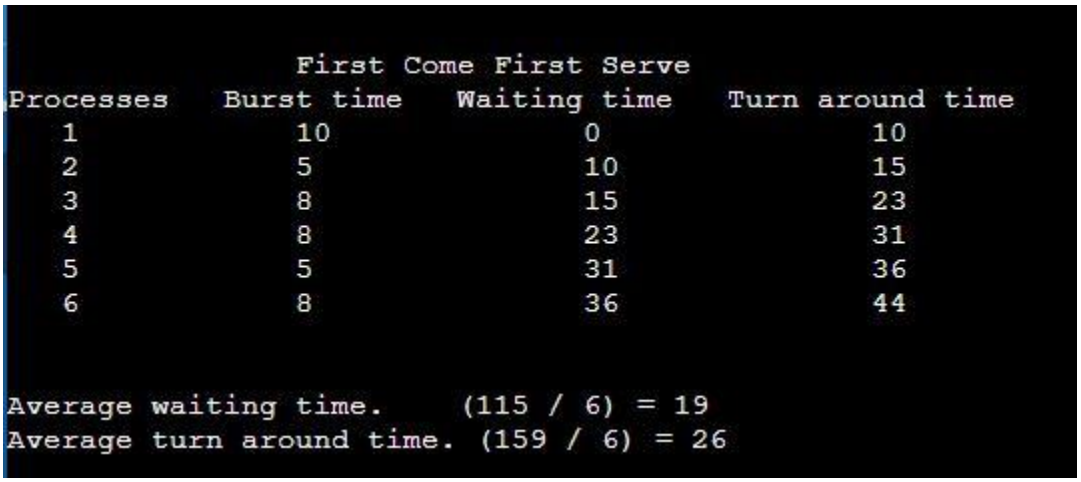
Muhammad Fahad
FA19-BSSE

First Come First Serve:

Code:

```c
#include<stdio.h>
void findWaitingTime(int processes[], int n, bt[], int wt[]){
    wt[0] = 0;
    for (int  i = 1; i < n ; i++ )
        wt[i] =  bt[i-1] + wt[i-1] ;
}

void findTurnAroundTime( int processes[], int n, int bt[], int wt[], int tat[]){
    for (int  i = 0; i < n ; i++)
        tat[i] = bt[i] + wt[i];
}

void findavgTime( int processes[], int n, int bt[]){
    int wt[n], tat[n], total_wt = 0, total_tat = 0;

    findWaitingTime(processes, n, bt, wt);
    findTurnAroundTime(processes, n, bt, wt, tat);

    printf("Processes   Burst time   Waiting time   Turn around time\n");
    for (int  i=0; i<n; i++){
        total_wt = total_wt + wt[i];
        total_tat = total_tat + tat[i];

        printf("   %d\t\t%d\t\t%d\t\t%d\n",(i+1), bt[i] ,wt[i] ,tat[i] );
    }
    int s=(float)total_wt / (float)n;
    int t=(float)total_tat / (float)n;

    printf("\n\nAverage waiting time. \t (%d / %d) = %d",total_wt, n, s);
    printf("\nAverage turn around time. (%d / %d) = %d",total_tat, n,t);
}

int main(){
    int processes[] = { 1, 2, 3,8,5,8};
    int  burst_time[] = {10, 5, 8,8,5,8};

    int n = sizeof processes / sizeof processes[0];

    findavgTime(processes, n,  burst_time);
    return 0;
}
```

Output:



```
                First Come First Serve
Processes   Burst time   Waiting time   Turn around time
   1            10              0              10
   2            5               10             15
   3            8               15             23
   4            8               23             31
   5            5               31             36
   6            8               36             44


Average waiting time.    (115 / 6) = 19
Average turn around time. (159 / 6) = 26
```

Muhammad Fahad
FA19-BSSE

Shortest Job First:

