**Batch: A2 Roll No.:16010123032**

**Experiment / assignment / tutorial No. 11**

**Grade: AA / AB / BB / BC / CC / CD /DD**

**Signature of the Staff In-charge with date**

|  |
| --- |
| **Title:**  Implementation of sorting Algorithms. |

**Objective:** To Understand and Implement Bubble & Shell Sort

**Expected Outcome of Experiment:**

|  |  |
| --- | --- |
| **CO** | **Outcome** |
| 4 | Demonstrate sorting and searching methods. |

**Books/ Journals/ Websites referred:**

1. *Fundamentals Of Data Structures In C –* Ellis Horowitz, Satraj Sahni, Susan Anderson-Fred
2. *An Introduction to data structures with applications –* Jean Paul Tremblay,

Paul G. Sorenson

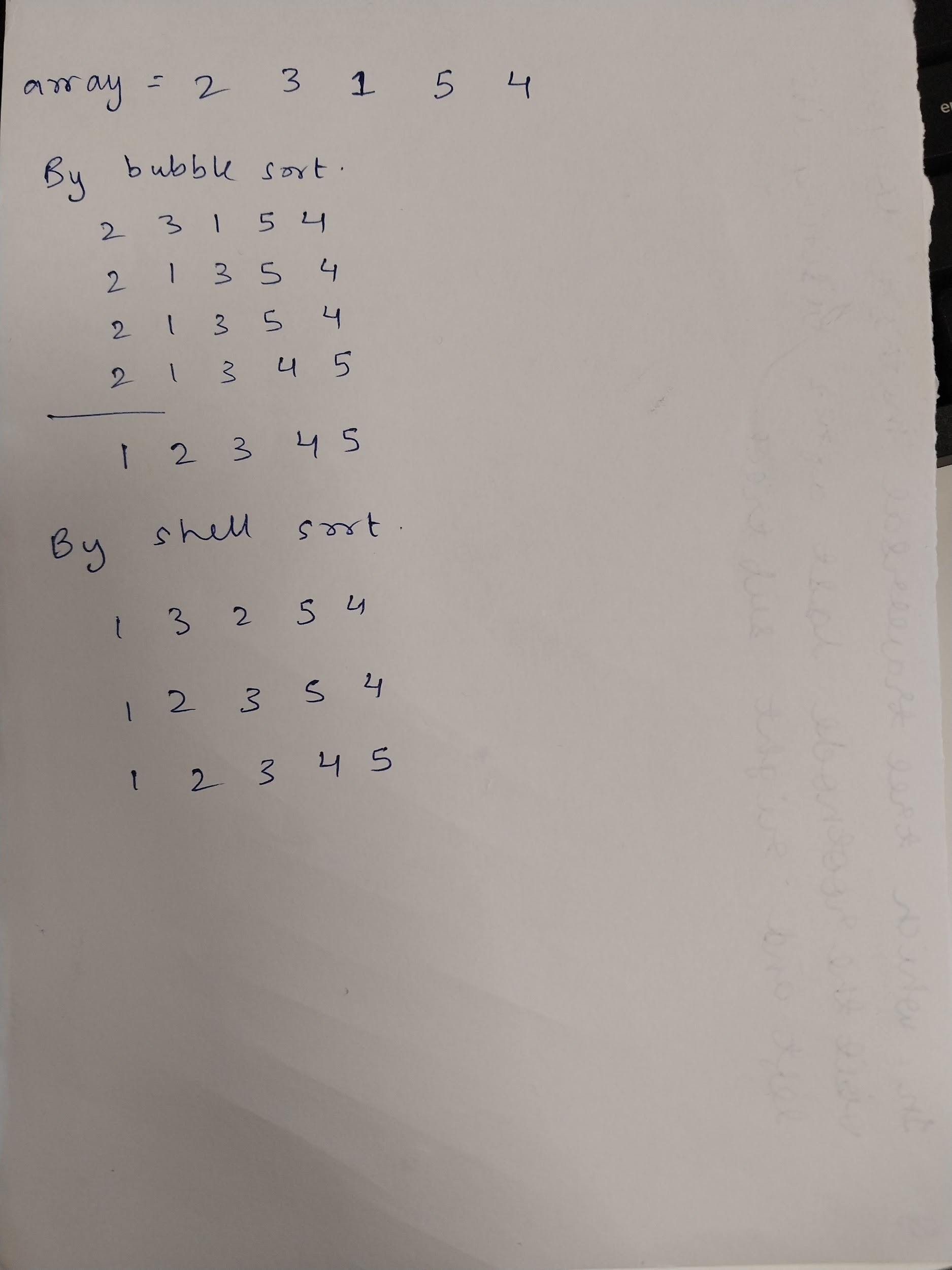
1. *Data Structures A Pseudo Approach with C –* Richard F. Gilberg & Behrouz A. Forouzan

