```
/* Shortest Job first
No pre-emption, so arrival time of processes has not been considered
If some processes have the same execution time then the program works like FCFS */
#include<stdio.h>
#include<unistd.h>

main()
{
    int n,i,j,t1,t2;
    char x[100],y[100],z[100];
    printf("\nEnter the number of processes:");
    scanf("%d",&n);
    int bt[n],p[n],pid[n],ft[n];

    for(i=0;i<n;i++)
        {
            printf("\nEnter the execution time for P%d:",i);
            scanf("%d",&bt[i]);
            p[i]=i; //P[i] contains process ID's
        }
    for(i=0;i<n;i++) //To sort processes in ascending order of execution time
        {
            for(j=i+1;j<n;j++)
                {
                    if(bt[i]>bt[j])
                        {
                            t1=bt[i];
                            t2=p[i]; //Swapping process ID also
                            bt[i]=bt[j];
                            p[i]=p[j];
                            bt[j]=t1;
                            p[j]=t2;
                        }
                }
        }

    ft[0]=bt[0];
    for(i=1;i<n;i++) //For calculating finishing time
        {
            ft[i]=ft[i-1]+bt[i];
        }

    for(i=0;i<n;i++)
        {
            pid[i]=fork();
            if(pid[i]==0)
                {
                    sprintf(x,"%d",p[i]);
                    sprintf(y,"%d",bt[i]);
                    sprintf(z,"%d",ft[i]);
                    execl("/cygdrive/c/users/pushpinder/desktop/child.exe","child.exe",x,y,z,
                    NULL);
```

```
                perror("Exec failed");
            }
        sleep(bt[i]); //So that the parent only creates a process after the previous one
        has executed
    }

}
```

```c
/*Code for child process used in SJF Scheduling */
#include<stdio.h>
#include<unistd.h>
#include<string.h>
#include<signal.h>
#include<stdlib.h>

int num,bt,ft;
char x[100],y[100];

void alarm_handler() //Used to handle SIGALRM
    {
      write(1,y,strlen(y));//Prints finishing message of process
    }


main(int argc,char *argv[])
    {
        num=atoi(argv[1]); //num stores process ID
        bt=atoi(argv[2]);  //bt stores execution time
        ft=atoi(argv[3]);  //ft stores finishing time

        sprintf(x,"\nProcess %d says:I will execute now for %d seconds",num,bt);
        sprintf(y,"\nProcess %d says:I have finished execution at t=%d",num,ft);

        write(1,x,strlen(x)); //Prints at starting of process execution
        signal(SIGALRM,alarm_handler); //Catches SIGALRM

        alarm(bt); //Self suspension till process is executed
        pause();

        exit(0);

    }
```

/cygdrive/c/users/pushpinder/Desktop

```
pushpinder@pushpinder-PC /cygdrive/c/users/pushpinder/Desktop
$ ./sjf

Enter the number of processes:5

Enter the execution time for P0:4

Enter the execution time for P1:5

Enter the execution time for P2:2

Enter the execution time for P3:6

Enter the execution time for P4:4

Process 2 says:I will execute now for 2 seconds
Process 2 says:I have finished execution at t=2
Process 0 says:I will execute now for 4 seconds
Process 0 says:I have finished execution at t=6
Process 4 says:I will execute now for 4 seconds
Process 4 says:I have finished execution at t=10
Process 1 says:I will execute now for 5 seconds
Process 1 says:I have finished execution at t=15
Process 3 says:I will execute now for 6 seconds
pushpinder@pushpinder-PC /cygdrive/c/users/pushpinder/Desktop
$
Process 3 says:I have finished execution at t=21
```

21:11
31-08-2014