Name and ID#:______

AMERICAN UNIVERSITY OF ARMENIA

College of Science and Engineering

CS 121 Data Structures and Algorithms

MIDTERM 1 EXAM

60/121

Date: Tuesday, October 18 2016

Starting time: 09:00

Duration: 1 hour 15 min

Attention: **ANY TYPE OF COMMUNICATION IS STRICTLY PROHIBITED**

Please write down your name and ID# at the top of all used pages

Problem 1: Consider below two recursive expressions:

$$a_n = 1 + a_1*b_1 + a_2*b_2 + a_3*b_3 + \ldots + a_{n-1}*b_{n-1}$$
$$b_n = 1 + 2*b_1 + 2*b_2 + 2*b_3 + \ldots + 2*b_{n-1} - b_{n-1}*b_{n-1}$$

The base cases are: $a_1 = b_1 = 1$.

Write an optimal C++ function or Java method that takes as its argument an int index ***int n*** and returns a_n.

```
double computeBn (int n)      n = 1
{
        if(            )
        {
                     b1=1
        return             ;

        }

    return bn = 2*bn-1 - bn-1*bn-1

}
```

```
double computeAn ( int n)
{
    if(n = 1)
        return An = 1;

    return An = ComputeAn(int n-1)*
                ComputeBn(int n-1);

}
```

nice try

4

$a_1 = 1 + a_1 * b_1$

$b_1 = 1 + 2*b_1 + 2b_0 - b_0*b_0$

$b_0 = 1$

$a_0 = 1$

$b_1 = 1 + 2b_1 + 2 - 1$

$-b_1 = 2$

$b_1 = -2$

$a_1 = 1 = 1 + a_1 b_1$

$b_1 = 1 = 1 + 2b_1$

$b_1 = 0$

$b_4 = 2*b_3 - b_3*b_3 =$

$= 2(2b_2 - b_2*b_2) - (2b_2 - b_2*b_2)$

$b_2 = 2b_1 - b_1*b_1 = 2\cdot1 - 1\cdot1$
$= 1$

$b_2 = 1 + 2 - 1*1 = 2$

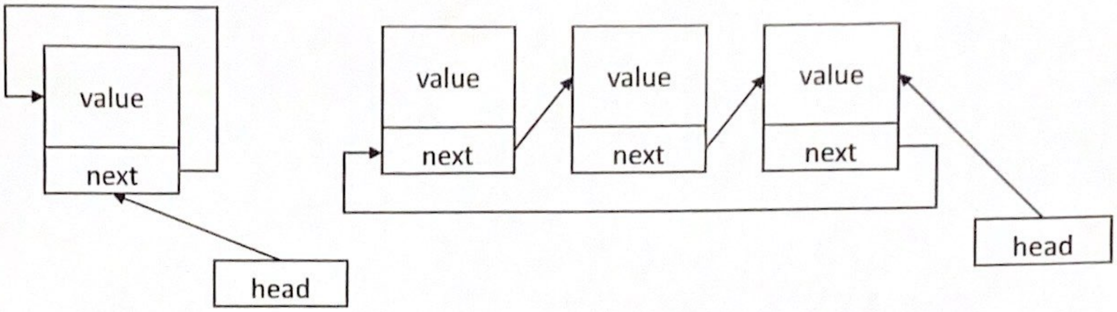
$b_2 = 1 + 2 - 1 = 2$

Use the backside, if needed

DS.MT1.181016.M152

Name and ID#:________

Problem 2: Consider the concept of a ***Circular Queue*** that implements ***Queue ADT*** as a circular list. It has a pointer ***node *head*** that points to the last node – the back, not the first one – the top. The address of the top, therefore, is kept in the pointer ***node *next*** of the last node – ***head->next***. If ***cqueue*** is empty, ***head = NULL***. If it has just one node, ***head = head->next***. Examples of one-node and three-node ***cqueue*** objects are shown below:



As usually, ***cqueue*** inserts a new value at the back, removes the top value and retrieves the top value. Derive a C++ ***class cqueue*** from ***class base*** that implements the concept of circular queue. Write the header and source files. The header file of ***class base*** and all its functions are implemented:

```
class base
{
public:
        base(); //the default constructor creates an empty base
        base(const base &that); //copy constructor
        ~base(); //destructor
        bool is_empty();
protected:
        struct node
        {
                int value;
                node *next;
                node(int new_value, node *new_next) : value(new_value), next(new_next){} ;
        };
        node *head;

        void insert(int new_value, int at, node* &from);
        bool remove(int at, node* &from);
        int retrieve(int at, node* from);
}
```

DS.MT1.181016.M152

Source file-?

1

header

```
class cqueue : base
{
  public:     cqueue();
              cqueue(const base &that);
              ~cqueue;
     void insert(int new_value, int at, node* &from);
     bool remove(int at, node* &from);
     int retrieve(int at, node* &from);


  protected:
      struct node
      { int value;
        node *next;
        node(int new-value, node* new_next): value(new-value),
                                                next(new-next){};
      }
      node *head;
}
```

Use the backside, if needed