Name - Harsh Rawat
Rollno - 202305 7
Course - BSC IT
Semester - 2

Subject - Operating system

Q 1 → # include <stdio.h>

unsigned int
Heap [100001], Index [100001], Position
[100001], size = 0;
unsigned int
Temp [100001], Temp 1 [100001];
unsigned int
~~Temp~~
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];
Temp 1 [k] = Cook_ Time [i];
i++;
k++;

}
else
{

DATE = 22/06/21

SIGN - Harsh

```
Temp[k] = Arr_Time[i];
Temp1[k] = Cook_Time[j];
j++;
k++;

}
}
if (i<=Mid)
{
int l;
for (l=i; l<=mid; l++)
{ Temp[k] = Arr_Time(l);
Temp1[k] = Cook_Time(l); k++;}
}
else if (j<=High)
{
int l;
for(l=j; l<=High; l++)
{Temp[k] = Arr_Time[l];
Temp1[k] = Cook_Time[l]; k++;}
}
k=0;
for (i=Low; i<=High; i++)
{
Arr_Time(i) = Temp[k];
Cook_Time[i] = Temp1[k];
k++;
}
}
```

DATE - 22/6/21
SIGN - Hemu

```
Void divide (int low, int High)
{
  if (low < High)
  { int mid = (low + High)/2;
    divide ( low, mid);
    divide ( mid+1, High);
    Merge (low, mid, High);
  }
}
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])
  {
```

DATE = 22/6/14

SIGN- Harsh

```
int t = Heap[s/2];
Heap[s/2] = Heap[s];
Heap[s] = t;

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;

}

}
int Extract-Min()
{
int N = Index[1];
int s = 1;
// printf("%d\n", Heap[1]);
position(N) = -1;
Index[1] = Index[size];
position[Index[size]] = 1;
Heap[1] = Heap[size--];
while(1)
{
int T;
if(Heap[s*2] < Heap[s] && s*2 <= Size)
```

DATE - 22/6/22

SIUN - Hari

```
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
  break;
  s = T;

}
  return N;

}
Void Init(int N)
{ int i;
for(i = 1; i <= N; i++)
{
```

DATE = 22/6/21

SILVI-Hash

```
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);
  // init(N);
for (i=0; i< Num; i++)
Scanf("%d %d", & All_Time[i], & Cost_Ti
me[i]);
divide(0, Num-1);
for (i= Num; i>=1; i--)
{
    All_Time[i] = Arr_Time[i-1];
Cost_Time[i] = Cost_Time[i-1];
// printf("%d %d
%d %d\n", Arr_Time[i], Cost_Time[i]);

}
Insert (1, Cost_Time[1]);

i=2;
While (i<= Num && All_Time[i] == Arr_Time[1])
  {
    Insert (i, Cost_Time[i]);
    i++;
    }
```

DATE = 22/6/21

SIGN = Harsh

```
While (size != 0)
{
  int i = Extract_Min();
  if (Time > Arr_Time[i])
  {
   Wait_Time += Time - Arr_Time[i]
+ Lost_Time[i];
          Time += Lost_Time[i];
          // Printf ("%d %d %d %d\n", i, Time, wait_Time);
  }
  else
  {
   Time = Arr_Time[i] + Lost_Time[i];
                  Wait_Time += Lost_Time[i];

    }
    // printf ("%d %d %d %d
\n", i, Time, Wait_Time);
      i = i;
While (i<= Num && Arr_Time[i] <= Time)
    {
    insert (i, Lost_Time[i]);
    i++;
  }
  if (i==i && i<= Num) // No job is before cull_time
    {
     Insert (i, Lost_Time[i]);
      i++;
While (i<= Num && All_Time[i] == All_Time[i])
    {
```

DATE = 22/06/21

SIGN-Harsh

```
Insert (i, Work_Time[i]);
        i++;
    }
    }
}
Wait_Time = Wait_Time / Num;
printf("%.11d", Wait_Time);
//system("Pause");
return 0;
}
```

DATE = 22/06/21

SIVAN - Harish

```
{
Insert(i,Cook_Time[i]);
i++;
}
if(I==i&&i<=Num)//No job is before curr_time
{
  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);
// system("pause");
```