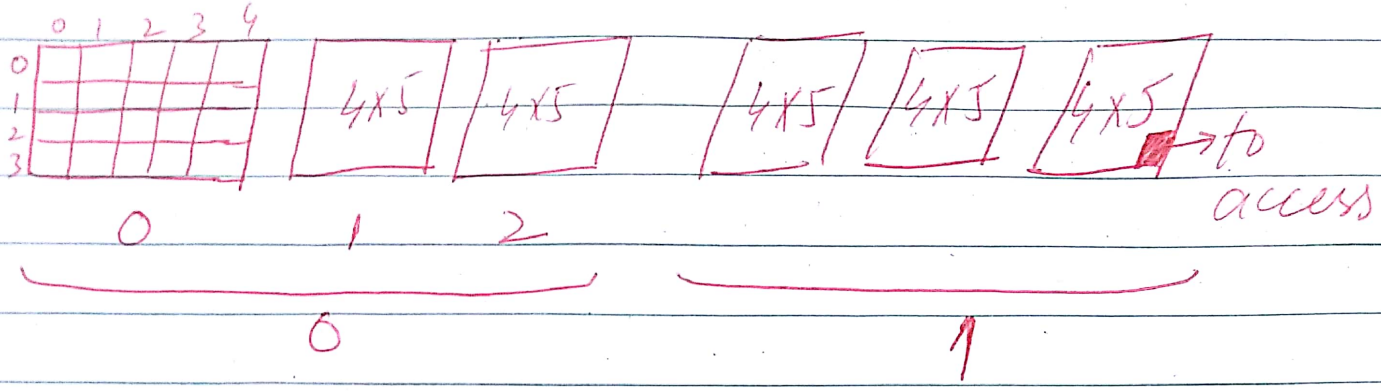
PART 01



Base Address + 1(Block Size) + 2(Squares Size) + 3(Row Size) + 4(Int Size)

$$= 1\left(\overbrace{3 \times 4 \times 5 \times 4}^{240}\right) + 2\left(\underbrace{4 \times 5 \times 4}_{80}\right) + 3\left(\underbrace{5 \times 4}_{20}\right) + \underbrace{4 \times 4}_{16}$$

$$= 460 + 16$$

$$= 476$$

PART 02

```
bool anagrams(char *s, char *t){
    if(strlen(s) != strlen(t))
        return false;
    int letters[256] = {0};
    for(int i=0; i < strlen(s); i++){
        letters[s[i]]++;
        letters(t[i])--;
    }
    for(int i=0; i<256; i++){
        if(letters[i] != 0)
    }        return true;
```

PART 03

Yes, we can generate the given sentence from the input data, using 2-Gram.
If we pick 3-Gram, the smallest possible gram with "Lahore" as last word, would be "is from Lahore" and we will not get our desired sentence.

PART 04

Only three values are out of order, so the "Insertion Sort" would be most efficient.

Variables	Array
i=1 j=1 temp=3	3,3,4,5,5,2,3,4,10,10
i=2 j=2 temp=4	3,3,4,5,5,2,3,4,10,10
i=3,4, j=3,4 temp=5,5	3,3,4,5,5,2,3,4,10,10
i=5, j=5 temp=2	3,3,4,5,5,5,3,4,10,10
i=5, j=4 temp=2	3,3,4,5,5,5,3,4,10,10
i=5, j=3 temp=2	3,3,4,4,5,5,3,4,10,10
i=5, j=2 temp=2	3,3,3,4,5,5,3,4,10,10
i=5, j=1 temp=2	3,3,3,4,5,5,3,4,10,10
i=5, j=0 temp=2	2,3,3,4,5,5,3,4,10,10
i=6, j=6 temp=3	2,3,3,4,5,5,5,4,10,10
i=6, j=5,4,3, temp=3	2,3,3,3,4,5,5,4,10,10
i=7, j=7,6,5 temp=4	2,3,3,3,4,4,5,5,10,10
i=8 temp=10	2,3,3,3,4,4,5,5,10,10
9 temp=10	2,3,3,3,4,4,5,5,10,10

PART 05

n	i	neron	A
8	1	0	5 6 3 1 8 7 2 4
//	2	0	5 3 6 1 8 7 2 4
//	3	2	5 3 1 6 8 7 2 4
//	4	3	5 3 1 6 8 7 2 4
//	5	3	5 3 1 6 7 8 2 4
//	6	5	5 3 1 6 7 2 8 4
//	7	6	5 3 1 6 7 2 4 8
7	1	0	3 5 1 6 7 2 4 8
//	2	1	3 1 5 6 7 2 4 8
//	3	2	3 1 5 6 7 2 4 8
//	4	2	3 1 5 6 7 2 4 8
//	5	2	3 1 5 6 2 7 4 8
//	6	5	3 1 5 6 2 4 7 8
6	1	0	1 3 5 6 2 4 7 8
//	2	0	1 3 5 6 2 4 7 8
//	3	0	1 3 5 6 2 4 7 8
//	4	0	1 3 5 2 6 4 7 8
//	5	4	1 3 5 2 4 6 7 8
5	1	0	1 3 5 2 4 6 7 8
//	2	0	1 3 5 2 4 6 7 8
//	3	0	1 3 2 5 4 6 7 8
//	4	3	1 3 2 4 5 6 7 8
4	1	0	1 3 2 4 5 6 7 8
//	2	0	1 2 3 4 5 6 7 8
//	3	2	1 2 3 4 5 6 7 8
2	1	0	1 2 3 4 5 6 7 8

