//Q1.Implement SJF with specified arrival time and burst time. Compute waiting time, turnaround and completion time.

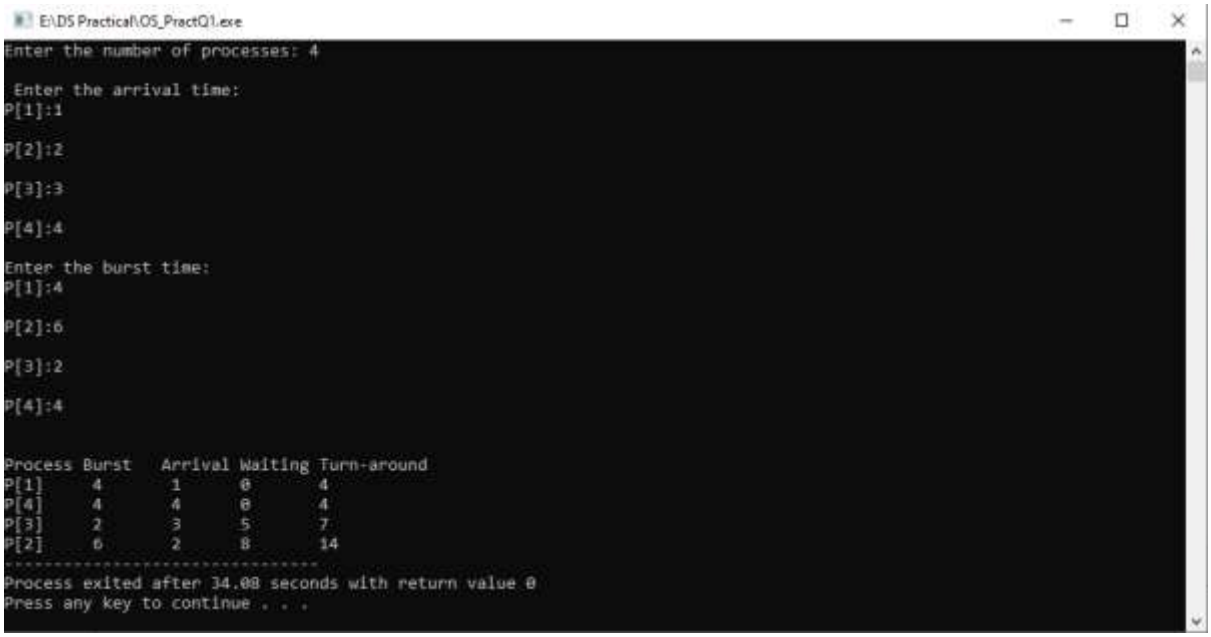
```
#include<stdio.h>
#include<iostream>
using namespace std;
int main() // main block
{
int i,n,proc_id[10],min,k=1,btime=0;
int bt[10],temp,j,at[10],wt[10],tt[10],a[10];
int t=0,m=0;
cout<<"Enter the number of processes: ";
cin>>n;
cout<<"\n Enter the arrival time: \n";
for(i=0;i<n;i++) //loop to input arrival time
{
cout<<"P["<<i+1<<"]:";
cin>>at[i];
cout<<endl;
}
cout<<"Enter the burst time:"<<endl;
for(i=0;i<n;i++) // loop to input burst time
{
cout<<"P["<<i+1<<"]:";
cin>>bt[i];
cout<<endl;
proc_id[i]=i+1;
}
for(i=0;i<n;i++)
{
for(j=0;j<n;j++)
{
```

```
if(at[i]<at[j])
{
temp=proc_id[j];
proc_id[j]=proc_id[i];
proc_id[i]=temp;
temp=at[j];
at[j]=at[i];
at[i]=temp;temp=bt[j];
bt[j]=bt[i];
bt[i]=temp;
  }
 }
}
t=0;
temp=0;
for(i=0;i<n;i++)
{
for(j=0;j<(n-1)-i;j++)
{
if(at[j]==at[j+1])
{
if(bt[j]>bt[j+1])
{
temp=at[j];
at[j]=at[j+1];
at[j+1]=temp;
t=bt[j];
bt[j]=bt[j+1];
bt[j+1]=t;
m=proc_id[j];
proc_id[j]=proc_id[j+1];proc_id[j+1]=m;
```

```
}
}
}
}
for(j=0;j<n;j++)
{
btime=btime+bt[j];
min=bt[k];
for(i=k;i<n;i++)
{
if (btime>=at[i] &&bt[i]<min)
{
temp=proc_id[k];
proc_id[k]=proc_id[i];
proc_id[i]=temp;
temp=at[k];
at[k]=at[i];
at[i]=temp;
temp=bt[k];
bt[k]=bt[i];
bt[i]=temp;
}
}
k++;}
wt[0]=0;
a[0]=0;
for(i=1;i<n;i++)
{
a[i]=a[i-1]+bt[i-1];
wt[i]=a[i]-at[i];
}
```

```
for(i=0;i<n;i++)
{
tt[i]=wt[i]+bt[i];
}
cout<<endl;
cout<<"Process\t"<<"Burst\t"<<"Arrival\t"<<"Waiting\t"<<"Turn-around" ;
for(i=0;i<n;i++)
{
cout<<"\n"<<"P["<<proc_id[i]<<"]"<<" \t "<<bt[i]<<" \t "<<at[i]<<" \t"<<wt[i]<<"\t" <<tt[i];
}
}
```

OUTPUT:



//Q2.Write a program to demonstrate fork where parent and child run different codes and parent process should be executed first.

```
#include <stdio.h>
```

```
#include <sys/types.h>
#include <unistd.h>

void forkexample()
{

   // child process because return value zero

   if (fork() == 0)

      printf("Hello from Child!\n");
   // parent process because return value non-zero.
   else
      printf("Hello from Parent!\n");
}
int main()
{
   forkexample();
   return 0;
}
```

OUTPUT:


OS_Practical_2_SET_1.cpp > forkdemo()
1 #include <stdio.h>
2 #include <sys/types.h>
PROBLEMS
OUTPUT
TERMINAL
DEBUG CONSOLE
Code
cd "/home/adarsh212/Desktop/OS_Practical/" && g++ OS_Practical_2_SET_1.cpp -o OS_Practical_2_SET_1 && "/home/adarsh212/Desktop/OS_Practical/"OS_Practiadarsh212@adarsh212:~/Desktop/OS_Practical$ cd "/home/adarsh212/Desktop/OS_Practical/" && g++ OS_Practical_2_SET_1.cpp -o OS_Practical_2_SET_1 && "/home/adarsh212/Desktop/OS_Practical/"OS_Practical_2_SET_1
Hello from Parent!
Hello from Child!
adarsh212@adarsh212:~/Desktop/OS_Practical$