Q1:- Implement the codes of Example#01 and Example#02. Discuss what you have learned after implementing and executing these two codes. Provide snapshots of the generated outputs.

***Code o1:-***

#include<stdio.h>

int main (void)

{

  int st[20], wt[20], tat[20], i, n;

  float wt\_avg, tat\_avg;

  printf("\nEnter the number of processes: ");

  scanf("%d", &n);

  for (i = 0; i < n; i++)

  {

    printf("\nEnter Burst/Service time for process%d: ", i);

    scanf("%d", &st[i]);

  }

  wt[0] = wt\_avg = 0;

  tat[0] = tat\_avg = st[0];

  for (i = 1; i < n; i++)

  {

    wt[i] = wt[i-1] + st[i-1];

    tat[i] = tat[i-1] + st[i];

    wt\_avg = wt\_avg + wt[i];

    tat\_avg = tat\_avg + tat[i];

  }

  printf("\n PROCESS \t SERVICE TIME \t WAITING TIME \t TURNAROUND TIME\n");

  for (i = 0; i < n; i++)

  {

    printf("\n\tP%d \t\t\t%d \t\t\t\t%d \t\t\t\t%d\n", i, st[i], wt[i], tat[i]);

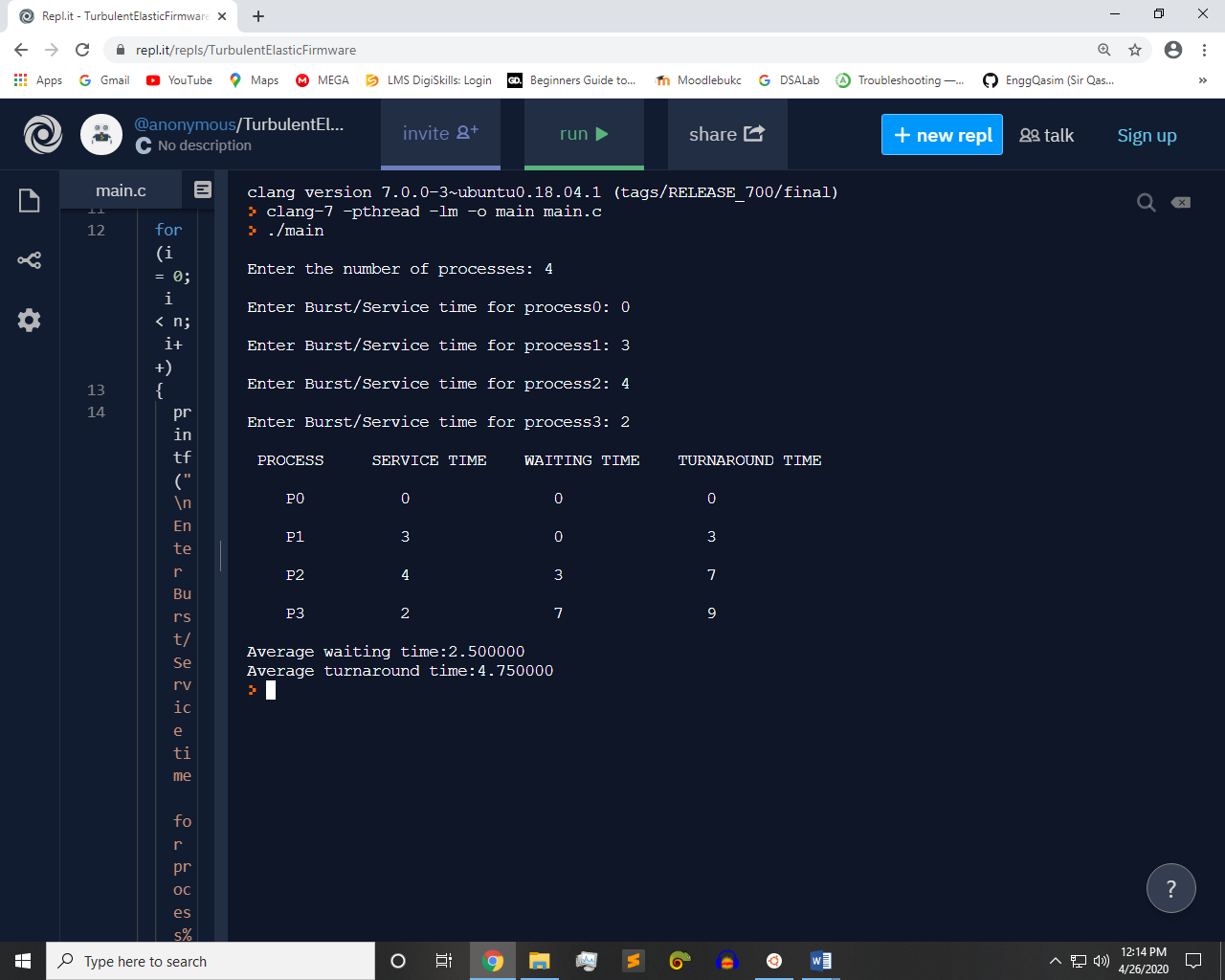
  }

  printf("\nAverage waiting time:%f ", wt\_avg/n);