**Abstract**: (Define sorting process, state applications of sorting)

**Example:**

*Take any random unsorted sequence of numbers and solve by using the Bubble and Shell Sort. Clearly showcase the sorted array after every pass.*

*The above is a pen-paper activity, take a picture of the solution and put it here.*



**Algorithm for Implementation:**

**Program:**

#include <stdio.h>

void Bubble(int ar[], int n) {

int i, j;

for (i = 0; i < n - 1; i++) {

for (j = 0; j < n - 1 - i; j++) {

if (ar[j] > ar[j + 1]) {

int temp = ar[j];

ar[j] = ar[j + 1];

ar[j + 1] = temp;

}

}

}

printf("\nSorted array by bubble sort: ");

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

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

}

printf("\n");

}

void Shell(int ar[], int n) {

int i, j, x;

int h = n;

while (h > 1) {

h = h / 2;

for (i = h; i <= n - 1; i++) {

j = i;

x = ar[i];

while (j > h - 1 && ar[j - h] > x) {

ar[j] = ar[j - h];

j = j - h;

}

ar[j] = x;

}

}

printf("\nSorted array by shell sort: ");

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

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

}

printf("\n");

}

void main() {

int n;

printf("Enter size: ");

scanf("%d", &n);

int ar[n];

printf("Enter elements\n");

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

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

}

printf("\nArray before sorting: ");

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

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

}

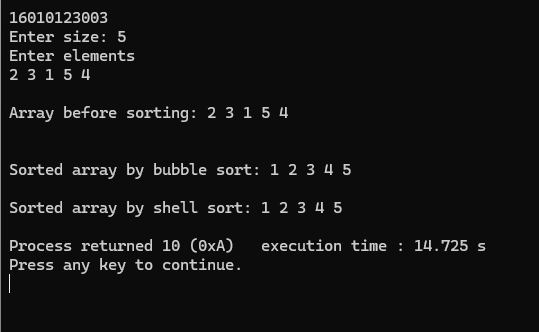
printf("\n\n");

Bubble(ar, n);

Shell(ar, n);

}

**Output screenshots:**

****

**Conclusion:-**

**Thus we implement bubble and shell sort.**

**Post Lab Questions:**

1. Describe how shell sort improves upon bubble sort. What are the main differences in their approaches?

Shell sort improves upon bubble sort by using gaps between elements to compare and move them, reducing the number of swaps and allowing faster movement of elements to their correct positions. While bubble sort compares only adjacent elements and has a time complexity of O(n²), Shell sort reduces the comparison scope with a shrinking gap sequence, often achieving better performance, sometimes close to O(n log n). This makes Shell sort more efficient, especially for larger or partially sorted data.

1. Explain the significance of the gap in shell sort. How does changing the gap sequence affect the performance of the algorithm?

The gap sequence directly influences both the number of comparisons and the number of element shifts in Shell sort.the most optimal sequence for a given problem may depend on the specific data set size and distribution.

1. In what scenarios would you choose shell sort over bubble sort? Discuss the types of datasets where shell sort performs better.

Choose Shell sort over Bubble sort for moderately large datasets, as it’s faster with better time complexity. It works well on partially sorted data or datasets with diverse values. Shell sort also handles small-to-medium arrays efficiently while being memory-friendly. It significantly outperforms Bubble sort, especially when space is limited but performance is needed.

1. Provide examples of real-world applications or scenarios where bubble sort or shell sort might be utilized, considering their characteristics.

* Bubble sort is often used in teaching basic sorting algorithms due to its simplicity and step-by-step swapping.
* Shell sort is useful when working with systems where cache locality is important, as it reduces cache misses compared to other algorithms.