Code:

```
#include <stdio.h>

void main()
{


  printf("\n\n\t\tShortest Job First\n");

  int bt[] = {10, 5, 8, 7, 2, 1},
      p[] = {1, 2, 3, 4, 5, 6},
      n = 6,
      wt[n], tat[n], total = 0,
      // loop varaibles
      i, j, pos, temp;

  float avg_wt, avg_tat;

  n = 6;

  //sorting burst time in ascending order using selection sort
  for (i = 0; i < n; i++)
  {
    pos = i;
    for (j = i + 1; j < n; j++)
    {
      if (bt[j] < bt[pos])
        pos = j;
    }

    temp = bt[i];
    bt[i] = bt[pos];
    bt[pos] = temp;

    temp = p[i];
    p[i] = p[pos];
    p[pos] = temp;
  }

  wt[0] = 0;
```

```
  //calculate waiting time
  for (i = 1; i < n; i++)
  {
    wt[i] = 0;
    for (j = 0; j < i; j++)
      wt[i] += bt[j];

    total += wt[i];
  }

  avg_wt = (float)total / n;
  int total_wt = total;
  total = 0;

  printf("\nProcess\t    Burst Time    \tWaiting Time\tTurnaround Time");
  for (i = 0; i < n; i++)
  {
    tat[i] = bt[i] + wt[i];
    total += tat[i];
    printf("\np%d\t\t  %d\t\t    %d\t\t\t%d", p[i], bt[i], wt[i], tat[i]);
  }

  avg_tat = (float)total / n;

  printf("\n\nAverage waiting time.  \t  (%d / %d) = %f",total_wt, n, avg_wt);
  printf("\nAverage turn around time. (%d / %d) = %f",total, n,avg_tat);
}
```

Output:

Muhammad Fahad
FA19-BSSE

```
                Shortest Job First

Process     Burst Time          Waiting Time    Turnaround Time
p6                  1                   0                   1
p5                  2                   1                   3
p2                  5                   3                   8
p4                  7                   8                   15
p3                  8                   15                  23
p1                  10                  23                  33

Average waiting time.      (50 / 6) = 8.333333
Average turn around time.  (83 / 6) = 13.833333
```

Shortest Job Remaining:

Code:

```
#include <stdio.h>
struct process
{

  int WT, AT, BT, TAT;
};

struct process a[10] = {};

int main(){
  printf("\n\n\t\tShortest Job Remaining\n");

  int n = 6, temp[10];
  int count = 0, t = 0, short_P;
  float total_WT = 0, total_TAT = 0, Avg_WT, Avg_TAT;

  int bt[] = {10, 5, 8, 7, 2, 1},
      p[] = {0, 0, 3, 4, 0, 6};

  printf("\n");

  for (int i = 0; i < n; i++){
    a[i].BT = bt[i];
    a[i].AT = p[i];
    temp[i] = a[i].BT;
  }

  a[9].BT = 10000; // temp for max
  for (t = 0; count != n; t++){

    short_P = 9;

    for (int i = 0; i < n; i++){
      if (a[i].BT < a[short_P].BT && (a[i].AT <= t && a[i].BT > 0))
        short_P = i;
    }

    a[short_P].BT = a[short_P].BT - 1;
```

```
    // if any process is completed
    if (a[short_P].BT == 0){

      // one process complete
      count++;

      a[short_P].WT = t + 1 - a[short_P].AT - temp[short_P];

      a[short_P].TAT = t + 1 - a[short_P].AT;

      // total calculation
      total_WT = total_WT + a[short_P].WT;

      total_TAT = total_TAT + a[short_P].TAT;
    }
  }

  Avg_WT = total_WT / n;

  Avg_TAT = total_TAT / n;

  // printing of the answer
  printf("Id.\tAT.\tBT.\tWT.\tTAT.\n");

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

    printf(" %d\t%d\t%d\t%d\t%d\n", (i + 1), a[i].AT,
           (a[i].TAT - a[i].WT), a[i].WT, a[i].TAT);
  }
  printf("Avg waiting time of the process is %f\n", Avg_WT);

  printf("Avg turn around time of the process %f\n", Avg_TAT);
}
```

Output:

```
          Shortest Job Remaining

Id.     AT.     BT.     WT.     TAT.
 1      0       10      23      33
 2      0       5       2       7
 3      3       8       12      20
 4      4       7       4       11
 5      0       2       0       2
 6      6       1       1       2
Avg waiting time of the process is 7.000000
Avg turn around time of the process 12.500000
```