Name - Disha Verma
Course - Bsc It   Section - 'B'
Student ID - 20052122
Roll no - 2023115
Subject - OS Practical

(1) Tim owns a pizza - - - - - - - - - - - time

```
# include <Stdio.h>
unsigned int Heap [100001], Index [100001], Position [100001],
  size = 0;
unsigned int Temp [100001], Temp [100001];
unsigned int Arr_Time [100001], Cook-Time [100001], Num;
void merge (int Low, int Mid, int High)
{
  int i = Low, j = Mid + 1, k = 0;
  while (i < Mid && j <= High)
  {
   If(Arr -Time[i]<= Arr_Time [j])
   {
     Temp[K] = Arr _Time [i];
     Temp1 [K] = Cook - Time[i];
     i++;
     K++;
   }
  else
  { Temp [K] = Arr_Time [j];
```

Dishaverma
22/06/21

```
Temp1[K] = cook - Time [j];
  j++;
  k++;
}
}
if (i<Mid)
{
  int I;
for(I = i , I< = Mid; I++)
{Temp[k] = Arr-Time[I] ,     Temp1[K] = cook -Time [I]; k++;}
 }
else if (j<= High)
{ int I ;
 for (I = j ; I<= High ; I++)
 {Temp [k] = Arr _Time [I];      Temp [k] = Cook _Time[I];
                                          k++; }

  }
  k=0;
for (i= Low ; i<=High; i++)
{
  Arr_Time [i] = Temp [k];
 cook - Time [i] = Temp 1[k];
   k++;
   }
}
void divide (int Low, int High)
{  if (Low < High>
  {  int Mid = (Low + High)/2;
       divide (Low , Mid);
       merge (Low, Mid, High);
   }
```

Disha Verma
22/06/21

```
}
void Insert ( int Node, unsigned int Value)
{
   int s;
   if (Position [Node] = = 0)
   {
    Heap [++ Size] = Value;
   Index [Size] = Node;
    Position [Node] = Size;
    S = Size;
   }
 else
 {
   Heap [Position [Node]] = Value;
      S = Position [Node];
    }
   while (S! = 1)
   {
       if (Heap [S/2] > Heap [S])
       {
         int t = Heap [S/2];
         Heap[S /2] = Heap [S];

         t= Index [S/2];

         Index [S/2 ] = Index [S];

         Index [S] = t;

         Position [Index [S/2]] = S/2;

         Position [Index [S] ] = S;

     }
   else
   break;
  } S = S/2;
}
```

Dishavarma
22/06/21

```
int Extract_Min ( )
{
  int N = Index [ 1 ] ;
   int S = 1 ;

  Position [ N ] = -1 ;
  Index [1 ]  = Index [ Size ] ;
  Position [ Index [ Size ] ] = 1 ;
  Heap [1 ] = Heap [ Size - - 1 ] ;
 while (1)
{   int T ;
  if (Heap [S * 2 ] < Heap [S] && S* 2 <= Size || Heap [ S * 2+1]
 < Heap  [S] && S * 2+1  <= Size)
 {
   if (Heap [ S * 2 ] < Heap [ S * 2+1 ])
  T = S * 2 ;
  else
   T = S * 2 + 1 ;

  int  t = Heap[T] ;
  Heap[T] = Heap [S] ;
   Heap [S ] = t ;
    t = Index[T ] ;
    Index [T ]  = Index [S ] ;
     Index [S ] = t ;

     Position [Index [T]] = T ;
     Position [Index [S]] = S ;
 }
  else
```

Disha Verma
22/06/21

```
break ;
 S= T;
}
return N ;
}
void Int (int N)
{
  int i ;
 for (i = 1; i<N;=i++ )
  {
    Position [i] = 0 ;
   Index [i] = 0;
   Heap [i]  =1000000001] ;
   }
   size = N;
 }
 int main ( )
{
   int A_T, C-T, i = 1 ;
   long long wait_Time = 0, Time = 0 ;
   Scanf ("%d", &Num);
for (i=0; i<Num ; i++ )
Scanf ("%lu%lu , & Arr_Time[i], & Cook - Time [i];
divide (0, Num-1);
 for (i = Num; i> =1 ; i - -1 )
{
   Arr_Time [i] = Arr_Time [i-1];
 Cook_Time [i] = Cook_Time [i-1];
}, insert (1, Cook_Time [1];
i = 2;
```

