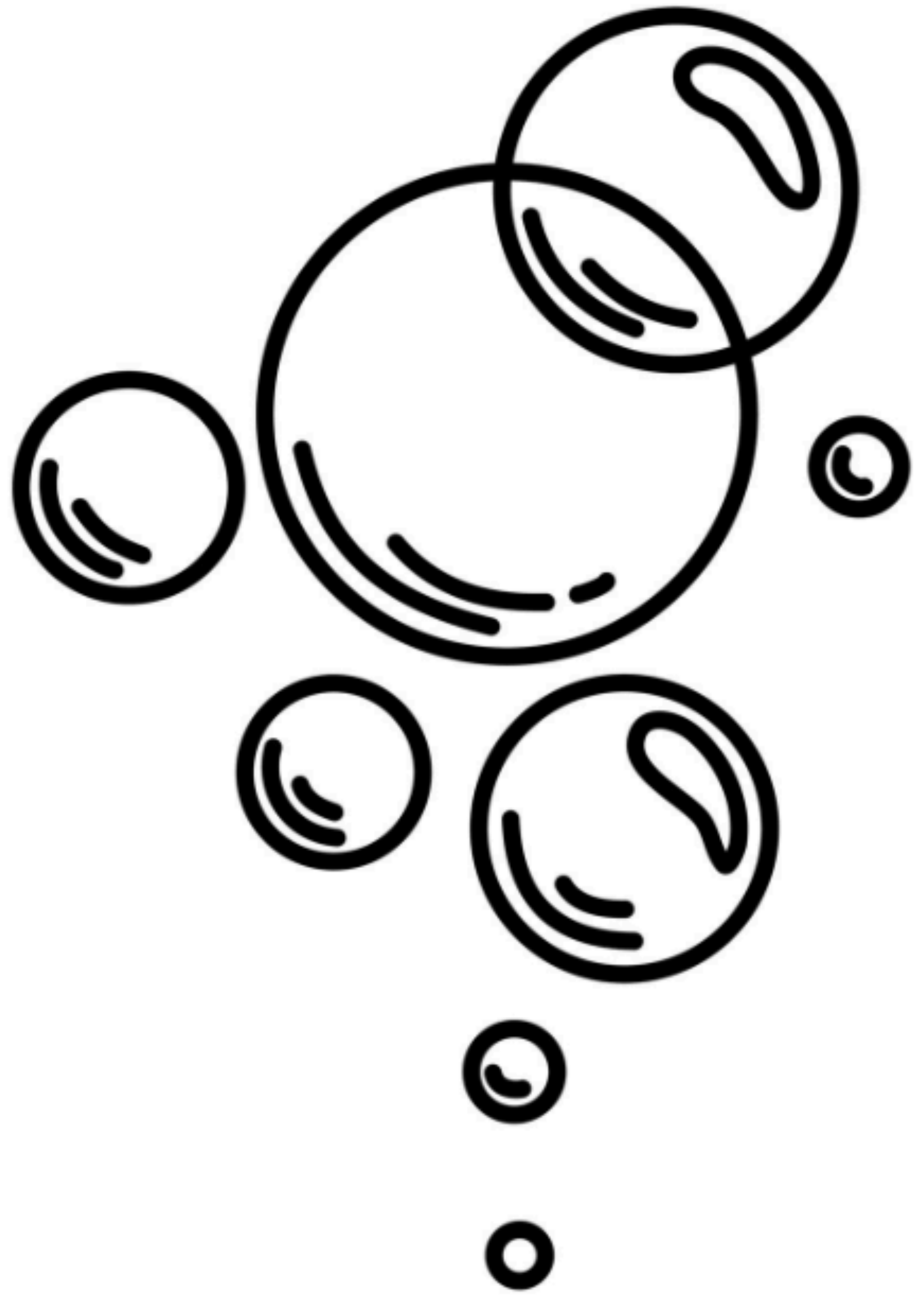
COSC 2436: Bubble Sort

J. Eoin Donovan - 2023

What is bubble sort?

Sorting algorithm that repeatdly swaps adjacent values if they are in wrong order. After every iteration, the largest value “bubbles” to the top.



BubbleSort - Code

```
for(int i = 0; i < size - 1; i++){
  for(int j = 0; j < size - i - 1; j++){
    if(arr[j] > arr[j+1]){
      swap(arr[j], arr[j+1]);
    }
  }
}
```

Bubble Sort - Enhanced Version

The enhanced version of bubble sort uses a boolean value that checks to see if any swaps take place. If no swaps occur in a given iteration, we know the array is sorted so we can break.

This will result in a best case time complexity of O(n) if the array is already sorted.

Enhanced BubbleSort - Code

```
int i = 0;
bool flag = false;
while(flag == false){
  flag = true;
  for(int j = 0; j < size - i - 1; j++){
    if(arr[j] > arr[j+1]){
      swap(arr[j], arr[j+1]);
      flag = false;
    }
  }
  i++;
}
```

Bubble Sort - Time Complexity

- Worst Case Time Complexity: $O(n^2)$
- Best Case Time Complexity: $O(n^2)$
- Enhanced Version Worst Case Time Complexity: $O(n^2)$
- Enhanced Version Best Case Time Complexity: $O(n)$

Bubble Sort for Linked List

```
void sort(){
  if(head == nullptr || head->next == nullptr)
    return;
  bool isSorted = false;
  node *cur;
  while(!isSorted){
    isSorted = true;
    cur = head;
    while(cur->next != nullptr){
      if(cur->data > cur->next->data){
        int tempData = cur->data;
        cur->data = cur->next->data;
        cur->next->data = tempData;
        isSorted = false;
      }
      cur = cur->next;
    }
  }
}
```