**TRƯỜNG ĐẠI HỌC QUỐC GIA TP.HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

🙤🙧🟍🙥🙦



**Trình Bày Bài Tập Về Các Thuật Toán Sắp Xếp Và Tìm Kiếm**

* **Sinh Viên Trình Bày: Võ Nhật Bảo**
* **MSSV: 17520277**
* **Lớp: TMĐT2017**

## Môn: [Cấu trúc dữ liệu và giải thuật - IT003.I21](https://courses.uit.edu.vn/course/view.php?id=3799)

**Chọn Ví Dụ Dãy Số:**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **EX:** | **5** | **25** | **15** | **8** | **7** | **28** | **1** | **4** | **10** | **9** |

**Tìm Kiếm Tuyến Tính:**

int LinearSearch(int a[], int n, int x)

{

int i = 0; a[n] = x;

while (a[i] != x)

i++;

if (i == n)

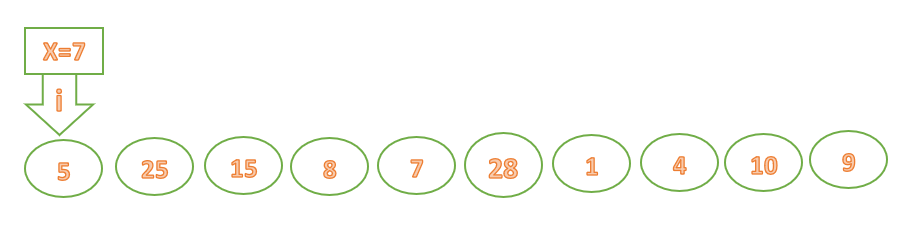
return 0;//Khong tim thay x

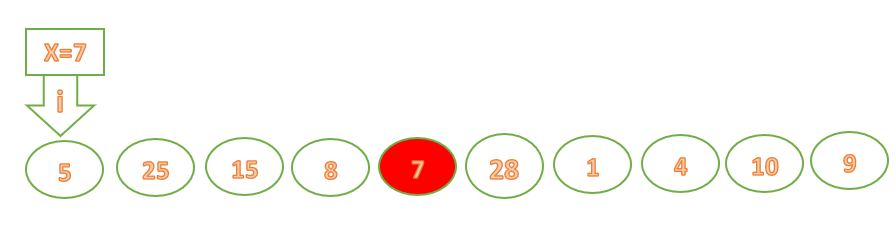
else

return 1;//Tim thay x

}

**Các Bước Thực Hiện: Tìm x=3 không tìm thấy x**

****

**Tìm x=7 tìm thấy x**

**Tìm Kiếm Nhị Phân:**

int BinarySearch(int a[], int n, int x)

{

int left, right, mid; left = 0;

right = n - 1;

do {

mid = (left + right) / 2;

if (a[mid] == x) return 1;

else if (a[mid]<x) left = mid - 1;

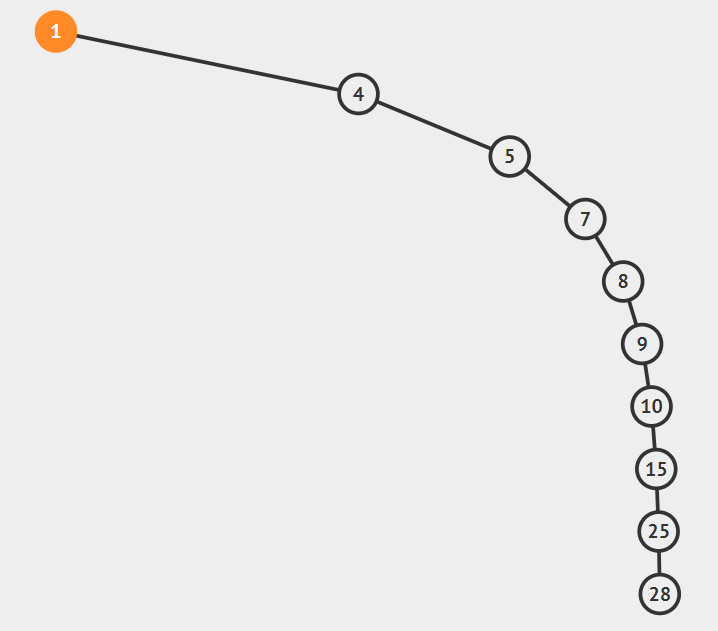
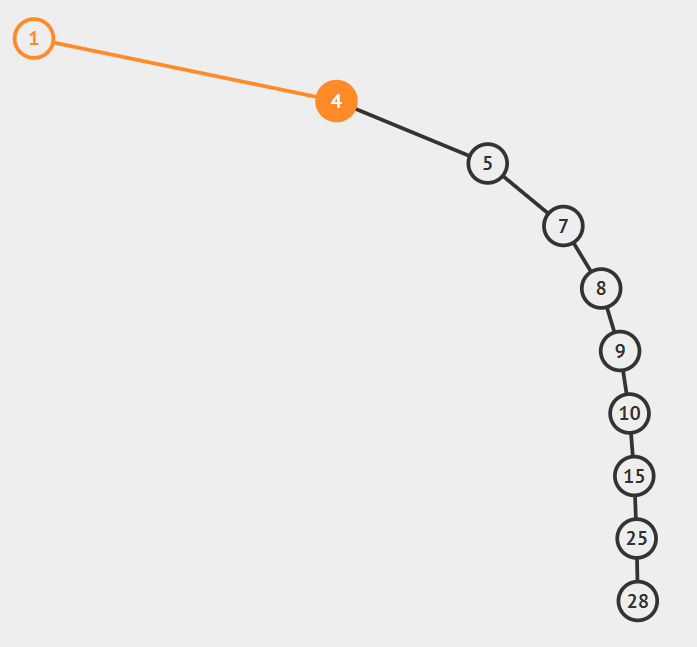
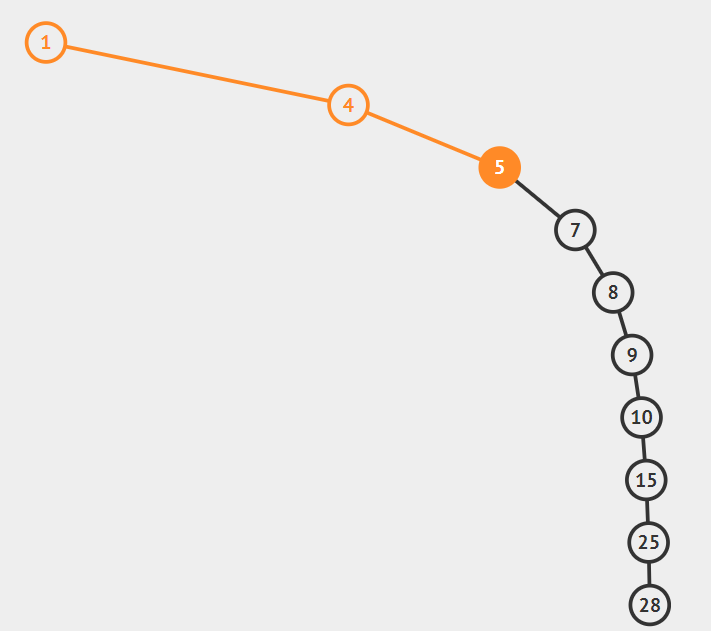
else if (a[mid]>x) right = mid + 1;

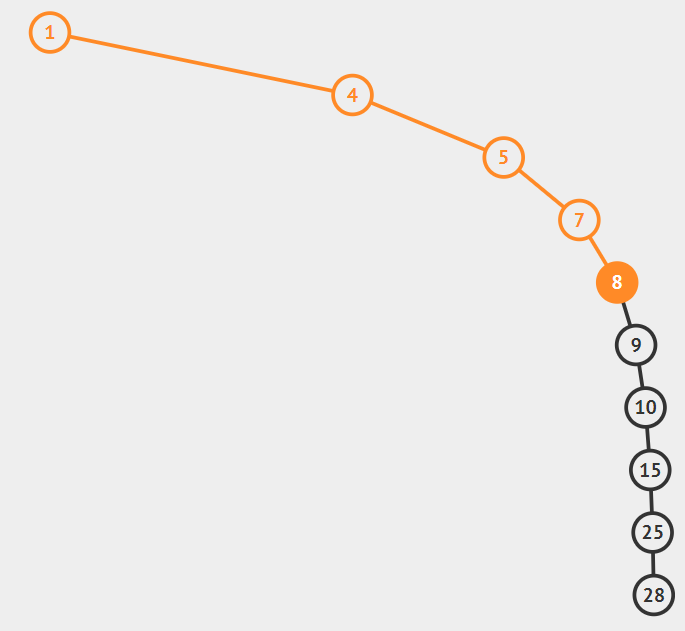
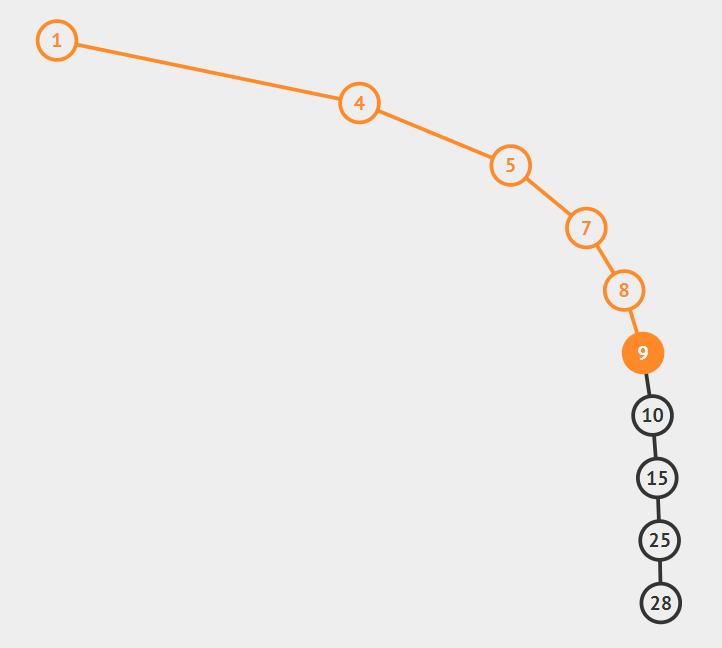
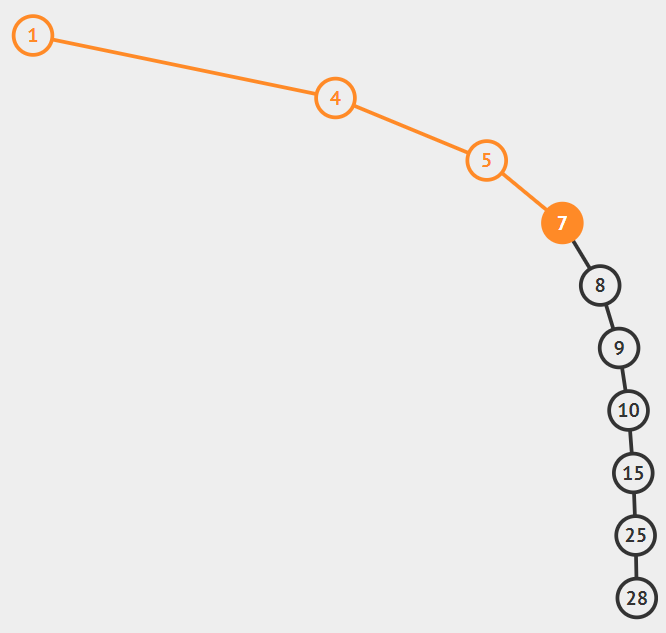
} while (left >= right);

return 0;

