[D] Attempt the following:

(a) Twenty-five numbers are entered from the keyboard into an array. The number to be searched is entered through the keyboard by the user. Write a program to find if the number to be searched is present in the array and if it is present, display the number of times it appears in the array.

(b) Implement the Selection Sort, Bubble Sort and Insertion sort algorithms on a set of 25 numbers. (Refer Figure 9.11 for the logic of the algorithms)

- Selection sort
- Bubble Sort
- Insertion Sort

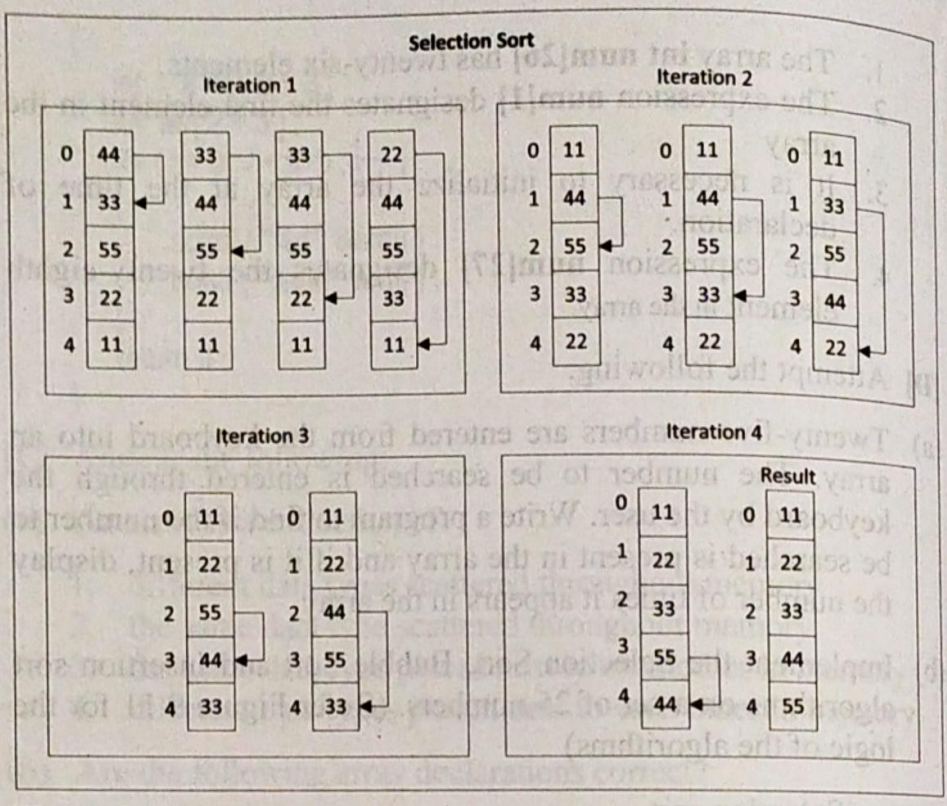
Selection Sort
Iteration 1
Iteration 2
Iteration 3
Iteration 4
Result

Figure 9.11 (a)