  printf("\nAverage turnaround time:%f \n", tat\_avg/n);

}

***Output:-***



***Code o1:-***

#include<stdio.h>

int main (void)

{

  int at[20], st[20], wt[20], ft[20], tat[20], i, n;

  float wt\_avg, tat\_avg;

  printf("\nEnter the number of processes: ");

  scanf("%d", &n);

  for (i = 0; i < n; i++)

  {

    printf("\nEnter the Arrival time for process%d: ", i);

    scanf("%d", &at[i]);

    printf("\nEnter Burst/Service time for process%d: ", i);

    scanf("%d", &st[i]);

  }

  wt[0] = wt\_avg = 0;

  tat[0] = tat\_avg = st[0];

  ft[0] = st[0];

  for (i = 1; i < n; i++)

  {

    if (at[i] <= at[i+1])

    {

      wt[i] = wt[i-1] + st[i-1];

      ft[i] = wt[i] + st[i];

      tat[i] = ft[i] - at[i];

      wt\_avg = wt\_avg + wt[i];

      tat\_avg = tat\_avg + tat[i];

    }

  }

  printf("\n PROCESS \t SERVICE TIME \t WAITING TIME \t FINISH TIME \t TURNAROUND TIME\n");

  for (i = 0; i < n; i++)

  {

    printf("\t\t\nP%d \t\t\t\t%d \t\t\t\t%d \t\t\t\t%d \t\t\t\t%d\n", i, st[i], wt[i], ft[i], tat[i]);

