Name – Himani Taneja
Student ID - - 20052047
University Roll no- 2023058

```
# include < stdio.h >
unsigned int
Heap [100001], Index[100001], Position [100001],
Size = 0;
Unsigned int Temp [100001], Temp [100001], Nom
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 [i];
i++;
k++;

}
else
{
Temp [k] = Arr_Time [j];
Temp 1 [k] = Cook_Time [j];
j++;
k ++;
}
}
if (i<=Mid)
}
```

Himani

Himani

```
int i;
for (i=1; i<=High; i++)
{
Arr_Time[i] = Temp[k];
Cook_Time[i] = Temp1[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)
{
```

Himani

```
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];
    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 ||
```

Himani

```
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;
  }
```

Himani

```
void Init (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);
 // int (N);
for (i=0; i<Num; i++)
Scanf ("%d %d", &Arr_Time [i], &Look_Time [i]);
divide (0, Num-1);
for [i= Num; i >= 1; i--)
{
```

Himani

```
        Arr_Time[i] = Arr_Time[i-1];
    Look_Time[i] = Look_Time[i-1];
    // printf("%d %d\n", Arr_Time[i], Look_
        Time[i]);
}
Insert(1, Look_Time[1]);
i = 2;
while (i <= Num && Arr_Time[i] == Arr_
    Time[0])
{
Insert(i, Look_Time[i]);
i++;
}
while (size != 0)
{
int i = Extract_Min();
if(Time > Arr_Time[i])
}
wait_Time += Time - Arr_Time[i] + Look_
    Time[i];
Time += Look_Time[i];
// Print("%d %d %d\n", i, Time, wait_
    Time);
}
```

Linear

```
else
{
                                        Himani
Time= Arr_Time[i]+ Look_Time[i];
Wait_Time+ = Look_Time[i];
}
// print ("%d %d %d %d \n", i, Time,
  Wait_Time);
  i = i ;
  while (i<=Num && Arr_Time[i] <= Time)
  {
Insert (i, Look_Time[i]);
i++;
}
if (i== i && i<= Num) // No job is before
 curr_time
{
Insert (i, Look_Time[i]);

 i++;
while (i<= Num && Arr_Time[i] == Arr_Time[i])
{
```

```
        Insert (i, cook Time[i]);
        i++;
      }
    }
}
wait_Time = wait_Time / Num;
Printf (" %d", wait_Time);
// System ("Pause");
  return 0;
}
```

Himani