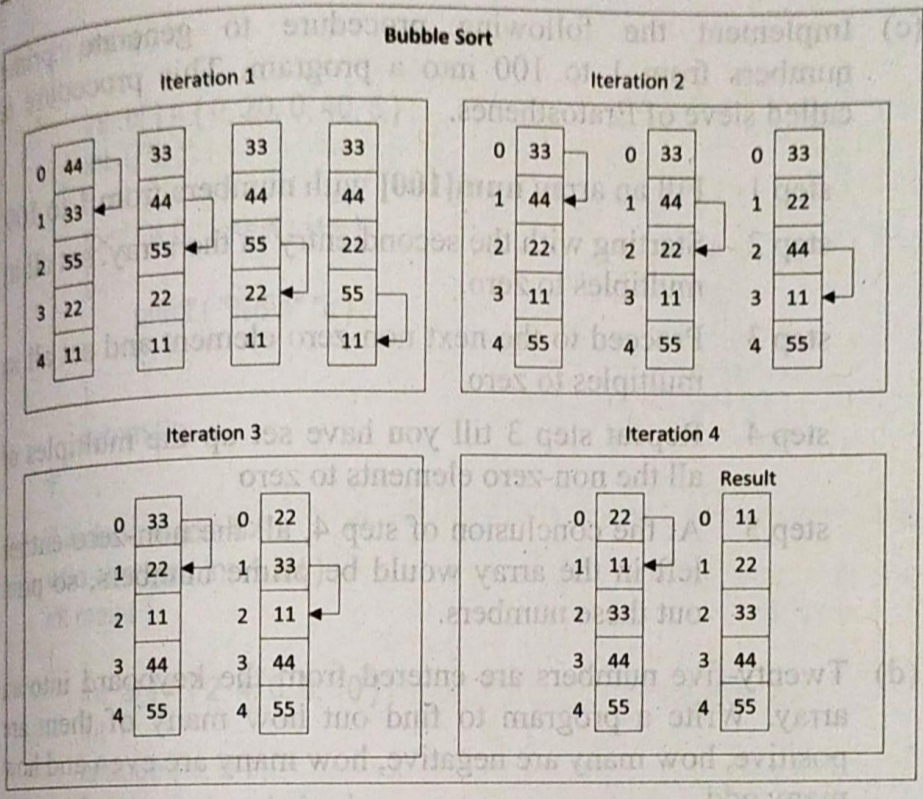

Figure 9.11 (b)

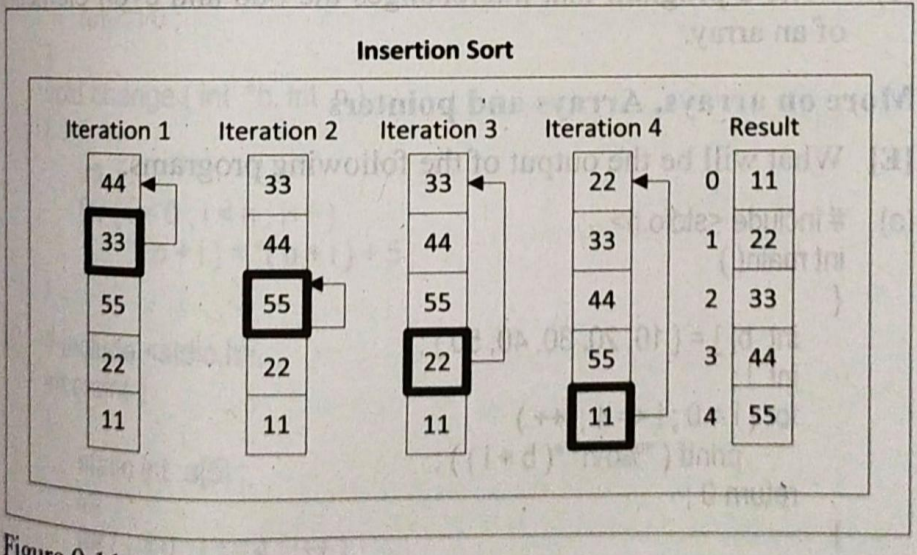

Figure 9.11 (c)

(c) Implement the following procedure to generate prime numbers from 1 to 100 into a program. This procedure is called sieve of Eratosthenes.

step 1 Fill an array **num[100]** with numbers from 1 to 100

step 2 Starting with the second entry in the array, set all its multiples to zero.

step 3 Proceed to the next non-zero element and set all its multiples to zero.

step 4 Repeat step 3 till you have set up the multiples of all the non-zero elements to zero

step 5 At the conclusion of step 4, all the non-zero entries left in the array would be prime numbers, so print out these numbers.

(d) Twenty-five numbers are entered from the keyboard into an array. Write a program to find out how many of them are positive, how many are negative, how many are even and how many odd.

(e) Write a program that interchanges the odd and even elements of an array.

(a) Write a program to copy the contents of one array into another in the reverse order.

(b) If an array **arr** contains **n** elements, then write a program to check if **arr[0] = arr[n-1]**, **arr[1] = arr[n-2]** and so on.

(c) Write a program using pointers to find the smallest number in an array of 25 integers.

(d) Write a program which performs the following tasks:

- initialize an integer array of 10 elements in **main( )**
- pass the entire array to a function **modify( )**
- in **modify( )** multiply each element of array by 3
- return the control to **main( )** and print the new array elements in **main( )**

[L] Attempt the following:

(a) How will you initialize a three-dimensional array **threed[3][2][3]**? How will you refer the first and last element in this array?