}

Tìm Kiếm X=9; khi đã sắp xếp các dẫy theo thứ tự tăng dần

Các Bước Thực Hiện:



**Tìm Thấy X=9 tại vị trí i=5.**

1. **Chức Năng Đổi chỗ trực tiếp – Interchange Sort**

**Code:**

void InterchangeSort(int a[], int n)

{

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

for (int j = 0; j < n; j++)

{

if (a[i] > a[j])

swap(a[i], a[j]);

}

}

**Các Bước Thực Hiện:**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  | **Step 1** |  |  |  |  |  |
| **5** | **25** | **15** | **8** | **7** | **28** | **1** | **4** | **10** | **9** |
| **1** | **25** | **15** | **8** | **7** | **28** | **5** | **4** | **10** | **9** |
|  |  |  |  | **Step 2** |  |  |  |  |  |
| **1** | **25** | **15** | **8** | **7** | **28** | **5** | **4** | **10** | **9** |
| **1** | **15** | **25** | **8** | **7** | **28** | **5** | **4** | **10** | **9** |
| **1** | **8** | **25** | **15** | **7** | **28** | **5** | **4** | **10** | **9** |
| **1** | **7** | **25** | **15** | **8** | **28** | **5** | **4** | **10** | **9** |
| **1** | **5** | **25** | **15** | **8** | **28** | **7** | **4** | **10** | **9** |
| **1** | **4** | **25** | **15** | **8** | **28** | **7** | **5** | **10** | **9** |
|  |  |  |  | **Step 3** |  |  |  |  |  |
| **1** | **4** | **25** | **15** | **8** | **28** | **7** | **5** | **10** | **9** |
| **1** | **4** | **15** | **25** | **8** | **28** | **7** | **5** | **10** | **9** |
| **1** | **4** | **8** | **25** | **15** | **28** | **7** | **5** | **10** | **9** |
| **1** | **4** | **7** | **25** | **15** | **28** | **8** | **5** | **10** | **9** |
| **1** | **4** | **5** | **25** | **15** | **28** | **8** | **7** | **10** | **9** |
|  |  |  |  | **Step 4** |  |  |  |  |  |
| **1** | **4** | **5** | **25** | **15** | **28** | **8** | **7** | **10** | **9** |
| **1** | **4** | **5** | **15** | **25** | **28** | **8** | **7** | **10** | **9** |
| **1** | **4** | **5** | **8** | **25** | **28** | **15** | **7** | **10** | **9** |
| **1** | **4** | **5** | **7** | **25** | **28** | **15** | **8** | **10** | **9** |
|  |  |  |  | **Step 5** |  |  |  |  |  |
| **1** | **4** | **5** | **7** | **25** | **28** | **15** | **8** | **10** | **9** |
| **1** | **4** | **5** | **7** | **15** | **28** | **25** | **8** | **10** | **9** |
| **1** | **4** | **5** | **7** | **8** | **28** | **25** | **15** | **10** | **9** |
|  |  |  |  | **Step 6** |  |  |  |  |  |
| **1** | **4** | **5** | **7** | **8** | **28** | **25** | **15** | **10** | **9** |
| **1** | **4** | **5** | **7** | **8** | **25** | **28** | **15** | **10** | **9** |
| **1** | **4** | **5** | **7** | **8** | **15** | **28** | **25** | **10** | **9** |
| **1** | **4** | **5** | **7** | **8** | **10** | **28** | **25** | **15** | **9** |
| **1** | **4** | **5** | **7** | **8** | **9** | **28** | **25** | **15** | **10** |
|  |  |  |  | **Step 7** |  |  |  |  |  |
| **1** | **4** | **5** | **7** | **8** | **9** | **28** | **25** | **15** | **10** |
| **1** | **4** | **5** | **7** | **8** | **9** | **25** | **28** | **15** | **10** |
| **1** | **4** | **5** | **7** | **8** | **9** | **15** | **28** | **25** | **10** |
| **1** | **4** | **5** | **7** | **8** | **9** | **10** | **28** | **25** | **15** |
|  |  |  |  | **Step 8** |  |  |  |  |  |
| **1** | **4** | **5** | **7** | **8** | **9** | **10** | **28** | **25** | **15** |
| **1** | **4** | **5** | **7** | **8** | **9** | **10** | **25** | **28** | **15** |
| **1** | **4** | **5** | **7** | **8** | **9** | **10** | **15** | **28** | **25** |
|  |  |  |  | **Step 9** |  |  |  |  |  |
| **1** | **4** | **5** | **7** | **8** | **9** | **10** | **15** | **28** | **25** |
| **1** | **4** | **5** | **7** | **8** | **9** | **10** | **15** | **25** | **28** |

* **Dãy đã sắp xếp theo thứ tự tăng dần**

1. **Chức Năng Chọn trực tiếp – Selection Sort**

**Code:**

void SelectionSort(int a[], int n)

{

int i, j, min\_idx;

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

{

min\_idx = i;

for (j = i + 1; j < n; j++)

if (a[j] > a[min\_idx])

min\_idx = j;

swap(a[min\_idx], a[i]);

}

}

**Các Bước Thực Hiện:**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  | **Step 1** |  |  |  |  |  |
| **5** | **25** | **15** | **8** | **7** | **28** | **1** | **4** | **10** | **9** |
| **1** | **25** | **15** | **8** | **7** | **28** | **5** | **4** | **10** | **9** |
|  |  |  |  | **Step 2** |  |  |  |  |  |
| **1** | **4** | **15** | **8** | **7** | **28** | **5** | **25** | **10** | **9** |
|  |  |  |  | **Step 3** |  |  |  |  |  |
| **1** | **4** | **5** | **8** | **7** | **28** | **15** | **25** | **10** | **9** |
|  |  |  |  |  |  |  |  |  |  |
| **1** | **4** | **5** | **7** | **8** | **28** | **15** | **25** | **10** | **9** |
|  |  |  |  | **Step 4** |  |  |  |  |  |
| **1** | **4** | **5** | **7** | **8** | **28** | **15** | **25** | **10** | **9** |
|  |  |  |  | **Step 5** |  |  |  |  |  |
| **1** | **4** | **5** | **7** | **8** | **9** | **15** | **25** | **10** | **28** |
|  |  |  |  | **Step 6** |  |  |  |  |  |
| **1** | **4** | **5** | **7** | **8** | **9** | **10** | **25** | **15** | **28** |
|  |  |  |  | **Step 7** |  |  |  |  |  |
| **1** | **4** | **5** | **7** | **8** | **9** | **10** | **15** | **25** | **28** |

* **Dãy đã sắp xếp theo thứ tự tăng dần**

1. **Chức Năng Nổi bọt – Bubble Sort**

**Code:**

void BubbleSort(int a[], int n)

{

int i, j;

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

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

{

if (a[j] > a[j + 1])

swap(a[j], a[j + 1]);

