**Experiment No. 09**

**Experiment Name: Implementation of SJF** **non-preemptive Scheduling algorithm .**

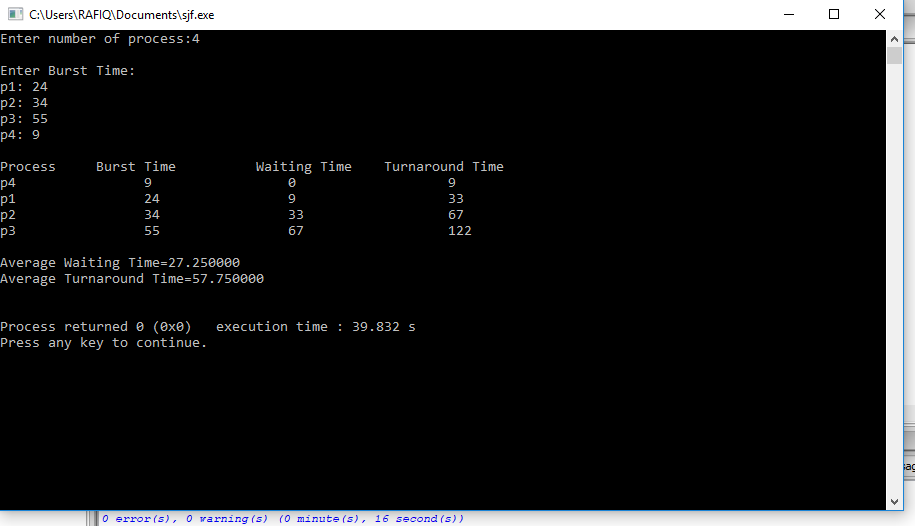
**Objectives:**

* What is SJF non-preemptive Scheduling algorithm.
* How to implementation in C

**Source code:**

|  |
| --- |
| #include<stdio.h>  int main()  {  int bt[20],p[20],wt[20],tat[20],i,j,n,total=0,pos,temp;  float avg\_wt,avg\_tat;  printf("Enter number of process:");  scanf("%d",&n);  printf("\nEnter Burst Time:\n");  for(i=0;i<n;i++)  {  printf("p%d: ",i+1);  scanf("%d",&bt[i]);  p[i]=i+1;  }  for(i=0;i<n;i++)  {  pos=i;  for(j=i+1;j<n;j++)  {  if(bt[j]<bt[pos])  pos=j;  }  temp=bt[i];  bt[i]=bt[pos];  bt[pos]=temp;  temp=p[i];  p[i]=p[pos];  p[pos]=temp;  }  wt[0]=0;  for(i=1;i<n;i++)  {  wt[i]=0;  for(j=0;j<i;j++)  wt[i]+=bt[j];  total+=wt[i];  }  avg\_wt=(float)total/n;  total=0;  printf("\nProcess\t Burst Time \tWaiting Time\tTurnaround Time");  for(i=0;i<n;i++)  {  tat[i]=bt[i]+wt[i];  total+=tat[i];  printf("\np%d\t\t %d\t\t %d\t\t\t%d",p[i],bt[i],wt[i],tat[i]);  }  avg\_tat=(float)total/n; //average turnaround time  printf("\n\nAverage Waiting Time=%f",avg\_wt);  printf("\nAverage Turnaround Time=%f\n",avg\_tat);  printf("\n");  return 0;  } |

**Output:**



**Conclusion:** Shortest-Job-First (SJF) is a non-preemptive discipline in which waiting job (or process) with the smallest estimated run-time-to-completion is run next. In other words, when CPU is available, it is assigned to the process that has smallest next CPU burst. The SJF scheduling is especially appropriate for batch jobs for which the run times are known in advance.