Name - Ayushi
Roll no. - 2023044
Course - BSC IT
Section - 'B'
~~Fathers~~ name - Sataya Narayan
Campus - D.Dun
Date - 22/06/2021

1 = Source Code :

```
# include <stdio.h>
  unsigned int Heap[100001], Index[100001], Position[10000],
   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];
  Temp[K]= Cook_Time[i];
  i++;
  k++;
```

Ayushi
22/06/2021

```
}
else
{
 Temp[K]= Arr_Time[j];
 Temp1[K] = Cook_Time[j];
  j++;
  K++;
  }
 }
 if (i<=Mid)
  {
   int I;
   for (I=i; I<=Mid; I++)
   { Temp[K]=Arr_Time[I];
     Temp[K]= Code_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++)
  {
```

Ayushi
22/06/2021

```
if (Heap [s/2] > Heap [s])
 {
  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;
  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 || Heap
     [s*2+1] < Heap [s] && s*2+1);
```

[illegible]
22/00/2021

```
Arr_Time [i] = Temp [K];
Cook_Time [i] = Temp [K];
 K++;
 }
}
 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)
```

[illegible]
24/06/20

```
{
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 = 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++)
 {
  Position[i]=0;
  Index[i]=0;
 Heap[i]= 1000000001;
 }
 Size=N;
```

Ayushi
22/06/2021

```
}
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("%u%u", & Arr_Time [i], & Cook_Time[i]);
 divide (0, Num-1);
 for (i= Num ; i>=1; i--)
  {
   Arr_Time[i]= Arr_Time [i-1];
  Cook_Time[i] = Cook_Time [i-1];
  }
  Insert (1, Cook_Time [i]);
  i=2;
  while (i<Num && Arr_Time [i]== Arr_Time[i])
  {
   Insert (1, Cook_Time[i]);
   i++;
   }
 while (size !=0)
  { int I= Extract_Min();
   if (Time> Arr_Time [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];
 }
```

Ayushi
22/06/2021