Code(Bubble Sort)

```
#include <iostream>
using namespace std;

void fillArray(int* a, int size, int& numberUsed);
void swapElements(int a[], int maxPos, int last);
void bubbleSortPhase(int a[], int last);
void bubbleSort(int a[], int n);

const int NSIZE = 10;

int main() {

  cout << "Sorting program" << endl;
  int sampleArray[NSIZE], numberUsed;
  fillArray(sampleArray, NSIZE, numberUsed);
  bubbleSort(sampleArray, numberUsed);

  cout << "Sorted results" << endl;
  for (int index = 0; index < numberUsed; index++)
    cout << sampleArray[index] << " ";
  cout << endl;

}

void fillArray(int* a, int size, int& numberUsed)
{
  cout << " enter up to " << size << " nonnegative whole numbers" << endl
    << "Mark the end of the list with a negative number" << endl;
  int next, index = 0;
  cin >> next;
  while ((next >= 0) && (index < size))
  {
    a[index] = next;
    index++;
    cin >> next;
  }
  numberUsed = index;
}

void swapElements(int a[], int maxPos, int last) {
  int temp = a[maxPos];
  a[maxPos] = a[last];
  a[last] = temp;
}


void bubbleSortPhase(int a[], int last) {
  // Precondition: a is in array indexed from a[0] to a[last]
  // Move the largest element between a[0] and a[last] into a[last]
  // by swapping out of order pairs
  int pos;
```

```
  for (pos = 0; pos < last; pos++) {
    if (a[pos] > a[pos + 1]) {
      swapElements(a, pos, pos + 1);
    }
    //Postconditions: a[0] ... a[last] contain the same elements
    //possibly reordered; a[last] >= a[0] ... a[last-1]
  }
}

void bubbleSort(int a[], int n) {
  // Precondition: a is an array indexed from a[0] to a[n-1]
  int i;
  for (i = n - 1; i > 0; i--) {
    bubbleSortPhase(a, i);
    cout << "Partially sorted results" << endl;
    for (int index = 0; index < n; index++)
      cout << a[index] << " ";
    cout << endl;
  }

  // Postcondition: a is sorted
}
```