}

}

**Các Bước Thực Hiện:**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  | **Step 1** |  |  |  |  |  |
| **5** | **25** | **15** | **8** | **7** | **28** | **1** | **4** | **10** | **9** |
| **5** | **25** | **15** | **8** | **7** | **28** | **1** | **4** | **9** | **10** |
| **5** | **25** | **15** | **8** | **7** | **28** | **1** | **4** | **9** | **10** |
| **5** | **25** | **15** | **8** | **7** | **28** | **1** | **4** | **9** | **10** |
| **5** | **25** | **15** | **8** | **7** | **1** | **28** | **4** | **9** | **10** |
| **5** | **25** | **15** | **8** | **1** | **7** | **28** | **4** | **9** | **10** |
| **5** | **25** | **15** | **1** | **8** | **7** | **28** | **4** | **9** | **10** |
| **5** | **25** | **1** | **15** | **8** | **7** | **28** | **4** | **9** | **10** |
| **5** | **1** | **25** | **15** | **8** | **7** | **28** | **4** | **9** | **10** |
| **1** | **5** | **25** | **15** | **8** | **7** | **28** | **4** | **9** | **10** |
|  |  |  |  | **Step 2** |  |  |  |  |  |
| **1** | **5** | **25** | **15** | **8** | **7** | **28** | **4** | **9** | **10** |
| **1** | **5** | **25** | **15** | **8** | **7** | **28** | **4** | **9** | **10** |
| **1** | **5** | **25** | **15** | **8** | **7** | **28** | **4** | **9** | **10** |
| **1** | **5** | **25** | **15** | **8** | **7** | **4** | **28** | **9** | **10** |
| **1** | **5** | **25** | **15** | **8** | **4** | **7** | **28** | **9** | **10** |
| **1** | **5** | **25** | **15** | **4** | **8** | **7** | **28** | **9** | **10** |
| **1** | **5** | **25** | **4** | **15** | **8** | **7** | **28** | **9** | **10** |
| **1** | **5** | **4** | **25** | **15** | **8** | **7** | **28** | **9** | **10** |
| **1** | **4** | **5** | **25** | **15** | **8** | **7** | **28** | **9** | **10** |
|  |  |  |  | **Step 3** |  |  |  |  |  |
| **1** | **4** | **5** | **25** | **15** | **8** | **7** | **28** | **9** | **10** |
| **1** | **4** | **5** | **25** | **15** | **8** | **7** | **28** | **9** | **10** |
| **1** | **4** | **5** | **25** | **15** | **8** | **7** | **9** | **28** | **10** |
| **1** | **4** | **5** | **25** | **15** | **8** | **7** | **9** | **28** | **10** |
| **1** | **4** | **5** | **25** | **15** | **7** | **8** | **9** | **28** | **10** |
| **1** | **4** | **5** | **25** | **7** | **15** | **8** | **9** | **28** | **10** |
| **1** | **4** | **5** | **7** | **25** | **15** | **8** | **9** | **28** | **10** |
| **1** | **4** | **5** | **7** | **25** | **15** | **8** | **9** | **28** | **10** |
|  |  |  |  | **Step 4** |  |  |  |  |  |
| **1** | **4** | **5** | **7** | **25** | **15** | **8** | **9** | **28** | **10** |
| **1** | **4** | **5** | **7** | **25** | **15** | **8** | **9** | **10** | **28** |
| **1** | **4** | **5** | **7** | **25** | **15** | **8** | **9** | **10** | **28** |
| **1** | **4** | **5** | **7** | **25** | **15** | **8** | **9** | **10** | **28** |
| **1** | **4** | **5** | **7** | **25** | **8** | **15** | **9** | **10** | **28** |
| **1** | **4** | **5** | **7** | **8** | **25** | **15** | **9** | **10** | **28** |
|  |  |  |  | **Step 5** |  |  |  |  |  |
| **1** | **4** | **5** | **7** | **8** | **25** | **15** | **9** | **10** | **28** |
| **1** | **4** | **5** | **7** | **8** | **25** | **15** | **9** | **10** | **28** |
| **1** | **4** | **5** | **7** | **8** | **25** | **15** | **9** | **10** | **28** |
| **1** | **4** | **5** | **7** | **8** | **25** | **9** | **15** | **10** | **28** |
| **1** | **4** | **5** | **7** | **8** | **9** | **25** | **15** | **10** | **28** |
| **1** | **4** | **5** | **7** | **8** | **9** | **25** | **15** | **10** | **28** |
|  |  |  |  | **Step 6** |  |  |  |  |  |
| **1** | **4** | **5** | **7** | **8** | **9** | **25** | **15** | **10** | **28** |
| **1** | **4** | **5** | **7** | **8** | **9** | **25** | **15** | **10** | **28** |
| **1** | **4** | **5** | **7** | **8** | **9** | **25** | **10** | **15** | **28** |
| **1** | **4** | **5** | **7** | **8** | **9** | **10** | **25** | **15** | **28** |
| **1** | **4** | **5** | **7** | **8** | **9** | **10** | **25** | **15** | **28** |
|  |  |  |  | **Step 7** |  |  |  |  |  |
| **1** | **4** | **5** | **7** | **8** | **9** | **10** | **25** | **15** | **28** |
| **1** | **4** | **5** | **7** | **8** | **9** | **10** | **25** | **15** | **28** |
| **1** | **4** | **5** | **7** | **8** | **9** | **10** | **15** | **25** | **28** |
| **1** | **4** | **5** | **7** | **8** | **9** | **10** | **15** | **25** | **28** |

* **Dãy đã sắp xếp theo thứ tự tăng dần**

**4. Chức Năng Shaker Sort**

**Code**

void ShakeSort(int a[], int n)

{

int l, r, k;

l = 0; r = n - 1; k = n - 1;

while (l < r)

{

for (int j = r; j > l; j--)

if (a[j]< a[j - 1])

{

swap(a[j], a[j - 1]);

k = j;

}

l = k;

for (int j = l; j < r; j++)

if (a[j]> a[j + 1])

{

swap(a[j], a[j + 1]);

k = j;

}

r = k;

}

} **Các Bước Thực Hiện Giống Bubble Sort.**

1. **Chức Năng Chèn trực tiếp – Insertion Sort**

**Code:**

void InsertionSort(int a[], int n)

{

int i, key, j;

for (i = 1; i < n; i++)

{

key = a[i];

j = i - 1;

while (j >= 0 && a[j] > key)

{

a[j + 1] = a[j];

j = j - 1;

}

a[j + 1] = key;

}

}

**Các Bước Thực Hiện:**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **5** | **25** | **15** | **8** | **7** | **28** | **1** | **4** | **10** | **9** |
| **5** | **25** | **15** | **8** | **7** | **28** | **1** | **4** | **10** | **9** |
| **5** | **15** | **25** | **8** | **7** | **28** | **1** | **4** | **10** | **9** |
| **5** | **8** | **15** | **25** | **7** | **28** | **1** | **4** | **10** | **9** |
| **5** | **7** | **8** | **15** | **25** | **28** | **1** | **4** | **10** | **9** |
| **5** | **7** | **8** | **15** | **25** | **28** | **1** | **4** | **10** | **9** |
| **1** | **5** | **7** | **8** | **15** | **25** | **28** | **4** | **10** | **9** |
| **1** | **4** | **5** | **7** | **8** | **15** | **25** | **28** | **10** | **9** |
| **1** | **4** | **5** | **7** | **8** | **10** | **15** | **25** | **28** | **9** |
| **1** | **4** | **5** | **7** | **8** | **9** | **10** | **15** | **25** | **28** |
| **1** | **4** | **5** | **7** | **8** | **9** | **10** | **15** | **25** | **28** |

* **Dãy đã sắp xếp theo thứ tự tăng dần**

1. **Chức Năng Chèn nhị phân – Binary Insertion Sort**

**Code:**

void BInsertionSort(int a[], int n)

{

int l, r, m, i;

int x;

for (int i = 1; i < n; i++)

{

x = a[i];

l = 0;

r = i;

while (l <= r)

{

m = (l + r) / 2;

if (x < a[m])

r = m - 1;

else

l = m + 1;

}

for (int j = i - 1; j >= l; j--)

a[j + 1] = a[j];

a[l] = x;

}

}

**Các Bước Thực Hiện Giống Insertion Sort**

1. **Chức Năng Shell Sort**

**Code:**

void ShellSort(int a[], int n)

{

for (int gap = n / 2; gap > 0; gap /= 2)

{

for (int i = gap; i < n; i += 1)

{

int temp = a[i];

int j;

for (j = i; j >= gap && a[j - gap] > temp; j -= gap)

a[j] = a[j - gap];

a[j] = temp;

}

}

}

**Các Bước Thực Hiện:**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  | **Step 1** |  |  |  |  |  |
| **5** | **25** | **15** | **8** | **7** | **28** | **1** | **4** | **10** | **9** |
| **1** | **4** | **10** | **8** | **7** | **28** | **5** | **25** | **15** | **9** |
| **1** | **4** | **10** | **5** | **7** | **15** | **8** | **25** | **28** | **9** |
|  |  |  |  | **Step 2** |  |  |  |  |  |
| **1** | **4** | **10** | **5** | **7** | **15** | **8** | **25** | **28** | **9** |
| **1** | **4** | **10** | **5** | **7** | **15** | **8** | **25** | **28** | **9** |
| **1** | **4** | **5** | **10** | **7** | **15** | **8** | **25** | **28** | **9** |
| **1** | **4** | **5** | **7** | **10** | **15** | **8** | **25** | **28** | **9** |
| **1** | **4** | **5** | **7** | **10** | **15** | **8** | **25** | **28** | **9** |
| **1** | **4** | **5** | **7** | **8** | **10** | **15** | **25** | **28** | **9** |
| **1** | **4** | **5** | **7** | **8** | **10** | **15** | **25** | **28** | **9** |
| **1** | **4** | **5** | **7** | **8** | **10** | **15** | **25** | **28** | **9** |
| **1** | **4** | **5** | **7** | **8** | **9** | **10** | **15** | **25** | **28** |
| **1** | **4** | **5** | **7** | **8** | **9** | **10** | **15** | **25** | **28** |

* **Dãy đã sắp xếp theo thứ tự tăng dần.**

1. **Chức Năng Heap Sort**

**Code:**

void HeapSort(int a[], int n)

{

int k, x, s, f, ivalue;

for (k = 1; k<n; k++)

{

x = a[k];

s = k;

f = (s - 1) / 2;

while (s>0 && a[f]<x)

{

a[s] = a[f];

s = f;

f = (s - 1) / 2;

}

a[s] = x;

}

for (k = n - 1; k>0; k--)

{

ivalue = a[k];

a[k] = a[0];

f = 0;

if (k == 1)

s = -1;

else

s = 1;

if (k>2 && a[2]>a[1])

s = 2;

while (s >= 0 && ivalue<a[s])

{

a[f] = a[s];

f = s;

s = 2 \* f + 1;

if (s + 1 <= k - 1 && a[s]<a[s + 1])

s = s + 1;

if (s>k - 1)

s = -1;

