1. Program to print all distinct elements of given input arrays. Also print the total of the distinct elements.
Input:
Arr1 = {1,2,3,4,5}
Arr 2 = {2,6,8,10}

2. Program to remove all vowels from a given string using pointers
Input: ACADEMY OF TECHNOLOGY
OUTPUT: CDMY F TCHNLGY

3. Find the occurrence of a substring in a parent string
Input:
aAbcDefabcAdf
Substring : abc

4. Find the number of all possible triplets in the array that can form the triangle(condition is a+b>c).

5. Print all the prime numbers which are below the given number separated by comma

Input: 50
Output: 2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47

6. Write a program to find the GCD of two Integers.

7. Write a program to reverse a string

8. Write a program to swap two numbers without using third variable.

9. Write a program to find out sum of digits of given number.

10. AMCAT automata questions – 10: Write a program to print Fibonacci series of given range.

11. Print the following pattern if the input is 5.

```
1
2*3
4*5*6
7*8*9*10
7*8*9*10
4*5*6
2*3
1
```

12. Print the pattern if Input: N=4 and S=3 where S is the first number.

```
3
44
555
6666
6666
555
44
3
```

13. Print the pattern If input is 5

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

14. Write a function to return a sorted array after merging two unsorted arrays, the parameters will be two integer pointers for referencing arrays and two int variable, the length of arrays (Hint: use malloc() to allocate memory for 3rd array):

AMCAT automata questions – 2: Mooshak the mouse has been placed in a maze. There is a huge chunk of cheese somewhere in the maze. The maze is represented as a two-dimensional array of integers, where 0 represents walls.1 represents paths where Mooshak can move and 9 represents the huge chunk of cheese. Mooshak starts in the top left corner at 0.

Write a method is Path of class Maze Path to determine if Mooshak can reach the huge chunk of cheese. The input to is Path consists of a two-dimensional array and for the maze matrix. the method should return 1 if there is a path from Mooshak to the cheese and 0 if not Mooshak is not allowed to leave the maze or climb on walls.

EX: 8 by 8(8*8) matrix maze where Mooshak can get the cheese.

```
1 0 1 1 1 0 0 1
1 0 0 0 1 1 1 1
1 0 0 0 0 0 0 0
1 0 1 0 9 0 1 1
1 1 1 0 1 0 0 1
1 0 1 0 1 1 0 1
1 0 0 0 0 1 0 1
1 1 1 1 1 1 1 1
```

Test Cases:

Test Case 1:
Input: [[0,0,0],[9,1,1],[0,1,1]]
Expected return value: 0

Test case 2:
Input:[[1,1,1,][9,1,1],[0,1,0]]
Expected return value :1

Explanation:
The piece of cheese is placed at(1,0) on the grid Mooshak can move from (0,0) to (1,0) to reach it or can move from (0,0) to (0,1) to (1,1) to (1,0)

15. There is a colony of 8 cells arranged in a straight line where each day every cell competes with its adjacent cells(neighbor). Each day, for each cell, if its neighbors are both active or both inactive, the cell becomes inactive the next day, otherwise, it becomes active the next day.
Assumptions: The two cells on the ends have a single adjacent cell, so the other adjacent cell can be assumed to be always inactive. Even after updating the cell state. Consider its previous state for updating the state of other cells. Update the cell information of all cells simultaneously.

Write a function cellCompete which takes one 8 element array of integers cells representing the current state of 8 cells and one integer days representing the number of days to simulate. An integer value of 1 represents an active cell and value of 0 represents an inactive cell.
Input: [1,0,0,0,0,1,0,0],1
Expected return value: [0,1,0,0,1,0,1,0]
16

A	****	B	****	C	****	D	******
	****		* *		****		******
	****		* *		****		******
	****		****		****		******
E	*	F	*	G	*	H	*******
	**		***		* *		*****
	***		*****		* *		***
	****		*******		*******		*
I	*******	J	*	K	*	L	*
	* *		**		**		***
	* *		***		***		*****
	*		****		****		*******
			***		***		*****
			**		**		***
			*		*		*

a.

1	1	1	1
1	1	1	1
1	1	1	1
1	1	1	1

b.

1	1	1	1
2	2	2	2
3	3	3	3
4	4	4	4

c.

3	3	3
3	1	3
3	2	3
3	3	3

d.

1			
2	3		
4	5	6	
7	8	9	10

e.

```
10      9       8       7
6       5       4
3       2
1
```

f.

```
6       6       6       6
5       5       5
4       4
3
```

g.

```
3
4       5
6       7       8
9       10      11      12
```

h.

```
3
4       5
6       7       8
9       10      11      12
6       7       8
4       5
3
```

i.

```
3
5       4
8       7       6
12      11      10      9
8       7       6
5       4
3
```

j.

```
2
3       4
5       6       7
8       9       10      11
8       9       10      11
5       6       7
3       4
2
```

k.

```
2
4       3
7       6       5
11      10      9       8
11      10      9       8
```

```
7       6       5
4       3
2
```

l.

```
1       *       2       *       3       *       4
5       *       6       *       7       *       8
9       *       10      *       11      *       12
13      *       14      *       15      *       16
```

m.

```
13      *       14      *       15      *       16
9       *       10      *       11      *       12
5       *       6       *       7       *       8
1       *       2       *       3       *       4
```

n.

```
1       *       2       *       3       *       4
9       *       10      *       11      *       12
5       *       6       *       7       *       8
13      *       14      *       15      *       16
```

o.

```
1       *       2       *       3       *       4
9       *       10      *       11      *       12
13      *       14      *       15      *       16
5       *       6       *       7       *       8
```

p.

```
4       *       3       *       2       *       1
12      *       11      *       10      *       9
8       *       7       *       6       *       5
16      *       15      *       14      *       13
```

q.

```
1       *       2       *       3       *       4
9       *       10      *       11      *       12
17      *       18      *       19      *       20
13      *       14      *       15      *       16
5       *       6       *       7       *       8
```

r.

```
1
4       *       5       *       6
2       *       3
7       *       8       *       9       *       10
```

s.

```
4       *       4       *       4       *       4
3       *       3       *       3
2       *       2
1
```

2	*	2				
3	*	3	*	3		
4	*	4	*	4	*	4

t.

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