Student Name: Vishal Gupta

Student ID: 11814743

Roll no-A14 Group-1

Email Address: vishalgupta88872@gmail.com

GitHub Link: <https://github.com/Vishal2-gupta/os-psa>

<https://github.com/Vishal2-gupta/os-psa/blob/master/os1.c>

Code: Mention solution code assigned to you

Question: Write a program to implement priority scheduling algorithm with context switching time. Prompt to user to enter the number of processes and then enter their priority, burst time and arrival time also. Now whenever operating system preempts a process and shifts cpu control to some another process of higher priority assume that it takes 2 seconds for context switching(dispatcher latency).Form a scenario, where we can give the processes are assigned with priority where the lower integer number is higher priority and then context switch .. as the process waits the priority of the process increase at rate of one per 2 time units of wait.

Calculate waiting time and turnaround time for each process.

1. Explain the problem in terms of operating system concept? (Max 200 word)

Description:

- Priority Scheduling Algorithm: Priority scheduling is a non-preemptive algorithm and one of the most common scheduling algorithms in batch systems. Each process is assigned a priority. Process with the highest priority is to be executed first and so on.

Processes with the same priority are executed on first come first served basis. Priority can be decided based on memory requirements, time requirements or any other resource requirement.

Priority Scheduling is a method of scheduling processes that is based on priority. In this algorithm, the scheduler selects the tasks to work as per the priority.

The processes with higher priority should be carried out first, whereas jobs with equal priorities are carried out on a round-robin or FCFS basis. Priority depends upon memory requirements, time.

## Types of Priority Scheduling

Priority scheduling divided into two main types:

### Preemptive Scheduling

In Preemptive Scheduling, the tasks are mostly assigned with their priorities. Sometimes it is important to run a task with a higher priority before another lower priority task, even if the lower priority task is still running. The lower priority task holds for some time and resumes when the higher priority task finishes its execution.

### Non-Preemptive Scheduling

In this type of scheduling method, the CPU has been allocated to a specific process. The process that keeps the CPU busy, will release the CPU either by switching context or terminating. It is the only method that can be used for various hardware platforms. That's because it doesn't need special hardware (for example, a timer) like preemptive scheduling.

* Characteristics of Priority Scheduling
* A CPU algorithm that schedules processes based on priority.
* It used in Operating systems for performing batch processes.
* If two jobs having the same priority are READY, it works on a FIRST COME, FIRST SERVED basis.
* In priority scheduling, a number is assigned to each process that indicates its priority level.
* Lower the number, higher is the priority.
* In this type of scheduling algorithm, if a newer process arrives, that is having a higher priority than the currently running process, then the currently running process is preempted.

## Advantages of priority scheduling

* Easy to use scheduling method
* Processes are executed on the basis of priority so high priority does not need to wait for long which saves time
* This method provides a good mechanism where the relative important of each process may be precisely defined.
* Suitable for applications with fluctuating time and resource requirements.

## Disadvantages of priority scheduling

* If the system eventually crashes, all low priority processes get lost.
* If high priority processes take lots of CPU time, then the lower priority processes may starve and will be postponed for an indefinite time.
* This scheduling algorithm may leave some low priority processes waiting indefinitely.
* A process will be blocked when it is ready to run but has to wait for the CPU because some other process is running currently.
* If a new higher priority process keeps on coming in the ready queue, then the process which is in the waiting state may need to wait for a long duration of time.

2. Write the algorithm for proposed solution of the assigned problem.

Algorithm:

- Priority Scheduling Algorithm: In this, each process is assigned a priority and processes are executed on the basis of their priority. We can either choose to set priority of the lowest number to be the first priority or vice versa. No other process can execute until the process with the highest priority has fully executed. If two processes have same priority, then process is executed on the basis of their arrival time.

3. Calculate complexity of implemented algorithm. (Student must specify complexity of each line of code along with overall complexity)

Description (purpose of use):

- Complexity of Priority Scheduling Algorithm: O(n)

4. Explain all the constraints given in the problem. Attach the code snippet of the implemented constraint.

Code snippet:

- Priority Scheduling Algorithm: (written in C):

#include<stdio.h>

#include<conio.h>

#include<string.h>

int main()

{

    int bt[20],at[10],n,i,j,temp,p[10],st[10],ft[10],wt[10],ta[10];

    int totwt=0,totta=0;

    float awt,ata;

    char pn[10][10],t[10];

    printf("Enter the number of process:");

    scanf("%d",&n);

    for(i=0; i<n; i++)

    {

        printf("Enter process name,arrivaltime,burst time & priority:");

        scanf("%s%d%d%d",pn[i],&at[i],&bt[i],&p[i]);

    }

    for(i=0; i<n; i++)

        for(j=0; j<n; j++)

        {

            if(p[i]<p[j])

            {

                temp=p[i];

                p[i]=p[j];

                p[j]=temp;

                temp=at[i];

                at[i]=at[j];

                at[j]=temp;

                temp=bt[i];

                bt[i]=bt[j];

                bt[j]=temp;

                strcpy(t,pn[i]);

                strcpy(pn[i],pn[j]);

                strcpy(pn[j],t);

