Day -2

## Assignment

## Menda Mani Sai

## 192111399

# 1. Write a program to perform Merge Sort.

#include<stdio.h>

#include <stdlib.h>

void merge(int arr[],int l, int m, int r)

{

int i,j,k;

int n1=m-l+1;

int n2=r-m;

int L[n1],R[n2];

for (i = 0; i < n1; i++)

L[i] = arr[l + i];

for (j = 0; j < n2; j++)

R[j] = arr[m + 1 +j];

i = 0;

j = 0;

k = l;

while (i < n1 && j < n2) {

if (L[i] <= R[j]) {

arr[k] = L[i];

i++;

}

else {

arr[k] = R[j];

j++;

}

k++;

}

while (i < n1) {

arr[k] = L[i];

i++;

k++;

}

while (j < n2) {

arr[k] = R[j];

j++;

k++;

}

}

void mergeSort(int arr[], int l, int r)

{

if (l < r) {

int m = l + (r - l) / 2;

mergeSort(arr, l, m);

mergeSort(arr, m + 1, r);

merge(arr, l, m, r);

}

}

void printArray(int a[], int n)

{

int i;

for (i = 0; i < n; i++)

printf("%d ", a[i]);

printf("\n");

}

int main()

{

int n;

printf("Enter the size of the array= ");

scanf("%d", &n);

int arr[n];

for(int i=0;i<n;i++){

scanf("%d", &arr[i]);

}

int arr\_n = sizeof(arr) / sizeof(arr[0]);

printf("Given array is \n");

printArray(arr, arr\_n);

mergeSort(arr, 0, arr\_n - 1);

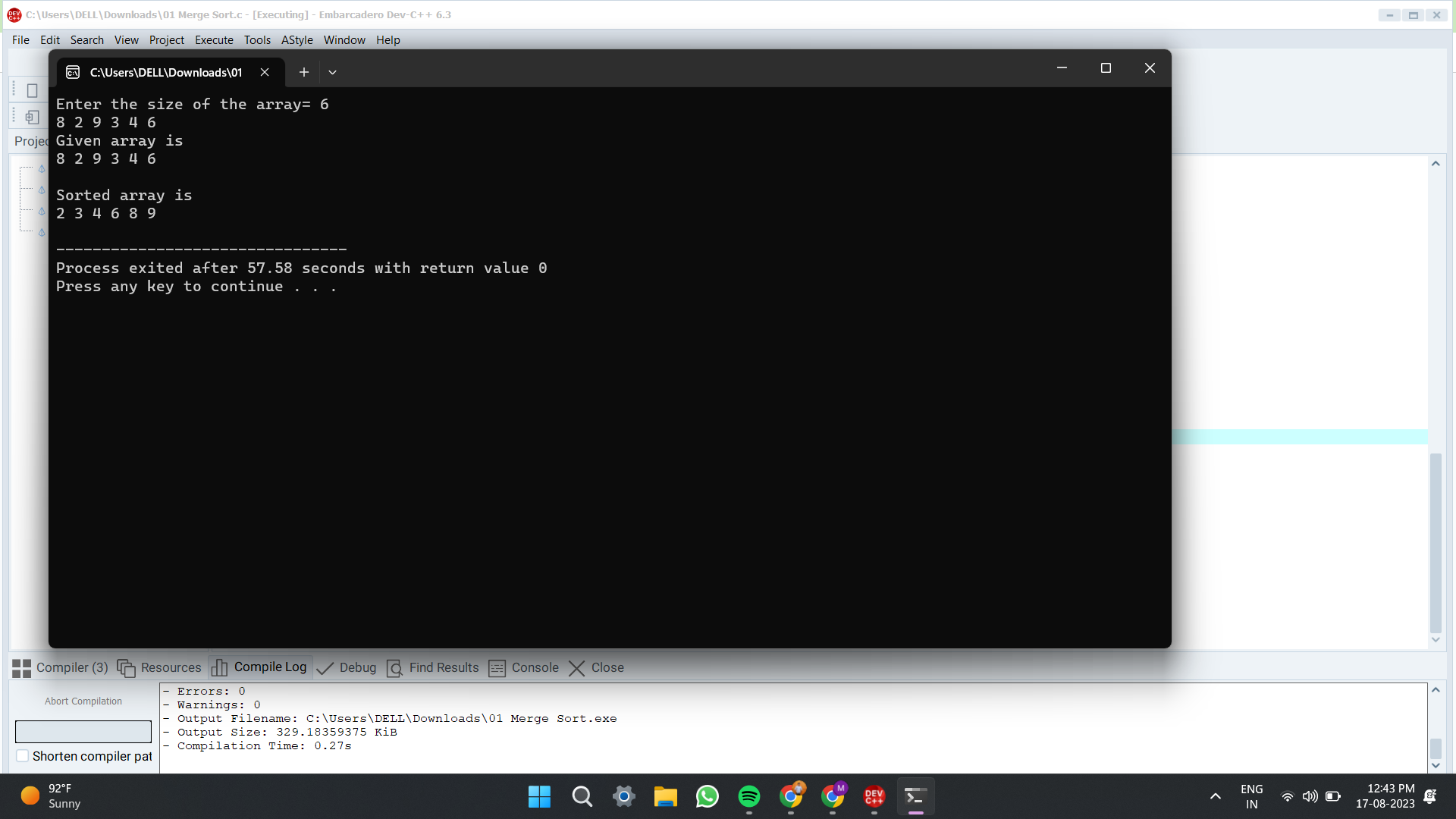
printf("\nSorted array is \n");