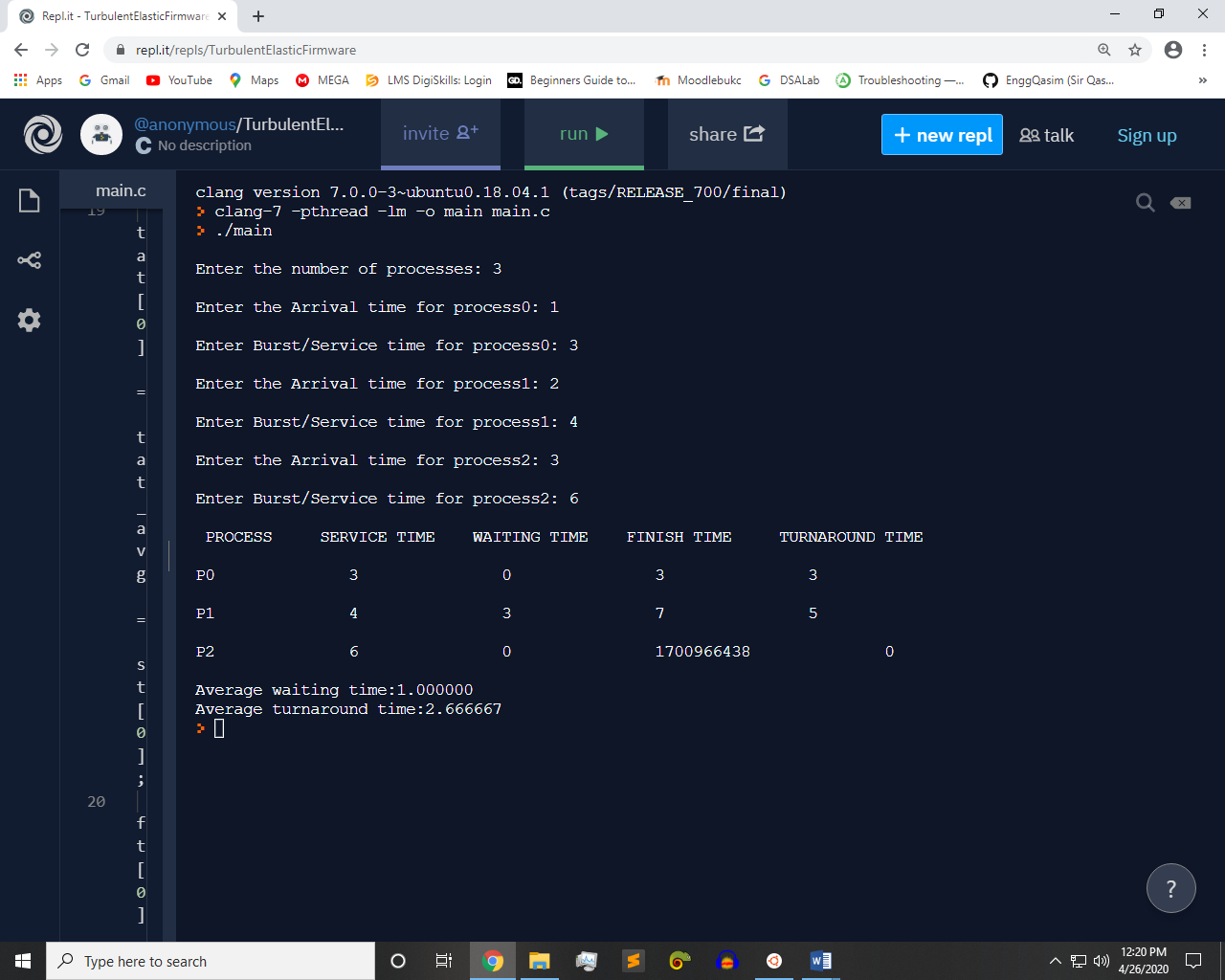
  }

  printf("\nAverage waiting time:%f ", wt\_avg/n);

  printf("\nAverage turnaround time:%f \n", tat\_avg/n);

}

***Output:-***



Q2:- The FCFS source code given above assumes that the arrival times for processes are provided in ascending order of time. Modify the code in C language for random process arrival times.

***Code :-***

#include <stdio.h>

int main(void) {

int N;

printf("Enter Number of Processes : \n");

scanf("%d",&N);

int arrival\_time[N];int service\_time[N];char names[N];

int start\_time[N];int finish\_time[N];int waiting\_time[N];

int turnaround\_time[N];int total\_time = 0;int previous\_time=0;

int i,j;

for(i=0;i<N;i++)

{

names[i]= (char)(65+i);

printf("Process %c \n",names[i]);

printf("Enter Arrival Time \n");

scanf("%d",&arrival\_time[i]);

printf("Enter Service Time \n");

scanf("%d",&service\_time[i]);

total\_time += service\_time[i];

}

start\_time[0]=0;

for(i=0;i<N;i++)

{

if(i>0)

{

for(j=i-1;j>=0;j--)

{

previous\_time += service\_time[j];

}

start\_time[i] = previous\_time;

}

waiting\_time[i] = start\_time[i] -arrival\_time[i];

finish\_time[i] = arrival\_time[i]+service\_time[i]+waiting\_time[i];

turnaround\_time[i]= service\_time[i]+waiting\_time[i];

}

int sumwait=0,sumturn=0;

for(i=0;i<N;i++)

