CPU SCHEDULING ALGORITHMS
2. Shortest Job First (SJF)

```
// Disk Scheduling - SJF
#include<stdio.h>
#include<stdbool.h>
typedef struct
{
int pid;
float at, wt, bt, ta, st;
bool isComplete;
}process;
void procdetail(int i, process p[])
{
printf("Process ID: ");
scanf("%d", &p[i].pid);
printf("Arrival Time: ");
scanf("%f", &p[i].at);
printf("Burst Time: ");
scanf("%f", &p[i].bt);
p[i].isComplete = false;
}//procdetail
void sort(process p[], int i, int start)
{
int k = 0, j;
process temp;
for (k = start; k<i; k++)
{
for (j = k+1; j<i; j++)
{
if(p[k].bt < p[j].bt)
continue;
else
{
temp = p[k];
p[k] = p[j];
p[j] = temp;
}
}
}
}//sort
void main()
{
int n, i, k = 0, j = 0;
float avgwt = 0.0, avgta = 0.0, tst = 0.0;
printf("Enter number of Processes: \n");
scanf("%d",&n);
process p[n];
for (i = 0; i<n; i++)
{
printf("\nEnter Process Details \n\n: ");
procdetail(i,p);
}
for (i = 0; i<n; i++)
{
if (p[i].isComplete == true)
continue;
else
{
k = i;
while (p[i].at<=tst && i<n)
i++;
sort (p,i,k);
i = k;
if(p[i].at<=tst)
p[i].st = tst;
else
p[i].st = p[i].at;
p[i].st = tst;
p[i].isComplete = true;
tst += p[i].bt;
p[i].wt = p[i].st - p[i].at;
p[i].ta = p[i].bt + p[i].wt;
avgwt += p[i].wt;
avgta += p[i].ta;
}
}
avgwt /= n;
avgta /= n;
printf("Process Schedule Table: \n");
printf("\tProcess ID\tArrival Time\tBurst
Time\tWait Time\tTurnaround Time\n");
for (i = 0; i<n; i++)
printf("\t%d\t\t%f\t%f\t%f\t%f\n", p[i].pid,p[i].at,
p[i].bt, p[i].wt, p[i].ta);
printf("\nAverage wait time: %f", avgwt);
printf("\nAverage turnaround time: %f\n",
avgta);
}//main
```