Pseudo code for a Round Robin Scheduler that computes the total runtime (total CPU time used) of the process.

Define new variable:
PCB->CPU_time: accumulates the total runtime of this process, initialized to 0.
start_time: global variable that records the beginning of a process CPU burst just before context switch to that process.

```
scheduler() {
    /* I am a Round Robin Scheduler */
    if (called by a process that is finished) {
        // process is done
        PCB->CPU_time += readtime() - start_time; /* add last burst */
        Append(Running_Process, Zombie_Queue); /* process is now dead */
        Running_Process = Head(Ready_Queue); /* RR scheduling */
        start_time = Read_time(); /* record begin of burst */
        if (Running_Process != NIL) Context_Switch (Running_Process, DELTA);
        else IDLE(); /* start next process or idle until next interrupt */
    }
    else if (called by I/O routines) { /* process requests I/O */
        PCB->CPU_time += readtime() - start_time; /* add last burst */
        Append(Running_Process, Wait_Queue);
        Running_Process = Head(Ready_Queue); /* RR scheduling */
        start_time = Read_time(); /* record beginning of burst */
        if (Running_Process != NIL) Context_Switch (Running_Process, DELTA);
        else IDLE(); /* start next process or idle until next interrupt */
    }
    else if (called by the I/O interrupt handler) { /* I/O is done for process IO_DonePID */
        PCB->CPU_time += readtime() - start_time; /* add last burst */
        Append(Running_Process, Ready_Queue); /* find process whose I/O just completed and remove from Wait_Queue */
        Running_Process = Select(Wait_Queue, IO_Done_PID); /* Schedule it next */
        start_time = Read_time(); /* record begin of burst */
        if (Running_Process != NIL) Context_Switch (Running_Process, TAU);
        else IDLE(); /* start next process or idle until next interrupt */
    }
    else if (called by the timer interrupt handler) {
        /* current process has used up it's time quantum */
        PCB->CPU_time += readtime() - start_time; /* add last burst */
        Append(Running_Process, Ready_Queue);
        Running_Process = Head(Ready_Queue);
        start_time = Read_time(); /* record beginning of burst */
        if (Running_Process != NIL) Context_Switch (Running_Process, TAU);
        else IDLE(); /* start next process or idle until next interrupt */
    }
}
```