Shortest Job First Scheduling (Preemptive):

```
import java.util.*;
public class SJF {

  public static void main(String args[]) {
    Scanner sc = new Scanner(System.in);
    System.out.println("*** Shortest Job First Scheduling (Preemptive) ***");
    System.out.print("Enter no of process:");
    int n = sc.nextInt();
    int process[] = new int[n];
    int arrivaltime[] = new int[n];
    int burstTime[] = new int[n];
    int completionTime[] = new int[n];
    int TAT[] = new int[n];
    int waitingTime[] = new int[n];
    int visit[] = new int[n];
    int remburstTime[] = new int[n];
    int temp, start = 0, total = 0;
    float avgwt = 0, avgTAT = 0;

    for (int i = 0; i < n; i++) {
      System.out.println(" ");
      process[i] = (i + 1);
      System.out.print("Enter Arrival Time for processor " + (i + 1) + ":");
      arrivaltime[i] = sc.nextInt();
      System.out.print("Enter Burst Time for processor " + (i + 1) + ": ");
      burstTime[i] = sc.nextInt();
      remburstTime[i] = burstTime[i];
      visit[i] = 0;
    }
```

```
for (int i = 0; i < n; i++) {
  for (int j = 0; j < n; j++) {
    if (arrivaltime[i] < arrivaltime[j]) {
      temp = process[j];
      process[j] = process[i];
      process[i] = temp;
      temp = arrivaltime[j];
      arrivaltime[j] = arrivaltime[i];
      arrivaltime[i] = temp;
      temp = remburstTime[j];
      remburstTime[j] = remburstTime[i];
      remburstTime[i] = temp;
      temp = burstTime[j];
      burstTime[j] = burstTime[i];
      burstTime[i] = temp;
    }
  }
}
while (true) {
  int min = 99, c = n;
  if (total == n) {
    break;
  }
  for (int i = 0; i < n; i++) {
    if ((arrivaltime[i] <= start) && (visit[i] == 0) && (burstTime[i] < min)) {
      min = burstTime[i];
      c = i;
    }
  }
 if (c == n)
```

```
      start++;
     else {
      burstTime[c]--;
      start++;
      if (burstTime[c] == 0) {
       completionTime[c] = start;
       visit[c] = 1;
       total++;
      }
     }
    }
    for (int i = 0; i < n; i++) {
     TAT[i] = completionTime[i] - arrivaltime[i];
     waitingTime[i] = TAT[i] - remburstTime[i];
     avgwt += waitingTime[i];
     avgTAT += TAT[i];
    }
    System.out.println("*** Shortest Job First Scheduling (Preemptive) ***");
    System.out.println("Processor\tArrival time\tBrust time\tCompletion Time\t\tTurn around time\tWaiting time");
    System.out.println(
      "-----------------------------------------------------------------------------------------------------");
    for (int i = 0; i < n; i++) {
     System.out.println("P" + process[i] + "\t\t" + arrivaltime[i] + "ms\t\t" + remburstTime[i] + "ms\t\t"
       + completionTime[i] + "ms\t\t\t" + TAT[i] + "ms\t\t\t" + waitingTime[i] + "ms");
    }
    avgTAT /= n;
    avgwt /= n;
```

```
    System.out.println("\nAverage turn around time is " + avgTAT);
    System.out.println("Average waiting time is " + avgwt);
    sc.close();
  }
}
```

Output:

```
*** Shortest Job First Scheduling (Preemptive) ***
Enter no of process:4

Enter Arrival Time for processor 1:1
Enter Burst Time for processor 1: 4

Enter Arrival Time for processor 2:2
Enter Burst Time for processor 2: 9

Enter Arrival Time for processor 3:3
Enter Burst Time for processor 3: 5

Enter Arrival Time for processor 4:4
Enter Burst Time for processor 4: 6
*** Shortest Job First Scheduling (Preemptive) ***
```

Processor	Arrival time	Brust time	Completion Time	Turn around time	Waiting time
P1	1ms	4ms	5ms	4ms	0ms
P2	2ms	9ms	25ms	23ms	14ms
P3	3ms	5ms	10ms	7ms	2ms
P4	4ms	6ms	16ms	12ms	6ms

```
Average turn around time is 11.5
Average waiting time is 5.5
```