{

printf("\n------------------Process %c ------------------\n",names[i]);

printf("Waiting Time = %d \t\t",waiting\_time[i]);

printf("Satrt Time = %d \t\t",start\_time[i]);

printf("Finish Time = %d \t\t",finish\_time[i]);

sumwait += waiting\_time[i];

printf("Turnaround Time = %d \t\t",turnaround\_time[i]);

sumturn += turnaround\_time[i];

}

double average\_wait=(double)sumwait/5;

double average\_turn=(double)sumturn/5;

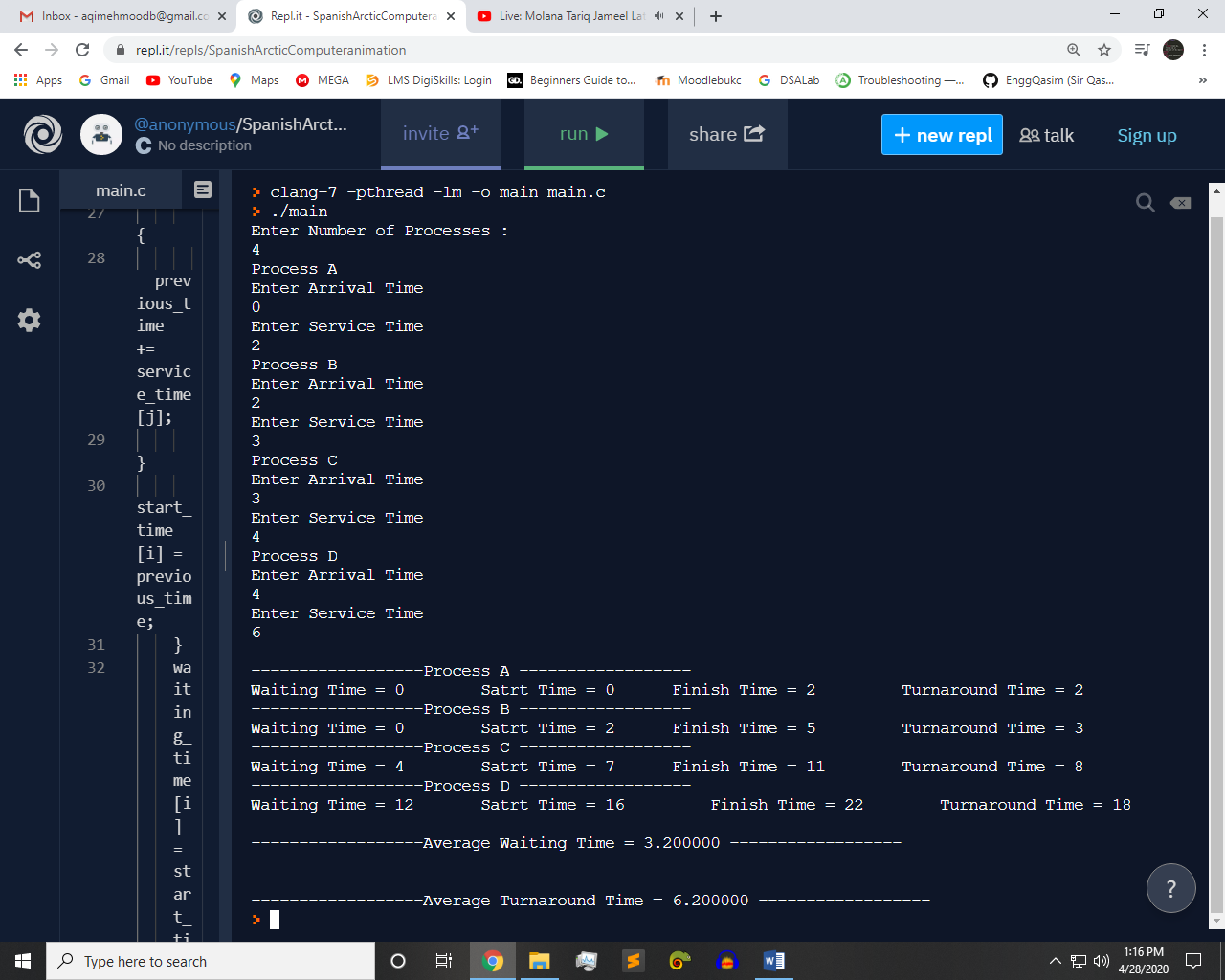
printf("\n\n------------------Average Waiting Time = %f ------------------\n",average\_wait);

printf("\n\n------------------Average Turnaround Time = %f ------------------\n",average\_turn);

return 0;

}

***Output:-***



Q3:- Implement Round Robin Algorithm in C language.