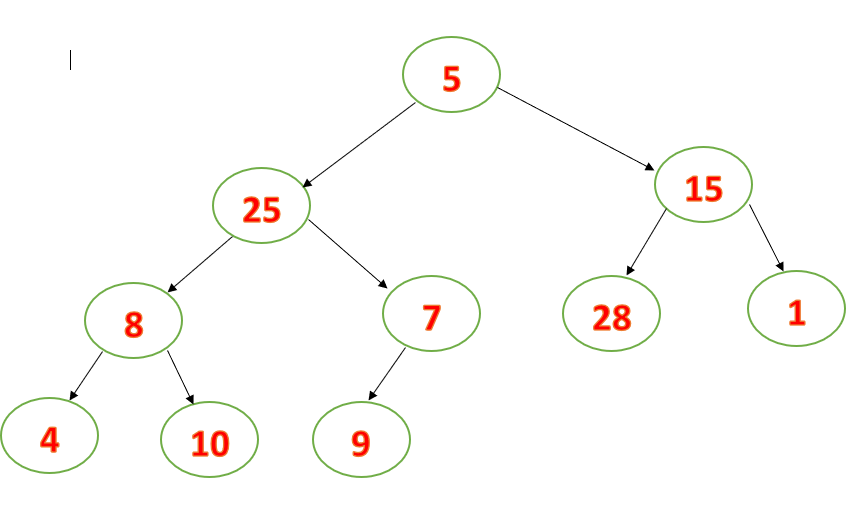
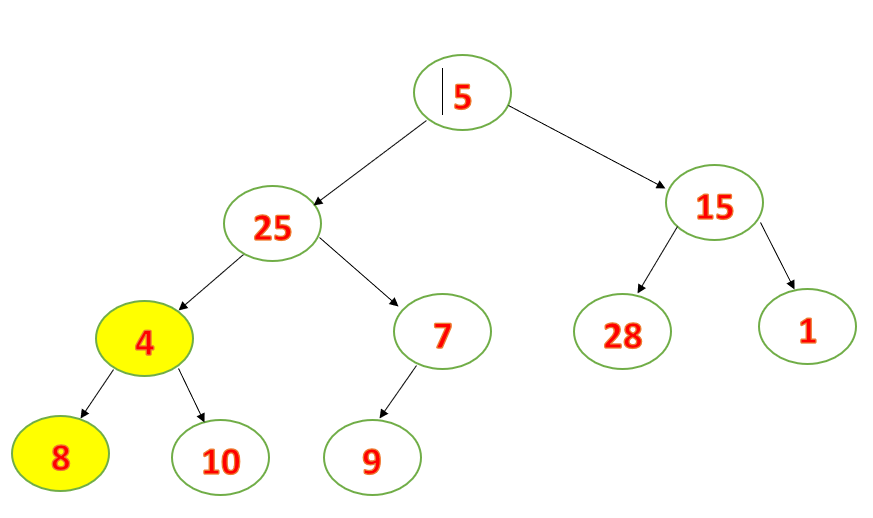
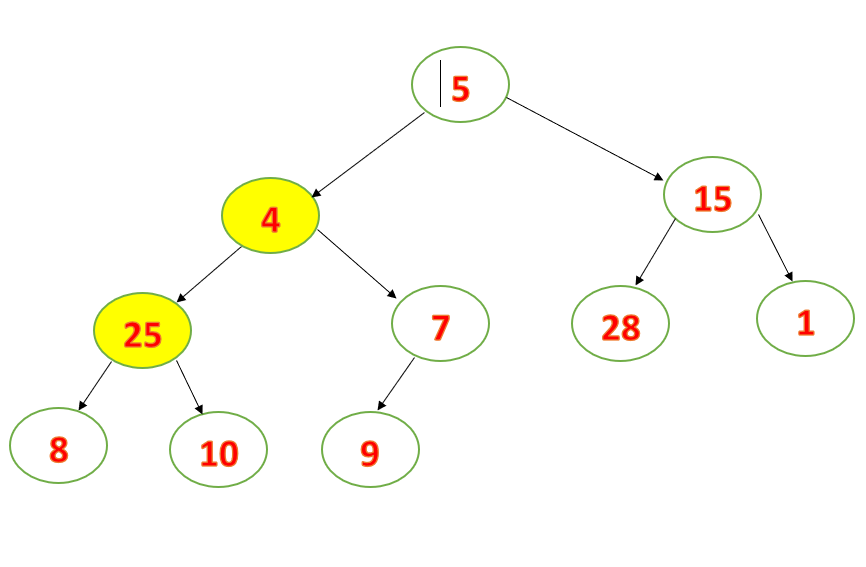
}

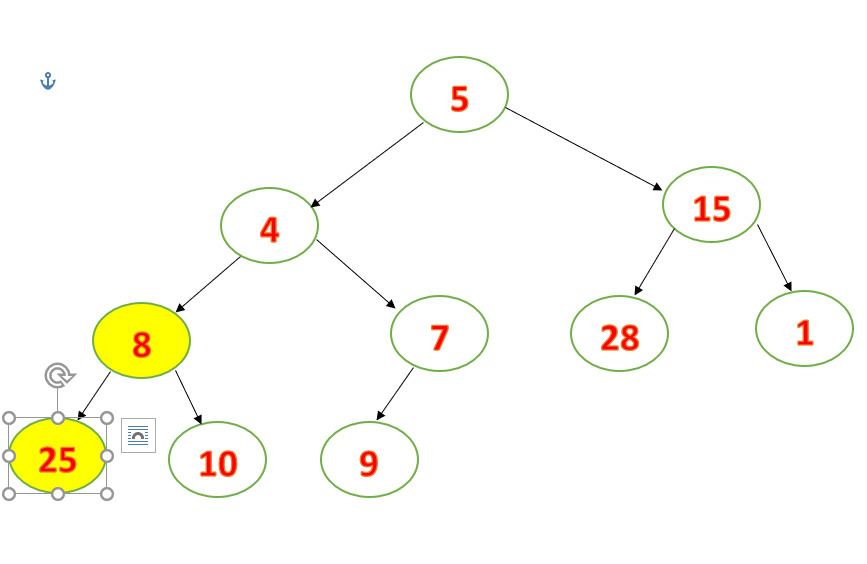
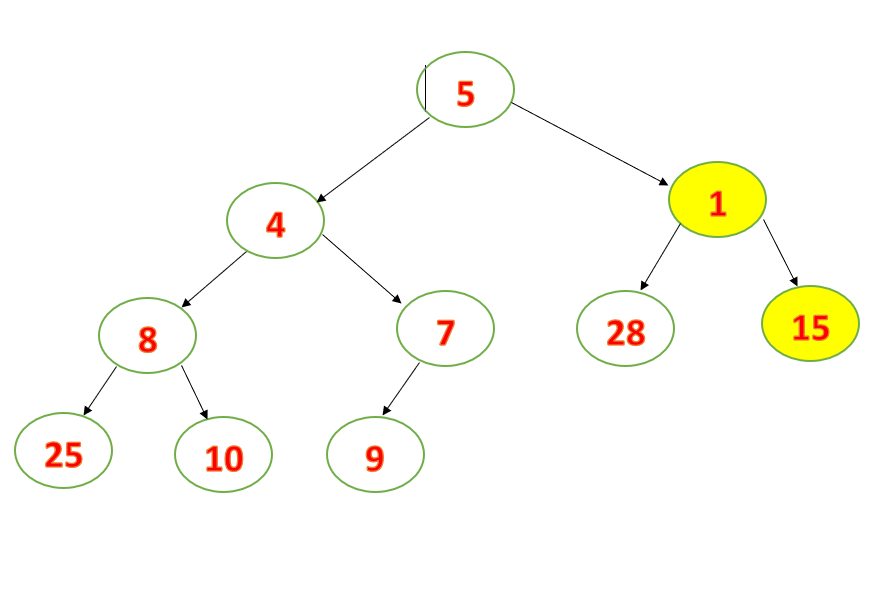
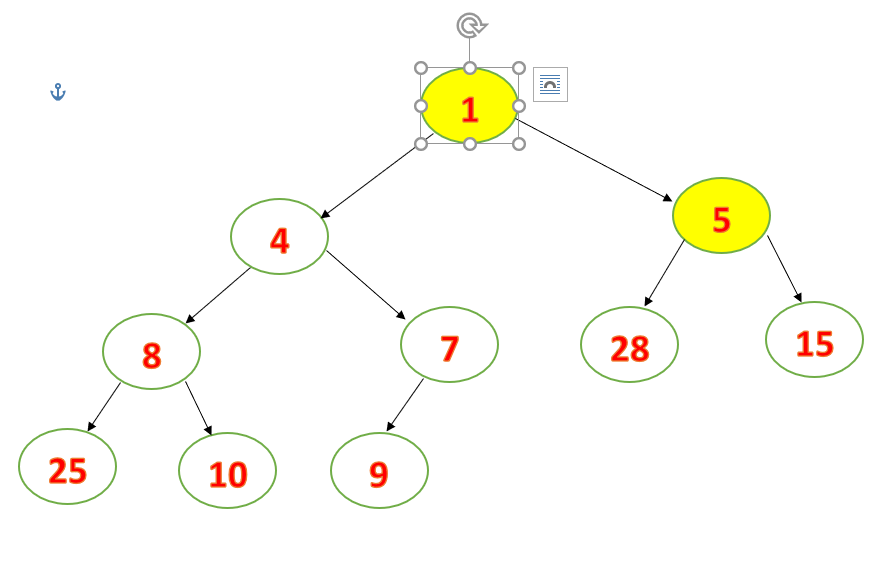
a[f] = ivalue;

}

}

**Các Bước Thực Hiện:**

****

****

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** |
| **KQ:** | **1** | **4** | **5** | **7** | **8** | **9** | **10** | **15** | **25** | **28** |

* **Dãy đã sắp xếp theo thứ tự tăng dần.**

1. **Chức Năng Quick Sort**

**Code:**

void QuickSort(int a[], int left, int right)

{

int i, j, x, y;

i = left; j = right;

x = a[(left + right) / 2];

do

{

while (a[i]<x) i++;

while (a[j]>x) j--;

if (i <= j)

{

swap(a[i], a[j]);

i++; j--;

}

} while (i < j);

if (left<j) QuickSort(a, left, j);

if (i<right) QuickSort(a, i, right);

}

void Quick\_Sort(int a[], int n)

{

QuickSort(a, 0, n - 1);

}

**Các Bước Thực Hiện:**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **5** | **25** | **15** | **8** | **7** | **28** | **1** | **4** | **10** | **9** |
| **X=5, [1,10]** | **5** | **25** | **15** | **8** | **7** | **28** | **1** | **4** | **10** | **9** |
|  | **5** | **4** | **15** | **8** | **7** | **28** | **1** | **25** | **10** | **9** |
|  | **5** | **4** | **1** | **8** | **7** | **28** | **15** | **25** | **10** | **9** |
|  | **5** | **4** | **1** | **7** | **8** | **28** | **15** | **25** | **10** | **9** |
|  | **5** | **4** | **1** | **7** | **8** | **28** | **15** | **25** | **10** | **9** |
| **X=2 ,[1,3]** | **5** | **4** | **1** | **7** | **8** | **28** | **15** | **25** | **10** | **9** |
|  | **1** | **4** | **5** | **7** | **8** | **28** | **15** | **25** | **10** | **9** |
| **X=7,[5,10]** | **1** | **4** | **5** | **7** | **8** | **28** | **15** | **25** | **10** | **9** |
|  | **1** | **5** | **6** | **7** | **8** | **9** | **15** | **25** | **10** | **28** |
|  | **1** | **5** | **6** | **7** | **8** | **9** | **10** | **25** | **15** | **28** |
| **X=9,[8,10]** | **1** | **5** | **6** | **7** | **8** | **9** | **10** | **25** | **15** | **28** |
|  | **1** | **5** | **6** | **7** | **8** | **9** | **10** | **15** | **25** | **28** |
| **KQ:** | **1** | **5** | **6** | **7** | **8** | **9** | **10** | **15** | **25** | **28** |

* **Dãy đã sắp xếp theo thứ tự tăng dần**

**10. Chức Năng Merge Sort**

**Code:**

void MergeSort(int a[], int n)

{

int i, j, k, low1, up1, low2, up2, size;

int dstam[100]; size = 1;

while (size<n)

{

low1 = 0; k = 0;

while (low1 + size <n)

{

low2 = low1 + size;

up1 = low2 - 1;

if (low2 + size - 1< n)

up2 = low2 + size - 1;

else

up2 = n - 1;

for (i = low1, j = low2; i <= up1 && j <= up2; k++)

{

if (a[i] < a[j])

dstam[k] = a[i++];

else

dstam[k] = a[j++];

}

for (; i <= up1; k++)

dstam[k] = a[i++];

for (; j <= up2; k++)

dstam[k] = a[j++];

low1 = up2 + 1;

}

for (i = low1; k<n; i++)

dstam[k++] = a[i];

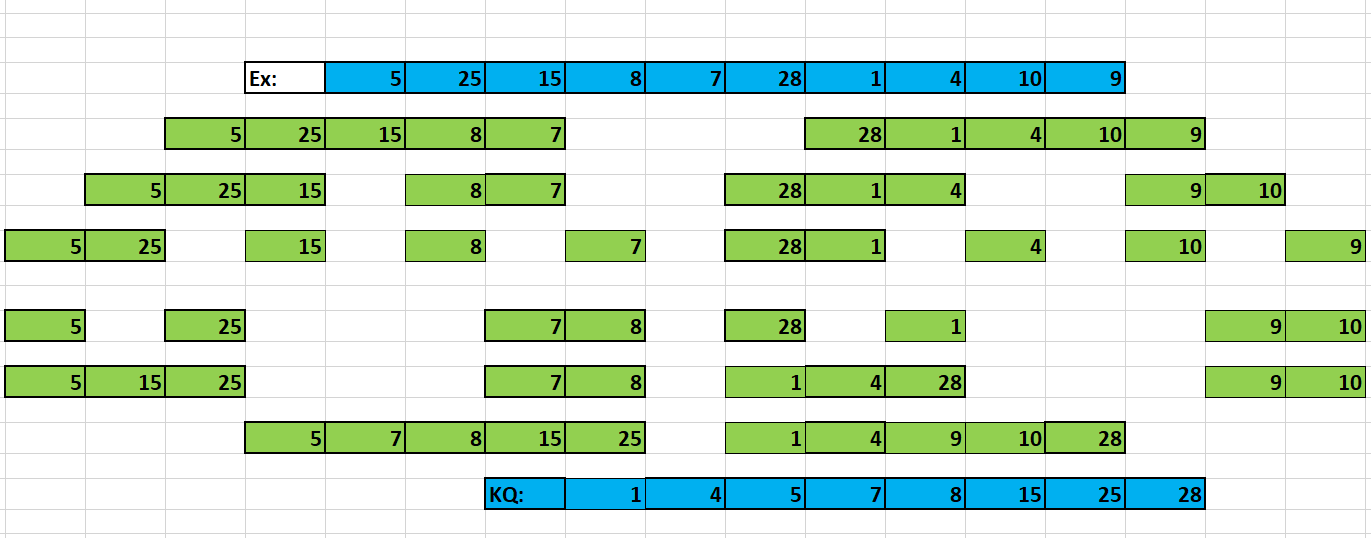
for (i = 0; i<n; i++)

a[i] = dstam[i];

size \*= 2;

}

}

**Các Bước Thực Hiện:**

1. **Chức Năng Radix Sort**

**Code:**

void RadixSort(int \*a, int n)

{

int i, b[100], m = a[0], exp = 1;

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

{

if (a[i] > m)

m = a[i];

}

while (m / exp > 0)

{

int bucket[10] = { 0 };

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

bucket[a[i] / exp % 10]++;

for (i = 1; i < 10; i++)

bucket[i] += bucket[i - 1];

for (i = n - 1; i >= 0; i--)

b[--bucket[a[i] / exp % 10]] = a[i];

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

a[i] = b[i];

exp \*= 10;

}

}

**Các Bước Thực Hiện:**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Ex:** | **5** | **25** | **15** | **8** | **7** | **28** | **1** | **4** | **10** | **9** |
|  |  |  |  |  | **Step 1** |  |  |  |  |  |
|  | **10** | **1** |  |  | **4** | **15 25 5** |  | **7** | **28 8** | **9** |
|  | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** |
|  |  |  |  |  | **Step 2** |  |  |  |  |  |
|  | **10** | **1** | **4** | **5** | **15** | **25** | **7** | **8** | **28** | **9** |
|  | **9**  **8 7**  **5 4 1** | **15 10** | **28 25** |  |  |  |  |  |  |  |
|  | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** |
|  |  |  |  |  | **Step 3** |  |  |  |  |  |
| **KQ:** | **1** | **4** | **5** | **7** | **8** | **9** | **10** | **15** | **25** | **28** |

**Thực Hiện Sắp Xếp 11 Chức Năng Theo Thứ Tự Giảm Dần**

1. **Chức Năng Đổi chỗ trực tiếp – Interchange Sort**

**Code:**

void InterchangeSort(int a[], int n)

{

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

for (int j = 0; j < n; j++)

{

if (a[i] > a[j])

swap(a[i], a[j]);

}

}

**Các Bước Thực Hiện:**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  | **Step 1** |  |  |  |  |  |
| **5** | **25** | **15** | **8** | **7** | **28** | **1** | **4** | **10** | **9** |
| **25** | **5** | **15** | **8** | **7** | **28** | **1** | **4** | **10** | **9** |
| **28** | **5** | **15** | **8** | **7** | **25** | **1** | **4** | **10** | **9** |
|  |  |  |  | **Step 2** |  |  |  |  |  |
| **28** | **5** | **15** | **8** | **7** | **25** | **1** | **4** | **10** | **9** |
| **28** | **15** | **5** | **8** | **7** | **25** | **1** | **4** | **10** | **9** |
| **28** | **25** | **5** | **8** | **7** | **15** | **1** | **4** | **10** | **9** |
|  |  |  |  | **Step 3** |  |  |  |  |  |
| **28** | **25** | **5** | **8** | **7** | **15** | **1** | **4** | **10** | **9** |
| **28** | **25** | **8** | **5** | **7** | **15** | **1** | **4** | **10** | **9** |
| **28** | **25** | **15** | **5** | **7** | **8** | **1** | **4** | **10** | **9** |
|  |  |  |  | **Step 4** |  |  |  |  |  |
| **28** | **25** | **15** | **5** | **7** | **8** | **1** | **4** | **10** | **9** |
| **28** | **25** | **15** | **7** | **5** | **8** | **1** | **4** | **10** | **9** |
| **28** | **25** | **15** | **8** | **5** | **7** | **1** | **4** | **10** | **9** |
| **28** | **25** | **15** | **10** | **5** | **7** | **1** | **4** | **8** | **9** |
|  |  |  |  | **Step 5** |  |  |  |  |  |
| **25** | **25** | **15** | **10** | **5** | **7** | **1** | **4** | **8** | **9** |
| **28** | **25** | **15** | **10** | **7** | **5** | **1** | **4** | **8** | **9** |
| **28** | **25** | **15** | **10** | **8** | **5** | **1** | **4** | **7** | **9** |
| **28** | **25** | **15** | **10** | **9** | **5** | **1** | **4** | **7** | **8** |
|  |  |  |  | **Step 6** |  |  |  |  |  |
| **28** | **25** | **15** | **10** | **9** | **5** | **1** | **4** | **7** | **8** |
| **28** | **25** | **15** | **10** | **9** | **7** | **1** | **4** | **5** | **8** |
| **28** | **25** | **15** | **10** | **9** | **8** | **1** | **4** | **5** | **7** |
|  |  |  |  | **Step 7** |  |  |  |  |  |
| **28** | **25** | **15** | **10** | **9** | **8** | **1** | **4** | **5** | **7** |
| **28** | **25** | **15** | **10** | **9** | **8** | **4** | **1** | **5** | **7** |
| **28** | **25** | **15** | **10** | **9** | **8** | **5** | **1** | **4** | **7** |
| **28** | **25** | **15** | **10** | **9** | **8** | **7** | **1** | **4** | **5** |
|  |  |  |  | **Step 8** |  |  |  |  |  |
| **28** | **25** | **15** | **10** | **9** | **8** | **7** | **1** | **4** | **5** |
| **28** | **25** | **15** | **10** | **9** | **8** | **7** | **4** | **1** | **5** |
| **28** | **25** | **15** | **10** | **9** | **8** | **7** | **5** | **1** | **4** |
|  |  |  |  | **Step 8** |  |  |  |  |  |
| **28** | **25** | **15** | **10** | **9** | **8** | **7** | **5** | **1** | **4** |
| **28** | **25** | **15** | **10** | **9** | **8** | **7** | **5** | **4** | **1** |
| **28** | **25** | **15** | **10** | **9** | **8** | **7** | **5** | **4** | **1** |

* **Dãy đã xếp theo thứ tự giảm dần**

1. **Chức Năng Chọn trực tiếp – Selection Sort**

**Code:**

void SelectionSort(int a[], int n)

{

int i, j, min\_idx;

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

{

min\_idx = i;

for (j = i + 1; j < n; j++)

if (a[j] > a[min\_idx])

min\_idx = j;

swap(a[min\_idx], a[i]);

}

}

**Các Bước Thực Hiện:**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  | **Step 1** |  |  |  |  |  |
| **5** | **25** | **15** | **8** | **7** | **28** | **1** | **4** | **10** | **9** |
| **28** | **25** | **15** | **8** | **7** | **5** | **1** | **4** | **10** | **9** |
|  |  |  |  | **Step 2** |  |  |  |  |  |
| **28** | **25** | **15** | **8** | **7** | **5** | **1** | **4** | **10** | **9** |
|  |  |  |  | **Step 3** |  |  |  |  |  |
| **28** | **25** | **15** | **8** | **7** | **5** | **1** | **4** | **10** | **9** |
|  |  |  |  | **Step 4** |  |  |  |  |  |
| **28** | **25** | **15** | **10** | **7** | **5** | **1** | **4** | **8** | **9** |
|  |  |  |  | **Step 5** |  |  |  |  |  |
| **28** | **25** | **15** | **10** | **9** | **5** | **1** | **4** | **8** | **7** |
|  |  |  |  | **Step 6** |  |  |  |  |  |
| **28** | **25** | **15** | **10** | **9** | **8** | **1** | **4** | **5** | **7** |
|  |  |  |  | **Step 7** |  |  |  |  |  |
| **28** | **25** | **15** | **10** | **9** | **8** | **7** | **4** | **5** | **1** |
|  |  |  |  | **Step 8** |  |  |  |  |  |
| **28** | **25** | **15** | **10** | **9** | **8** | **7** | **5** | **4** | **1** |
| **28** | **25** | **15** | **10** | **9** | **8** | **7** | **5** | **4** | **1** |

* **Dãy đã sắp xếp theo thứ tự giảm dần**

1. **Chức Năng Nổi bọt – Bubble Sort**

**Code:**

void BubbleSort(int a[], int n)

{

int i, j;

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

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

{

if (a[j] < a[j + 1])

swap(a[j], a[j + 1]);

}

}

**Các Bước Thực Hiện**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  | **Step 1** |  |  |  |  |  |
| **5** | **25** | **15** | **8** | **7** | **28** | **1** | **4** | **10** | **9** |
| **5** | **25** | **15** | **8** | **7** | **28** | **1** | **4** | **10** | **9** |
| **5** | **25** | **15** | **8** | **7** | **28** | **1** | **10** | **4** | **9** |
| **5** | **25** | **15** | **8** | **7** | **28** | **10** | **1** | **4** | **9** |
| **5** | **25** | **15** | **8** | **7** | **28** | **10** | **1** | **4** | **9** |
| **5** | **25** | **15** | **8** | **28** | **7** | **10** | **1** | **4** | **9** |
| **5** | **25** | **15** | **28** | **8** | **7** | **10** | **1** | **4** | **9** |
| **5** | **25** | **28** | **15** | **8** | **7** | **10** | **1** | **4** | **9** |
| **5** | **28** | **25** | **15** | **8** | **7** | **10** | **1** | **4** | **9** |
| **28** | **5** | **25** | **15** | **8** | **7** | **10** | **1** | **4** | **9** |
|  |  |  |  | **Step 2** |  |  |  |  |  |
| **28** | **5** | **25** | **15** | **8** | **7** | **10** | **1** | **9** | **4** |
| **28** | **5** | **25** | **15** | **8** | **7** | **10** | **9** | **1** | **4** |
| **28** | **5** | **25** | **15** | **8** | **7** | **10** | **9** | **1** | **4** |
| **28** | **5** | **25** | **15** | **8** | **10** | **7** | **9** | **1** | **4** |
| **28** | **5** | **25** | **15** | **10** | **8** | **7** | **9** | **1** | **4** |
| **28** | **5** | **25** | **15** | **10** | **8** | **7** | **9** | **1** | **4** |
| **28** | **5** | **25** | **15** | **10** | **8** | **7** | **9** | **1** | **4** |
| **28** | **25** | **5** | **15** | **10** | **8** | **7** | **9** | **1** | **4** |
|  |  |  |  | **Step 3** |  |  |  |  |  |
| **28** | **25** | **5** | **15** | **10** | **8** | **7** | **9** | **4** | **1** |
| **28** | **25** | **5** | **15** | **10** | **8** | **7** | **9** | **4** | **1** |
| **28** | **25** | **5** | **15** | **10** | **8** | **9** | **7** | **4** | **1** |
| **28** | **25** | **5** | **15** | **10** | **9** | **8** | **7** | **4** | **1** |
| **28** | **25** | **5** | **15** | **10** | **9** | **8** | **7** | **4** | **1** |
| **28** | **25** | **5** | **15** | **10** | **9** | **8** | **7** | **4** | **1** |
| **28** | **25** | **15** | **5** | **10** | **9** | **8** | **7** | **4** | **1** |
|  |  |  |  | **Step 4** |  |  |  |  |  |
| **28** | **25** | **15** | **5** | **10** | **9** | **8** | **7** | **4** | **1** |
| **28** | **25** | **15** | **5** | **10** | **9** | **8** | **7** | **4** | **1** |
| **28** | **25** | **15** | **5** | **10** | **9** | **8** | **7** | **4** | **1** |
| **28** | **25** | **15** | **5** | **10** | **9** | **8** | **7** | **4** | **1** |
| **28** | **25** | **15** | **5** | **10** | **9** | **8** | **7** | **4** | **1** |
| **28** | **25** | **15** | **10** | **5** | **9** | **8** | **7** | **4** | **1** |
|  |  |  |  | **Step 5** |  |  |  |  |  |
| **28** | **25** | **15** | **10** | **5** | **9** | **8** | **7** | **4** | **1** |
| **28** | **25** | **15** | **10** | **5** | **9** | **8** | **7** | **4** | **1** |
| **28** | **25** | **15** | **10** | **5** | **9** | **8** | **7** | **4** | **1** |
| **28** | **25** | **15** | **10** | **5** | **9** | **8** | **7** | **4** | **1** |
| **28** | **25** | **15** | **10** | **9** | **5** | **8** | **7** | **4** | **1** |
|  |  |  |  |  |  |  |  |  |  |
| **28** | **25** | **15** | **10** | **9** | **5** | **8** | **7** | **4** | **1** |
| **28** | **25** | **15** | **10** | **9** | **5** | **8** | **7** | **4** | **1** |
| **28** | **25** | **15** | **10** | **9** | **8** | **5** | **7** | **4** | **1** |
|  |  |  |  |  |  |  |  |  |  |
| **28** | **25** | **15** | **10** | **9** | **8** | **5** | **7** | **4** | **1** |
| **28** | **25** | **15** | **10** | **9** | **8** | **7** | **5** | **4** | **1** |
| **28** | **25** | **15** | **10** | **9** | **8** | **7** | **5** | **4** | **1** |

* **Dãy đã sắp xếp theo thứ tự giảm dần**

1. **Chức Năng Shaker Sort**

**Code:**

void ShakeSort(int a[], int n)

{

int l, r, k;

l = 0; r = n - 1; k = n - 1;

while (l < r)

{

for (int j = r; j > l; j--)

if (a[j]> a[j - 1])

{

swap(a[j], a[j - 1]);

k = j;

}

l = k;

for (int j = l; j < r; j++)

if (a[j]< a[j + 1])

{

swap(a[j], a[j + 1]);

k = j;

}

r = k;

}

}

**Các Bước Thực Hiện Giống Bubble Sort.**

1. **Chức Năng Chèn trực tiếp – Insertion Sort**

**Code:**

void InsertionSort(int a[], int n)

{

int i, key, j;

for (i = 1; i < n; i++)

{

key = a[i];

j = i - 1;

while (j >= 0 && a[j] < key)

{

a[j + 1] = a[j];

j = j - 1;

}

a[j + 1] = key;

}

}

**Các Bước Thực Hiện:**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **5** | **25** | **15** | **8** | **7** | **28** | **1** | **4** | **10** | **9** |
| **25** | **5** | **15** | **8** | **7** | **28** | **1** | **4** | **10** | **9** |
| **25** | **15** | **5** | **8** | **7** | **28** | **1** | **4** | **10** | **9** |
| **25** | **15** | **8** | **5** | **7** | **28** | **1** | **4** | **10** | **9** |
| **25** | **15** | **8** | **7** | **5** | **28** | **1** | **4** | **10** | **9** |
| **28** | **25** | **15** | **8** | **7** | **5** | **1** | **4** | **10** | **9** |
| **28** | **25** | **15** | **8** | **7** | **5** | **1** | **4** | **10** | **9** |
| **28** | **25** | **15** | **8** | **7** | **5** | **4** | **1** | **10** | **9** |
| **28** | **25** | **15** | **10** | **8** | **7** | **5** | **4** | **1** | **9** |
| **28** | **25** | **15** | **10** | **9** | **8** | **7** | **5** | **4** | **1** |
| **28** | **25** | **15** | **10** | **9** | **8** | **7** | **5** | **4** | **1** |

* **Dãy đã xếp theo thứ tự giảm dần**

1. **Chức Năng Chèn nhị phân – Binary Insertion Sort**

**Code:**

void BInsertionSort(int a[], int n)

{

int l, r, m, i;

int x;

for (int i = 1; i < n; i++)

{

x = a[i];

l = 0;

r = i;

while (l <= r)

{

m = (l + r) / 2;

if (x >= a[m])

r = m - 1;

else

l = m + 1;

}

for (int j = i - 1; j >= l; j--)

a[j + 1] = a[j];

a[l] = x;

}

}

**Các Bước Thực Hiện Giống Insertion Sort**

1. **Chức Năng Shell Sort**

**Code:**

void ShellSort(int a[], int n)

{

for (int gap = n / 2; gap > 0; gap /= 2)

{

for (int i = gap; i < n; i += 1)

{

int temp = a[i];

int j;

for (j = i; j >= gap && a[j - gap] < temp; j -= gap)

a[j] = a[j - gap];

a[j] = temp;

}

}

}

**Các Bước Thực Hiện:**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  | **Step 1** |  |  |  |  |  |
| **5** | **25** | **15** | **8** | **7** | **28** | **1** | **4** | **10** | **9** |
| **5** | **25** | **15** | **9** | **7** | **28** | **1** | **4** | **10** | **8** |
| **9** | **25** | **28** | **5** | **7** | **15** | **1** | **4** | **10** | **8** |
| **9** | **25** | **28** | **5** | **7** | **15** | **1** | **8** | **10** | **4** |
| **9** | **25** | **28** | **8** | **7** | **15** | **1** | **5** | **10** | **4** |
|  |  |  |  | **Step 2** |  |  |  |  |  |
| **9** | **25** | **28** | **8** | **7** | **15** | **1** | **5** | **10** | **4** |
| **25** | **9** | **28** | **8** | **7** | **15** | **1** | **5** | **10** | **4** |
| **28** | **25** | **9** | **8** | **7** | **15** | **1** | **5** | **10** | **4** |
| **28** | **25** | **9** | **8** | **7** | **15** | **1** | **5** | **10** | **4** |
| **28** | **25** | **9** | **8** | **7** | **15** | **1** | **5** | **10** | **4** |
| **28** | **25** | **15** | **9** | **8** | **7** | **1** | **5** | **10** | **4** |
| **28** | **25** | **15** | **9** | **8** | **7** | **1** | **5** | **10** | **4** |
| **28** | **25** | **15** | **9** | **8** | **7** | **5** | **1** | **10** | **4** |
| **28** | **25** | **15** | **10** | **9** | **8** | **7** | **5** | **1** | **4** |
| **28** | **25** | **15** | **10** | **9** | **8** | **7** | **5** | **4** | **1** |
| **28** | **25** | **15** | **10** | **9** | **8** | **7** | **5** | **4** | **1** |

* **Dãy đã săp xếp theo thứ tự giảm dần**

1. **Chức Năng Heap Sort**

**Code:**

void Heapify(int a[], int n, int vt)

{

while (vt <= n / 2 - 1)

{

int child1 = 2 \* vt + 1;

int child2 = 2 \* vt + 2;

int lc = child1;

if (child2<n&&a[child2]>a[lc])

lc = child2;

if (a[vt] < a[lc])

swap(a[vt], a[lc]);

vt = lc;

}

}

void BuildHeap(int a[], int n)

{

for (int i = n / 2 - 1; i >= 0; i--)

Heapify(a, n, i);

}

void HeapSort(int a[], int n)

{

BuildHeap(a, n);

for (int i = n - 1; i >= 0; i--)

{

swap(a[0], a[i]);

Heapify(a, n, 0);

}

}

void xuat(int a[], int n)

{

for (int i = n + 1; i < n; i++)

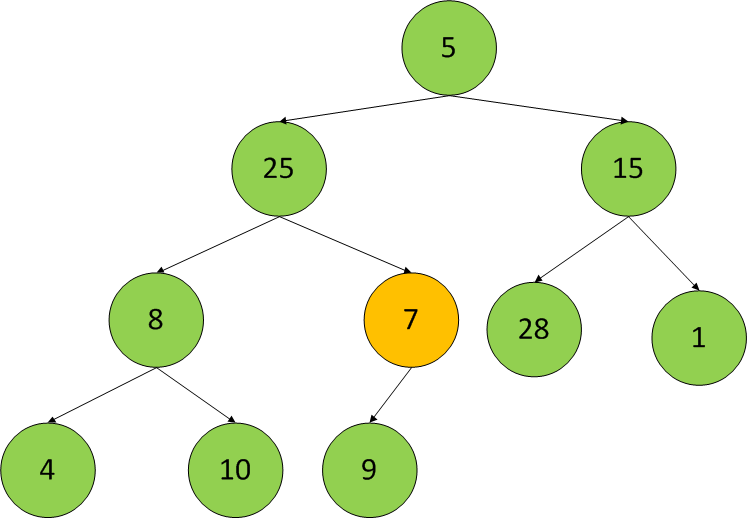
{

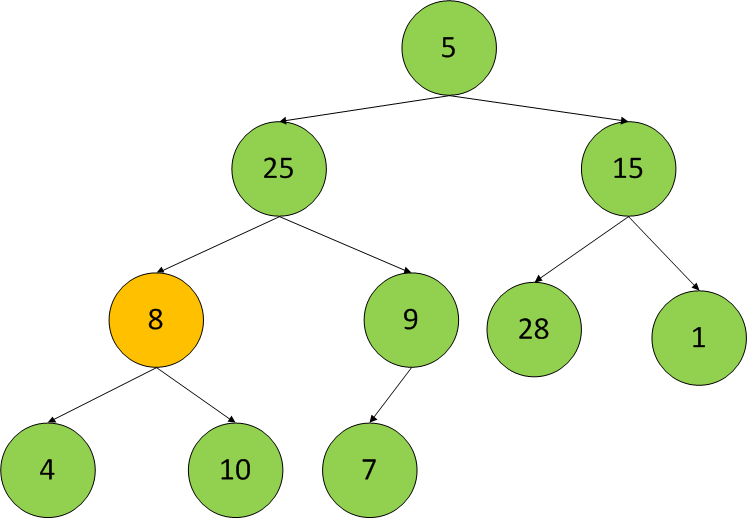
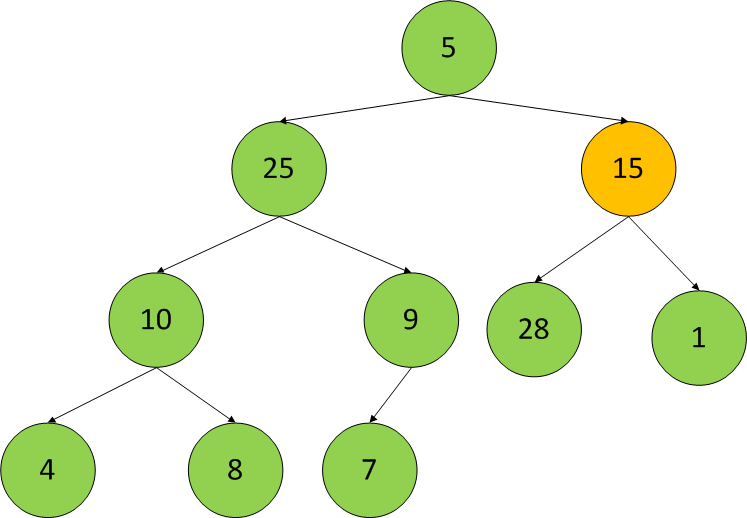
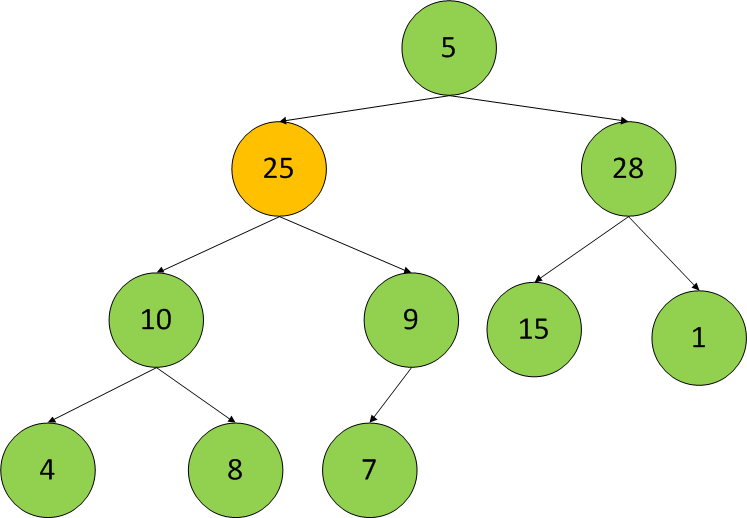
cout << a[i] << " ";

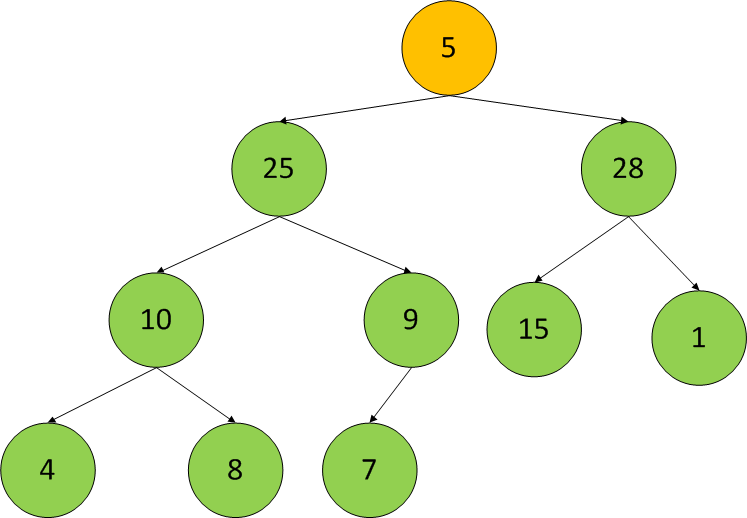
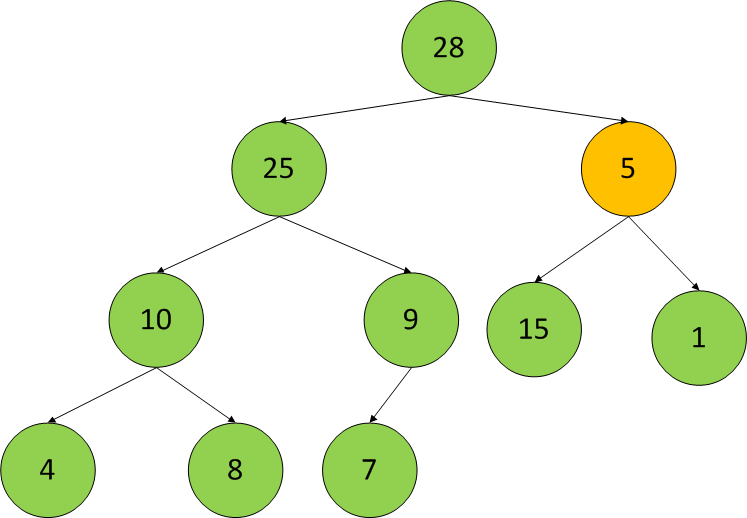
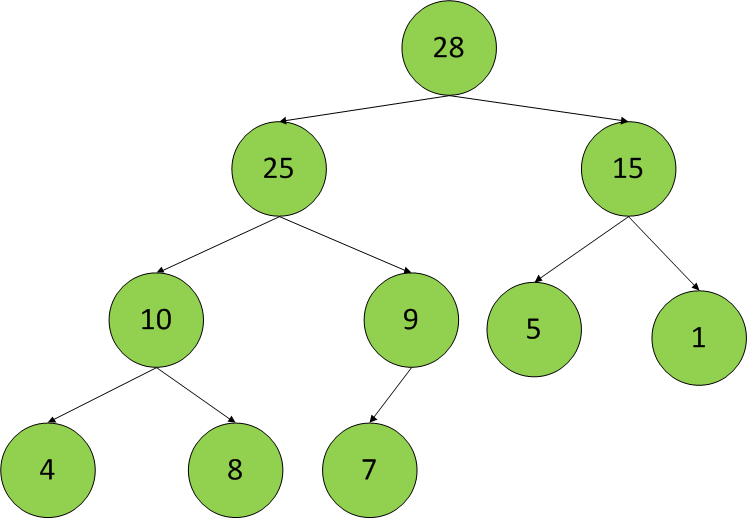
break;

}

}

**Các Bước Thực Hiện:**

****

****

* **Dãy đã săp xếp theo thứ tự giảm dần**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Ex:** | **5** | **25** | **15** | **8** | **7** | **28** | **1** | **4** | **10** | **9** |
| **KQ:** | **28** | **25** | **15** | **10** | **9** | **8** | **7** | **5** | **4** | **1** |

1. **Chức Năng Quick Sort**

**Code:**

void QuickSort(int a[], int left, int right)

{

int i, j, x, y;

i = left; j = right;

x = a[(left + right) / 2];

do

{

while (a[i]>x) i++;

while (a[j]<x) j--;

if (i <= j)

{

swap(a[i], a[j]);

i++; j--;

}

} while (i < j);

if (left<j) QuickSort(a, left, j);

if (i<right) QuickSort(a, i, right);

