Q1. Read random numbers from a file and arrange odd numbers together and then the even numbers.
For example if the file "input.txt" contains the following numbers:

2 3 4 12 56 33 77 89

The output file "output.txt" will contain the numbers in the following order:

2 4 12 56 3 33 77 89

You need to do the following:
a. Use the function rand() to write 100 random numbers into the file "input.txt";
b. Read the file "input.txt" and put the content into an array.
c. Rearrange the content of the array in a way such that the even numbers are before the odd numbers.
d. Put the content of the array into the file "output.txt".

Q2. Given two arrays of size N and M , construct an array of size (N+M).
For example if the first array A is the following:
A:{1,2,3,4}
and the second array B is the following the following:
B:{7,8,9,10,11,12}
write a function stitchArray, that will take the two arrays A and B by reference and join array A with B so that array A looks like the following:
A:{1,2,3,4,7,8,9,10,11,12}

After calling stitchArray(A,B,N,M) from the main function and then iterating the array A in the main function for N+M times, the content of both the arrays will be observed. Remember to take memory leakage into account.

1.Write a program which will create a dynamically allocated 1D array of length n (given by the user) such that, 100 > n > 1000. Fill in the array with random variables using rand() %1000. Now print all the numbers, which appears more than once in the array.

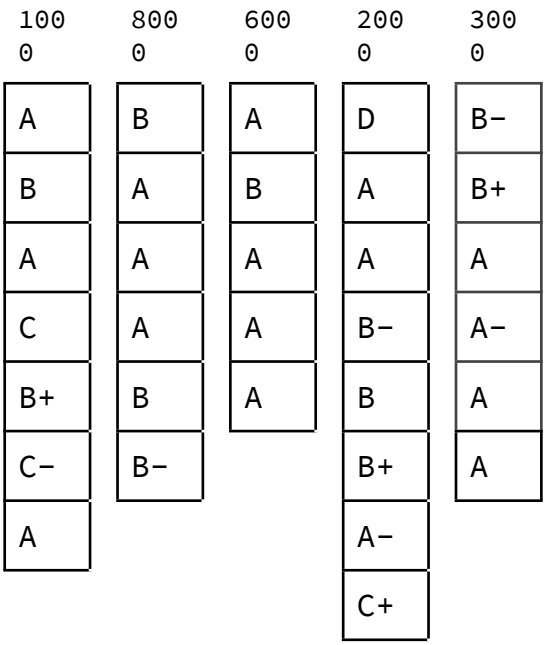
For example, if your program generates the following 100 random numbers:
41 467 334 500 169 724 478 358 962 464 705 145 281 827 961 491
995 942 827 436 391 604 902 153 292 382 421 716 718 895 447
726 771 538 869 912 667 299 35 894 703 811 322 333 673 664 141
711 253 868 547 644 662 757 37 859 723 741 529 778 316 35 190
842 288 106 40 942 264 648 446 805 890 729 370 350 6 101 393
548 629 623 84 954 756 840 966 376 931 308 944 439 626 323 537
538 118 82 929 541

Then the output of your program should be: 827 2 538 35

2.The final grades of a student in his 5 years of honours degree at IUB are given below:
1st Year: A B A C B+ C- A
2nd Year: B A A A B B-

3rd Year: A B A A A
4th Year: D A A B- B B+ A- C+
5th Year: B- B+ A A- A A
Use ideas of dynamic memory and struct where appropriate to encode the data given above. Your data structure should look theoretically like the one given below:

Num of Courses per year	Pointer to the array
7	1000
6	8000
5	6000
8	2000
6	3000



Hint: You can create an array of struct ‘StudentGrades’ which can hold two variables i.e., number of courses ‘numCourses’ taken by students in each year and a pointer ‘p’ to the base address of array of size equal to the number of courses, which holds the grades earned by the student in different years.
Definition of the struct:

```
struct student {
     int numOfCourses;
```

```
        int* grades;
};

int main() {
        //For one student taking 4 courses
        //it would look like
        student *s = new student;
        s->numOfCourses = 4;
        s->grades = new int[s->numOfCourses];

        return 0;
}
```

Write a program which will take marks of three quizzes as input, then call a function void ScaleUp(float q1, float q2, float q3) to check if the marks of any of the quiz 'q' is less than the average of rest of two quizzes 'a'. If marks of any quiz is less than the average of the rest of two marks then add the difference between the average of the two quizzes and the quiz itself to q i.e., q = a - q, to scale up the quiz marks.

For example, if q1 = 5, q2 = 8 and q3 = 9 then your function will change the marks of quiz 1 using the steps below:

A = (q3 + q2)/2 => (8+9)/2=> 8.5 Diff = A - q1 => 8.5 - 5 => 3.5
Q1 = Q1 + Diff => 5 + 3.5 => 8.5

Note that the marks are input from the user, so there might be three cases I. if q1 < q2 + q3 or
Ii. if q2 < q1 + q3 or
Iii. if q3 < q1 + q2

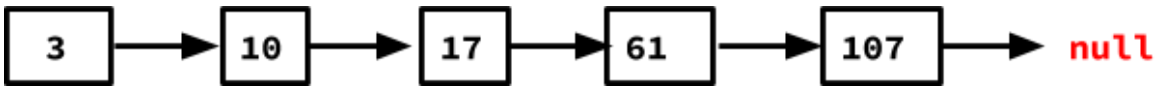
1. write a function bool insertByValueBefore (node *&sll, int svalue, int newvalue).

   a) The function will search the svalue in the linked list. If found, it will add newvalue before the node with svalue and the function will return true.

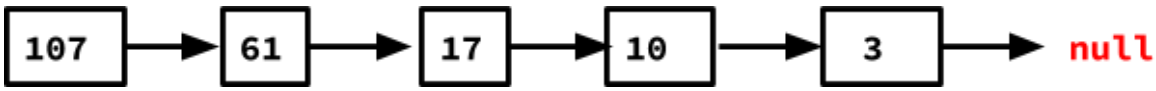
   b) if svalue is not found in the linked list, the function will return false.

2. Rewrite the function for problem 1 in such a way that if svalue is ound multiple times, the function will add the newvalue before each node with svalue.
3. write a function insertByPositionBefore (node *&sll, int pos, int newvalue). The function will add the newvalue in front of the node of position pos.
4. write a function insertByPositionAfter (node *&sll, int pos, int newvalue). The function will add the newvalue the node of position pos

5. Assume there is a singly linked list. HEAD is the pointer for head node. Write a function reverseList(node* &list) to reverse the linked list. It means, reverse the links.
Then print the reversed list.
If the list is as follow:
head
`



the output will be as follows:
head



Practice Problem using recursion.

1. Write a recursive function float power(int x, int n) that finally return the value x^n
2. Take n from user. Then write a recursive function int sum(int n) to calculate the sum 1 + 2 + 3 + ... + n
3. Take n from user. Then write a recursive function int sumPowers(int n) to calculate the sum 1^n + 2^n + 3^n + ...n^n
4. Write a recursive sum function for a linkedlist that return the sum of all nodes: int sumRec(node* sll)
5. Write a recursive function for a linkedlist that return the number of nodes: int countNodeRec(node* sll)
6. Write a recursive function bool chkPalindrome(string str) that will check whether the string **str** palindrome or not.