Name → Nimeesha Agrawal

Course - BSc IT [Sem 2]

Date

Page No.

Question -1:-

```
#include <stdio.h>
unsigned int
Heap [100001], Index [100001], Position
[100001], size = 0;
unsigned int
Temp [10000], Temp1 [10000];
unsigned int
Arr_Time [100001], Cook_Time
    [10001], 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];
Temp [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];
    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)
        {
```

Date
Page No

```
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])
{
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]] = s;
    }
    else
      break;
      s = s/2;
      }
    }

int Extract_Min()
  {
   int M = Index[2];
     int s = 1;
     int T;
   {
  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];
```

```
Position[Index[T]] = T;
Position[Index[S]] = S;
}
else
    break;
    S = S/2;
}
}
int Extract_Min()
{
int M = Index[1];
int S = 1;
while(1)
{
int T;
}
if (Heap[S*2] < Heap[S*2+1])
T = S*2
else
T = S*2+1;

int t = Heap[T];
Heap[T] = Heap[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;
    }
    int main ()
{
  int A_T, C_T, i=1;

  divide (0, Num -1);
  for(i=Num; i>=1;i--)
{
Arr_Time [i]=Arr_Time [i-1];
Cook_Time [i]=Cook_Time [i-1];
//printf("%u
%u \n", Arr_Time [i], Cook_Time(i));
}
  Insert (1, Cook_Time[1]);

  i=2;
```

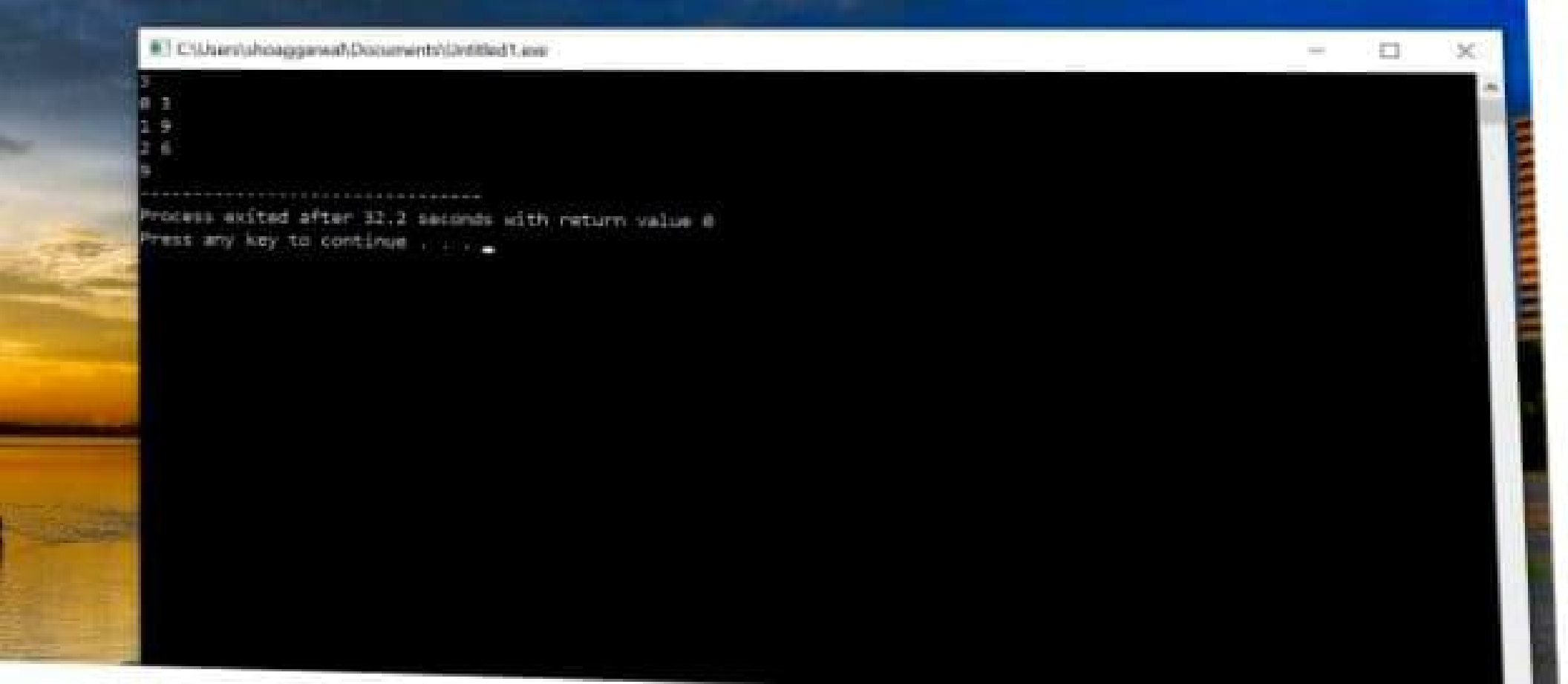
```
while(i<=Num && Arr_Time[i]
    == Arr_Time
  e[i])
  {
  Insert(i, Cook_Time[i]);
    i++;
    }
    while ( Size != 0)
  {
   int l= Extract_Min();
  if (Time > Arr_Time[l])
  Time += Cook_Time[l];
    }
    else
    {
     Time = Arr_Time[l]
  + Cook_Time[l];
  }
while(i<= Num && Arr_Time[i]<=
                    Time)
  {
  Insert( i, Cook_Time[i]);
    i++;
    }
if (l== i && i<=Num) //No job is
    before curr_time
```

Date
Page No

```
{
    Insert(i, Cook_Time[i]);
    i++;
while(i<NumOfArr_Time[i]==
        Arr_Time[i])
    {
    Insert(i, Cook_Time[i]);
        i++;
    }
    }
}
Wait_Time = WaitTime/Num;
printf("%d", Wait_Time);
// system("pause")
    return 0;
}
```

— x —

Khureeda


C:\Users\shoaggarwal\Documents\Untitled1.exe
3
0 1
1 9
2 6
5
Process exited after 31.2 seconds with return value 0
Press any key to continue . . .