***Code :-***

#include<stdio.h>

int main()

{

int i, limit, total = 0, x, counter = 0, time\_quantum;

int wait\_time = 0, turnaround\_time = 0, arrival\_time[10], burst\_time[10], temp[10];

float average\_wait\_time, average\_turnaround\_time;

printf("\nEnter Total Number of Processes:\t");

scanf("%d", &limit);

x = limit;

for(i = 0; i < limit; i++)

{

printf("\nEnter Details of Process[%d]\n", i + 1);

printf("Arrival Time:\t");

scanf("%d", &arrival\_time[i]);

printf("Burst Time:\t");

scanf("%d", &burst\_time[i]);

temp[i] = burst\_time[i];

}

printf("\nEnter Time Quantum:\t");

scanf("%d", &time\_quantum);

printf("\nProcess\_ID \t\tBurst\_Time \t\tTurnaround\_Time \t\tWaiting Time\n");

for(total = 0, i = 0; x != 0;)

{

if(temp[i] <= time\_quantum && temp[i] > 0)

{

total = total + temp[i];

temp[i] = 0;

counter = 1;

}

else if(temp[i] > 0)

{

temp[i] = temp[i] - time\_quantum;

total = total + time\_quantum;

}

if(temp[i] == 0 && counter == 1)

{

x--;

printf("\nProcess[%d] \t\t\t%d \t\t\t %d \t\t\t\t %d", i + 1, burst\_time[i], total - arrival\_time[i], total - arrival\_time[i] - burst\_time[i]);

wait\_time = wait\_time + total - arrival\_time[i] - burst\_time[i];

turnaround\_time = turnaround\_time + total - arrival\_time[i];

counter = 0;

}

if(i == limit - 1)

{

i = 0;

}

else if(arrival\_time[i + 1] <= total)

{

i++;

}

else

{

i = 0;

}

}

average\_wait\_time = wait\_time \* 1.0 / limit;

average\_turnaround\_time = turnaround\_time \* 1.0 / limit;