            }

        }

    for(i=0; i<n; i++)

    {

        if(i==0)

        {

            st[i]=at[i];

            wt[i]=st[i]-at[i];

            ft[i]=st[i]+bt[i];

            ta[i]=ft[i]-at[i];

        }

        else

        {

            st[i]=ft[i-1];

            wt[i]=st[i]-at[i];

            ft[i]=st[i]+bt[i];

            ta[i]=ft[i]-at[i];

        }

        totwt+=wt[i];

        totta+=ta[i];

    }

    awt=(float)totwt/n;

    ata=(float)totta/n;

    printf("\nPname\tarrivaltime\tbursttime\tpriority\twaitingtime\ttatime");

    for(i=0; i<n; i++)

        printf("\n%s\t%5d\t\t%5d\t\t%5d\t\t%5d\t\t%5d",pn[i],at[i],bt[i],p[i],wt[i],ta[i]);

    printf("\nAverage waiting time is:%f",awt);

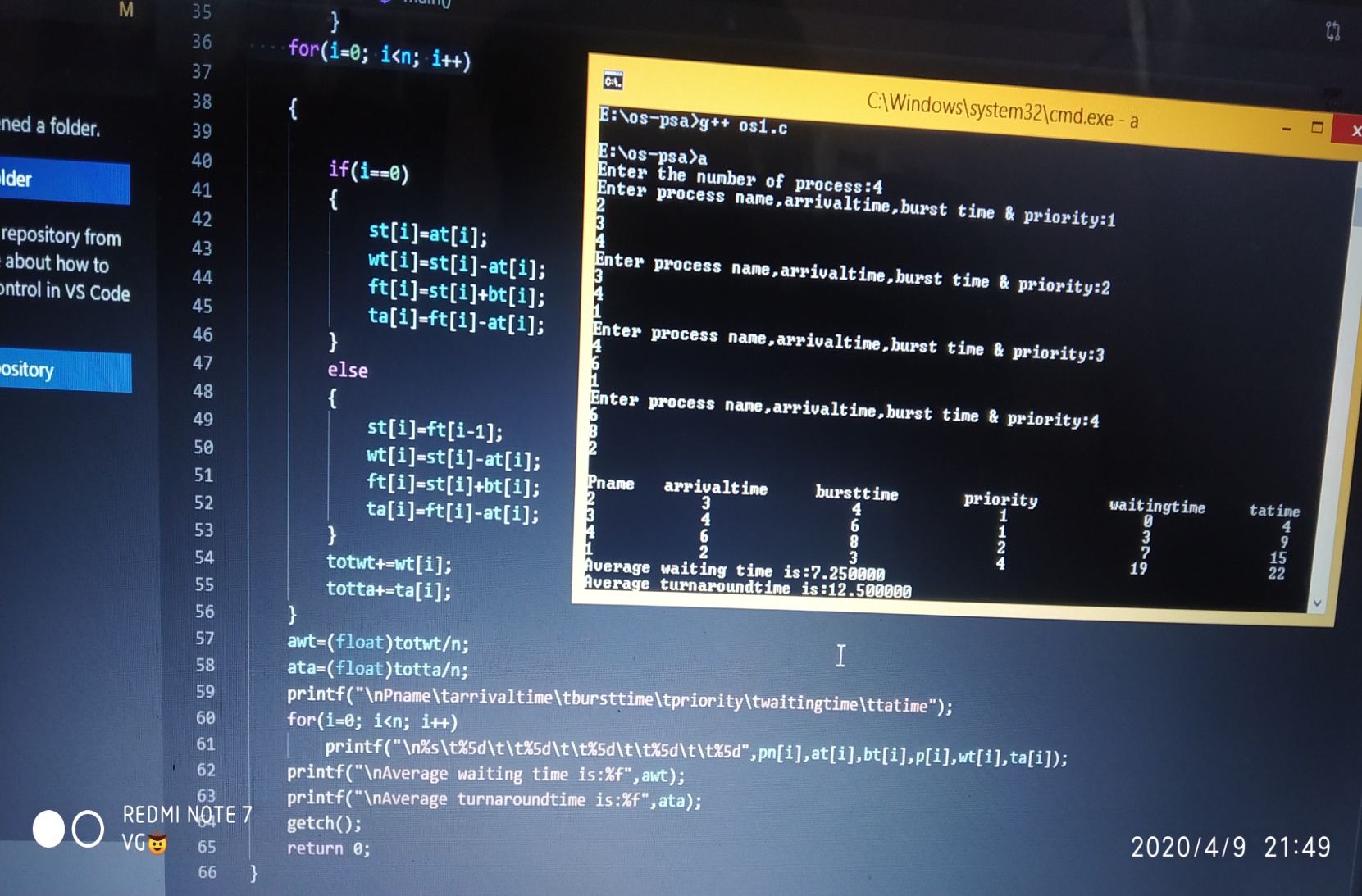
    printf("\nAverage turnaroundtime is:%f",ata);