22/06/21

```
while (i<=Num&& Arr_Time[i]==Arr_Time[i])
{
 Insert (i, Cook_Time [i]);
  i++;
 }
while (Size!=0)
{
   int I = Extract_Min ( );
  if (Time > Arr [I])
  {
   Wait_Time += Time - Arr_Time [I] + Cook_Time [I];
   Time += Cook_Time [I];
  }
else
 {
    Time= Arr_Time [I] + Cook_Time [I];
    wait_Time += Cook_Time[I];
   }
  J = i;
  while (i<=Num && Arr_Time [i] <= Time)
 {
  insert ( i, Cook_Time [i]);
   i++;
   }
   if (J==i && i<=Num)
   { Insert (i, Cook_Time [i]);

     i++;
  while (i<=Num && ~~Arr~~ Arr_Time [i] = ~~Arr~~ Arr_Time
                                                     [I]);
 { Insert (i, Cook_Time [i]);
    i++;
  }
```

Dishaverma
22/06/21

```
if (I==P && i<=Num)
 {
   Insert (i, Cook_Time [i]);
    i++;
   }
  }
}
wait-Time = wait-Time /Num;
printf ("%ld", wait-Time);
 return 0;
}
```

22/06/21

Name- Disha Verma

Course- BSc It

Student ID- 20052122

Roll no- 2023115

2

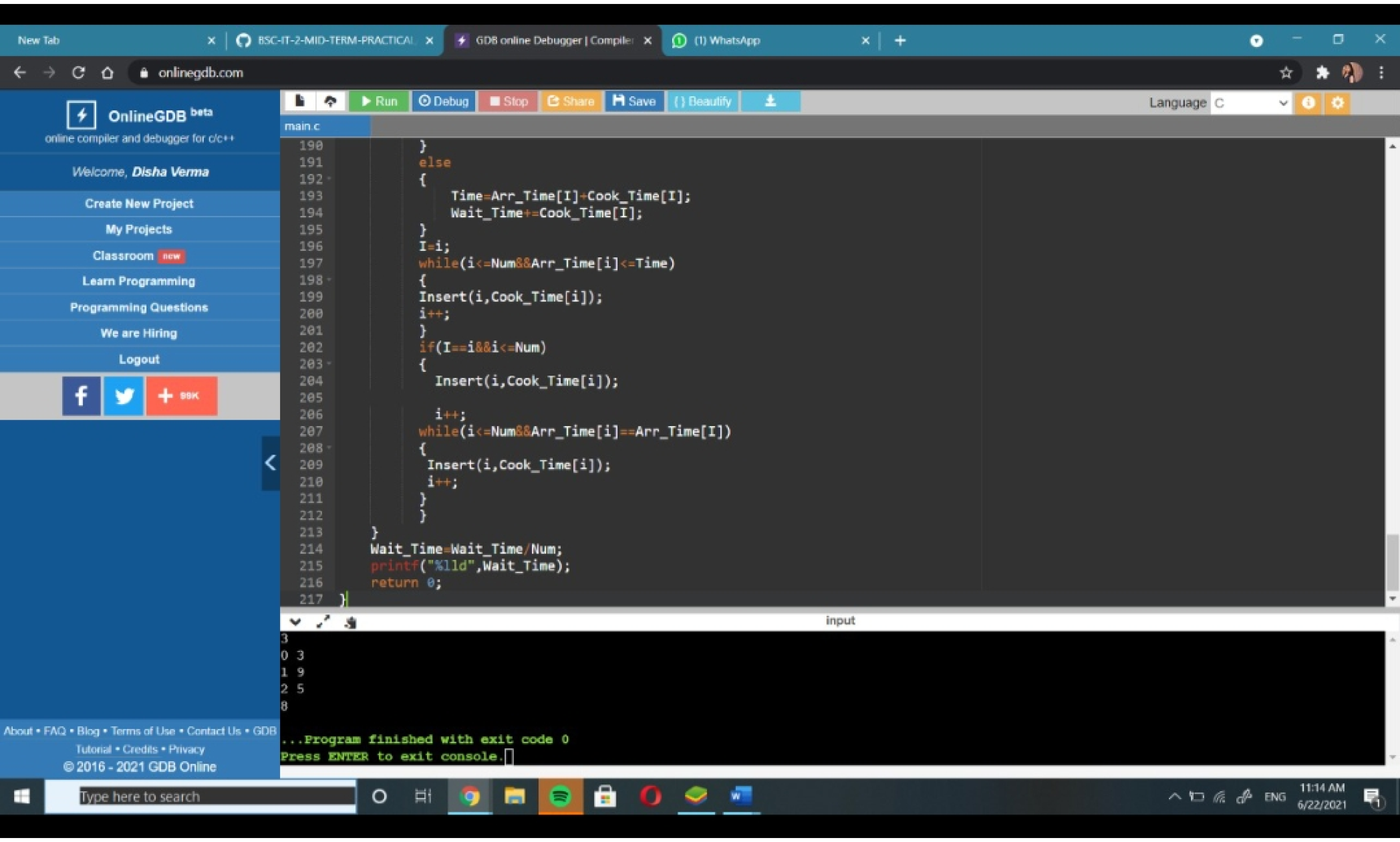
```
# include <stdio.h>
int main()
{ int bt[20], p[20], wt [20], tat [20], i, j, n,
   total = 0, pos , temp;
 float avg-wt, avg-tat;
 printf ("Enter number of processes:");
 scanf ("%d", &n);
 printf ("\nEnter Burst Time:\n");
  for (i=0; i<n; i++){
 {
   printf ("P%d :", i+1);
   scanf (" %d", & bt[i]);
   p[i] = i + 1;
 }
  for (i = 0; i<n; i++)
   { if (bt [j] < bt [pos])
       pos = j;
    }
   temp = bt[i];
   bt[i]= bt[pos];
     bt [pos]= temp;
     temp = p[i];
      bt [i] = bt [pos];
```

