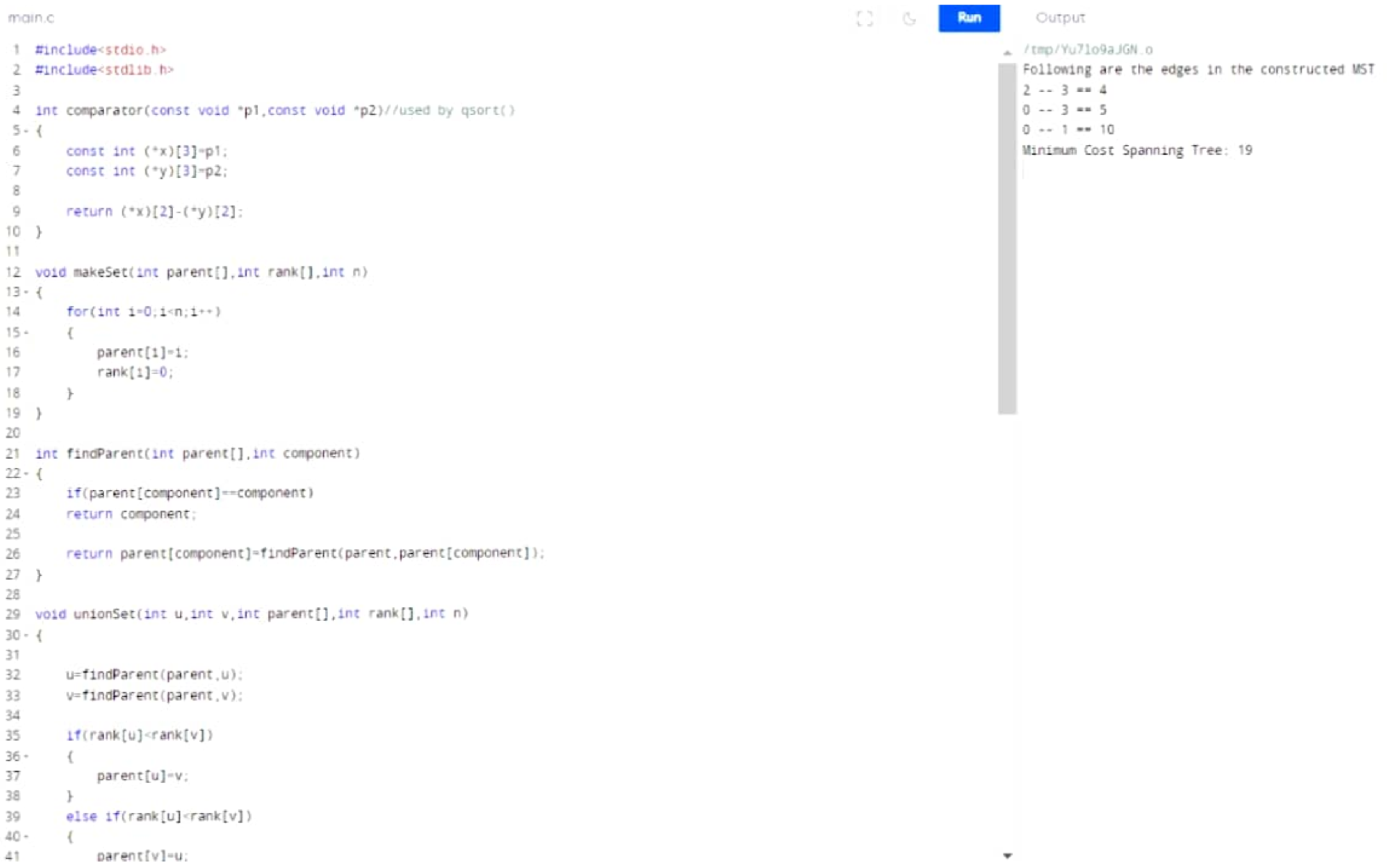
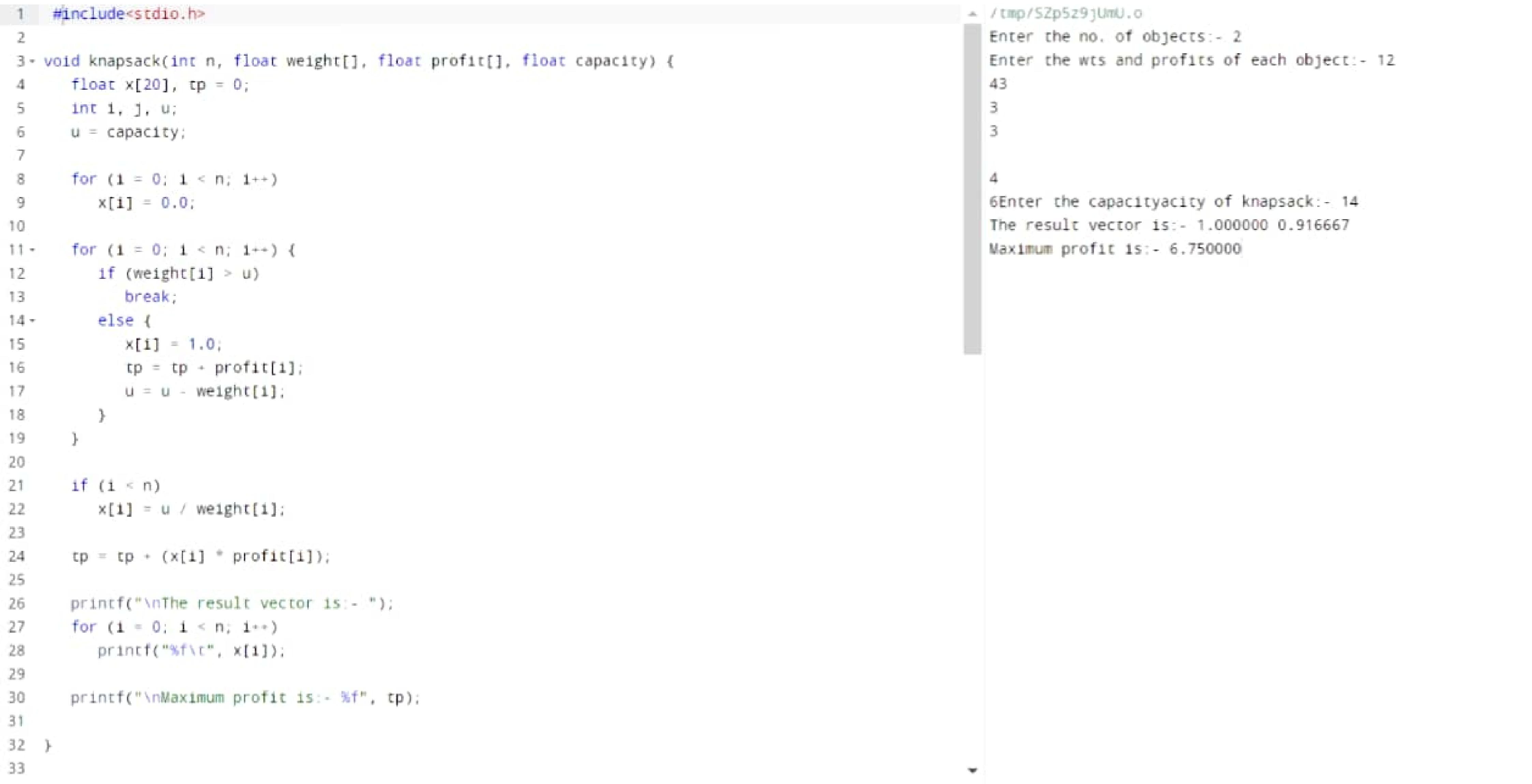


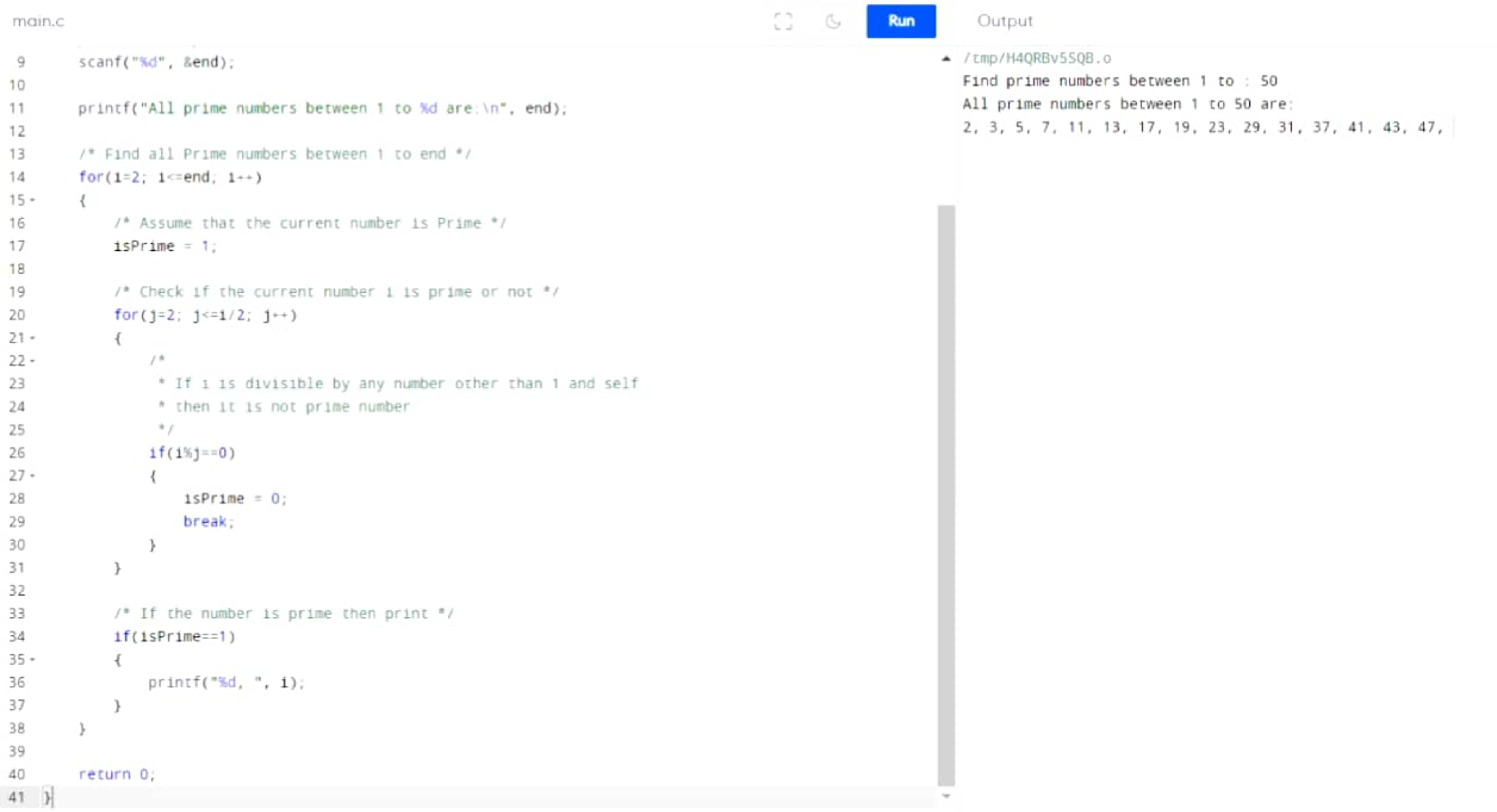
```c
#include<stdio.h>
#include<stdlib.h>

int comparator(const void *p1,const void *p2)//used by qsort()
{
    const int (*x)[3]=p1;
    const int (*y)[3]=p2;

    return (*x)[2]-(*y)[2];
}

void makeSet(int parent[],int rank[],int n)
{
    for(int i=0;i<n;i++)
    {
        parent[i]=i;
        rank[i]=0;
    }
}

int findParent(int parent[],int component)
{
    if(parent[component]==component)
    return component;

    return parent[component]=findParent(parent,parent[component]);
}

void unionSet(int u,int v,int parent[],int rank[],int n)
{

    u=findParent(parent,u);
