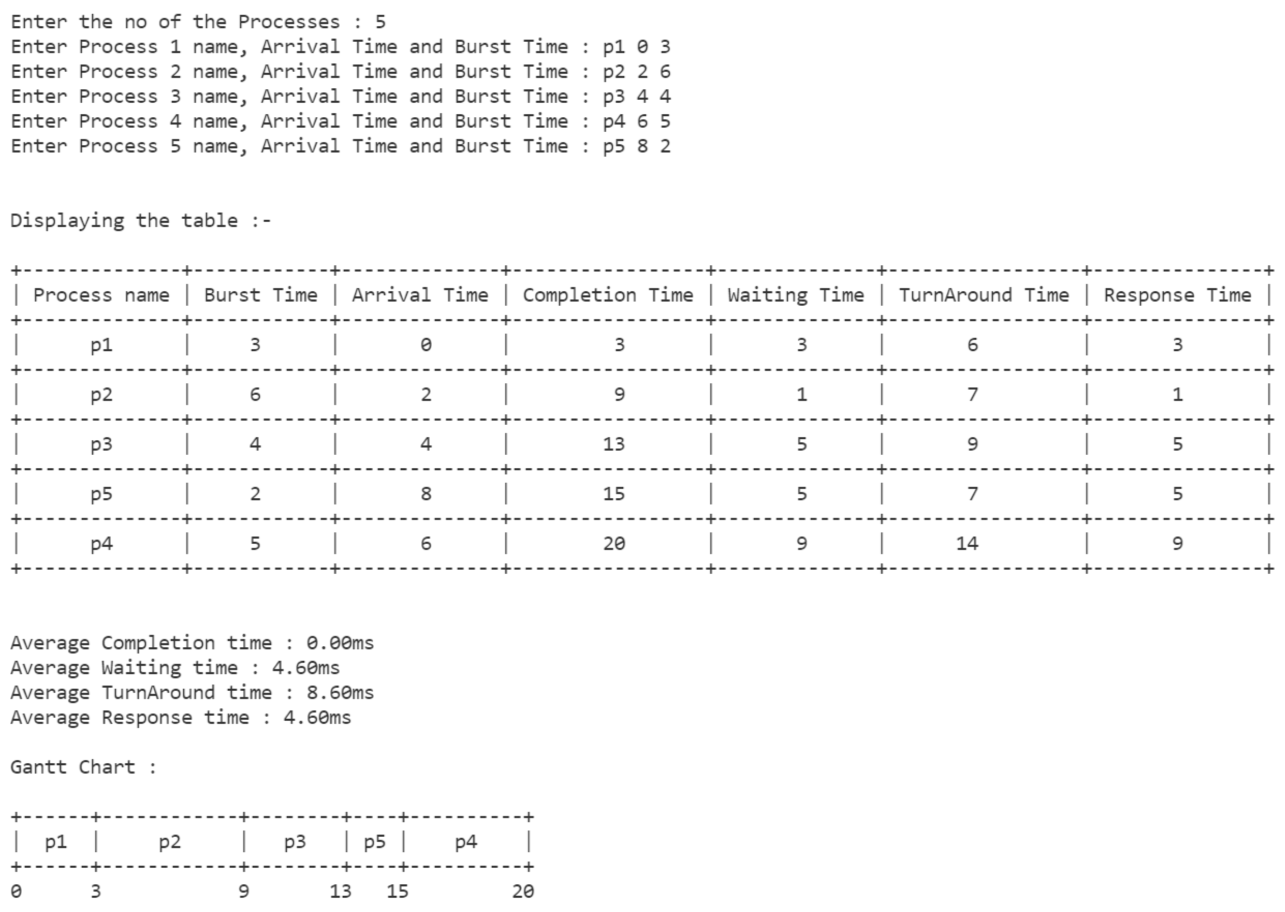
```
#include<iostream>
#include<vector>
#include<queue>
using namespace std;

struct Process{
    char Pname[3];
    int id;
    int Times[6];
};

int n;
float avgc, avgw, avgt;
vector<Process> input;
vector<Process> v;
vector<bool> visited;

bool completed(){
    for(auto b:visited) if(!b) return false;
    return true;
}

void HRRN(){
    int currentTime=0, mx=-1, idx=-1;
    float ResponseRatio;

    while(!completed()){
        for(auto p:input){
            if(!visited[p.id] && p.Times[0]<=currentTime){
                ResponseRatio = (1.00)*(currentTime - p.Times[0] +
p.Times[1])/p.Times[1];
                if(ResponseRatio>mx){
                    idx = p.id;
                    mx=ResponseRatio;
                }
            }
        }

        if(idx!=-1){
            visited[idx]=true;
            currentTime+=input[idx].Times[1];
            input[idx].Times[2]=currentTime;
            v.push_back(input[idx]);
```

```
                mx=-1;idx=-1;
            }
            else{
                currentTime++;
            }
        }
}

void calculateTimes(){
    v.front().Times[3]=0;
    float sumc=0, sumw=0, sumt=0;

    //calculating waiting time and Response Time
    int prv = v.front().Times[2];
    for(auto &p:v){
        p.Times[3] = prv - p.Times[0];
        p.Times[5] = p.Times[3];
        prv = p.Times[2];
        sumw += p.Times[3];
    }

    //calculating turn around time
    for(auto &p:v){
        p.Times[4] = p.Times[1] + p.Times[3];
        sumt += p.Times[4];
    }

    //calculating avg(s) time
    avgc = sumc/n;
    avgw = sumw/n;
    avgt = sumt/n;
}

void display(){
    cout<<"\n\nDisplaying the table :- ";

    cout<<"\n\n+--------------+------------+--------------+-----------------
+--------------+-----------------+---------------+";
    cout<<"\n| Process name | Burst Time | Arrival Time | Completion Time |
Waiting Time | TurnAround Time | Response Time |";
    cout<<"\n+--------------+------------+--------------+-----------------+-
-------------+-----------------+---------------+";

    for(auto i:v){
        printf("\n|      %s      |    %2d      |      %2d      |        %2d
      |      %2d      |      %2d         |      %2d       |"
```

```
            ,i.Pname, i.Times[1], i.Times[0], i.Times[2], i.Times[3],
i.Times[4], i.Times[5]);
    cout<<"\n+--------------+------------+--------------+-----------------+-
-------------+-----------------+---------------+";
    }

    cout<<"\n\n";
    printf("\nAverage Completion time : %.2fms", avgc);
    printf("\nAverage Waiting time : %.2fms", avgw);
    printf("\nAverage TurnAround time : %.2fms", avgt);
    printf("\nAverage Response time : %.2fms", avgw);
}

void printFree1(int x, int y, char a, char b){
    if(x==y) return;

    for(int i=0; i<2*(x-y); i++){
        cout<<a;
    }
    cout<<b;
}

void printFree2(int x, int y, int z){
    if((x-y)==z) return;

    // x-z to be printed
    int gap = x-y-z;
    for(int i=0; i<2*gap-1; i++){
        cout<<" ";
    }
    printf("%2d", x-z);
}

void printGantt(){

    cout<<"\n\nGantt Chart : ";

    // printing the upper part of Gantt Chart
    cout<<"\n\n+";
    int prv = 0;
    for(auto p:v){
        printFree1(p.Times[2]-prv, p.Times[1], '-', '+');
        for(int i=0; i<2*p.Times[1]; i++){
            cout<<"-";
        }
        cout<<"+";
```

```cpp
        prv = p.Times[2];
    }

    // Printing the middle one
    cout<<"\n|";
