**Lab Report-10**

**Experiment Name:**

Shortest-Job-First (SJF) Preemptive Scheduling Algorithm.

**Aim and Object:**

1. To know about the working procedure of Shortest-Job-First (SJF) Preemptive Scheduling Algorithm.
2. To implement the Shortest-Job-First (SJF) Preemptive Scheduling Algorithm program in C++ and check its following output.

**Theory:**

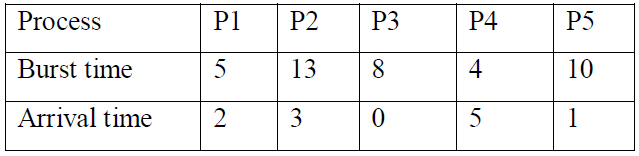
**Shortest-Job-First (SJF) Preemptive Algorithm**

* This is also known as **shortest job first**, or SJF
* This is a non-preemptive, pre-emptive scheduling algorithm.
* Best approach to minimize waiting time.
* Easy to implement in Batch systems where required CPU time is known in advance.
* Impossible to implement in interactive systems where required CPU time is not known.
* The processer should know in advance how much time process will take.

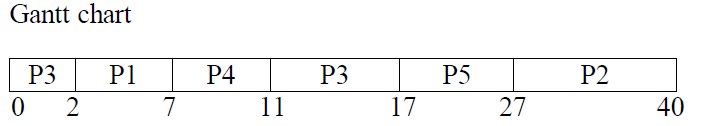
**Example:**

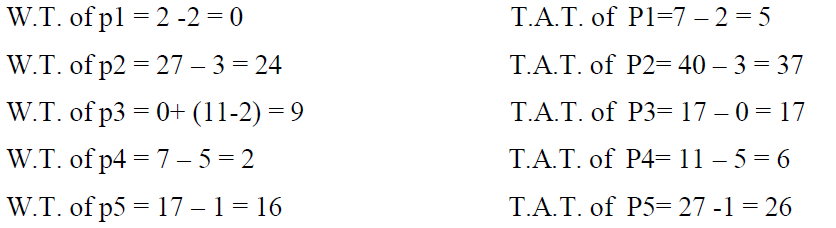
Find the average waiting time (A.W.T) and average turnaround time (A.T.A.T) for

Executing the following process using Preemptive short-job first short-job first



**Solution:**

****

****

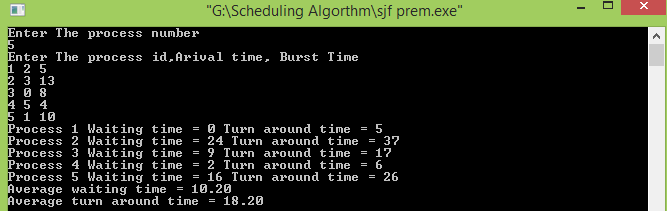
So, A.W.T = (0+24+9+2+16)/5=10.2

And A.T.A.T = (5+37+17+6+26)/5=18.2

**Source code of SJF Preemptive:**

|  |  |
| --- | --- |
| #include<bits/stdc++.h>  #define ll long long int  #define pb push\_back  using namespace std;  struct st  {  ll proc,arrive,burst;  };  vector<st>v;  vector<ll>wait,turn;  ll process;  void findwait()  {  vector<ll>rt;  for(ll i=0; i<process; i++)  rt.pb(v[i].burst);  ll x,complete=0,t=0,mn=LONG\_LONG\_MAX,shortest=0,finish\_time;  bool check=false;  st a;  while(complete<process)  {  for(ll i=0; i<process; i++)  {  a=v[i];  if(a.arrive<=t&&(rt[i]<mn&&rt[i]>0))  {  mn=rt[i];  shortest=i;  check=true;  }  }  if(check==false)  {  t++;  continue;  }  rt[shortest]--;  mn=rt[shortest];  if(mn==0)  mn=LONG\_LONG\_MAX;  if(rt[shortest]==0)  {  complete++;  finish\_time=t+1;  wait[shortest]=finish\_time-v[shortest].burst-v[shortest].arrive;  if(wait[shortest]<0)  wait[shortest]=0;  }  t++;  }  } | void findturnaround()  {  st a;  for(ll i=0; i<process; i++)  {  a=v[i];  turn[i]+=a.burst+wait[i];  }  }  int main()  {  cout<<"Enter The process number"<<endl;  while(cin>>process)  {  cout<<"Enter The process id,Arival time, Burst Time"<<endl;  v.clear();  wait.clear();  turn.clear();  for(ll i=0; i<=process; i++)  {  wait.pb(0);  turn.pb(0);  }  st a;  ll x;  for(ll i=0; i<process; i++)  {  cin>>x;  a.proc=x;  cin>>x;  a.arrive=x;  cin>>x;  a.burst=x;  v.pb(a);  }  findwait();  findturnaround();  double avg\_wait=0.0,avg\_turn=0.0;  for(ll i=0; i<process; i++)  {  a=v[i];  cout<<"Process "<<a.proc<<" Waiting time = "<<wait[i]<<" Turn around time = "<<turn[i]<<endl;  avg\_wait+=(double)wait[i];  avg\_turn+=(double)turn[i];  }  printf("Average waiting time = %.2lf\n",avg\_wait/process);  printf("Average turn around time = %.2lf\n",avg\_turn/process);  }  } |

**Output:**



**Conclusion:** From this Lab I know about Shortest-Job-First (SJF) Preemptive Scheduling Algorithm. I also know about the working procedure of Shortest-Job-First (SJF) Preemptive Scheduling Algorithm. Then I implement the Shortest-Job-First (SJF) Preemptive Scheduling Algorithm. Then I write the Program of SJF Preemptive Algorithm and check the output with the following example. Finally I successfully done this lab.