CPU scheduling algorithms

a) FCFS

```
#include<stdio.h>
int main()
{
int bt[20], wt[20], tat[20], i, n;
float wtavg, tatavg;
printf("\nEnter the number of processes -- ");
scanf("%d", &n);
for(i=0;i<n;i++)
{
printf("\nEnter Burst Time for Process %d -- ", i);
scanf("%d", &bt[i]);
}
wt[0] = wtavg = 0;
tat[0] = tatavg = bt[0];
for(i=1;i<n;i++)
{
wt[i] = wt[i-1] +bt[i-1];
tat[i] = tat[i-1] +bt[i];
wtavg = wtavg + wt[i];
tatavg = tatavg + tat[i];
}
printf("\t PROCESS \tBURST TIME \t WAITING TIME\t TURNAROUND TIME\n");
for(i=0;i<n;i++)
printf("\n\t P%d \t\t %d \t\t %d \t\t %d", i, bt[i], wt[i], tat[i]);
printf("\nAverage Waiting Time -- %f", wtavg/n);
printf("\nAverage Turnaround Time -- %f", tatavg/n);
return 0;
}
```

b)SJF (Non-Pre-emptive)

```
#include<stdio.h>
int main()
{
int p[20], bt[20], wt[20], tat[20], i, k, n, temp;
 float wtavg, tatavg;
printf("\nEnter the number of processes -- ");
scanf("%d", &n);
for(i=0;i<n;i++)
{
p[i]=i;
printf("Enter Burst Time for Process %d -- ", i);
scanf("%d", &bt[i]);
}
for(i=0;i<n;i++)
for(k=i+1;k<n;k++)
if(bt[i]>bt[k])
{
temp=bt[i];
bt[i]=bt[k];
bt[k]=temp;

temp=p[i];
p[i]=p[k];
p[k]=temp;
}
wt[0] = wtavg = 0;
tat[0] = tatavg = bt[0];
for(i=1;i<n;i++)
{
wt[i] = wt[i-1] +bt[i-1];
tat[i] = tat[i-1] +bt[i];
wtavg = wtavg + wt[i];
tatavg = tatavg + tat[i];
}
printf("\n\t PROCESS \tBURST TIME \t WAITING TIME\t TURNAROUND TIME\n");
for(i=0;i<n;i++)
printf("\n\t P%d \t\t %d \t\t %d \t\t %d", p[i], bt[i], wt[i], tat[i]);
printf("\nAverage Waiting Time -- %f", wtavg/n);
printf("\nAverage Turnaround Time -- %f", tatavg/n);
return 0;
}
```

When Arrival Time is User should provide in SJF:

```
#include<stdio.h>

int main(){
  int p[20], bt[20],at[20],wt[20],tat[20],i,n,k,temp;
  float wtavg,tatavg;

  printf("Enter the no of process --");
  scanf("%d",&n);

  for(i=0;i<n;i++)
  {
    p[i]=i;
    printf("Enter the Arrival time for process %d --",i);
    scanf("%d",&at[i]);
}

for(i=0;i<n;i++)
{
    p[i]=i;
    printf("Enter the Burst time for process %d --",i);
    scanf("%d",&bt[i]);
}
for(i=0;i<n;i++)
    for(k=i+1;k<n;k++)
    if(bt[i]>bt[k])
{
    temp=bt[i];
    bt[i]=bt[k];
    bt[k]=temp;

    temp=p[i];
    p[i]=p[k];
    p[k]=temp;

}

wt[0]=wtavg=0;
tat[0]=tatavg=bt[0];
for(i=1;i<n;i++){
    wt[i]=wt[i-1]+bt[i-1];
    tat[i]=wt[i]+bt[i];
    wtavg=wtavg+wt[i];
    tatavg=tatavg+tat[i];
    }


    printf("\t PROCESS\tARRIVAL TIME\tBURST TIME\t WAITING TIME\t TURN AROUNT
TIME\n");
    for(i=0;i<n;i++)
        printf("\n\tP%d\t\t%d\t\t%d\t\t%d\t\t%d",i,at[i],bt[i],wt[i],tat[i]);


    printf("\n Average waiting time -- %f",wtavg/n);
    printf("\n Average Turn Around Time -- %f",tatavg/n);


    return 0;
}
```

