**BSc(H) Computer Science**

**Data Structures**

**Assignment-1**

**Q1)** Write recursive functions for the following:

1. A method that prints a pattern of 2\*(n-m+1) lines to the standard output. The first line contains m asterisks; the next line contains m+1 asterisks, and so on up to a line with n asterisks. Then the pattern is repeated backwards, going n back down to m.

Example output:

triangle(3, 5) will print this:

\*\*\*

\*\*\*\*

\*\*\*\*\*

\*\*\*\*\*

\*\*\*\*

\*\*\*

1. This question involves a game with teddy bears. The game starts when I give you some bears. You can then give back some bears, but you must follow these rules (where n is the number of bears that you have):
2. If n is even, then you may give back exactly n/2 bears.
3. If n is divisible by 3 or 4, then you may multiply the last two digits of n and give back this many bears. (By the way, the last digit of n is n%10, and the next-to-last digit is ((n%100)/10).
4. If n is divisible by 5, then you may give back exactly 42 bears.

The goal of the game is to end up with EXACTLY 42 bears. For example, suppose that you start with 250 bears. Then you could make these moves:

--Start with 250 bears.

--Since 250 is divisible by 5, you may return 42 of the bears, leaving you with 208 bears.

--Since 208 is even, you may return half of the bears, leaving you with 104 bears.

--Since 104 is even, you may return half of the bears, leaving you with 52 bears.

--Since 52 is divisible by 4, you may multiply the last two digits (resulting in 10) and return these 10 bears. This leaves you with 42 bears.

--You have reached the goal!

The function returns true if it is possible to win the game starting with n bears and false otherwise.

1. A method to print a pattern of letters as follows:

1. If the parameter c is 'A', then the output is 'A'.

2. For other values of c, the output consists of three parts:

-- the output for the previous letter (c-1);

-- followed by the letter c itself;

-- followed by a second copy of the output for the previous letter.

There is no '\n' printed at the end of the output.

Example output:

letters('D') will print : ABACABADABACABA

1. Print all permutations of a given string.

**Q2)** Write a program to implement a queue using two stacks. (Show enqueue and dequeue op)

**Q3)** Given a null terminated linked list, *in*, create a new null-terminated linked, list *out*, of the same length, such that node *i* of out contains the sum of the data in *in*'s nodes up to and including node *i* of list *in.*

**Q4)** Given a null terminated linked list, rearrange its nodes into two lists: <first node, third node, fifth node, ...> and <second node, fourth node, sixth node, ...>. Do not allocate any new nodes.

**Q5)** Implement a program which reads a file of infix arithmetic expressions as input, evaluates all of the expressions, and writes the resulting answers to the standard output. The filename should be specified on the command line as a parameter to the program.

**Q6)**  We have two queues, the input queue Q1 and the output queue Q2 and one stack S. We are only allowed to dequeue from Q1 and enqueue into the output queue. We are allowed to both push into and pop from the stack.

Consider that we have the numbers 1 2 3 4 5 6 enqueued in Q1. Suppose we perform the following sequence of operations:

enqueue(Q2, dequeue(Q1))  
push(S, dequeue(Q1))  
enqueue(Q2, dequeue(Q1))  
push(S, dequeue(Q1))  
enqueue(Q2, pop(S))  
enqueue(Q2, pop(S))  
enqueue(Q2, dequeue(Q1))  
enqueue(Q2, dequeue(Q1))

then the output queue contains 1 3 4 2 5 6

This is an example of what we call a \*stack permutation\* i.e. A stack permutation is a ordering of numbers from 1 to n that can be obtained from the initial ordering 1, 2, ... n by a sequence of stack operations as described above.

To clarify this, note that 1 5 3 4 2 6 is not a stack permutation. Intuitively this is because to enqueue 5 into Q2 after 1 we would have to push 2 3 4 into the stack which would then be output in the order 4 3 2, not in the order 3 4 2.

In this question you will be given a permutation of n numbers and asked to check if it is a stack permutation or not. Input is the number n and a permutation. If it is a stack permutation your program will have to return the sequence of operations that formed that permutation. If it is not a permutation, your program will have to say it is not a permutation.

**Q7)** Write a program that can be used as a database of student’s information for a department.

The program should be able to dynamically allocate or deallocate storage for the student’s records using linked lists.

The database should have the following fields: the first and last names, a course code and a grade for a student.

The program should display the following menu and give below functionalities:

Press 1 to insert a new record

Press 2 to delete a record

Press 3 to search the database (by last name)

Press 4 to print records in the database

Press 5 to find the class average for a course

Press 9 to quit