    v=findParent(parent,v);

    if(rank[u]<rank[v])
    {
        parent[u]=v;
    }
    else if(rank[u]<rank[v])
    {
        parent[v]=u;
```

```
/tmp/Yu7lo9aJGN.o
Following are the edges in the constructed MST
2 -- 3 == 4
0 -- 3 == 5
0 -- 1 == 10
Minimum Cost Spanning Tree: 19
```



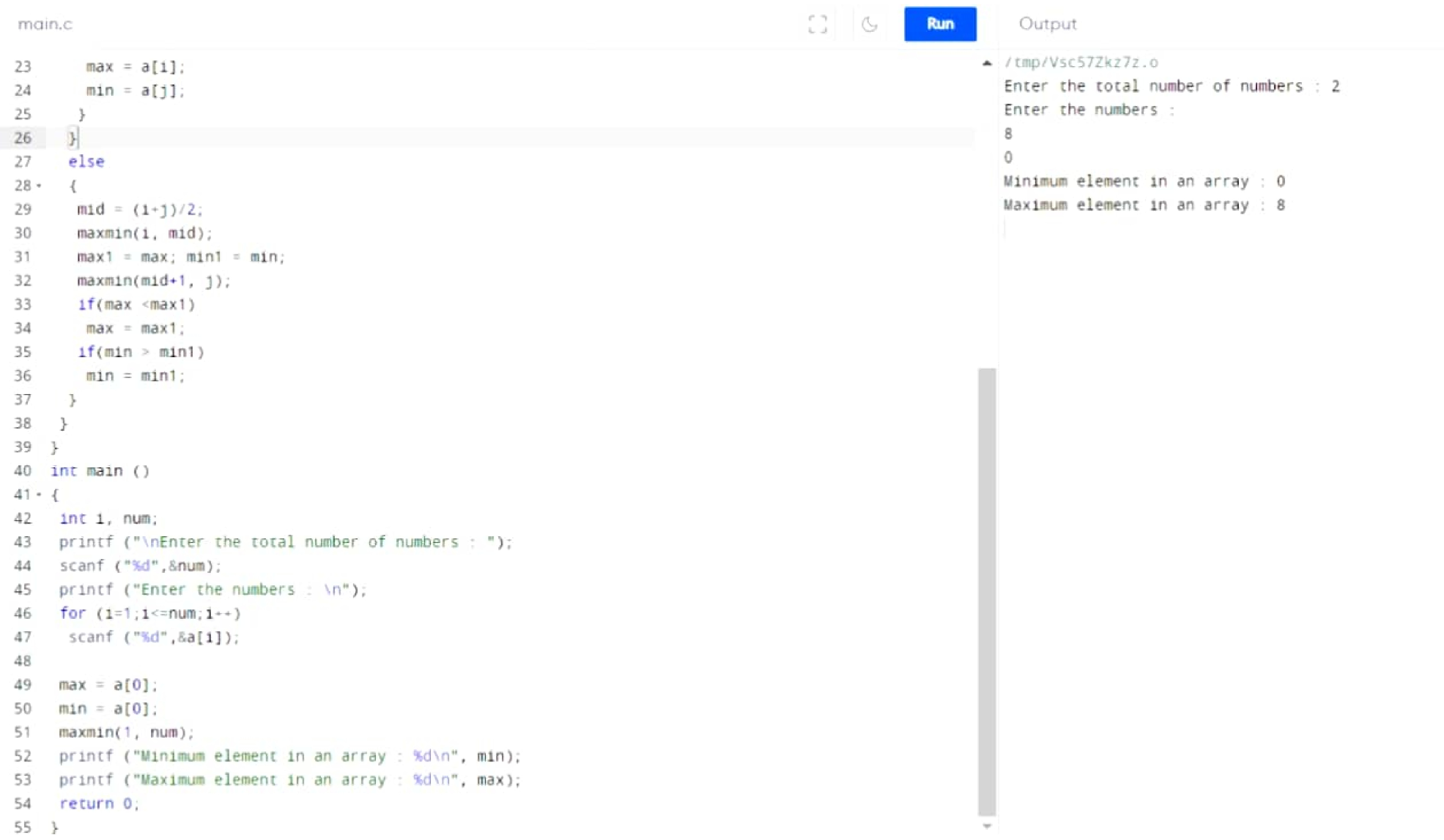
```c
#include<stdio.h>

void knapsack(int n, float weight[], float profit[], float capacity) {
   float x[20], tp = 0;
   int i, j, u;
   u = capacity;

   for (i = 0; i < n; i++)
      x[i] = 0.0;

   for (i = 0; i < n; i++) {
      if (weight[i] > u)
         break;
      else {
         x[i] = 1.0;
         tp = tp + profit[i];
         u = u - weight[i];
      }
   }

   if (i < n)
