Question 1

{179,721,639,549,292,427,335,435,62}, Radix = 9

Key % 9

0 – 639, 549 5 –

1 – 721 6 –

2 – 335 7 –

3 – 435 8 – 179, 62

4 – 292, 427 9 –

639, 549, 721, 335, 435, 292, 427, 179, 62

(Key / 9) % 9

0 –62 5 – 292

1 – 335, 179 6 –

2 – 427 7 – 549

3 – 435 8 – 639, 721

4 – 9 –

62, 335, 179, 427, 435, 292, 549, 639, 721

(key / 81)

0 – 62 5 – 427, 435

1 – 6 – 549

2 – 179 7 – 639

3 – 292 8 – 721

4 – 335 9 –

62, 179, 292, 335, 427, 435, 549, 639, 721 (Sorted)

—-------------------------------—-------------------------------—-------------------------------

Question 2

Since the best that can be done for comparison-based algorithms is nlogn, which in our case would be 8 i.e , it is impossible to devise an algorithm that can sort 4 elements using exactly 5 comparisons in the worst case.

—-------------------------------—-------------------------------—-------------------------------

Question 3

Algorithm makeFBS(S)  
 Input array S with n elements  
 output S in FBS format.  
 S ← sort(S) O(nlogn)

evenIdx← 0 1

oddIdx←new int[S.length] 2

fbsAry←new int[S.length] 3

stop← S.length-1 1

for i←0 to stop do 1+(n+1)

if i%2=0 then n

fbsAry[even]←S[i] 3n

even←even+2 2n

else

fbsAry[odd]←S[i] 3n

odd←odd-2 2n

{increment counter i} n

Asymptotic running time for this algorithm is O(nlogn),l which is the largest of all running times listed above.  
If S or the input array is already sorted the best case or the fastest possible asymptotic running time would be O(n).