printArray(arr, arr\_n);

return 0;

}

# OUTPUT:



# 2. Using Divide and Conquer strategy to find Max and Min value in the list.

#include<stdio.h>

#include<stdio.h>

int max, min;

int a[100];

void maxmin(int i, int j)

{

int max1, min1, mid;

if(i==j)

{

max = min = a[i];

}

else

{

if(i == j-1)

{

if(a[i] <a[j])

{

max = a[j];

min = a[i];

}

else

{

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);

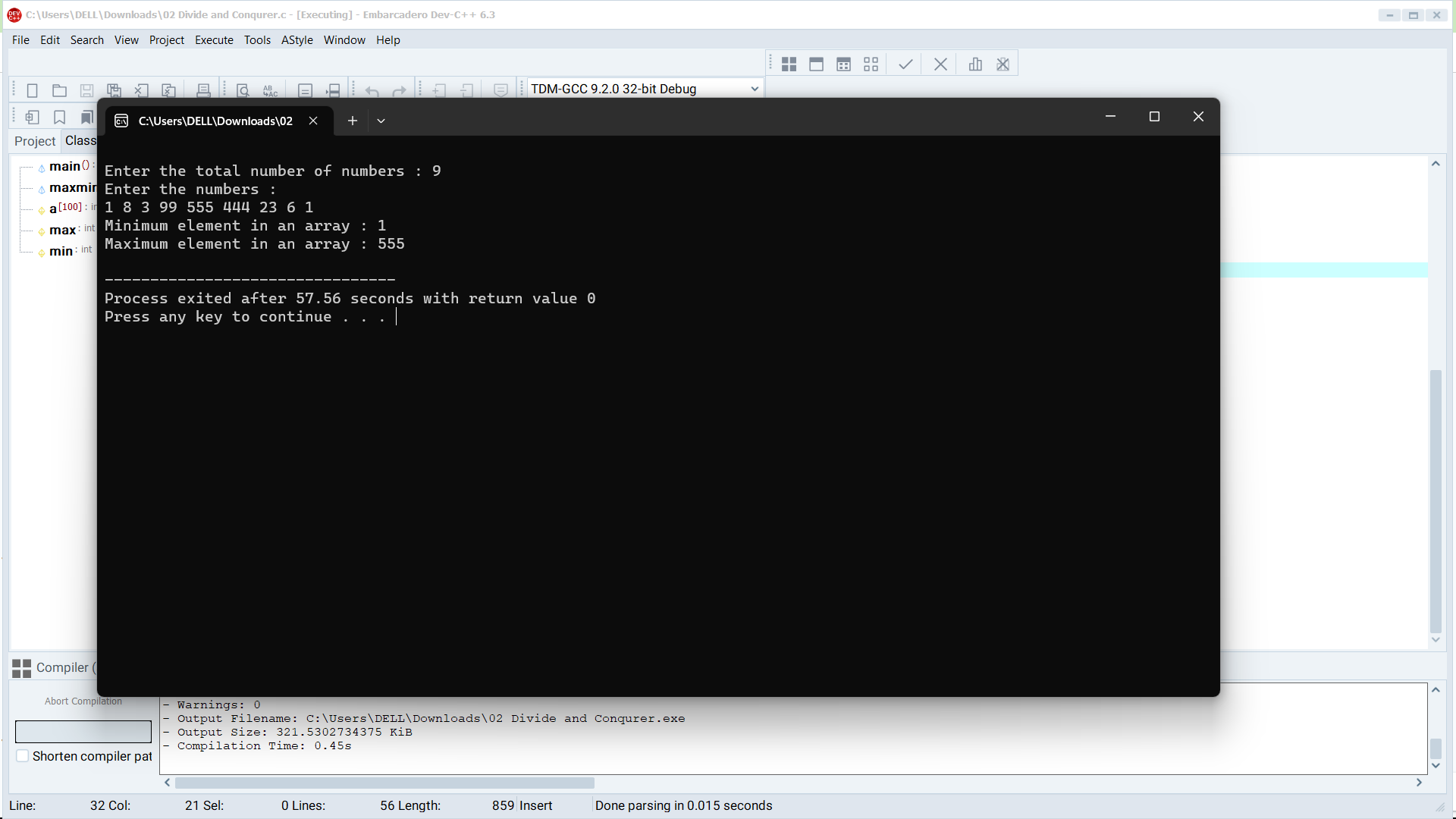
printf ("Minimum element in an array : %d\n", min);