      x[i] = u / weight[i];

   tp = tp + (x[i] * profit[i]);

   printf("\nThe result vector is:- ");
   for (i = 0; i < n; i++)
      printf("%f\t", x[i]);

   printf("\nMaximum profit is:- %f", tp);

}
```

```
/tmp/SZp5z9jUmU.o
Enter the no. of objects:- 2
Enter the wts and profits of each object:- 12
43
3
3

4
6Enter the capacityacity of knapsack:- 14
The result vector is:- 1.000000 0.916667
Maximum profit is:- 6.750000
```



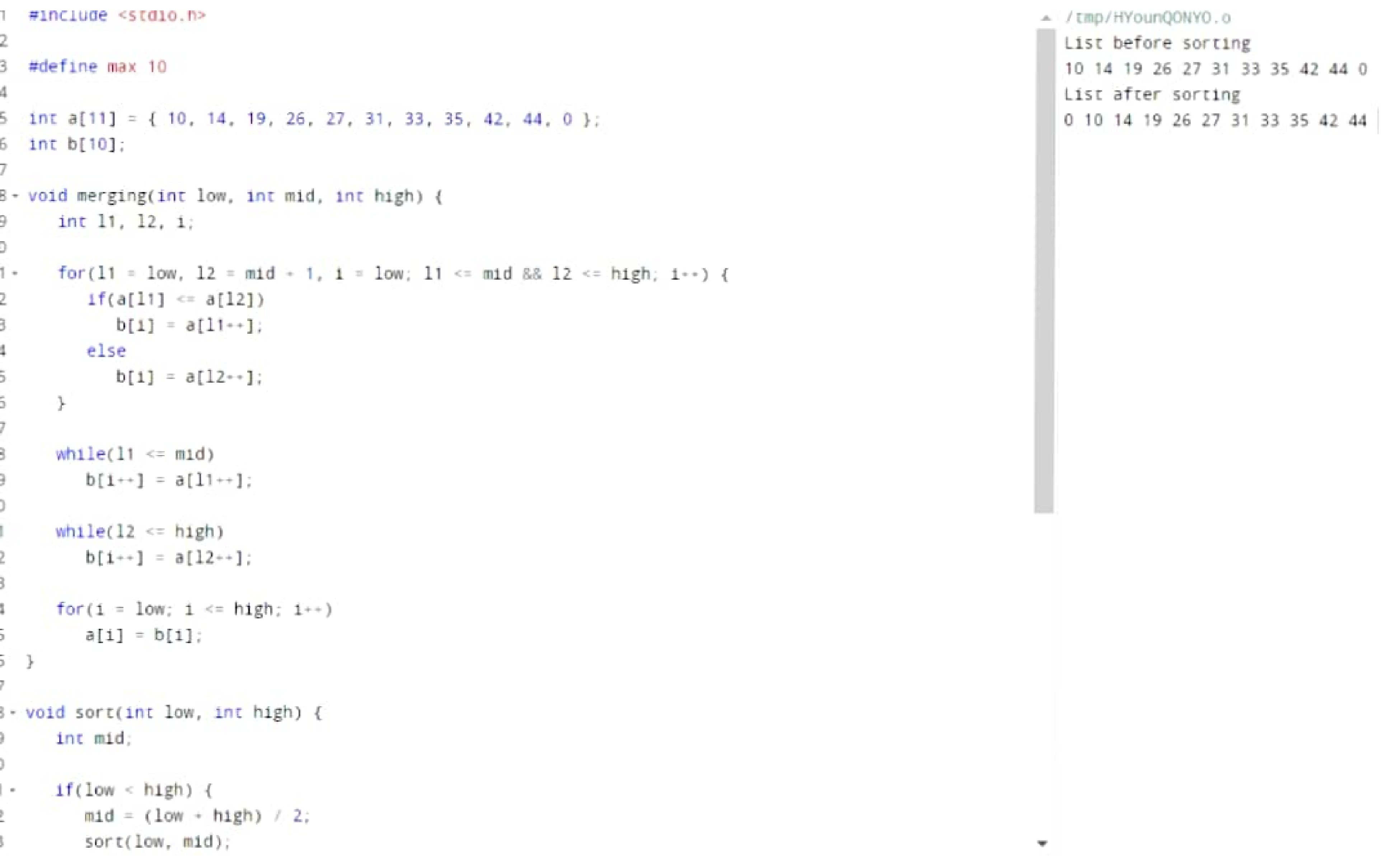
```c
    scanf("%d", &end);

    printf("All prime numbers between 1 to %d are:\n", end);

    /* Find all Prime numbers between 1 to end */
    for(i=2; i<=end; i++)
    {
        /* Assume that the current number is Prime */
