```c
#include<stdio.h>
#include<time.h>
#include<stdlib.h>
void selsort(int n,int a[]);

void main()
{
   int a[15000],n,i,j,ch,temp;
   clock_t start,end;

   while(1)
   {
 printf("\n1:For manual entry of N value and array elements");
 printf("\n2:To display time taken for sorting number of elements N in the range 500 to 14500");
 printf("\n3:To exit");
    printf("\nEnter your choice:");
    scanf("%d", &ch);
    switch(ch)
    {
      case 1:  printf("\nEnter the number of elements: ");
       scanf("%d",&n);
       printf("\nEnter array elements: ");
       for(i=0;i<n;i++)
        {
         scanf("%d",&a[i]);
        }
       start=clock();
       heapsort(a,n);
       end=clock();
       printf("\nSorted array is: ");
       for(i=0;i<n;i++)
       printf("%d\t",a[i]);
printf("\n Time taken to sort %d numbers is %f Secs",n, (((double)(end-start))/CLOCKS_PER_SEC));
       break;
    case 2:
          n=500;
          while(n<=14500) {
          for(i=0;i<n;i++)
         {
```

```
          while(n<=14500) {
          for(i=0;i<n;i++)
          {

            a[i]=n-i;
          }
          start=clock();
          heapsort(a,n);

      for(j=0;j<500000000;j++){ temp=38/600;}
             end=clock();
printf("\n Time taken to sort %d numbers is %f Secs",n, (((double)(end-start))/CLOCKS_PER_SEC));
              n=n+1000;
          }
          break;
   case 3: exit(0);
   }
   getchar();
    }
}


void heapify(int arr[], int n, int i) {
    int temp, maximum, left_index, right_index;
    maximum = i;
    right_index = 2 * i + 2;
    left_index = 2 * i + 1;

    if (left_index < n && arr[left_index] > arr[maximum])
        maximum = left_index;
    if (right_index < n && arr[right_index] > arr[maximum])
        maximum = right_index;

    if (maximum != i) {
        temp = arr[i];
        arr[i] = arr[maximum];
        arr[maximum] = temp;
        heapify(arr, n, maximum);
    }
}
```

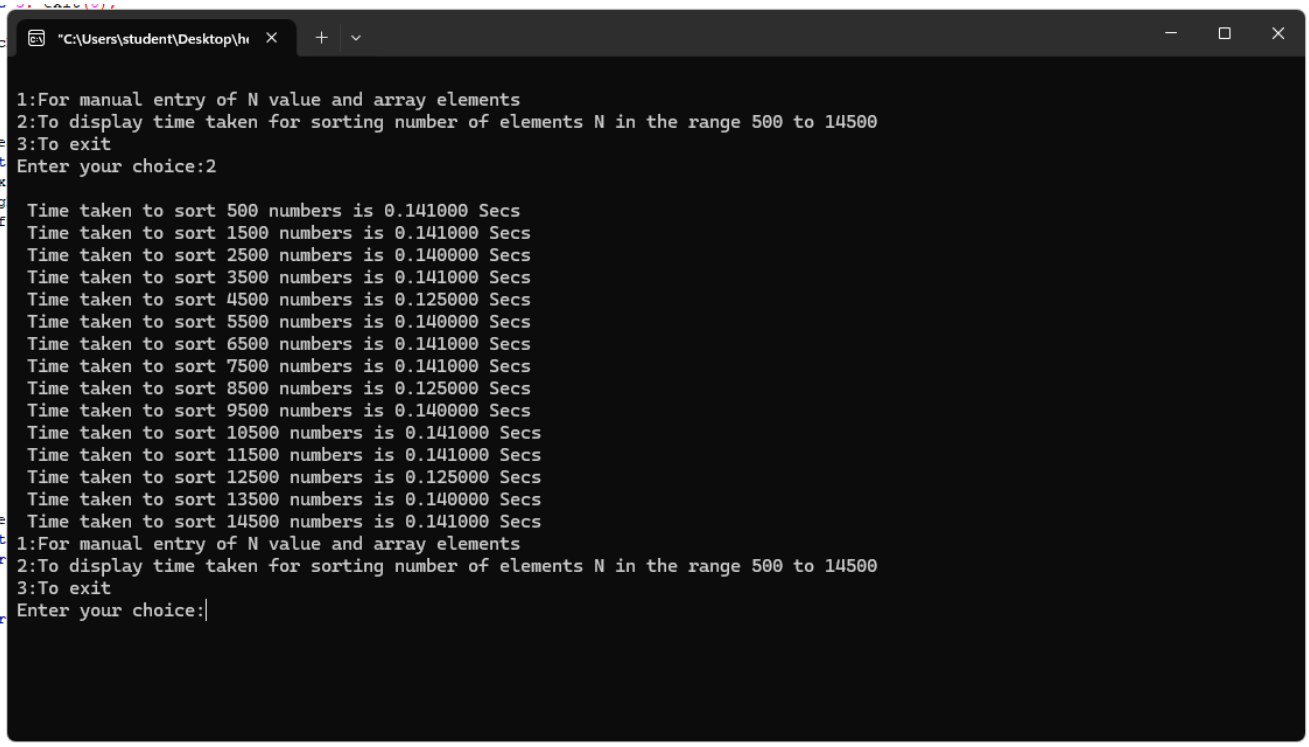
```
        break;
   case 3: exit(0);
   }
   getchar();
    }
}


void heapify(int arr[], int n, int i) {
    int temp, maximum, left_index, right_index;
    maximum = i;
    right_index = 2 * i + 2;
    left_index = 2 * i + 1;

    if (left_index < n && arr[left_index] > arr[maximum])
        maximum = left_index;
    if (right_index < n && arr[right_index] > arr[maximum])
        maximum = right_index;

    if (maximum != i) {
        temp = arr[i];
        arr[i] = arr[maximum];
        arr[maximum] = temp;
        heapify(arr, n, maximum);
    }
}

void heapsort(int arr[], int n) {
    int i, temp;
    for (i = n / 2 - 1; i >= 0; i--) {
        heapify(arr, n, i);
    }
    for (i = n - 1; i > 0; i--) {
        temp = arr[0];
        arr[0] = arr[i];
        arr[i] = temp;
        heapify(arr, i, 0);
    }
}
```