printf ("Maximum element in an array : %d\n", max);

return 0;

}

# OUTPUT:



# 3. Write a program to generate all the prime numbers.

#include<stdio.h>

void main(){

int i, num, n, count;

printf("Enter the range: ");

scanf("%d", &n);

printf("The prime numbers in between the range 1 to %d:",n);

for(num = 1;num<=n;num++){

count = 0;

for(i=2;i<=num/2;i++){

if(num%i==0){

count++;

break;

}

}

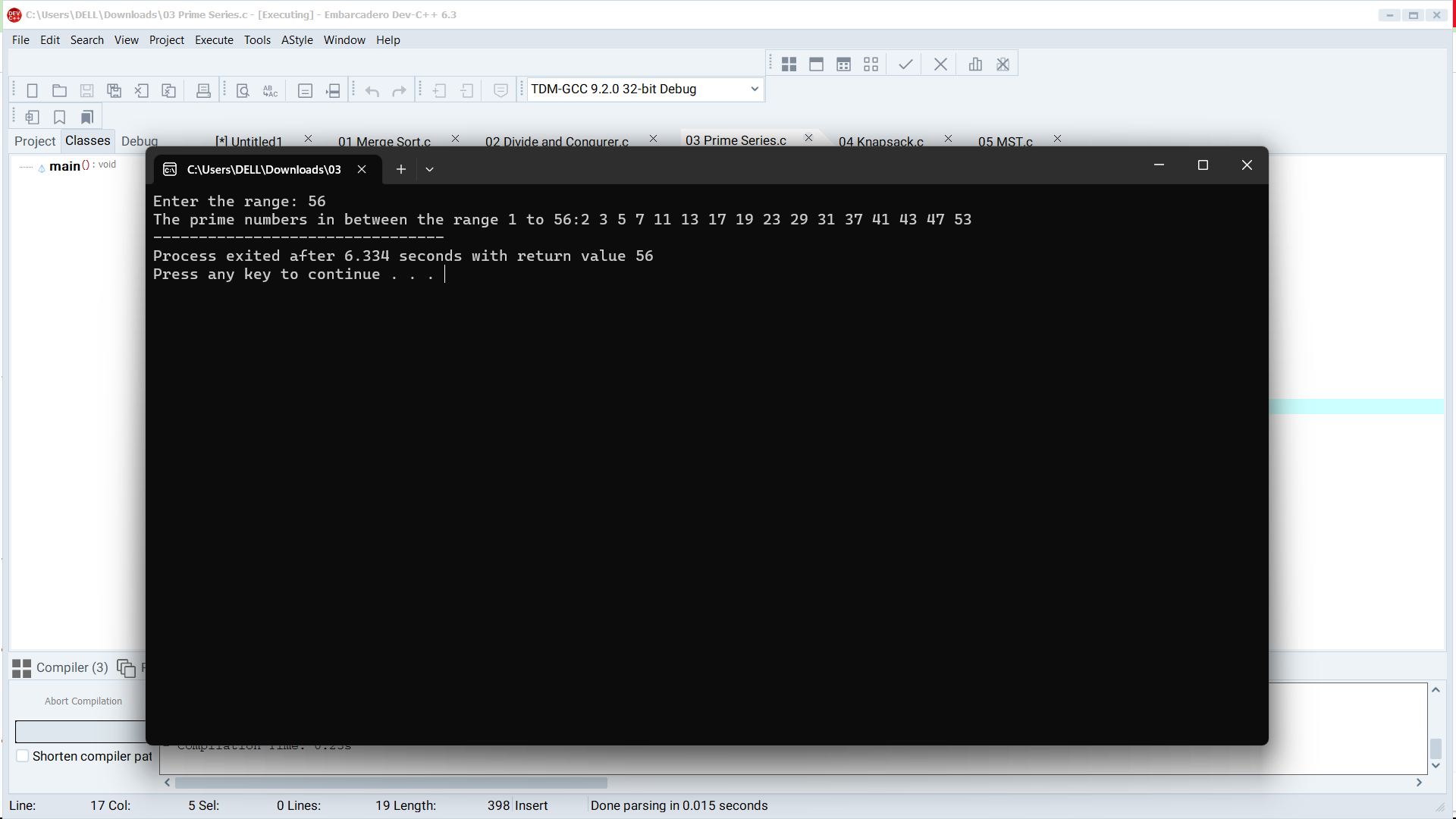
if(count==0 && num!= 1)