        isPrime = 1;

        /* Check if the current number i is prime or not */
        for(j=2; j<=i/2; j++)
        {
            /*
             * If i is divisible by any number other than 1 and self
             * then it is not prime number
             */
            if(i%j==0)
            {
                isPrime = 0;
                break;
            }
        }

        /* If the number is prime then print */
        if(isPrime==1)
        {
            printf("%d, ", i);
        }
    }

    return 0;
}
```

```
/tmp/H4QRBv5SQB.o
Find prime numbers between 1 to : 50
All prime numbers between 1 to 50 are:
2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47,
```



```c
    max = a[i];
    min = a[j];
   }
  }
  else
  {
   mid = (i+j)/2;
   maxmin(i, mid);
   max1 = max; min1 = min;
   maxmin(mid+1, j);
   if(max <max1)
    max = max1;
   if(min > min1)
    min = min1;
  }
 }
}
int main ()
{
 int i, num;
 printf ("\nEnter the total number of numbers : ");
 scanf ("%d",&num);
 printf ("Enter the numbers : \n");
 for (i=1;i<=num;i++)
  scanf ("%d",&a[i]);

 max = a[0];
 min = a[0];
 maxmin(1, num);
 printf ("Minimum element in an array : %d\n", min);
 printf ("Maximum element in an array : %d\n", max);
 return 0;
}
```

```
/tmp/Vsc57Zkz7z.o
Enter the total number of numbers : 2
Enter the numbers :
8
0
Minimum element in an array : 0
Maximum element in an array : 8
```



```c
#include <stdio.h>

#define max 10

int a[11] = { 10, 14, 19, 26, 27, 31, 33, 35, 42, 44, 0 };
int b[10];

void merging(int low, int mid, int high) {
   int l1, l2, i;

   for(l1 = low, l2 = mid + 1, i = low; l1 <= mid && l2 <= high; i++) {
      if(a[l1] <= a[l2])
         b[i] = a[l1++];
      else
         b[i] = a[l2++];
   }

   while(l1 <= mid)
      b[i++] = a[l1++];

   while(l2 <= high)
      b[i++] = a[l2++];

   for(i = low; i <= high; i++)
      a[i] = b[i];
}

void sort(int low, int high) {
   int mid;

   if(low < high) {
      mid = (low + high) / 2;
      sort(low, mid);
```

```
/tmp/HYounQONYO.o
List before sorting
10 14 19 26 27 31 33 35 42 44 0
List after sorting
0 10 14 19 26 27 31 33 35 42 44
```