In case that multiple elements at the end of the list are sorted, then newn will keep the index of last swap. For next iteration of while loop, the n is initialized from newn, i.e. last swap index and results in the improvement as many elements are ignored from the end.

$O(n^2)$ no improvement here (worst case)

PART 06

sorted	i	list	
false		5, 6, 3, 1, 8, 7, 2, 4	Initial
True	1	5, 3, 6, 1, 8, 7, 2, 4	
false	3	5, 3, 6, 1, 8, 7, 2, 4	
false	5	5, 3, 6, 1, 8, 2, 7, 4	
false	0	3, 5, 6, 1, 8, 2, 7, 4	
false	2	3, 5, 1, 6, 8, 2, 7, 4	
false	4	3, 5, 1, 6, 2, 8, 7, 4	
false	6	3, 5, 1, 6, 2, 8, 4, 7	1st While
true	1	3, 1, 5, 6, 2, 8, 4, 7	
false	3	3, 1, 5, 2, 6, 8, 4, 7	
false	5	3, 1, 5, 2, 6, 4, 8, 7	
false	0	1, 3, 5, 2, 6, 4, 8, 7	
false	2	1, 3, 2, 5, 6, 4, 8, 7	
false	4	1, 3, 2, 5, 4, 6, 8, 7	
false	6	1, 3, 2, 5, 4, 6, 7, 8	2nd While
true	1	1, 2, 3, 5, 4, 6, 7, 8	
false	3	1, 2, 3, 4, 5, 6, 7, 8	
false	5	1, 2, 3, 4, 5, 6, 7, 8	
false	0	1, 2, 3, 4, 5, 6, 7, 8	
false	2	1, 2, 3, 4, 5, 6, 7, 8	
false	4,6	1, 2, 3, 4, 5, 6, 7, 8	3rd While
true		〃	Sorted
〃	3	〃	
〃	5	〃	
〃	0	〃	
〃	2,46	〃	

Interesting Bubble Sort moves largest and second largest elements towards the end of the list, hence sorts early.
Secondly, it can run two separate sortings on two separate machines (distribute

$O(n^2)$ still the worst case.

PART 07

swapped	start	end	i	a
true	0	7		5, 6, 3, 1, 8, 7, 2, 4
false	0	7	0	5, 6, 3, 1, 8, 7, 2, 4
false	0	7	1	5, 3, 6, 1, 8, 7, 2, 4
true	0	7	2	5, 3, 1, 6, 8, 7, 2, 4
"	0	7	3	5, 3, 1, 6, 8, 7, 2, 4
"	0	7	4	5, 3, 1, 6, 7, 8, 2, 4
"	0	7	5	5, 3, 1, 6, 7, 2, 8, 4
"	0	7	6	5, 3, 1, 6, 7, 2, 4, 8 ←
false	0	6	5	5, 3, 1, 6, 7, 2, 4, 8
"	0	6	4	5, 3, 1, 6, 2, 7, 4, 8
true	0	6	3	5, 3, 1, 2, 6, 7, 4, 8
"	0	6	2	5, 3, 1, 2, 6, 7, 4, 8
"	0	6	1	5, 1, 3, 2, 6, 7, 4, 8 ←
"	0	6	0	1, 5, 3, 2, 6, 7, 4, 8
false	1	6	1	1, 3, 5, 2, 6, 7, 4, 8
true	1	6	2	1, 3, 2, 5, 6, 7, 4, 8
"	1	6	3	1, 3, 2, 5, 6, 7, 4, 8
"	1	6	4	1, 3, 2, 5, 6, 7, 4, 8
"	1	6	5	1, 3, 2, 5, 6, 4, 7, 8 ←
false	1	5	4	1, 3, 2, 5, 4, 6, 7, 8
true	1	5	3	1, 3, 2, 4, 5, 6, 7, 8
"	1	5	2	1, 3, 2, 4, 5, 6, 7, 8
"	1	5	1	1, 2, 3, 4, 5, 6, 7, 8 ←
false	2	5	2, 3, 4	1, 2, 3, 4, 5, 6, 7, 8

Cocktail sort is better as it moves (slower moving) smaller elements towards the start in every reverse loop, hence terminates early.
(Turtles Vs. Rabbits concept)

$O(n^2)$ is still the worst case.