Round Robin:

```
#include<stdio.h>


int main(){
    int p[20], bt[20], at[20], wt[20], tat[20], rt[20], i, n, k, temp, t, q;
    float wtavg, tatavg;


    printf("Enter the no of process --");
    scanf("%d",&n);
    for(i=0;i<n;i++)
    {
        p[i]=i;
        printf("Enter the Arrival time for process %d --",i);
        scanf("%d",&at[i]);
    }

    for(i=0;i<n;i++)
    {
        p[i]=i;
        printf("Enter the Burst time for process %d --",i);
        scanf("%d",&bt[i]);
        rt[i] = bt[i]; // initialize remaining time
    }

    printf("Enter the time quantum --");
    scanf("%d",&q);

    t = 0;
    while(1) {
        int flag = 0;
        for(i=0;i<n;i++) {
            if(at[i] <= t && rt[i] > 0) {
                if(rt[i] > q) {
                    rt[i] -= q;
                    t += q;
                }
else {
                    t += rt[i];
                    rt[i] = 0;
                    flag = 1;
                }
            }
        }

        if(flag == 0) t++;

        int all_done = 1;
        for(i=0;i<n;i++) {
            if(rt[i] > 0) {
                all_done = 0;
                break;
            }
        }

        if(all_done) break;
    }

    wtavg = tatavg = 0;
    for(i=0;i<n;i++) {
        tat[i] = t - at[i];
        wt[i] = tat[i] - bt[i];
        wtavg += wt[i];
        tatavg += tat[i];
    }

    printf("\t PROCESS\tARRIVAL TIME\tBURST TIME\t WAITING TIME\t TURN AROUNT
TIME\n");
    for(i=0;i<n;i++)
        printf("\n\tP%d\t\t%d\t\t%d\t\t%d\t\t%d",i,at[i],bt[i],wt[i],tat[i]);
    printf("\n Average waiting time -- %f",wtavg/n);
    printf("\n Average Turn Around Time -- %f",tatavg/n);

    return 0;
}
```

Priority:

```
#include<stdio.h>

int main(){
    int p[20], bt[20], at[20], pr[20], wt[20], tat[20], i, n, k, temp, t;
    float wtavg, tatavg;

    printf("Enter the no of process --");
    scanf("%d",&n);

    for(i=0;i<n;i++)
    {
        p[i]=i;
        printf("Enter the Arrival time for process %d --",i);
        scanf("%d",&at[i]);
    }

    for(i=0;i<n;i++)
    {
        p[i]=i;
        printf("Enter the Burst time for process %d --",i);
        scanf("%d",&bt[i]);
    }
```

```
for(i=0;i<n;i++)
{
    p[i]=i;
    printf("Enter the Priority for process %d --",i);
    scanf("%d",&pr[i]);
}

// Sorting based on priority in descending order (higher priority first)
for(i=0;i<n;i++)
{
    for(k=i+1;k<n;k++)
    {
        if(pr[i] < pr[k])
        {
            temp = p[i];
            p[i] = p[k];
            p[k] = temp;

            temp = at[i];
            at[i] = at[k];
            at[k] = temp;

            temp = bt[i];
            bt[i] = bt[k];
            bt[k] = temp;

            temp = pr[i];
            pr[i] = pr[k];
            pr[k] = temp;
            }
        }
    }

    t = 0;
    wtavg = tatavg = 0;
    for(i=0;i<n;i++)
    {
        t += bt[i];
        tat[i] = t - at[i];
        wt[i] = tat[i] - bt[i];
        wtavg += wt[i];
        tatavg += tat[i];
    }

    printf("\t PROCESS\tARRIVAL TIME\tBURST TIME\tPRIORITY\t WAITING TIME\t TURN
AROUNT TIME\n");
    for(i=0;i<n;i++)
        printf("\n\tP%d\t\t%d\t\t%d\t\t%d\t\t%d\t\t%d",p[i],at[i],bt[i],pr[i],wt[i],tat[i]);

    printf("\n Average waiting time -- %f",wtavg/n);
    printf("\n Average Turn Around Time -- %f",tatavg/n);

    return 0;
}
```