printf("%d ",num);

}

}

# OUTPUT:



# 4. Write a program to perform Knapsack problem using greedy techniques.

# 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);

}

int main() {

float weight[20], profit[20], capacity;

int num, i, j;

float ratio[20], temp;

printf("\nEnter the no. of objects:- ");

scanf("%d", &num);

printf("\nEnter the wts and profits of each object:- ");

for (i = 0; i < num; i++) {

scanf("%f %f", &weight[i], &profit[i]);

}

printf("\nEnter the capacityacity of knapsack:- ");

scanf("%f", &capacity);

for (i = 0; i < num; i++) {

ratio[i] = profit[i] / weight[i];

}

for (i = 0; i < num; i++) {

for (j = i + 1; j < num; j++) {

if (ratio[i] < ratio[j]) {

temp = ratio[j];

ratio[j] = ratio[i];

ratio[i] = temp;

temp = weight[j];

weight[j] = weight[i];

weight[i] = temp;

temp = profit[j];

profit[j] = profit[i];

profit[i] = temp;

}

}

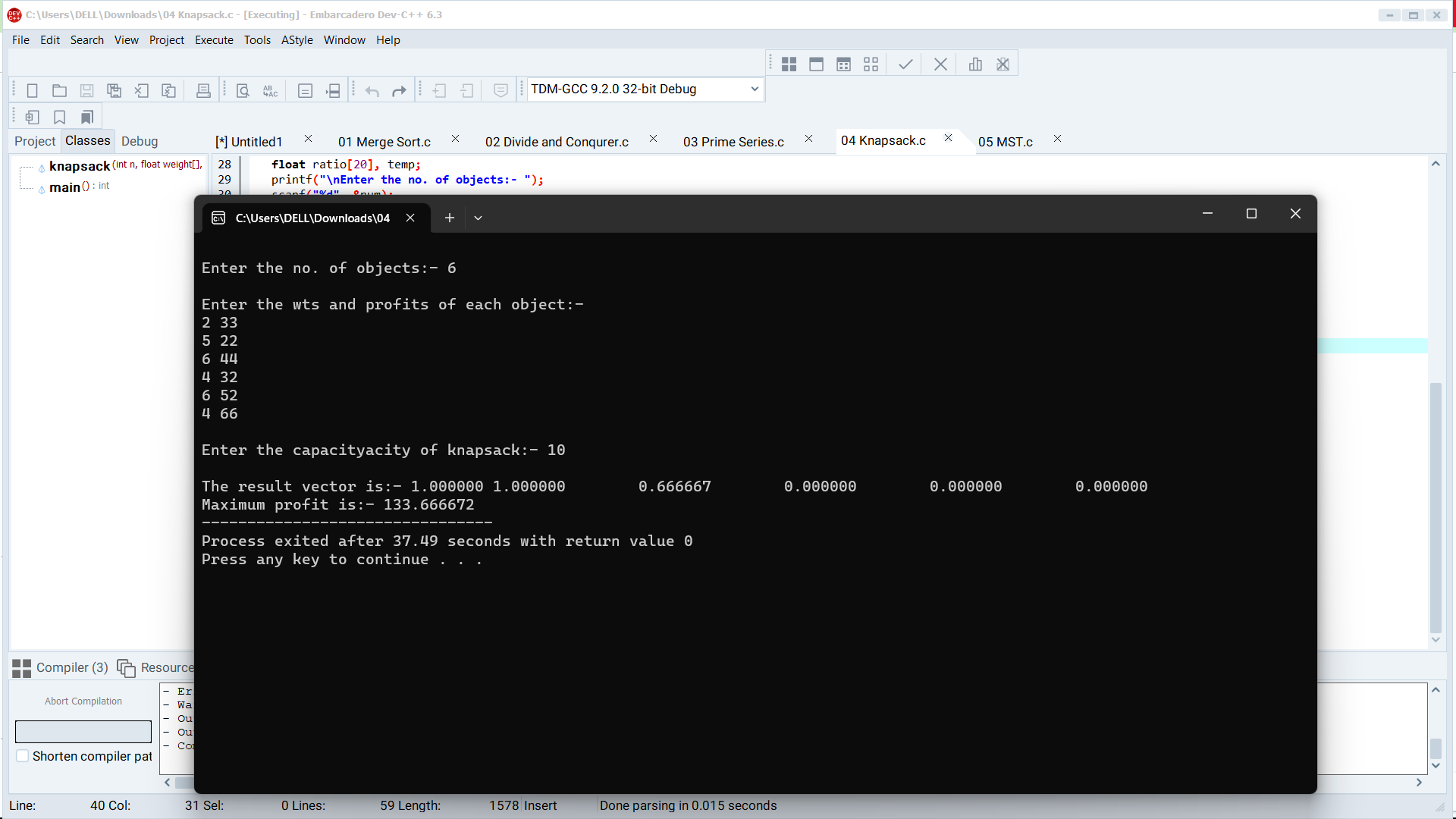
}

knapsack(num, weight, profit, capacity);

return(0);

}

# OUTPUT:



# 5. Write a program to perform MST using greedy techniques.

#include<stdio.h>

#include<conio.h>

int a,b,u,v,n,i,j,ne=1;

int visited[10]={0},min,mincost=0,cost[10][10];

void main()

{

printf("\nEnter the number of nodes:");

scanf("%d",&n);

printf("\nEnter the adjacency matrix:\n");

for(i=1;i<=n;i++)

for(j=1;j<=n;j++)

{

scanf("%d",&cost[i][j]);

if(cost[i][j]==0)

cost[i][j]=999;

}

visited[1]=1;

printf("\n");

while(ne < n)

{

for(i=1,min=999;i<=n;i++)

for(j=1;j<=n;j++)

if(cost[i][j]< min)

if(visited[i]!=0)

{

min=cost[i][j];

a=u=i;

b=v=j;

}

if(visited[u]==0 || visited[v]==0)

{

printf("\n Edge %d:(%d %d) cost:%d",ne++,a,b,min);

mincost+=min;

visited[b]=1;

}

cost[a][b]=cost[b][a]=999;

