First Midterm Exam (b) MCS 360 Jan Verschelde Friday 13 October 2017

NAME : ***answers***

The exam is closed book, no notes and no computer.
All your answers to the questions below must be submitted on paper.
Write your name on this sheet and submit it with your answers.
Please do not ask questions during the exam.

1. Describe the Abstract Data Type of a queue to store elements of any type.

   In your description of the operations on the queue,
   specify all relevant preconditions and postconditions.

   ***Answer:***

```
abstract <typename T> queue;
/* a queue is a sequence of elements,
   FIFO = First In First Out */

abstract bool empty ( queue q );
postcondition: empty(s)
   == true if queue is empty,
   == false otherwise;

abstract T pop ( queue q );
precondition: not empty(q);
postcondition: first element is removed from q;

abstract void push ( queue q, elements e );
postcondition: element e is at end of queue q;
```

2. Consider the problem of computing the sum of an array of arrays of doubles.

   - Write a C++ function `summate` to compute the sum of the numbers in `double A[n][n]`.
     In addition to `A`, the other input argument is `n`, of type `size_t`.
     The function returns a `double`, the sum of the numbers in `A`.
   - The cost of this problem is the number of addition operations. Count the number of additions for general `n` and use the big $O(\cdot)$ notation to express the cost.
     Justify your count, *writing complete sentences*, and the cost of this problem.

***Answer:***

- A function to summate an array of arrays of doubles:

```
double summate ( size_t n, double* A[n] )
{
   double result = 0.0;

   for(int i=0; i<n; i++)
      for(int j=0; j<n; j++)
         result = result + A[i][j];

   return result;
}
```

- The cost of this problem is $O(n^2)$.
  The statement `result = result = A[i][j]` is executed for every `i` from 0 to `n-1` and for every `j` from 0 to `n-1`. The number of additions thus equals $n \times n = n^2$.

3. Consider two scenarios to process a number of integers:

   (a) The number of integers is unpredictable with each run of the program.
   Sometimes only a couple of integers are stored, while at other times, the number of integers equals several thousands.

   (b) In each run of the program, the number of integers ranges between 20 and 30.

   Which data structure, vector or list, is best for which scenario?
   Justify your choice of data structure and *write complete sentences* for your justification.
   In your arguments, refer to the advantages and disadvantages of vectors and lists.

   ***Answer:***

   A list is best for (a) while for (a) a vector is best.

   A list is a dynamically allocated data structure, which allows us to be frugal with memory. If only a couple of numbers will be processed, then we will store only those couple of numbers. If we were to use a vector, then we always would need to allocate space for several thousands of numbers, which would be wasteful for the cases in which only a couple of numbers are processed.

   The advantage of a vector is fast access: we can retrieve every element in constant time. This fast access comes at the expense of static allocation, we have to allocate all elements in advance. In scenario (a) we can always allocate space for 30 elements and there will not be a lot of waste as the number of numbers is never less than 20.

4. The code

```
class List
{
   struct Node
   {
      string data;
      Node *next; // pointer to the next node
      Node(const string& item, Node* ptr = NULL) :
         data(item), next(ptr) {}
   };
   Node *first;   // pointer to the first node
   Node *last;    // pointer to the last node
```

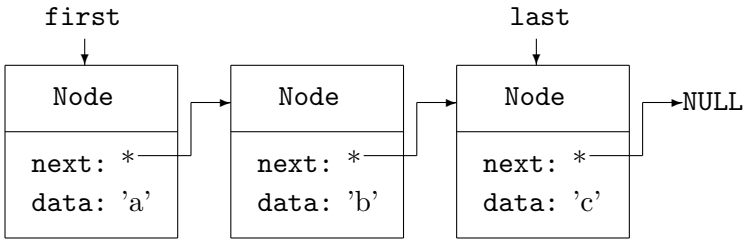
defines a linked list where `first` points to the first node and `last` points to the last node in the list. Consider the operation to insert a string to the front of the list.

(a) Draw a list of three elements and illustrate your drawing to show the steps in the insert operation of a string to the front of the list.

