Round Robin:

Program:

```
#include<stdio.h>

int main() {
	int N = 3;
	int TQ = 3;
	int pid, i;

	int id[N];
	int at[N];
	int bt[N];

	id[0] = 2;
	id[1] = 1;
	id[2] = 3;

	at[0] = 3;
	at[1] = 0;
	at[2] = 4;

	bt[0] = 3;
	bt[1] = 5;
	bt[2] = 4;

	int startingTime[N];
	int endingTime[N] ;
	int btt[N];
	for(int i = 0; i < N; i++) {
	startingTime[i] = -1;
	endingTime[i] = -1;
	btt[i] = bt[i];
	}
	double TTAT = 0; // total turnaround time
	double TWT = 0; // total waiting time
	int cycle = 0;

	int totalTime = 0;

	//sorting
	for (int j = 0; j < N; j++) {
	for (int k = 0; k < N - j - 1; k++) {
```

```
if (at[k] > at[k + 1]) {
        int temp = at[k];
        at[k] = at[k + 1];
        at[k + 1] = temp;

        temp = bt[k];
        bt[k] = bt[k + 1];
        bt[k + 1] = temp;

        temp = id[k];
        id[k] = id[k + 1];
        id[k + 1] = temp;
}
}
}

printf("\nProcesses:\n");
printf("ID\tAT\tBT\n");
printf("------------------\n");
for (int i = 0; i < N; i++) {
printf("%d\t", id[i]);
printf("%d\t", at[i]);
printf("%d\n", bt[i]);
totalTime += bt[i];
}
printf("\n\nTotal time required to run all processes: %d\n", totalTime);

int queue[N*TQ];
int head = 0, tail = 0;
for (int i = 0; i < N*TQ; i++) {
queue[i] = -1;
}

for (int i = 0 ; cycle < totalTime; i++) {
printf("\n\n");
if (i < N) { // not all process are arrived yet
pid = id[i] - 1;
if (TQ >= bt[pid]) {
        if (startingTime[pid] == -1) {
        startingTime[pid] = cycle;
        }

        cycle += bt[pid];
        bt[pid] = 0;
```

```
            if (endingTime[pid] == -1) {
            endingTime[pid] = cycle;
            }
            printf("completed\n");
    } else {
            if (startingTime[pid] == -1) {
            startingTime[pid] = cycle;
            }

            cycle += TQ;
            bt[pid] -= TQ;

            printf("remaining\n");
    }
    if (bt[pid] != 0) {
            queue[tail++] = pid+1;
    }
    printf("cycle=%d, pid=%d, bt=%d, rbt=%d", cycle, pid+1, btt[pid], bt[pid]);

    printf("\nqueue:\t");
    for (int i = head; i < N*TQ; i++) {
            printf("%d\t", queue[i]);
    }
    } else { // all process arrived
    printf("----");
    pid = queue[head++] - 1;
    if (TQ >= bt[pid]) {
            cycle += bt[pid];
            bt[pid] = 0;

            if(endingTime[pid] == -1) {
            endingTime[pid] = cycle;
            }
            printf("completed\n");
    } else {
            cycle += TQ;
            bt[pid] -= TQ;

            printf("remaining\n");
    }
    if (bt[pid] != 0) {
            queue[tail++] = pid+1;
    }
```

```
        printf("cycle=%d, pid=%d, bt=%d, rbt=%d", cycle, pid+1, btt[pid], bt[pid]);

        printf("\nqueue:\t");
        for (int i = head; i < N*TQ; i++) {
                printf("%d\t", queue[i]);
        }
        }
        }


printf("\n\n\n============================================================
=========================\n");
        printf("\nProcesses:\n");
        printf("ID\tAT\tBT\tST\tET\tTT\tWT\n");
        printf("-------------------------------------------------\n");
        int tt; // turnaround time
        int wt;
        for (int i = 0; i < N; i++) {
        printf("%d\t", id[i]);
        printf("%d\t", at[i]);
        printf("%d\t", btt[i]);
        printf("%d\t", startingTime[i]);
        printf("%d\t", endingTime[i]);

        tt = endingTime[i] - at[i];
        TTAT += tt;

        wt = abs(tt - btt[i]);
        TWT += wt;
        printf("%d\t", tt);
        printf("%d\n", wt);
        }

        // getting average
        TTAT = TTAT / N;
        TWT = TWT / N;
        printf("\nTotal Turnaround Time : %f", TTAT);
        printf("\nTotal Waiting Time : %f", TWT);
        printf("\nTotal time required to run all processes: %d\n", totalTime);

printf("\n=====================================================\n");
        printf("\n");
        return 0;
}
```

Output:

```
i-raj-shinobi-47@irajshinobi47-Lenovo-G50-80:~/Desktop/OSL Practicals/Assignment 3/Round Robin$ ./a.out

Processes:
ID    AT    BT
------------------
1    0    5
2    3    3
3    4    4

Total time required to run all processes: 12

remaining
cycle=3, pid=1, bt=3, rbt=2
queue:    1    -1    -1    -1    -1    -1    -1    -1    -1

completed
cycle=6, pid=2, bt=5, rbt=0
queue:    1    -1    -1    -1    -1    -1    -1    -1    -1

remaining
cycle=9, pid=3, bt=4, rbt=1
queue:    1    3    -1    -1    -1    -1    -1    -1    -1

----completed
cycle=11, pid=1, bt=3, rbt=0
queue:    3    -1    -1    -1    -1    -1    -1    -1

----completed
cycle=12, pid=3, bt=4, rbt=0
queue:    -1    -1    -1    -1    -1    -1    -1

========================================
Processes:
ID    AT    BT    ST    ET    TT    WT
--------------------------------------------------
1    0    3    0    11    11    8
2    3    5    3    6    3    2
3    4    4    6    12    8    4

Total Turnaround Time : 7.333333
Total Waiting Time : 4.666667
Total time required to run all processes: 12

========================================
```