Disha Verma
22/06/21

```
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 Burst Time \waiting Time \ Turnaround
    Time");
  for (i=0; i<n; i++)
  {
    total [i] = bt [i] +wt [i],
     total += tat [i] ;
     printf ("\np%d\t\t %d \t\t %d\t\t\t %d",
        p[i], bt [i], wt [i], tat [i]);
   }

     avg_tat = (float) total /n;
    printf ("\n\n Average waiting Time = %f", avg_wt);
    printf ("\n average Turnaround time = %f\n", avg tat);
}
```

22/06/21

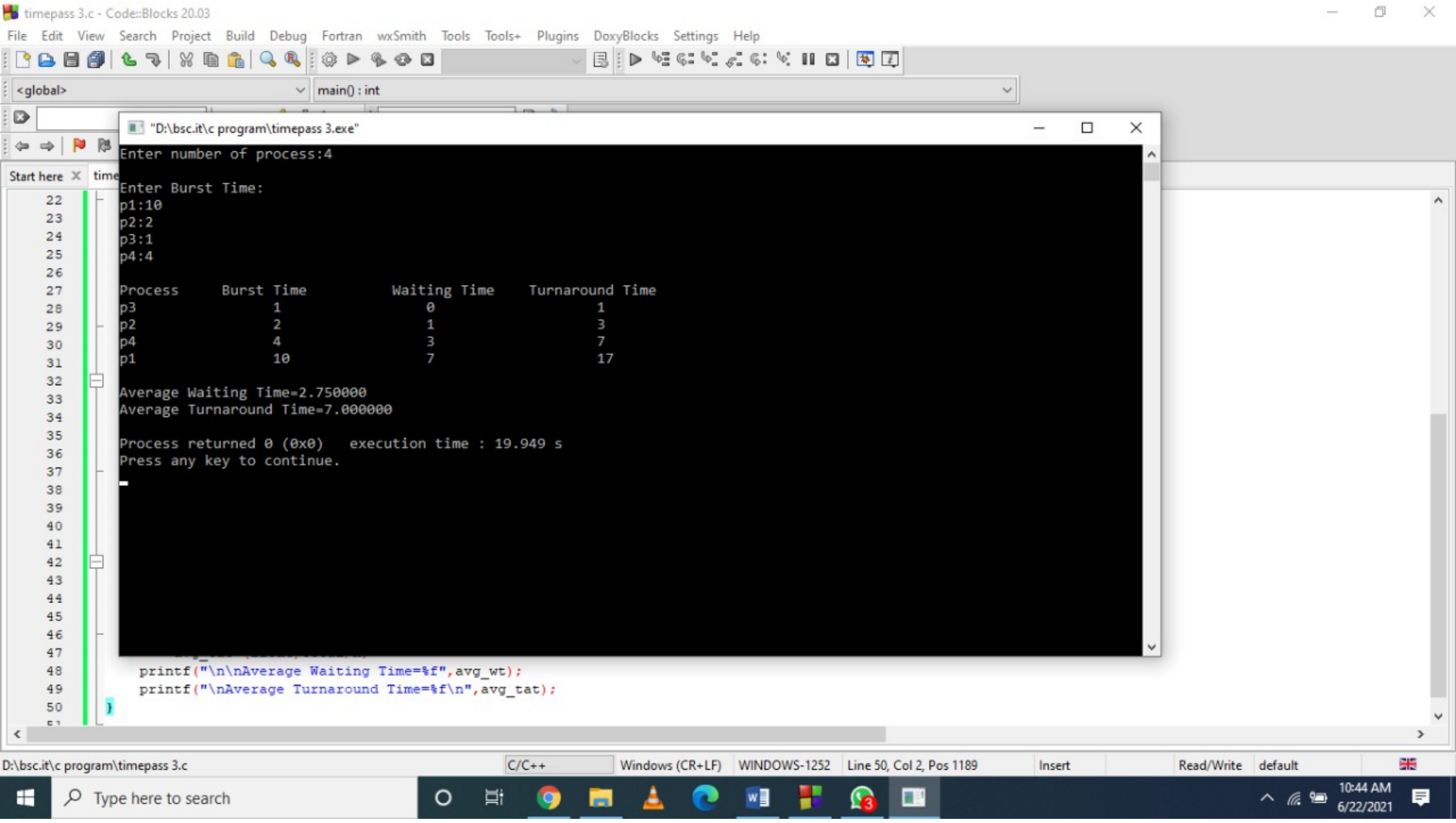


```c
          }
          else
          {
              Time=Arr_Time[I]+Cook_Time[I];
              Wait_Time+=Cook_Time[I];
          }
          I=i;
          while(i<=Num&&Arr_Time[i]<=Time)
          {
          Insert(i,Cook_Time[i]);
          i++;
          }
          if(I==i&&i<=Num)
          {
            Insert(i,Cook_Time[i]);

            i++;
          while(i<=Num&&Arr_Time[i]==Arr_Time[I])
          {
           Insert(i,Cook_Time[i]);
           i++;
          }
          }
    }
    Wait_Time=Wait_Time/Num;
    printf("%lld",Wait_Time);
    return 0;
}
```

```
3
0 3
1 9
2 5
8

...Program finished with exit code 0
Press ENTER to exit console.
```


timepass 3.c - Code::Blocks 20.03
File Edit View Search Project Build Debug Fortran wxSmith Tools Tools+ Plugins DoxyBlocks Settings Help
<global>
main() : int
Start here
time
D:\bsc.it\c program\timepass 3.c
C/C++
Windows (CR+LF)
WINDOWS-1252
Line 50, Col 2, Pos 1189
Insert
Read/Write
default
Type here to search
10:44 AM
6/22/2021

"D:\bsc.it\c program\timepass 3.exe"

```
Enter number of process:4

Enter Burst Time:
p1:10
p2:2
p3:1
p4:4

Process     Burst Time          Waiting Time    Turnaround Time
p3                1                 0                   1
p2                2                 1                   3
p4                4                 3                   7
p1                10                7                   17

Average Waiting Time=2.750000
Average Turnaround Time=7.000000

Process returned 0 (0x0)   execution time : 19.949 s
Press any key to continue.
```

```
    printf("\n\nAverage Waiting Time=%f",avg_wt);
    printf("\nAverage Turnaround Time=%f\n",avg_tat);
}
```