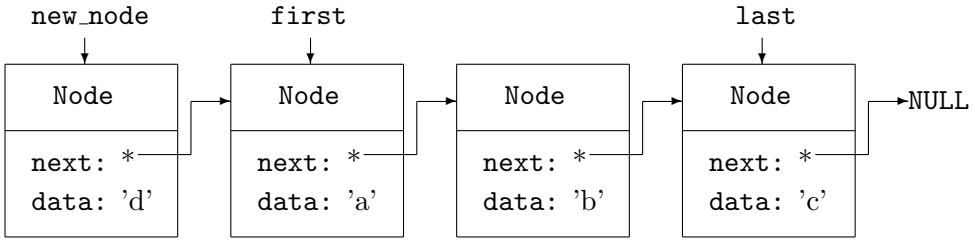
(b) Write code for a function to insert a string to the front of the list.

***Answer:***

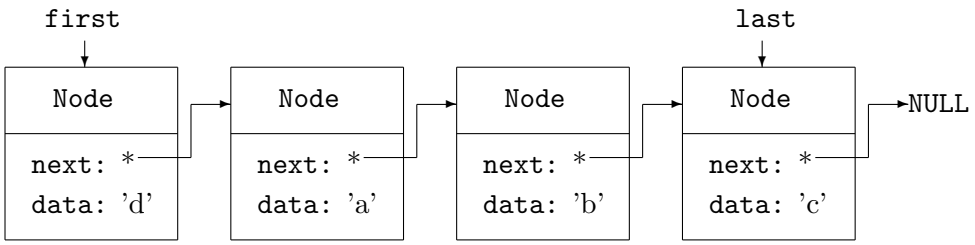
(a) A drawing of a linked list of three elements:




First step: make a new node with as data a string and `new_node->next = first`.




Second step: `first = new_node`.



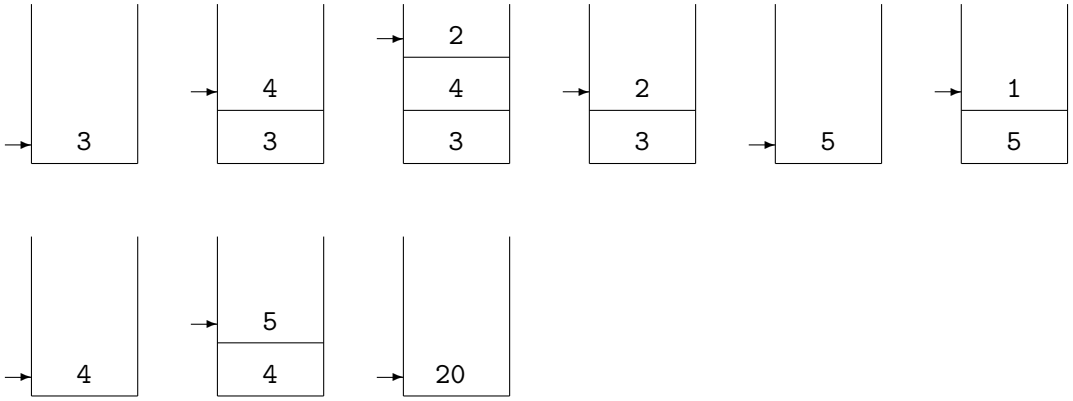
(b) Code for a method to insert a string to the front of a list.

```
void List::insert(string item)
{
   if(first == NULL)
   {
      first = new Node(item);
      last = first;
   }
   else if(first == last)
   {
      first->next = new Node(item);
      last = first->next;
   }
   else
   {
      Node* new_node = new Node(item);
      new_node->next = first;
      first = new_node;
   }
}
```

5. Evaluate the postfix expression `3 4 2 / + 1 - 5 *` using a stack.

   Draw all intermediate stages of the stack. Write the corresponding infix expression.

   ***Answer:***




The infix expression corresponding to `3 4 2 / + 1 - 5 *` is `( 3 + 4 / 2 - 1 ) * 5`.