**Lab Report #06**

**Experiment Name #**  Short Job First (SJF) Non-preemptive Scheduling Algorithm.

**Aim and Objective:**

The purpose of the experiment is to learn Short Job First (SJF) non-preemptive scheduling algorithm and execute a code by using C.

* To learn SJF non-preemptive scheduling algorithm.
* To implement SJF using C.

**Code:**

#include<stdio.h>

int main()

{

int i,n,j,min,line[10],line1[10],p[10]={1,2,3,4,5,6,7,8,9,10};

int B\_T[1000],A\_V\_T[1000],W\_T[1000],temp,T\_A\_T[10];

int bt=0,ta=0,sum=0,k=1;

float AVR\_W\_T=0,AVR\_T\_A\_T=0,T\_A\_T\_SUM=0,W\_T\_SUM=0;

printf("Number of processes : ");

scanf("%d",&n);

for(i=0;i<n;i++)

{

printf("\nArrival Time Of Process %d : ",i+1);

scanf(" %d",&A\_V\_T[i]);

printf("Burst Time Of Process%d : ",i+1);

scanf(" %d",&B\_T[i]);

}

for(i=0;i<n;i++)

{

for(j=0;j<n;j++)

{

if(A\_V\_T[i]<A\_V\_T[j])

{

temp=p[j];

p[j]=p[i];

p[i]=temp;

temp=A\_V\_T[j];

A\_V\_T[j]=A\_V\_T[i];

A\_V\_T[i]=temp;

temp=B\_T[j];

B\_T[j]=B\_T[i];

B\_T[i]=temp;

}

}

}

for(j=0;j<n;j++)

{

bt=bt+B\_T[j];

min=B\_T[k];

for(i=k;i<n;i++)

{

if (bt>=A\_V\_T[i] && B\_T[i]<min)

{

temp=p[k];

p[k]=p[i];

p[i]=temp;

temp=A\_V\_T[k];

A\_V\_T[k]=A\_V\_T[i];

A\_V\_T[i]=temp;

temp=B\_T[k];

B\_T[k]=B\_T[i];

B\_T[i]=temp;

}

}

k++;

}

W\_T[0]=0;

for(i=1;i<n;i++)

{

sum=sum+B\_T[i-1];

W\_T[i]=sum-A\_V\_T[i];

line[i]=W\_T[i];

W\_T\_SUM=W\_T\_SUM+W\_T[i];

}

AVR\_W\_T=(W\_T\_SUM/n);

for(i=0;i<n;i++)

{

ta=ta+B\_T[i];

T\_A\_T[i]=ta-A\_V\_T[i];

line1[i]=T\_A\_T[i];

T\_A\_T\_SUM=T\_A\_T\_SUM+T\_A\_T[i];

}

AVR\_T\_A\_T=(T\_A\_T\_SUM/n);

printf("\nProcess Name\tBurst Time\t Arrival Time\t Waiting Time\t Turn Around Time" );

for(i=0;i<n;i++)

{

printf("\np%d\t\t%d\t\t %d\t\t %d\t\t %d",p[i],B\_T[i],A\_V\_T[i],W\_T[i],T\_A\_T[i]);

}

printf("\n\nAverage Waiting Time: %.2f",AVR\_W\_T);

printf("\nAverage Turn Around Time: %.2f\n",AVR\_T\_A\_T);

printf("\n");

for(i=0;i<T\_A\_T\_SUM/2;i++)

{

printf(" ");

}

printf("gantt chart\n\n");

line[0]=0;

for(i=0;i<n;i++)

{

for(j=0;j<(line[i]+line1[i])/2;j++)

{

printf(" ");

}

printf("P%d",p[i]);

}

printf("\n");

printf("0");

for(i=0;i<n; i++)

{

for(j=0;j<line1[i];j++)

{

printf("-");

}

printf("%d",line1[i]);

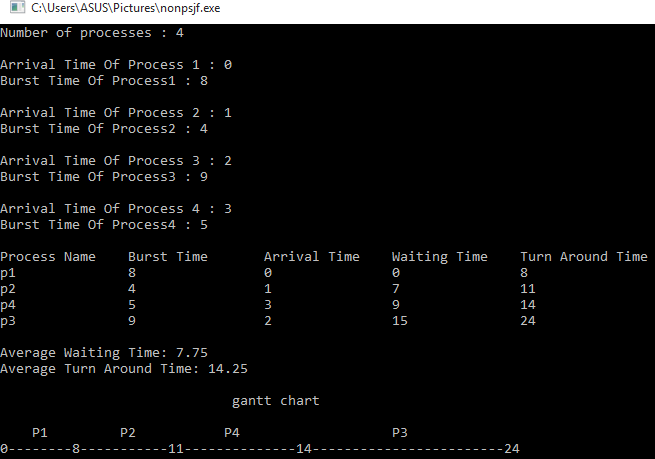
}

printf("\n");

getch();

return 0;}

**Output:**



**Conclusion:**

By following these steps one can perform short job first non-preemptive using c. In this algorithm the process which has shortest burst time has to do the job first and holds CPU until it executes total time. It is quite important CPU scheduling in operation system.