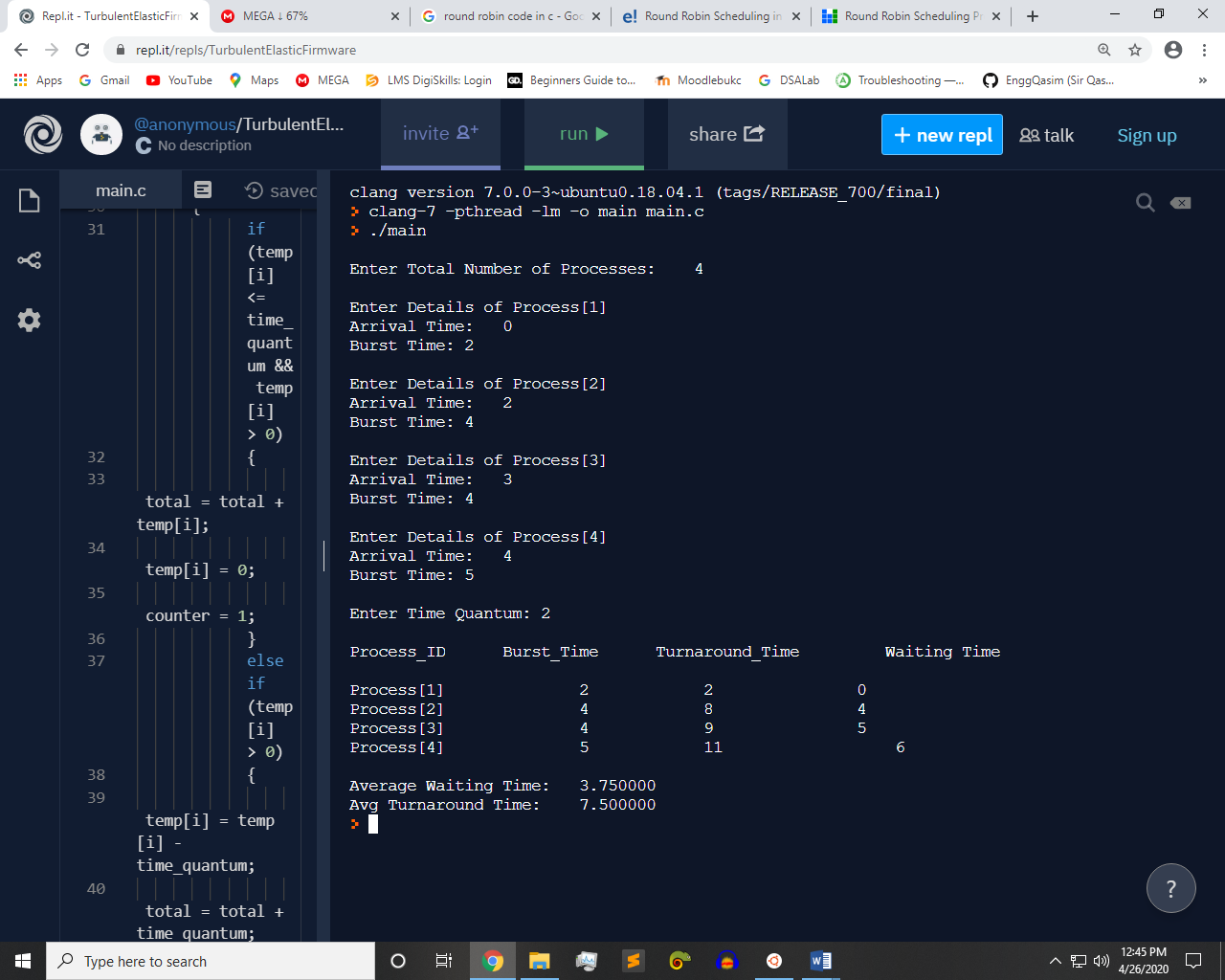
printf("\n\nAverage Waiting Time:\t%f", average\_wait\_time);

printf("\nAvg Turnaround Time:\t%f\n", average\_turnaround\_time);

return 0;

}

***Output:-***



Q4:- Implement SPN algorithm in C.

***Code :-***

#include<stdio.h>

int main()

{

int bt[20],p[20],wt[20],tat[20],i,j,n,total=0,pos,temp;

float avg\_wt,avg\_tat;

printf("Enter number of process:");

scanf("%d",&n);

printf("\nEnter Burst Time:\n");

for(i=0;i<n;i++)

{

printf("p%d:",i+1);

scanf("%d",&bt[i]);

p[i]=i+1;

}

for(i=0;i<n;i++)

{

pos=i;

for(j=i+1;j<n;j++)

{

if(bt[j]<bt[pos])

pos=j;

}

temp=bt[i];

bt[i]=bt[pos];

bt[pos]=temp;

temp=p[i];

p[i]=p[pos];

p[pos]=temp;

}

wt[0]=0;

for(i=1;i<n;i++)

{

wt[i]=0;

for(j=0;j<i;j++)

wt[i]+=bt[j];

total+=wt[i];

}

avg\_wt=(float)total/n;

total=0;

printf("\nProcess\t\tBurst\_Time\t\tWaiting\_Time \t\tTurnaround\_Time");

for(i=0;i<n;i++)

{

tat[i]=bt[i]+wt[i];

total+=tat[i];

printf("\n\tp%d\t\t\t\t%d\t\t\t%d\t\t\t\t\t%d",p[i],bt[i],wt[i],tat[i]);

