Bubble Sort Program in C

We shall see the implementation of **bubble sort** in C programming language here

Implementation in C

```
#include <stdio.h>
#include <stdbool.h>

#define MAX 10

int list[MAX] = {1,8,4,6,0,3,5,2,7,9};

void display() {
   int i;
   printf("[");

   // navigate through all items
   for(i = 0; i < MAX; i++) {
      printf("%d ",list[i]);
   }

   printf("]\n");
}

void bubbleSort() {
   int temp;
   int i,j;

   bool swapped = false;

   // loop through all numbers
   for(i = 0; i < MAX-1; i++) {
      swapped = false;

      // loop through numbers falling ahead
      for(j = 0; j < MAX-1-i; j++) {
         printf("     Items compared: [ %d, %d ] ", list[j],list[j+1]);

         // check if next number is lesser than current no
         //   swap the numbers.
         //  (Bubble up the highest number)

         if(list[j] > list[j+1]) {
            temp = list[j];
            list[j] = list[j+1];
            list[j+1] = temp;

            swapped = true;
            printf(" => swapped [%d, %d]\n",list[j],list[j+1]);
         } else {
            printf(" => not swapped\n");
         }

      }

      // if no number was swapped that means
      //  array is sorted now, break the loop.
      if(!swapped) {
         break;
      }

      printf("Iteration %d#: ",(i+1));
      display();
   }

}

void main() {
```

```
   printf("Input Array: ");
   display();
   printf("\n");

   bubbleSort();
   printf("\nOutput Array: ");
   display();
}
```

If we compile and run the above program, it will produce the following result –

Output

```
Input Array: [1 8 4 6 0 3 5 2 7 9 ]
     Items compared: [ 1, 8 ]  => not swapped
     Items compared: [ 8, 4 ]  => swapped [4, 8]
     Items compared: [ 8, 6 ]  => swapped [6, 8]
     Items compared: [ 8, 0 ]  => swapped [0, 8]
     Items compared: [ 8, 3 ]  => swapped [3, 8]
     Items compared: [ 8, 5 ]  => swapped [5, 8]
     Items compared: [ 8, 2 ]  => swapped [2, 8]
     Items compared: [ 8, 7 ]  => swapped [7, 8]
     Items compared: [ 8, 9 ]  => not swapped
Iteration 1#: [1 4 6 0 3 5 2 7 8 9 ]
     Items compared: [ 1, 4 ]  => not swapped
     Items compared: [ 4, 6 ]  => not swapped
     Items compared: [ 6, 0 ]  => swapped [0, 6]
     Items compared: [ 6, 3 ]  => swapped [3, 6]
     Items compared: [ 6, 5 ]  => swapped [5, 6]
     Items compared: [ 6, 2 ]  => swapped [2, 6]
     Items compared: [ 6, 7 ]  => not swapped
     Items compared: [ 7, 8 ]  => not swapped
Iteration 2#: [1 4 0 3 5 2 6 7 8 9 ]
     Items compared: [ 1, 4 ]  => not swapped
     Items compared: [ 4, 0 ]  => swapped [0, 4]
     Items compared: [ 4, 3 ]  => swapped [3, 4]
     Items compared: [ 4, 5 ]  => not swapped
     Items compared: [ 5, 2 ]  => swapped [2, 5]
     Items compared: [ 5, 6 ]  => not swapped
     Items compared: [ 6, 7 ]  => not swapped
Iteration 3#: [1 0 3 4 2 5 6 7 8 9 ]
     Items compared: [ 1, 0 ]  => swapped [0, 1]
     Items compared: [ 1, 3 ]  => not swapped
     Items compared: [ 3, 4 ]  => not swapped
     Items compared: [ 4, 2 ]  => swapped [2, 4]
     Items compared: [ 4, 5 ]  => not swapped
     Items compared: [ 5, 6 ]  => not swapped
Iteration 4#: [0 1 3 2 4 5 6 7 8 9 ]
     Items compared: [ 0, 1 ]  => not swapped
     Items compared: [ 1, 3 ]  => not swapped
     Items compared: [ 3, 2 ]  => swapped [2, 3]
     Items compared: [ 3, 4 ]  => not swapped
     Items compared: [ 4, 5 ]  => not swapped
Iteration 5#: [0 1 2 3 4 5 6 7 8 9 ]
     Items compared: [ 0, 1 ]  => not swapped
     Items compared: [ 1, 2 ]  => not swapped
     Items compared: [ 2, 3 ]  => not swapped
     Items compared: [ 3, 4 ]  => not swapped

Output Array: [0 1 2 3 4 5 6 7 8 9 ]
```

12/5/2020

Selection Sort Program in C

Selection sort is a simple sorting algorithm. This sorting algorithm is an in-place comparison-based algorithm in which the list is divided into two parts, the sorted part at the left end and the unsorted part at the right end. Initially, the sorted part is empty and the unsorted part is the entire list.

Implementation in C

```
#include <stdio.h>
#include <stdbool.h>

#define MAX 7

int intArray[MAX] = {4,6,3,2,1,9,7};

void printline(int count) {
   int i;

   for(i = 0;i < count-1;i++) {
      printf("=");
   }

   printf("=\n");
}

void display() {
   int i;
   printf("[");

   // navigate through all items
   for(i = 0;i < MAX;i++) {
      printf("%d ", intArray[i]);
   }

   printf("]\n");
}

void selectionSort() {
   int indexMin,i,j;

   // loop through all numbers
   for(i = 0; i < MAX-1; i++) {

      // set current element as minimum
      indexMin = i;

      // check the element to be minimum
      for(j = i+1;j < MAX;j++) {
         if(intArray[j] < intArray[indexMin]) {
            indexMin = j;
         }
      }

      if(indexMin != i) {
         printf("Items swapped: [ %d, %d ]\n" , intArray[i], intArray[indexMin]);

         // swap the numbers
         int temp = intArray[indexMin];
         intArray[indexMin] = intArray[i];
         intArray[i] = temp;
      }

      printf("Iteration %d#:",(i+1));
      display();
   }
}

void main() {
```

```
   printf("Input Array: ");
   display();
   printline(50);
   selectionSort();
   printf("Output Array: ");
   display();
   printline(50);
}
```

If we compile and run the above program, it will produce the following result –

Output

```
Input Array: [4 6 3 2 1 9 7 ]
==================================================
Items swapped: [ 4, 1 ]
Iteration 1#:[1 6 3 2 4 9 7 ]
Items swapped: [ 6, 2 ]
Iteration 2#:[1 2 3 6 4 9 7 ]
Iteration 3#:[1 2 3 6 4 9 7 ]
Items swapped: [ 6, 4 ]
Iteration 4#:[1 2 3 4 6 9 7 ]
Iteration 5#:[1 2 3 4 6 9 7 ]
Items swapped: [ 9, 7 ]
Iteration 6#:[1 2 3 4 6 7 9 ]
Output Array: [1 2 3 4 6 7 9 ]
```