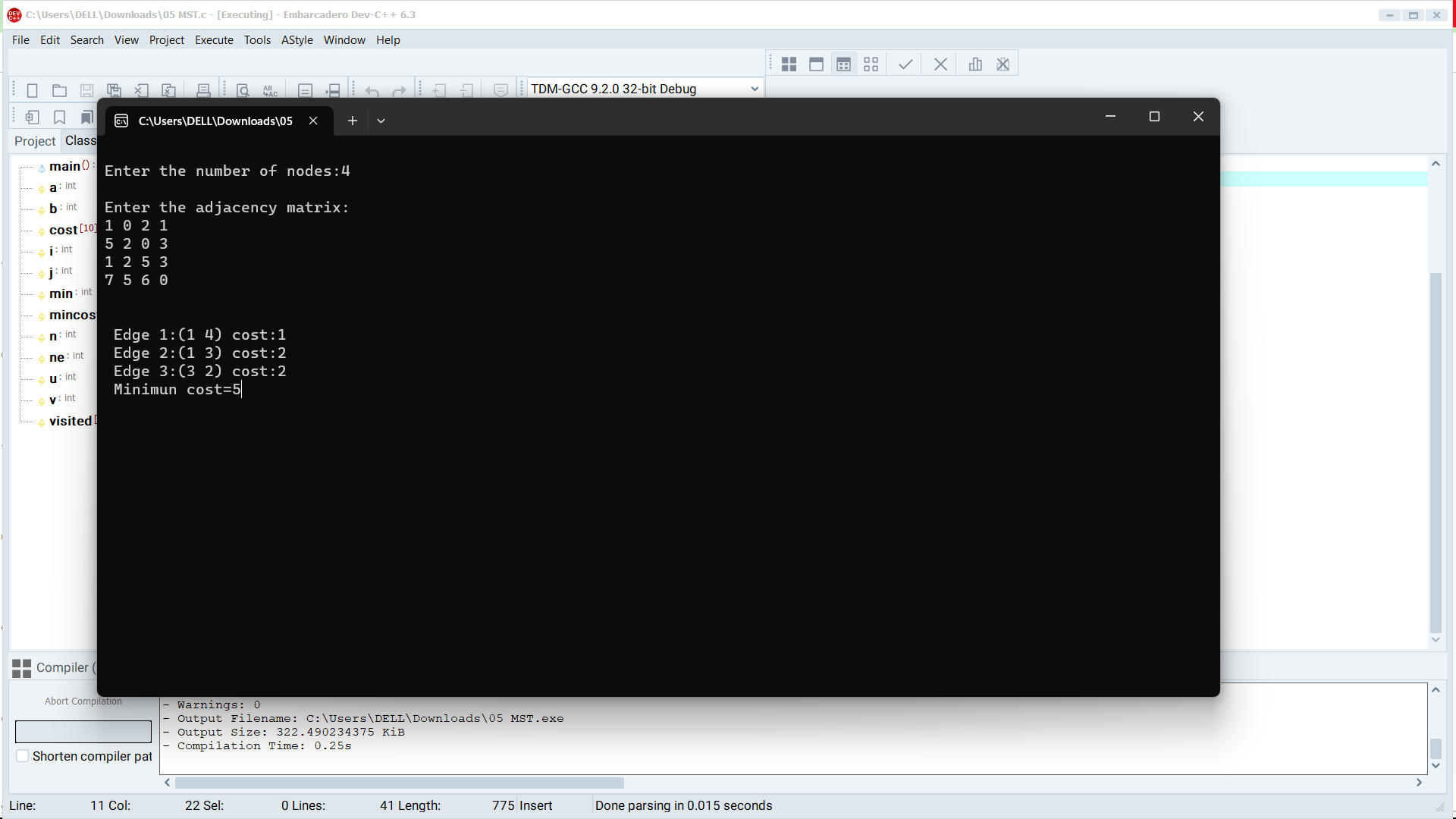
}

printf("\n Minimun cost=%d",mincost);

getch();

}

# OUTPUT:



# 6. Using Dynamic programming concept to find out optimal binary search tree.

#include <stdio.h>

#include <limits.h>

#define MAX\_KEYS 100

int sum(int freq[], int i, int j) {

int s = 0;

for (int k = i; k <= j; k++) {

s += freq[k];

}

return s;

}

int optimalBST(int keys[], int freq[], int n) {

int cost[n][n];

for (int i = 0; i < n; i++) {

cost[i][i] = freq[i];

}

for (int length = 2; length <= n; length++) {

for (int i = 0; i <= n - length + 1; i++) {

int j = i + length - 1;

cost[i][j] = INT\_MAX;

for (int r = i; r <= j; r++) {

int c = ((r > i) ? cost[i][r - 1] : 0) +

((r < j) ? cost[r + 1][j] : 0) +

sum(freq, i, j);

if (c < cost[i][j]) {

cost[i][j] = c;

}

}

}

}

return cost[0][n - 1];

}

int main() {

int n;

printf("Enter the number of keys: ");

scanf("%d", &n);

int keys[MAX\_KEYS];

int freq[MAX\_KEYS];

printf("Enter the keys:\n");

for (int i = 0; i < n; i++) {

scanf("%d", &keys[i]);

}

printf("Enter the frequencies:\n");

for (int i = 0; i < n; i++) {

scanf("%d", &freq[i]);