"C:\Users\student\Desktop\h
1:For manual entry of N value and array elements
2:To display time taken for sorting number of elements N in the range 500 to 14500
3:To exit
Enter your choice:2
Time taken to sort 500 numbers is 0.141000 Secs
Time taken to sort 1500 numbers is 0.141000 Secs
Time taken to sort 2500 numbers is 0.140000 Secs
Time taken to sort 3500 numbers is 0.141000 Secs
Time taken to sort 4500 numbers is 0.125000 Secs
Time taken to sort 5500 numbers is 0.140000 Secs
Time taken to sort 6500 numbers is 0.141000 Secs
Time taken to sort 7500 numbers is 0.141000 Secs
Time taken to sort 8500 numbers is 0.125000 Secs
Time taken to sort 9500 numbers is 0.140000 Secs
Time taken to sort 10500 numbers is 0.141000 Secs
Time taken to sort 11500 numbers is 0.141000 Secs
Time taken to sort 12500 numbers is 0.125000 Secs
Time taken to sort 13500 numbers is 0.140000 Secs
Time taken to sort 14500 numbers is 0.141000 Secs
1:For manual entry of N value and array elements
2:To display time taken for sorting number of elements N in the range 500 to 14500
3:To exit
Enter your choice:

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

int cost[1000][1000];

void floyd(int n){
    int d[n][n];
    for(int i = 0; i < n; i++){
        for(int j = 0; j < n; j++){
            d[i][j] = cost[i][j];
        }
    }
    for(int k = 0; k < n; k++){
        for(int i = 0; i < n; i++){
            for(int j = 0; j < n; j++){
                if(d[i][j] > d[i][k] + d[k][j]){
                    d[i][j] = d[i][k] + d[k][j];
                }
            }
        }
    }
    printf("Output: \n");
    for(int i = 0; i < n; i++){
        for(int j = 0; j < n; j++){
            printf("%d ", d[i][j]);
        }
        printf("\n");
    }
}

int main(){
    int n;
    printf("Enter number of elements: ");
    scanf("%d", &n);
    printf("Enter elements: \n");
    for(int i = 0; i < n; i++){
        for(int j = 0; j < n; j++){
            scanf("%d", &cost[i][j]);
        }
    }
```

```
void floyd(int n){
    int d[n][n];
    for(int i = 0; i < n; i++){
        for(int j = 0; j < n; j++){
            d[i][j] = cost[i][j];
        }
    }
    for(int k = 0; k < n; k++){
        for(int i = 0; i < n; i++){
            for(int j = 0; j < n; j++){
                if(d[i][j] > d[i][k] + d[k][j]){
                    d[i][j] = d[i][k] + d[k][j];
                }
            }
        }
    }
    printf("Output: \n");
    for(int i = 0; i < n; i++){
        for(int j = 0; j < n; j++){
            printf("%d ", d[i][j]);
        }
        printf("\n");
    }
}

int main(){
    int n;
    printf("Enter number of elements: ");
    scanf("%d", &n);
    printf("Enter elements: \n");
    for(int i = 0; i < n; i++){
        for(int j = 0; j < n; j++){
            scanf("%d", &cost[i][j]);
        }
    }
    floyd(n);
    return 0;
}
```