}

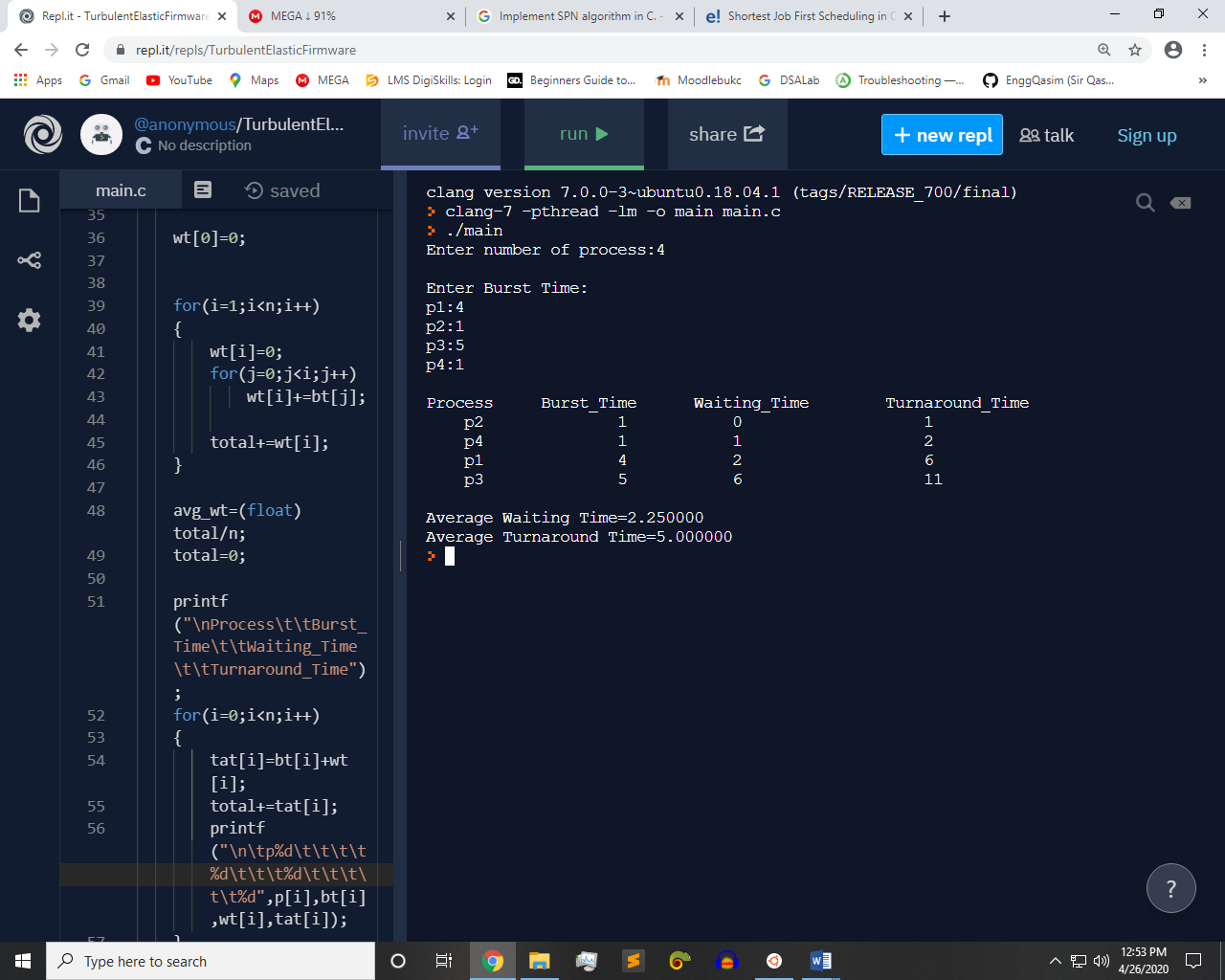
avg\_tat=(float)total/n;

printf("\n\nAverage Waiting Time=%f",avg\_wt);

printf("\nAverage Turnaround Time=%f\n",avg\_tat);

}

***Output:-***



~~~~~~\*\*/**THE END**/\*\*~~~~~~