LAB ASSIGNMENT - 5

Aim: To create C programs for the different scheduling algorithms.

To perform: Create and execute C programs for following CPU Scheduling Algorithms:

1. First Come First Serve (FCFS)

2. Shortest Job First (SJF)

3. Round Robin Scheduling.

```
#include<stdio.h>
#include<vector>
#include<algorithm>
#include<iostream>
#include <climits>
#include<queue>
using namespace std;

struct Process{
   int number;
   int arrival_time;
   int execution_time;
   int completion_time;
   int turnaround_time;
   int waiting_time;
};

void display(Process p[],int n){
   cout<< "\nNO\tArr\texe\tComp\tTAT\tWaiting" <<endl;
   for (int i = 0; i < n; i++) {
      cout << p[i].number<< "\t" << p[i].arrival_time << "\t" << p[i].execution_time << "\t"
      << p[i].completion_time<< "\t" << p[i].turnaround_time<< "\t" << p[i].waiting_time<< endl;}}
```

```
void FCFS(Process p[],int n){//1 first come first serve
  vector<vector<int>>v;
  for(int i=0;i<n;i++){
    v.push_back({p[i].arrival_time,i});
  }
  sort(v.begin(),v.end());
  int total_time=0;
  for(int i=0;i<n;i++){
     total_time+=p[v[i][1]].execution_time;
     p[v[i][1]].completion_time=total_time;
     p[v[i][1]].turnaround_time=total_time-p[v[i][1]].arrival_time;
     p[v[i][1]].waiting_time=p[v[i][1]].turnaround_time-p[v[i][1]].execution_time;
  }
  display(p,n);
}
//2 shortest job first
void SJF(Process p[],int n){
    bool visited[n]={false};
    int remaining=n;
    int total_time=0;
    while(remaining!=0){
       int idx=-1;
       int exTime=INT_MAX;
       for(int i=0;i<n;i++){
        if(p[i].arrival_time<=total_time && !visited[i] && p[i].execution_time<exTime){
            idx=i;
            exTime=p[i].execution_time;
        }
       }
       visited[idx]=true;
       total_time+=p[idx].execution_time;
```

```
      p[idx].completion_time=total_time;
      p[idx].turnaround_time=total_time-p[idx].arrival_time;
      p[idx].waiting_time=p[idx].turnaround_time-p[idx].execution_time;
      remaining--;
    }
    display(p,n);
}
//3 Round-Robin
void roundRobin(Process p[], int n) {
    int quant=5;
    int total_time = 0;
    queue<int> q;
    int remaining[n];
    bool inQueue[n] = {false};
    for (int i = 0; i < n; i++) {
        remaining[i] = p[i].execution_time;
        if(p[i].arrival_time==0){
            q.push(i);
            inQueue[i] = true;}
    }
    while (!q.empty()) {
        int idx = q.front();
        q.pop();
        if (remaining[idx] > quant) {
            total_time += quant;
            remaining[idx] -= quant;
        }

        else {
            total_time += remaining[idx];
            remaining[idx] = 0;
```

```
            p[idx].completion_time = total_time;
            p[idx].turnaround_time = total_time - p[idx].arrival_time;
            p[idx].waiting_time = p[idx].turnaround_time - p[idx].execution_time;
        }

        for (int i = 0; i < n; i++) {
            if (!inQueue[i] && remaining[i] > 0 && p[i].arrival_time <= total_time) {
                q.push(i);
                inQueue[i] = true; }}

        if (remaining[idx] > 0) {
            q.push(idx);}}
    display(p, n);
}

int main(){
int n;
cout<<"enter no of processes ";
cin>>n;
Process p[n];
for(int i=0;i<n;i++){
    p[i].number=i+1;
    cout<<"enter process "<<i+1<<" arrival time "<<endl;
    cin>>p[i].arrival_time;
    cout<<"enter process "<<i+1<<" execution time "<<endl;
    cin>>p[i].execution_time;
}
FCFS(p,n);
SJF(p,n);
roundRobin(p,n);
}
```

```
PS C:\Users\varun\Desktop\C++\OS algo> cd "c:\Users\varun\Desktop\C++\OS algo\" ; if ($?) { g++ Algorithm.cpp -o Algorithm } ; if ($?) { .\Algorithm }
enter no of processes 6
enter process 1 arrival time
15
enter process 1 execution time
14
enter process 2 arrival time
22
enter process 2 execution time
17
enter process 3 arrival time
0
enter process 3 execution time
12
enter process 4 arrival time
5
enter process 4 execution time
10
enter process 5 arrival time
10
enter process 5 execution time
5
enter process 6 arrival time
30
enter process 6 execution time
3
```

```
Applying FCFS algorithm.....

NO      Arr     exe     Comp    TAT     Waiting
1       15      14      41      26      12
2       22      17      58      36      19
3       0       12      12      12      0
4       5       10      22      17      7
5       10      5       27      17      12
6       30      3       61      31      28
Applying SJF algorithm.....

NO      Arr     exe     Comp    TAT     Waiting
1       15      14      41      26      12
2       22      17      61      39      22
3       0       12      12      12      0
4       5       10      27      22      12
5       10      5       17      7       2
6       30      3       44      14      11
Applying roundrobin algorithm.....

NO      Arr     exe     Comp    TAT     Waiting
1       15      14      54      39      25
2       22      17      61      39      22
3       0       12      32      32      20
4       5       10      25      20      10
5       10      5       20      10      5
6       30      3       40      10      7
PS C:\Users\varun\Desktop\C++\OS algo>
```