    prv=0;
    for(auto p:v){
        printFree1(p.Times[2]-prv, p.Times[1], ' ', '|');
        for(int i=0; i<p.Times[1]-1; i++){
            cout<<" ";
        }
        cout<<p.Pname;
        for(int i=0; i<p.Times[1]-1; i++){
            cout<<" ";
        }
        cout<<"|";
        prv = p.Times[2];
    }

    // Printing the bottom one
    cout<<"\n+";
    prv = 0;
    for(auto p:v){
        printFree1(p.Times[2]-prv, p.Times[1], '-', '+');
        for(int i=0; i<2*p.Times[1]; i++){
            cout<<"-";
        }
        cout<<"+";
        prv = p.Times[2];
    }

    // Printing the indexes of times
    cout<<"\n0";
    prv=0;
    for(auto p:v){
        printFree2(p.Times[2], prv, p.Times[1]);
        for(int i=0; i<2*p.Times[1]-1; i++){
            cout<<" ";
        }
        printf("%2d", p.Times[2]);
        prv = p.Times[2];
    }
    cout<<"\n\n";
}

int main(){
    cout<<"Enter the no of the Processes : ";
    cin>>n;

    for(int i=0; i<n; i++){
        struct Process p;
        cout<<"Enter Process "<<i+1<<" name, Arrival Time and Burst Time :
";
        cin>>p.Pname>>p.Times[0]>>p.Times[1];
        p.id=i;
        visited.push_back(false);
        input.push_back(p);
    }

    HRRN();
    calculateTimes();
    display();
    printGantt();
    return 0;
}
```

OUTPUT



```
Enter the no of the Processes : 5
Enter Process 1 name, Arrival Time and Burst Time : p1 0 3
Enter Process 2 name, Arrival Time and Burst Time : p2 2 6
Enter Process 3 name, Arrival Time and Burst Time : p3 4 4
Enter Process 4 name, Arrival Time and Burst Time : p4 6 5
Enter Process 5 name, Arrival Time and Burst Time : p5 8 2


Displaying the table :-

+--------------+------------+--------------+-----------------+--------------+-----------------+---------------+
| Process name | Burst Time | Arrival Time | Completion Time | Waiting Time | TurnAround Time | Response Time |
+--------------+------------+--------------+-----------------+--------------+-----------------+---------------+
|      p1      |     3      |      0       |        3        |      3       |        6        |       3       |
+--------------+------------+--------------+-----------------+--------------+-----------------+---------------+
|      p2      |     6      |      2       |        9        |      1       |        7        |       1       |
+--------------+------------+--------------+-----------------+--------------+-----------------+---------------+
|      p3      |     4      |      4       |       13        |      5       |        9        |       5       |
+--------------+------------+--------------+-----------------+--------------+-----------------+---------------+
|      p5      |     2      |      8       |       15        |      5       |        7        |       5       |
+--------------+------------+--------------+-----------------+--------------+-----------------+---------------+
|      p4      |     5      |      6       |       20        |      9       |       14        |       9       |
+--------------+------------+--------------+-----------------+--------------+-----------------+---------------+


Average Completion time : 0.00ms
Average Waiting time : 4.60ms
Average TurnAround time : 8.60ms
Average Response time : 4.60ms

Gantt Chart :

+------+------------+--------+----+----------+
|  p1  |     p2     |   p3   | p5 |    p4    |
+------+------------+--------+----+----------+
0      3            9        13   15         20
```

Thank you