**Lab Report-09**

**Experiment Name:**

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

**Aim and Object:**

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

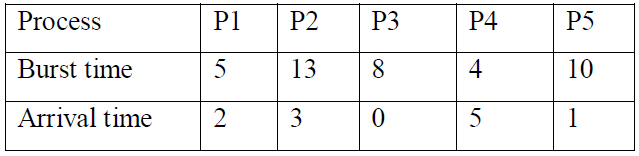
**Theory:**

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

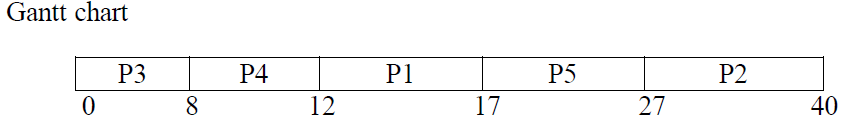
* This is also known as **shortest job first**, or SJF
* This is a non-preemptive, pre-emptive scheduling algorithm.
* Best approach to minimize waiting time.
* Easy to implement in Batch systems where required CPU time is known in advance.
* Impossible to implement in interactive systems where required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

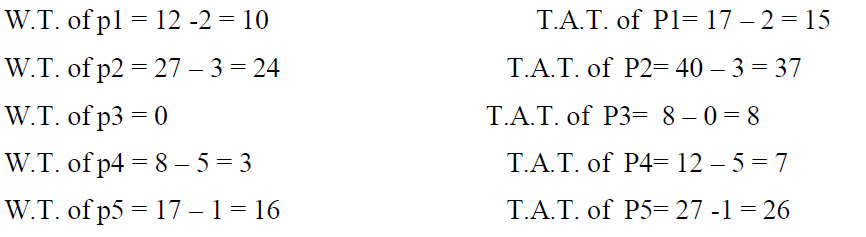
**Example:**

Find the average waiting time (A.W.T) and average turnaround time (A.T.A.T) for executing the following process using Non-preemptive short-job first.



**Solution:**

****

****

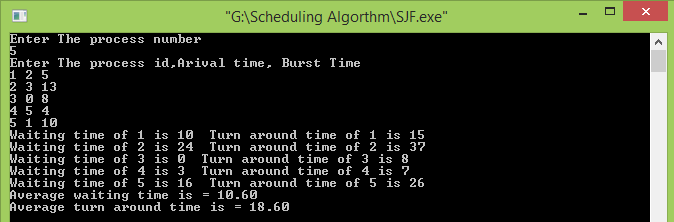
So, A.W.T = (10+24+0+3+16)/5=10.6

And A.T.A.T = (15+37+8+7+26)/5=18.6

**Source code of SJF Non Preemptive:**

|  |  |
| --- | --- |
| #include<bits/stdc++.h>  #define ll long long int  #define pb push\_back  using namespace std;  struct st  {  ll proc,arrive,burst;  };  struct stq  {  ll pr,start,last;  };  bool cmp(st a,st b)  {  if(a.burst==b.burst)return a.arrive<b.arrive;  return a.burst<b.burst;  }  bool cmp1(stq a,stq b)  {  return a.pr<b.pr;  }  int main()  {  ll process,p,pp,ppp;  cout<<"Enter The process number"<<endl;  while(cin>>process)  {  cout<<"Enter The process id,Arival time, Burst Time"<<endl;  vector<st>v,rs;  st x,y;  x.proc=-1;  x.arrive=-1;  x.burst=-1;  v.pb(x);  for(ll i=0; i<process; i++)  {  cin>>ppp;  cin>>p;  cin>>pp;  if(p==0)  {  y.proc=ppp;  y.arrive=p;  y.burst=pp;  rs.pb(y);  }  else  { | x.proc=ppp;  x.arrive=p;  x.burst=pp;  v.pb(x);  rs.pb(x);  }  }  v[0]=y;  sort(v.begin()+1,v.end(),cmp);  vector<stq>V;  stq xx;  x=v[0];  xx.pr=x.proc;  xx.start=0;  xx.last=x.burst;  V.pb(xx);  ll sum=x.burst;  for(ll i=1;i<process;i++)  {  x=v[i];  xx.start=sum;  sum+=x.burst;  xx.pr=x.proc;  xx.last=sum;  V.pb(xx);  }  sort(V.begin(),V.end(),cmp1);  double avg\_wait=0.0,avg\_turn=0.0;  ll t;  for(ll i=0;i<process;i++)  {  xx=V[i];  x=rs[i];  t=xx.start-x.arrive;  avg\_wait+=(double)t;  printf("Waiting time of %lld is %lld ",x.proc,t);  t=xx.last-x.arrive;  avg\_turn+=(double)t;  printf("Turn around time of %lld is %lld\n",x.proc,t);  }  avg\_turn/=process;  avg\_wait/=process;  printf("Average waiting time is = %.2lf\n",avg\_wait);  printf("Average turn around time is = %.2lf\n",avg\_turn);  }  } |

**Output:**



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