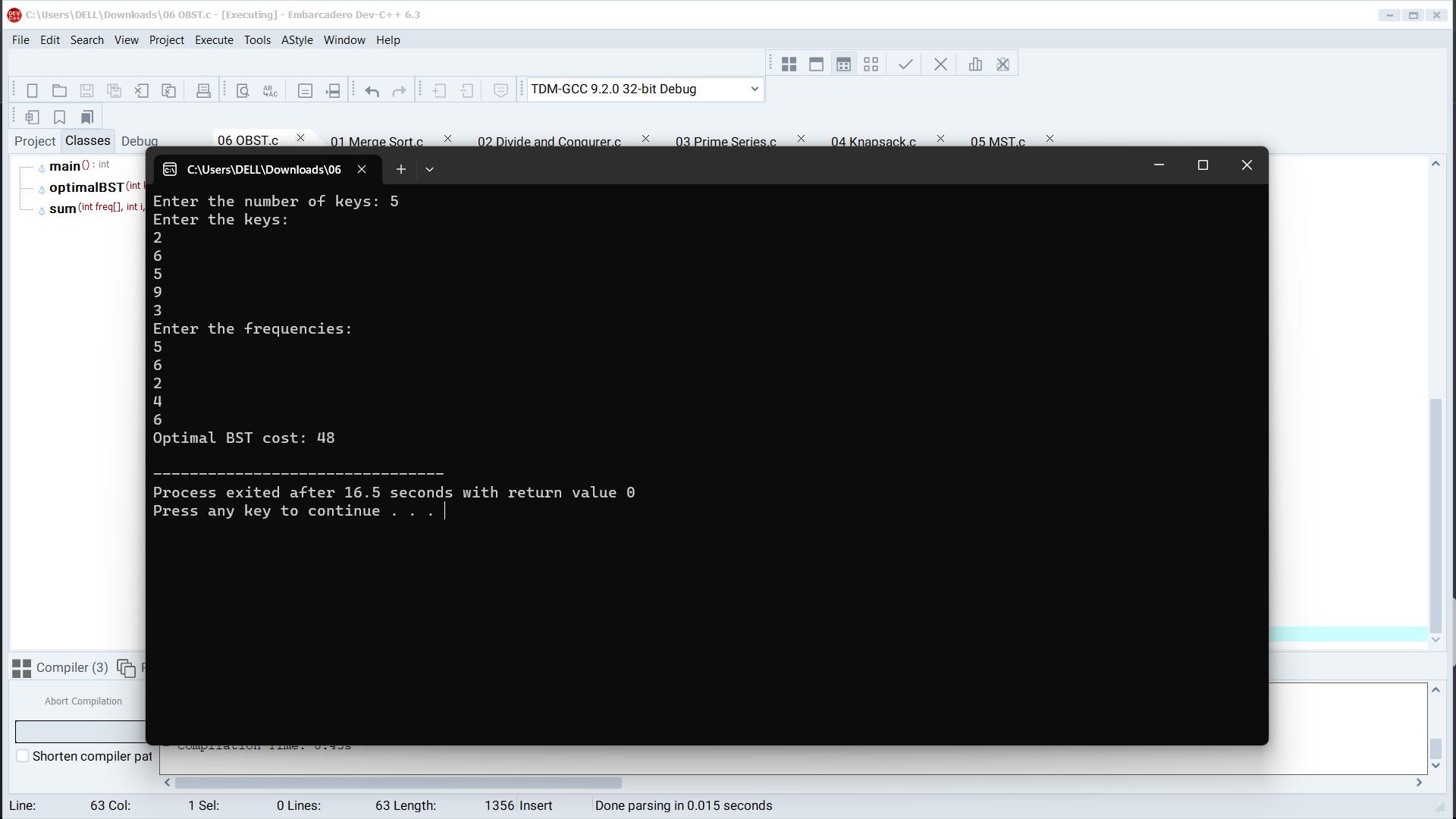
}

printf("Optimal BST cost: %d\n", optimalBST(keys, freq, n));

return 0;

}

# OUTPUT:



# 7. Using Dynamic programming techniques to find binomial coefficient of a given number

#include <stdio.h>

int binomialCoeff(int n, int k) {

int C[n + 1][k + 1];

for (int i = 0; i <= n; i++) {

for (int j = 0; j <= min(i, k); j++) {

if (j == 0 || j == i)

C[i][j] = 1;

else

C[i][j] = C[i - 1][j - 1] + C[i - 1][j];

}

}

return C[n][k];

}

int min(int a, int b) {

return (a < b) ? a : b;

}

int main() {

int n, k;

printf("Enter values of n and k: ");

scanf("%d %d", &n, &k);

printf("Binomial Coefficient C(%d, %d) = %d\n", n, k, binomialCoeff(n, k));

return 0;

}

# OUTPUT:

