**Bài tập về nhà: (Cô có bổ sung thêm câu 5)**

**Đề gồm 4 câu.**

**Câu 1:**

1. Hãy trình bày ý tưởng thực hiện thuật toán Quick Sort. Thuật toán QuickSort tốt hơn thuật toán nào đã học (So sánh ngắn gọn với ít nhất 2 thuật toán khác).
2. Thể hiện **từng bước** kết quả chạy thuật toán QuickSort (**dùng thuật toán Quick sort trong slide**) với các mảng sau.

Mảng 1: 22, 12, 9, 7, 31, 19, 2, 27

Mảng 2: 27, 32, 12, 52, 39, 76, 18, 4, 25, 69, 10, 8, 16

**Câu 2**: Hãy nêu ý tưởng của thuật toán MergeSort.

**Câu 3**: Thực hiện sắp xếp dãy sau bằng thuật toán MergeSort: 14, 2, 13, 21, 16, 9, 10, 4 , 8, 17. Sử dụng 2 phương pháp đệ quy và không đệ quy.

**Câu 4**: Thực hiện sắp xếp dãy sau bằng thuật toán MergeSort: 7, 9, 10, 15, 8, 5, 6, 1, 4. Sử dụng 2 phương pháp đệ quy và không đệ quy.

Lưu ý:

* Sinh viên đọc thêm phương pháp không đệ quy (bottom up merge sort) để làm bài tập.
* Merge sort: chỉ cần vẽ hình kết quả từng bước như trong slide

**Câu 5: (mở rộng cho bài 1b)**

Cho 2 cài đặt Hoare’s Partition (chia mảng ban đầu thành 2 mảng con dựa trên phần tử pivot, với pivot là phần tử giữa mảng) như bên dưới. Hãy chạy tay mảng ví dụ bên dưới và cho biết kết quả 2 mảng con tương ứng sau lần partition đầu tiên với l=0, r=n-1.

Mảng 1: 22, 12, 9, 7, 31, 19, 2, 27

Mảng 3: 1, 2, 3, 4, 5, 6, 7, 8, 9, 0

Mảng 2: 12, 2, 8, 5, 1, 6, 4, 15

Cài đặt thuật toán Hoare’s Partition 1 (trong slide):

void Partition1(int \*a, int l, int r, int &i, int &j){

int pivot=a[(l+r)/2];

i=l; j=r;

while (i <= j) {

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

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

if (i <= j) {

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

i++; j--;

}

}

}

Cài đặt thuật toán Hoare’s Partition 2 (thuật toán Hoare’s Partition trên wiki, nhưng lưu ý vẫn chọn pivot là phần tử giữa):

int Partition2(int \*a, int l, int r){

int pivot=a[(l+r)/2];

int i=l-1, j=r+1;

while (true) {

do {i++;} while (a[i] < pivot);

do {j--;} while (a[j] > pivot);

if (i >= j) return j;

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

}

return j;

}