}

void Quick\_Sort(int a[], int n)

{

QuickSort(a, 0, n - 1);

}

**Các Bước Thực Hiện:**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **5** | **25** | **15** | **8** | **7** | **28** | **1** | **4** | **10** | **9** |
| **X=5, [1,10]** | **5** | **25** | **15** | **8** | **7** | **28** | **1** | **4** | **10** | **9** |
|  | **28** | **25** | **15** | **8** | **7** | **5** | **1** | **4** | **10** | **9** |
|  | **28** | **25** | **15** | **8** | **10** | **5** | **1** | **4** | **7** | **9** |
|  | **28** | **25** | **15** | **8** | **10** | **5** | **1** | **4** | **9** | **7** |
|  | **28** | **25** | **15** | **8** | **10** | **5** | **1** | **4** | **9** | **7** |
| **X=5,[1,9]** | **28** | **25** | **15** | **8** | **10** | **5** | **1** | **4** | **9** | **7** |
|  | **28** | **25** | **15** | **8** | **10** | **5** | **1** | **4** | **9** | **7** |
|  | **28** | **25** | **15** | **10** | **8** | **5** | **1** | **4** | **9** | **7** |
|  | **28** | **25** | **15** | **10** | **8** | **5** | **1** | **4** | **9** | **7** |
| **X=7,[5,10]** | **28** | **25** | **15** | **10** | **8** | **5** | **1** | **4** | **9** | **7** |
|  | **28** | **25** | **15** | **10** | **8** | **5** | **4** | **1** | **9** | **7** |
|  | **28** | **25** | **15** | **10** | **8** | **5** | **4** | **9** | **1** | **7** |
|  | **28** | **25** | **15** | **10** | **8** | **5** | **4** | **9** | **7** | **1** |
| **X=5,[1,9]** | **28** | **25** | **15** | **10** | **8** | **5** | **4** | **9** | **7** | **1** |
|  | **28** | **25** | **15** | **10** | **9** | **5** | **4** | **8** | **7** | **1** |
| **X=7,[6,8]** | **28** | **25** | **15** | **10** | **9** | **5** | **4** | **8** | **7** | **1** |
|  | **28** | **25** | **15** | **10** | **9** | **8** | **4** | **5** | **7** | **1** |
| **X=9,[8,10]** | **28** | **25** | **15** | **10** | **9** | **8** | **4** | **5** | **7** | **1** |
|  | **28** | **25** | **15** | **10** | **9** | **8** | **4** | **7** | **5** | **1** |
|  | **28** | **25** | **15** | **10** | **9** | **8** | **7** | **4** | **5** | **1** |
|  | **28** | **25** | **15** | **10** | **9** | **8** | **7** | **4** | **5** | **1** |
|  | **28** | **25** | **15** | **10** | **9** | **8** | **7** | **5** | **4** | **1** |
| **KQ:** | **28** | **25** | **15** | **10** | **9** | **8** | **7** | **5** | **4** | **1** |

1. **Chức Năng Merge Sort**

**Code:**

void MergeSort(int a[], int n)

{

int i, j, k, low1, up1, low2, up2, size;

int dstam[100]; size = 1;

while (size<n)

{

low1 = 0; k = 0;

while (low1 + size <n)

{

low2 = low1 + size;

up1 = low2 - 1;

if (low2 + size - 1< n)

up2 = low2 + size - 1;

else

up2 = n - 1;

for (i = low1, j = low2; i <= up1 && j <= up2; k++)

{

if (a[i] >= a[j])

dstam[k] = a[i++];

else

dstam[k] = a[j++];

}

for (; i <= up1; k++)

dstam[k] = a[i++];

for (; j <= up2; k++)

dstam[k] = a[j++];

low1 = up2 + 1;

}

for (i = low1; k<n; i++)

dstam[k++] = a[i];

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

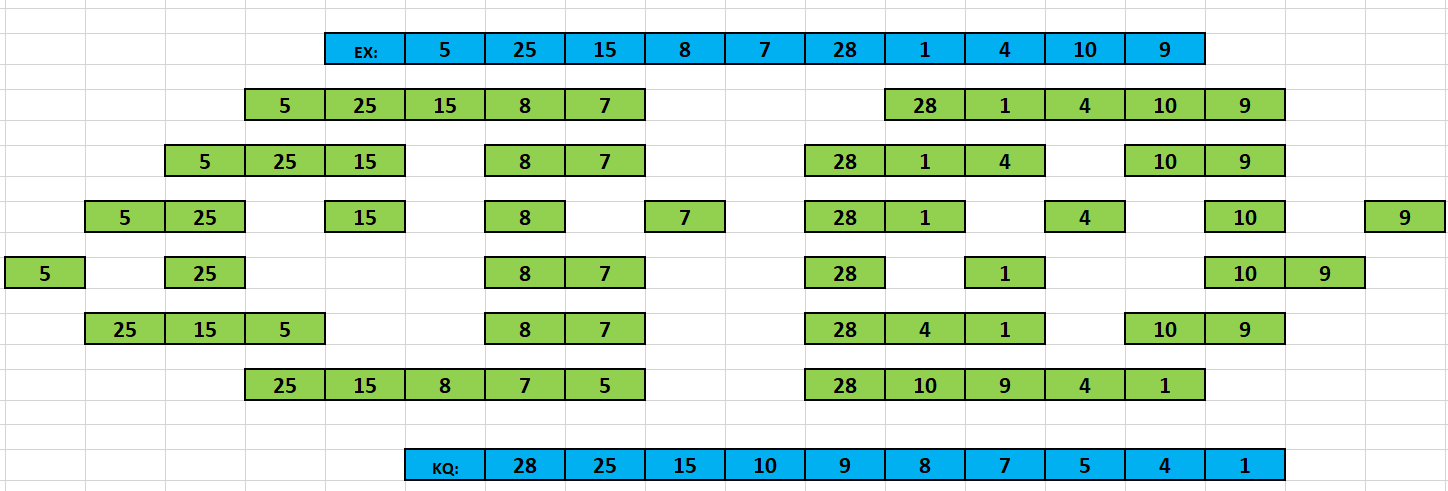
a[i] = dstam[i];

size \*= 2;

}

}

**Các Bước Thực Hiện:**



1. **Chức Năng Radix Sort**

**Code:**

void RadixSort(int \*a, int n)

{

int i, b[100], m = a[0], exp = 1;

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

{

if (a[i] <= m)

m = a[i];

}

while (m / exp <= 0)

{

int bucket[10] = { 0 };

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

bucket[a[i] / exp % 10]++;

for (i = 1; i < 10; i++)

bucket[i] += bucket[i - 1];

for (i = n - 1; i >= 0; i--)

b[--bucket[a[i] / exp % 10]] = a[i];

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

a[i] = b[i];

exp \*= 10;

}

}

**Các Bước Thực Hiện:**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Ex:** | **5** | **25** | **15** | **8** | **7** | **28** | **1** | **4** | **10** | **9** |
|  |  |  |  |  | **Step 1** |  |  |  |  |  |
|  | **9** | **28**  **8** | **7** |  | **15 25 5** | **4** |  |  | **1** | **10** |
|  | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
|  |  |  |  |  | **Step 2** |  |  |  |  |  |
|  | **9** | **8** | **28** | **7** | **5** | **25** | **15** | **4** | **1** | **10** |
|  |  |  |  |  |  |  |  | **25 28** | **10 15** | **1**  **4**  **5**  **7**  **8**  **9** |
|  | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
|  |  |  |  |  | **Step 3** |  |  |  |  |  |
| **KQ:** | **28** | **25** | **15** | **10** | **9** | **8** | **7** | **5** | **4** | **1** |

**Các Hàm Bổ Trợ:**

void swap(int &x, int &y)

{

int temp = x;

x = y;

y = temp;

}

int KtraTangGiam(int a[], int n)

{

int tang = 0, giam = 0;

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

{

if (a[i] < a[i + 1])

tang++;

if (a[i] > a[i + 1])

giam++;

}

if (tang == n - 1) return 1;

else if (giam == n - 1) return 2;

else return 0;

}

**Hàm Main:**

int main()

{

int a[10] = { 5,25,15,8,7,28,1,4,10,9 };

int n = 10, x, l, r;

int chon;

do

{

chon = menu();

switch (chon)

{

case 0:return 0;

case 1:

{

cout << "Nhap x: ";

cin >> x;

if (LinearSearch(a, n, x))

cout << "Tim thay Gia Tri " << x << endl;

else

cout << "Khong Tim Thay Gia Tri " << x << endl;

break;

}

case 2:

{

if (KtraTangGiam(a, n) == 2)

{

cout << "Nhap x: ";

cin >> x;

if (BinarySearch(a, n, x))

cout << "Tim thay Gia Tri " << x;

else

cout << "Khong Tim Thay Gia Tri " << x;

break;

}

else

cout << "Mang Khong Theo Thu Tu Giam Dan Nen Khong Theo Tim Kiem Nhi Phan" << endl;

break;

}

case 3:

{

InterchangeSort(a, n);

cout << "Day sau khi sap xep: ";

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

cout << a[i] << " ";

break;

}

case 4:

{

SelectionSort(a, n);

cout << "Day sau khi sap xep: ";

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

cout << a[i] << " ";

break;

}

case 5:

{

BubbleSort(a, n);

cout << "Day sau khi sap xep: ";

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

cout << a[i] << " ";

break;

}

case 6:

{

InsertionSort(a, n);

cout << "Day sau khi sap xep: ";

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

cout << a[i] << " ";

break;

}

case 7:

{

BInsertionSort(a, n);

cout << "Day sau khi sap xep: ";

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

cout << a[i] << " ";

break;

}

case 8:

{

ShakeSort(a, n);

cout << "Day sau khi sap xep: ";

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

cout << a[i] << " ";

break;

}

case 9:

{

ShellSort(a, n);

cout << "Day sau khi sap xep: ";

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

cout << a[i] << " ";

break;

}

case 10:

{

HeapSort(a, n);

cout << "Day sau khi sap xep: ";

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

cout << a[i] << " ";

break;

}

case 11:

{

Quick\_Sort(a, n);

cout << "Day sau khi sap xep: ";

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

cout << a[i] << " ";

break;

}

case 12:

{

MergeSort(a, n);

cout << "Day sau khi sap xep: ";

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

cout << a[i] << " ";

break;

}

case 13:

RadixSort(a, n);

cout << "Day sau khi sap xep: ";

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

{

cout << a[i] << " ";

}

break;

}

system("pause");

} while (1);

}