**Program: Round Robin Scheduling With Arrival Time.**

**CODE:**

```
#include #include using namespace std;

struct Process { int id, at, bt, remaining_bt, wt, tat, ct; };

void roundRobin(Process p[], int n, int quantum) { queue q; int time =
0, completed = 0; float total_wt = 0, total_tat = 0; bool inQueue[n] =
{false}; // To check if process is already in queue

// Push processes into queue based on arrival time
while (completed < n) {
    for (int i = 0; i < n; i++) {
        if (p[i].at <= time && !inQueue[i] && p[i].remaining_bt > 0) {
          q.push(i);
            inQueue[i] = true;
        }
    }

    if (q.empty()) {
        time++;
        continue;
    }
 int i = q.front();
    q.pop();

    if (p[i].remaining_bt > quantum) {
        time += quantum;
        p[i].remaining_bt -= quantum;
    } else {
        time += p[i].remaining_bt;
        p[i].remaining_bt = 0;
        p[i].ct = time;
        p[i].tat = p[i].ct - p[i].at;
        p[i].wt = p[i].tat - p[i].bt;
        total_wt += p[i].wt;
        total_tat += p[i].tat;
        completed++;
    }
    // Re-add process if still has remaining time
    for (int j = 0; j < n; j++) {
        if (p[j].at <= time && !inQueue[j] && p[j].remaining_bt > 0) {
            q.push(j);
            inQueue[j] = true;
        }
    }

    if (p[i].remaining_bt > 0) {
        q.push(i);
    }
}

cout << "\nProcess\tAT\tBT\tWT\tTAT\tCT\n";
for (int i = 0; i < n; i++) {
    cout << p[i].id << "\t" << p[i].at << "\t" << p[i].bt << "\t"
         << p[i].wt << "\t" << p[i].tat << "\t" << p[i].ct << endl;
}

cout << "\nAverage Waiting Time: " << (total_wt / n);
cout << "\nAverage Turnaround Time: " << (total_tat / n) << endl;


}

int main() { int n, quantum; cout << "Enter number of processes: ";
cin >> n;

Process p[n];

for (int i = 0; i < n; i++) {
    cout << "Enter arrival time and burst time for process " << i + 1
<< ": ";
    cin >> p[i].at >> p[i].bt;
    p[i].id = i + 1;
    p[i].remaining_bt = p[i].bt; // Initialize remaining burst time
}

cout << "Enter time quantum: ";
```

```
cin >> quantum;

roundRobin(p, n, quantum);
return 0;
 }
```

**OUTPUT:**

```
Enter number of processes: 3

Enter arrival time and burst time for process 1: 0 5

Enter arrival time and burst time for process 2: 1 3

Enter arrival time and burst time for process 3: 2 8

Enter time quantum: 2

 Process        AT      BT      WT      TAT     CT

1       0       5       7       12      12

2       1       3       5       8       9

3       2       8       6       14      16


Average Waiting Time: 6

Average Turnaround Time: 11.3333
```

**Conclusion: The program Round Robin was implemented successfully.**