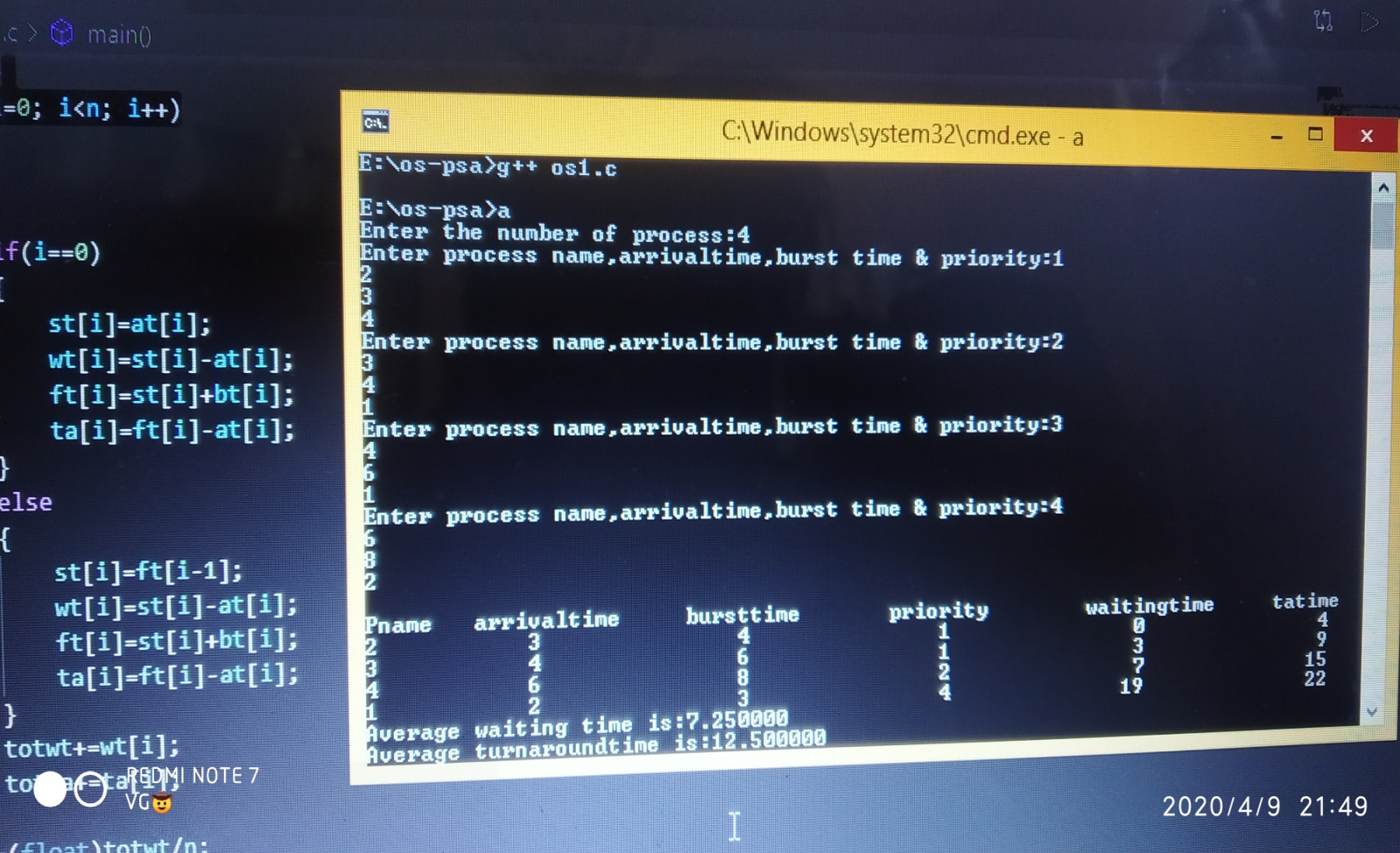
    getch();

    return 0;

}

OUTPUT:





5. If you have implemented any additional algorithm to support the solution, explain the need and usage of the same.

Description: No additional algorithm used.

6. Explain the boundary conditions of the implemented code.

Description:

- Constraints for Priority Scheduling Algorithm:

1. Process with highest priority gets preference first.

2. If two or more processes have same priority, then process if executed on their arrival time.

3. If there is already a process running and another process comes with higher priority, then the running process is pre-empted.

7. Explain all the test cases applied on the solution of assigned problem.

Description:

- Priority Scheduling Algorithm:

1. Process 1 comes -> starts getting executed

2. Next process comes with lower priority -> Waits till execution finishes

3. Another process comes with higher priority -> Pre-empts running process and starts executing

4. If two process have similar priority -> process that arrives first, gets executed

8. Have you made minimum 5 revisions of solution on GitHub?

- Yes, I have

GitHub Link: <https://github.com/Vishal2-gupta/os-psa>

<https://github.com/Vishal2-gupta/os-psa/blob/master/os1.c>

<https://github.com/Vishal2-gupta/os-psa/blob/master/os.c>