250000). The output window shows the process details and the calculated times.

```
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 }
```

Output:

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

P\tBT\tWT\tTAT
P4\t2\t0\t2
P2\t5\t2\t7
P1\t7\t7\t14
P3\t8\t14\t22

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++ program in Dev-C++ implementing Round Robin scheduling. The program prompts the user to enter the arrival and burst times for four processes. It then calculates the Turn Around Time (TAT) and Waiting Time for each process based on a time quantum of 4. The output shows the TAT and Waiting Time for each process, along with the average values.

```
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[%d]:", i);
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)
```

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

Line: 16 Col: 23 Sel: 0 Lines: 61 Length: 1565 Insert Done parsing in 0.016 seconds

36°C Rain showers

Search

ENG IN 14:15 29-04-2023