SEARCHING

Linear Searching

Program:

#include <stdio.h>

int Linear\_Search(int arr[], int n, int item)

{

int i = 0;

while (i < n && arr[i] != item)

i++;

if (i < n)

return i;

return -1;

}

int main(int argc, char const \*argv[])

{

int n,i;

printf("Enter number: ");

scanf("%d", &n);

int arr[n];

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

{

printf("Enter number %d: ", i + 1);

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

}

printf("Enter item to be searched: ");

int item, loc;

scanf("%d", &item);

if ((loc = Linear\_Search(arr, n, item)) != -1)

printf("%d found at index %d\n", item, loc);

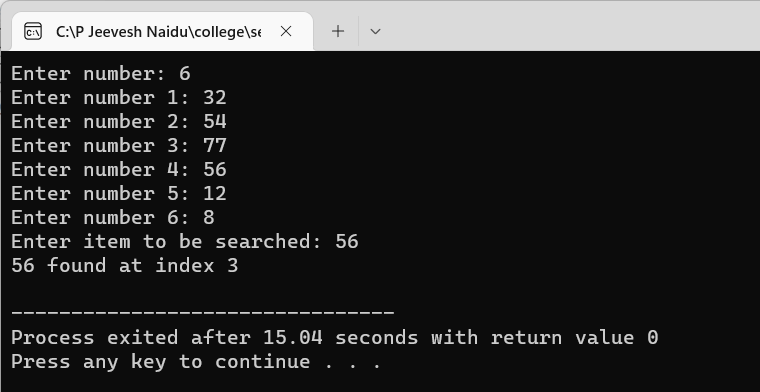
else

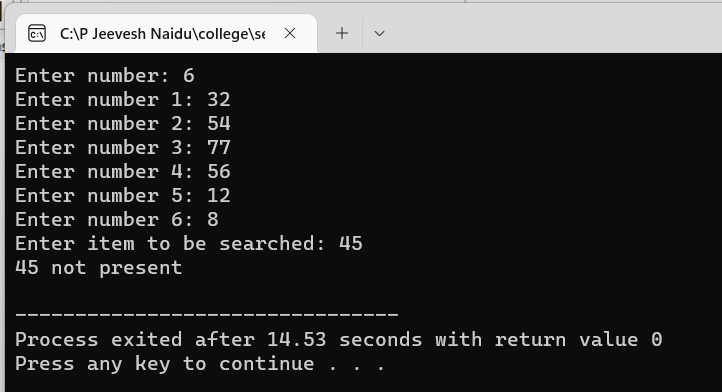
printf("%d not present\n", item);

return 0;

}

Output:





Binary Search

Program:

|  |
| --- |
| #include <stdio.h> |
|  |  |
|  | int binarySearchNR(int arr[], int item, int n, int low, int up) |
|  | { |
|  | int mid; |
|  | while (low <= up) |
|  | { |
|  | mid = (low + up) / 2; |
|  | if (item < arr[mid]) |
|  | up = mid - 1; |
|  | else if (item > arr[mid]) |
|  | low = mid + 1; |
|  | else |
|  | return mid; |
|  | } |
|  | return -1; |
|  | } |
|  |  |
|  | void Insertion\_Sort(int arr[], int n) |
|  | { |
|  | for (int i = 1; i < n; i++) |
|  | { |
|  | int j = i - 1, k = arr[i]; |
|  | while (j >= 0 && k < arr[j]) |
|  | { |
|  | arr[j + 1] = arr[j]; |
|  | j--; |
|  | } |
|  | arr[j + 1] = k; |
|  | } |
|  | } |
|  |  |
|  | int binarySearchR(int arr[], int item, int low, int up) |
|  | { |
|  | int mid; |
|  | if (low > up) |
|  | return -1; |
|  | mid = (low + up) / 2; |
|  | if (item < arr[mid]) |
|  | binarySearchR(arr, item, low, mid - 1); |
|  | else if (item > arr[mid]) |
|  | binarySearchR(arr, item, mid + 1, up); |
|  | else |
|  | return mid; |
|  | } |
|  |  |
|  | int main(int argc, char const \*argv[]) |
|  | { |
|  | int n; |
|  | printf("Enter number: "); |
|  | scanf("%d", &n); |
|  | int arr[n], loc; |
|  | for (int i = 0; i < n; i++) |
|  | { |
|  | printf("Enter number %d: ", i + 1); |
|  | scanf("%d", &arr[i]); |
|  | } |
|  | Insertion\_Sort(arr, n); |
|  | printf("After Sorting: "); |
|  | for (int i = 0; i < n; i++) |
|  | printf("%3d", arr[i]); |
|  | printf("\nEnter item to be searched: "); |
|  | int item; |
|  | scanf("%d", &item); |
|  | if ((loc = binarySearchNR(arr, item, n, 0, n - 1)) != -1) |
|  | printf("%d found at index %d\n", item, loc); |
|  | else |
|  | printf("%d not found\n", item); |
|  | printf("Enter item to be searched: "); |
|  | scanf("%d", &item); |
|  | if ((loc = binarySearchR(arr, item, 0, n - 1)) != -1) |
|  | printf("%d found at index %d\n", item, loc); |
|  | Else |
|  | printf("%d not found\n", item); |
|  | return 0; |
|  | }  Output: |
|  |  |

