```
#include <stdio.h>

#define max 20

void sjf() {
   int A[max][4];
   int i, j, n, total = 0, index, temp;
   float avg_wt, avg_tat;

   printf("Enter number of processes: ");
   scanf("%d", &n);
   printf("Enter Burst Time:\n");
   for (i = 0; i < n; i++) {
      printf("P%d: ", i + 1);
      scanf("%d", &A[i][1]);
      A[i][0] = i + 1;
   }

   // Sorting based on burst time using selection sort
   for (i = 0; i < n; i++) {
      index = i;
      for (j = i + 1; j < n; j++)
         if (A[j][1] < A[index][1])
            index = j;
      temp = A[i][1];
      A[i][1] = A[index][1];
      A[index][1] = temp;
      temp = A[i][0];
      A[i][0] = A[index][0];
      A[index][0] = temp;
   }
```

```
    A[0][2] = 0;
    total = 0;
    for (i = 1; i < n; i++) {
        A[i][2] = 0;
        for (j = 0; j < i; j++)
            A[i][2] += A[j][1];
        total += A[i][2];
    }

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

    printf("P BT WT TAT\n");
    for (i = 0; i < n; i++) {
        A[i][3] = A[i][1] + A[i][2];
        total += A[i][3];
        printf("P%d %d %d %d\n", A[i][0], A[i][1], A[i][2], A[i][3]);
    }

    avg_tat = (float)total / n;
    printf("Average Waiting Time= %f\n", avg_wt);
    printf("Average Turnaround Time= %f\n", avg_tat);
}

void rr() {
    int i, burstTime[max], remainTime[max], remainProcess, arrivalTime[max];
    int totalExecutionTime = 0, timeQuantum, flag = 0, n;
    float totalWaitingTime = 0;

    printf("Enter the Number of Processes (max 20): ");
```

```
scanf("%d", &n);
remainProcess = n;

printf("Enter Arrival Time\n");
for (i = 0; i < n; i++) {
   printf("For P[%d]: ", i + 1);
   scanf("%d", &arrivalTime[i]);
}

printf("Enter Burst Time\n");
for (i = 0; i < n; i++) {
   printf("For P[%d]: ", i + 1);
   scanf("%d", &burstTime[i]);
   remainTime[i] = burstTime[i];
}

printf("Enter Time Quantum: ");
scanf("%d", &timeQuantum);

for (i = 0; remainProcess != 0;) {
   if (remainTime[i] <= timeQuantum && remainTime[i] > 0) {
      totalExecutionTime += remainTime[i];
      remainTime[i] = 0;
      flag = 1;
   } else if (remainTime[i] > 0) {
      remainTime[i] -= timeQuantum;
      totalExecutionTime += timeQuantum;
   }

   if (flag == 1 && remainTime[i] == 0) {
      printf("P[%d] | Waiting Time: %d\n", i + 1, totalExecutionTime - arrivalTime[i] - burstTime[i]);
```

```
            totalWaitingTime += totalExecutionTime - arrivalTime[i] - burstTime[i];
            flag = 0;
            remainProcess--;
        }

        if (i == n - 1)
            i = 0;
        else if (arrivalTime[i + 1] <= totalExecutionTime)
            i++;
        else
            i = 0;
    }

    totalWaitingTime = (float)totalWaitingTime / n;
    printf("The Average Waiting Time: %.2f\n", totalWaitingTime);
}

int main() {
    int choice;
    printf("********* Menu **********\n");
    printf("1. Shortest Job First (SJF)\n");
    printf("2. Round Robin (RR)\n");
    printf("Enter your choice: ");
    scanf("%d", &choice);

    switch (choice) {
        case 1:
            sjf();
            break;
        case 2:
            rr();
```

```
            break;
        default:
            printf("Invalid choice!\n");
            break;
    }

    return 0;
}
```