(b) Write a program to pick up the largest number from any 5 row by 5 column matrix.

(c) Write a program to obtain transpose of a 4 x 4 matrix. The transpose of a matrix is obtained by exchanging the elements of each row with the elements of the corresponding column.

(d) Very often in fairs we come across a puzzle that contains 15 numbered square pieces mounted on a frame. These pieces can be moved horizontally or vertically. A possible arrangement of these pieces is shown below:

1	4	15	7
8	10	2	11
14	3	6	13
12	9	5	

Figure 9.12

As you can see there is a blank at bottom right corner. Implement the following procedure through a program:

Draw the boxes as shown above. Display the numbers in the above order. Allow the user to hit any of the arrow keys (up, down, left, or right). If you are using Turbo C/C++, use the library function **gotoxy( )** to position the cursor on the screen while drawing the boxes. If you are using Visual Studio then use the following function to position the cursor.

```
#include <windows.h>
void gotoxy ( short col, short row )
{
   HANDLE  h = GetStdHandle ( STD_OUTPUT_HANDLE ) ;
   COORD  position = { col, row } ;
   SetConsoleCursorPosition ( h, position ) ;
}
```

If user hits say, right arrow key then the piece with a number 5 should move to the right and blank should replace the original position of 5. Similarly, if down arrow key is hit, then 13 should move down and blank should replace the original position of 13. If left arrow key or up arrow key is hit then no action should be taken.

The user would continue hitting the arrow keys till the numbers aren't arranged in ascending order.

Keep track of the number of moves in which the user manages to arrange the numbers in ascending order. The user who manages it in minimum number of moves is the one who wins.

How do we tackle the arrow keys? We cannot receive them using **scanf( )** function. Arrow keys are special keys which are identified by their 'scan codes'. Use the following function in your program. It would return the scan code of the arrow key being hit. The scan codes for the arrow keys are:

up arrow key – 72 down arrow key – 80
left arrow key – 75 right arrow key – 77

```
# include <conio.h>
int getkey( )
{
    int ch ;
    ch = getch( ) ;
    if ( ch == 0 )
    {
        ch = getch( ) ;
        return ch ;
    }
    return ch ;
}
```

(e) Match the following with reference to the following program segment:

```
int i, j, = 25;
int *pi, *pj = & j;
.....
..... /* more lines of program */
.....
*pj = j + 5;
j = *pj + 5 ;
pj = pj ;
*pi = i + j
```

Each integer quantity occupies 2 bytes of memory. The value assigned to **i** begin at (hexadecimal) address F9C and the value assigned to j begins at address F9E. Match the value represented by left hand side quantities with the right.

1.	&i	a.	30
2.	&j	b.	F9E
3.	pj	c.	35
4.	*pj	d.	FA2

5.	i	e.	F9C
6.	pi	f.	67
7.	*pi	g.	unspecified
8.	(pi + 2)	h.	65
9.	(*pi + 2)	i.	F9E
10.	* (pi + 2)	j.	F9E
		k.	FAO
		l.	F9D

(f) Match the following with reference to the following program segment:

```
int x[3][5] = {
                { 1, 2, 3, 4, 5 },
                { 6, 7, 8, 9, 10 },
                { 11, 12, 13, 14, 15 }
              }, *n = &x ;
```

1.	*(*(x + 2) + 1)	a.	9
2.	*(*x + 2) + 5	b.	13
3.	*(*(x + 1))	c.	4
4.	*(*(x) + 2) + 1	d.	3
5.	* (*(x + 1) + 3)	e.	2
6.	*n	f.	12
7.	*(n +2)	g.	14
8.	(*(n + 3) + 1	h.	7
9.	*(n + 5)+1	i.	1
10.	++*n	j.	8
		k.	5
		l.	10
		m.	6

(g) Match the following with reference to the following program segment:

```
unsigned int  arr[3][3] = {
                            2, 4, 6,
                            9, 1, 10,
                            16, 64, 5
```

```
};
```

1.	**arr	a.	64
2.	**arr < *(*arr + 2)	b.	18
3.	*(arr + 2) / (*(*arr + 1) > **arr)	c.	6
4.	*(arr[1] + 1) | arr[1][2]	d.	3
5.	*(arr[0]) | *(arr[2])	e.	0
6.	arr[1][1] < arr[0][1]	f.	16
7.	arr[2][[1] & arr[2][0]	g.	1
8.	arr[2][2] | arr[0][1]	h.	11
9.	arr[0][1] ^ arr[0][2]	i.	20
10.	++**arr + --arr[1][1]	j.	2
		k.	5
		l.	4

(h) Write a program to find if a square matrix is symmetric.