SORTING

Bubble Sort

Program:

#include <stdio.h>

void show(int arr[], int n)

{

int i;

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

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

printf("\n");

}

int main(int argc, char const \*argv[])

{

int n, temp, i, j;

printf("Enter number: ");

scanf("%d", &n);

int arr[n];

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

{

printf("Enter number %d: ", i + 1);

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

}

show(arr, n);

for ( i = 0; i < n - 1; i++) // no of passes

{

int swaps = 0;

for ( j = 0; j < n - i - 1; j++) // exclusing the last elements cuz they sorted

{

if (arr[j] > arr[j + 1])

{

temp = arr[j];

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

arr[j + 1] = temp;

swaps++;

}

}

if (!swaps)

break;

}

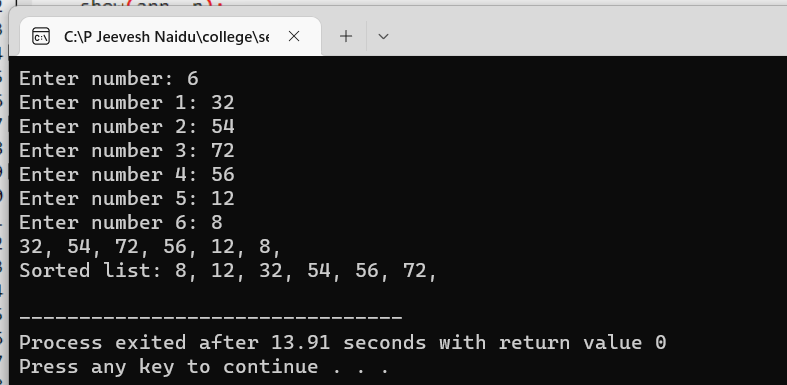
printf("Sorted list: ");

show(arr, n);

return 0;

}

Output:

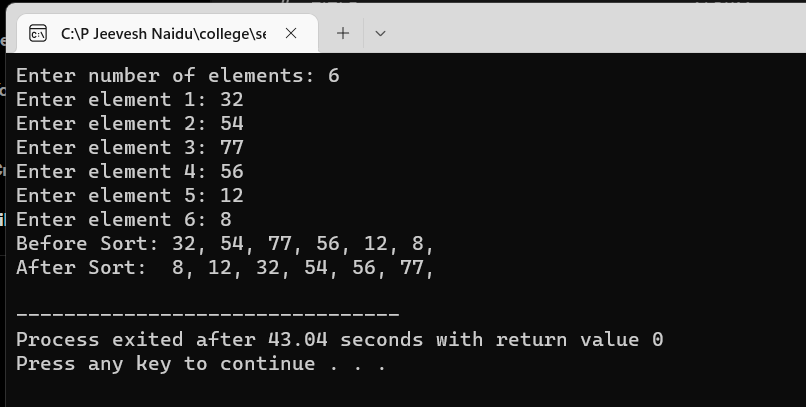


Selection Sort

Program:

|  |
| --- |
| #include <stdio.h> |
|  |  |
|  | int isMin(int a, int b) |
|  | { |
|  | return (a < b); |
|  | } |
|  |  |
|  | void swapnum(int \*a, int \*b) |
|  | { |
|  | int temp = \*a; |
|  | \*a = \*b; |
|  | \*b = temp; |
|  | } |
|  |  |
|  | void show(int arr[], int n) |
|  | { |
|  | for (int i = 0; i < n; i++) |
|  | printf("%d, ", arr[i]); |
|  | printf("\n"); |
|  | } |
|  |  |
|  | int main(int argc, char const \*argv[]) |
|  | { |
|  | int n; |
|  | printf("Enter number of elements: "); |
|  | scanf("%d", &n); |
|  | int arr[n]; |
|  | for (int i = 0; i < n; i++) |
|  | { |
|  | printf("Enter element %d: ", i + 1); |
|  | scanf("%d", &arr[i]); |
|  | } |
|  | printf("Before Sort: "); |
|  | show(arr, n); |
|  | int min; |
|  | for (int i = 0; i < n; i++) |
|  | { |
|  | min = i; // holds index of min element |
|  | for (int j = i + 1; j < n; j++) |
|  | { |
|  | if (isMin(arr[j], arr[min])) |
|  | { |
|  | min = j; |
|  | } |
|  | } |
|  | swapnum(&arr[i], &arr[min]); |
|  | } |
|  | printf("After Sort: "); |
|  | show(arr, n); |
|  | return 0; |
|  | } |

Output:



Insertion Sort

Program:

#include <stdio.h>

int main(int argc, char const \*argv[])

{

int n,i;

printf("Enter number of elements: ");

scanf("%d", &n);

int arr[n];

printf("Insert elements\n");

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

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

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

{

int j = i - 1, k = arr[i];

while (j >= 0 && k < arr[j])

{

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

j--;

}

arr[j + 1] = k;

}

printf("Sorted array is: ");

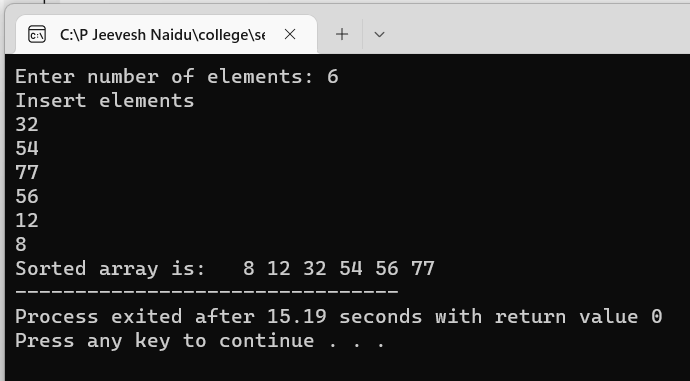
for ( i = 0; i < n; i++)

printf("%3d", arr[i]);

return 0;

}

Output:



Merge Sort

Program:

#include <stdio.h>

void merge(int arr[], int left, int mid, int right)

{

int i, j, k;

int n1 = mid - left + 1;

int n2 = right - mid; // right minus mid because mid actually lies before the right part

int L[n1], R[n2];

// splitting the parent array

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

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

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

R[j] = arr[mid + 1 + j]; // for right half left is mid+1

i = 0;

j = 0;

k = left;

// entering elements from each subarray in a sorted manner while (i < n1 && j < n2)

while (i < n1 && j < n2)

{

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

{

arr[k] = L[i];

i++;

}

else

{

arr[k] = R[j];

j++;

}

k++;

}

// remaining elements

while (i < n1)

{

arr[k] = L[i];

i++;

k++;

}

while (j < n2)

{

arr[k] = R[j];

j++;

k++;

}

}

void mergesort(int arr[], int left, int right)

{

if (left < right)

{

int mid = left + (right - left) / 2;

mergesort(arr, left, mid);

mergesort(arr, mid + 1, right);

merge(arr, left, mid, right);

}

}

void printArray(int arr[], int n)

{

int i;

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

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

printf("\n");

}

int main(int argc, char const \*argv[])

{

// printf("Enter number of elements: ");

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

int arr[] = {28, 66, 16, 76, 71, 86, 94, 97, 56, 95};

int n = sizeof(arr) / sizeof(arr[0]);

// printf("Insert elements\n");

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

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

printArray(arr, n);

printf("Sorted array: ");

mergesort(arr, 0, n - 1);

printArray(arr, n);

return 0;

}

Output:

