DSA Assignment

Nalini.Nerusu

Class:CSE-F

Roll no:Ap19110010409

1. Write a program for the Insertion sort algorithm.

```
#include <stdio.h>

int main()

{

  int n, array[1000], c, d, t, flag = 0;

  printf("Enter number of elements\n");

  scanf("%d", &n);

  printf("Enter %d integers\n", n);

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

    scanf("%d", &array[c]);

  for (c = 1 ; c <= n - 1; c++) {

    t = array[c];

    for (d = c - 1 ; d >= 0; d--) {

      if (array[d] > t) {

        array[d+1] = array[d];

        flag = 1;
      }
      else
        break;
    }
    if (flag)
      array[d+1] = t;
  }
  printf("Sorted list in ascending order:\n");
  for (c = 0; c <= n - 1; c++) {
    printf("%d\n", array[c]);
  }
  return 0;
}
```

Output:

```
Enter number of elements
5
Enter 5 integers
12 1 34 65 24
Sorted list in ascending order:
1
12
24
34
65
```

2. Write a program for the Selection sort algorithm.

```
#include <stdio.h>
int main()
```

```
{
  int array[100], n, c, d, position, t;
  printf("Enter number of elements\n");
  scanf("%d", &n);
  printf("Enter %d integers\n", n);
  for (c = 0; c < n; c++)
    scanf("%d", &array[c]);
  for (c = 0; c < (n - 1); c++) // finding minimum element (n-1) times
  {
    position = c;
    for (d = c + 1; d < n; d++)
    {
      if (array[position] > array[d])
        position = d;
    }
    if (position != c)
    {
      t = array[c];
      array[c] = array[position];
      array[position] = t;
    }
  }
  printf("Sorted list in ascending order:\n");

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

    printf("%d\n", array[c]);

  return 0;

}
```

Output:

```
Enter number of elements
5
Enter 5 integers
12
34
56
24
45
Sorted list in ascending order:
12
24
34
45
56
```

.

3. Write a program for the Bubble sort algorithm.

```
#include <stdio.h>

int main()


{

  int array[100], n, c, d, swap;

  printf("Enter number of elements\n");

  scanf("%d", &n);

  printf("Enter %d integers\n", n);
  for (c = 0; c < n; c++)

    scanf("%d", &array[c]);

  for (c = 0 ; c < n - 1; c++)

 {

    for (d = 0 ; d < n - c - 1; d++)

    {

      if (array[d] > array[d+1]) /* For decreasing order use < */

      {

        swap       = array[d];

        array[d]   = array[d+1];

        array[d+1] = swap;

      }

    }

  }

  printf("Sorted list in ascending order:\n");

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

     printf("%d\n", array[c]);

  return 0;

}
```

Output:

```
Enter number of elements

5 3

Enter 3 integers

34

12

1

Sorted list in ascending order:

1

12

34
```

4. Write a program for the Merge sort algorithm

```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);
      sort(mid+1, high);
      merging(low, mid, high);
   } else {
      return;
   }
}

int main() {
   int i;

   printf("List before sorting\n");

   for(i = 0; i <= max; i++)
      printf("%d ", a[i]);

   sort(0, max);
```

```
  printf("\nList after sorting\n");

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

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

}
```

Output:

```
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
```