(i) Write a program to add two 6 x 6 matrices.

(j) Write a program to multiply any two 3 x 3 matrices.

(k) Given an array **p[5]**, write a function to shift it circularly left by two positions. Thus, if p[0] = 15, p[1]= 30, p[2] = 28, p[3]= 19 and p[4] = 61 then after the shift p[0] = 28, p[1] = 19, p[2] = 61, p[3] = 15 and p[4] = 30. Call this function for a (4 x 5) matrix and get its rows left shifted.

(l) A 6 x 6 matrix is entered through the keyboard. Write a program to obtain the Determinant value of this matrix.

(m) For the following set of sample data, compute the standard deviation and the mean.

-6, -12, 8, 13, 11, 6, 7, 2, -6, -9, -10, 11, 10, 9, 2

The formula for standard deviation is

$$\frac{\sqrt{(x_i - \bar{x})^2}}{n}$$

where x_i is the data item and $\bar{x}$ is the mean.

(n) The area of a triangle can be computed by the sine law when 2 sides of the triangle and the angle between them are known.

Area = (1 / 2) ab sin (angle)

Given the following 6 triangular pieces of land, write a program to find their area and determine which is largest.

Plot No.	a	b	angle
1	137.4	80.9	0.78
2	155.2	92.62	0.89
3	149.3	97.93	1.35
4	160.0	100.25	9.00
5	155.6	68.95	1.25
6	149.7	120.0	1.75

(o) For the following set of n data points (x, y), compute the correlation coefficient r, given by

$$r = \frac{\sum xy - \sum x \sum y}{\sqrt{[n\sum x^2 - (\sum x)^2][n\sum y^2 - (\sum y)^2]}}$$

x	y
34.22	102.43
39.87	100.93
41.85	97.43
43.23	97.81
40.06	98.32
53.29	98.32
53.29	100.07
54.14	97.08
49.12	91.59
40.71	94.85
55.15	94.65

(p) For the following set of point given by **(x, y)** fit a straight line given by

$y = a + bx$

where,

$a = \bar{y} - b\bar{x}$ and

$$b = \frac{n\sum yx - \sum x \sum y}{[n\sum x^2 - (\sum x)^2]}$$

x	Y
3.0	1.5
4.5	2.0
5.5	3.5
6.5	5.0
7.5	6.0
8.5	7.5
8.0	9.0
9.0	10.5
9.5	12.0
10.0	14.0

(q) The **X** and **Y** coordinates of 10 different points are entered through the keyboard. Write a program to find the distance of last point from the first point (sum of distances between consecutive points).

(r) A dequeue is an ordered set of elements in which elements may be inserted or retrieved from either end. Using an array simulate a dequeue of characters and the operations retrieve left, retrieve right, insert left, insert right. Exceptional conditions such as dequeue full or empty should be indicated. Two pointers (namely, left and right) are needed in this simulation.

(s) Sudoku is a popular number-placement puzzle. The objective is to fill a 9×9 grid with digits so that each column, each row, and each of the nine 3×3 sub-grids that compose the grid contains all of the digits from 1 to 9. The puzzle setter provides a partially completed grid, which typically has a unique solution. One such solution is given below. Write a program to check whether the solution is correct or not.

5	3	4	6	7	8	9	1	2
6	7	2	1	9	5	3	4	8
1	9	8	3	4	2	5	6	7
8	5	9	7	6	1	4	2	3
4	2	6	8	5	3	7	9	1
7	1	3	9	2	4	8	5	6
9	6	1	5	3	7	2	8	4
2	8	7	4	1	9	6	3	5
3	4	5	2	8	6	1	7	9