DATE:
PAGE NO

Swapnil Kumar
B.SC. I.T
20051110

Sunopril DATE: / / B+
PAGE NO.

```
#include<Stdio.h>
unsigned int Heap[100001], Index[100001], Position
[100001], Size=0;
unsigned int Temp[100001], Temp1[100001];
unsigned int Arr_Time[100001], Cook_Time[10001],
 num;
void merge(int low, ind mid, int high)
{
 int i= low, j=mid+1, K=0;

 while(i<=mid&&j<=high)
 {
 if(Arr_Time[i]< [j])
 {
  Temp[K]=Arr_Time[i];
  Temp1[K]=Cook_Time[i];
  i++;
  K++;
}
else
{
 Temp[K]=Arr_Time[j];
 Temp1[K]=Cook_Time[j];
  j++;
 K++;
}
}
if(i<=mid)
{
 int I;
```

SUCCESS

```
for (I = i ; I <= Mid ; I ++ )
{Temp (K] = Arr_Time[I] ; Temp1 [K] cook_Tim
                              [I]; K++; }
}
else if (j <= High )
{
 int I ;
for (I = j ; I <= High ; I ++ )
{Temp [K] = Arr_Time [I]; Temp1 [K] = cook
                      _time [I]; K++; }
}
K = 0;
for (i = low ; 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 ) ;
 }
}
```

Swapnil

DATE: __/__/__ B+
PAGE NO.______

```
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]
            Heap [S] = t ;

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

            Position [Order [S/2] ] = S/2 ;
            Position [Index [S] ] = S ;
}
```

Sumapril

```
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 || Heap [S*2+] < Heap [S] && S*
          2+1 <= Size)
{
    if (Heap [S*2] < Heap [S*2+1])
    T = S*2 ;
    else
    T = S*2+1 ;
```

Swapnil

```
        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++)
  {
    Position [i] = 0 ;
    Index [i] = 0 ;
    Heap [i] = 100000001 ;
  }
   Size = N ;
}
  Size = N
}
```

Swapnil

DATE: / /
PAGE NO.

B+

```
int main ()
{
    int A_T, C_T, i = 1;

    long long wait_time = 0, Time = 0 ;
   Scanf ("%d", &Num) ;
   // int (N) ;
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 ] ;
```