First Midterm Exam (a) 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 stack to store elements of any type.

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

   Answer:

```
abstract <typename T> stack;
/* a stack is a sequence of elements,
   stored as Last In First Out (LIFO) */

abstract bool empty ( stack s );
postcondition: empty(s)
   == true if s is empty,
   == false if s is not empty;

abstract T pop ( stack s );
precondition: not empty(s);
postcondition: push(s,pop(s)) == s;

abstract void push ( stack s, T e );
postcondition: push(s,e); pop(s) == e;
```

2. Consider the problem of finding the minimum value in an array of arrays of integers.

   - Write a C++ function `find_min` to compute the minimum value of `int A[n][n]`.
     In addition to `A`, the other input argument is `n`, of type `size_t`.
     The function returns an `int`, the smallest element in `A`.
   - The cost of this problem is the number of comparison operations. Count the number of comparisons 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 C++ function to compute the minimum value of a matrix:

```
int find_min ( size_t n, int* A[n] )
{
   int result = A[0][0];

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

   return result;
}
```

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

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

   (a) In each run of the program, the number of strings ranges between 10 and 20.

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

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

   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 20 elements and there will not be a lot of waste as the number of strings is never less than 10.

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

4. The code

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

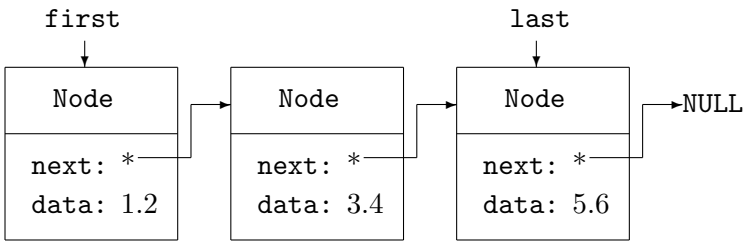
defines a linked list where `first` points to the first node and `last` points to the last node in the list. Consider the operation to append a number to the end of the list.

(a) Draw a list of three elements and illustrate your drawing to show the steps in the append operation of a number to the end of the list.

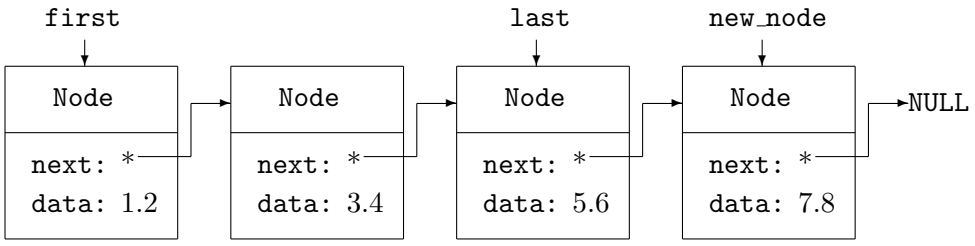
(b) Write code for a function to append a number to the end of the list.

Answer:

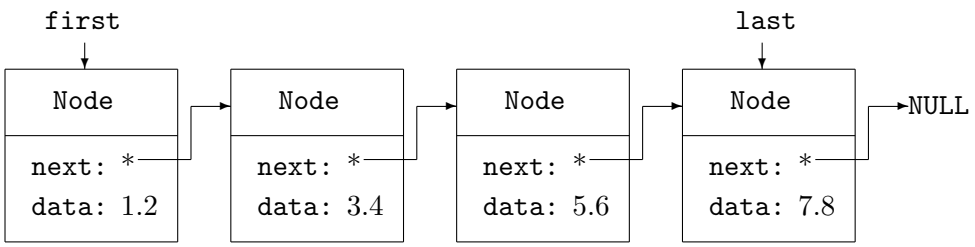
(a) A drawing of a linked list of three elements:



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



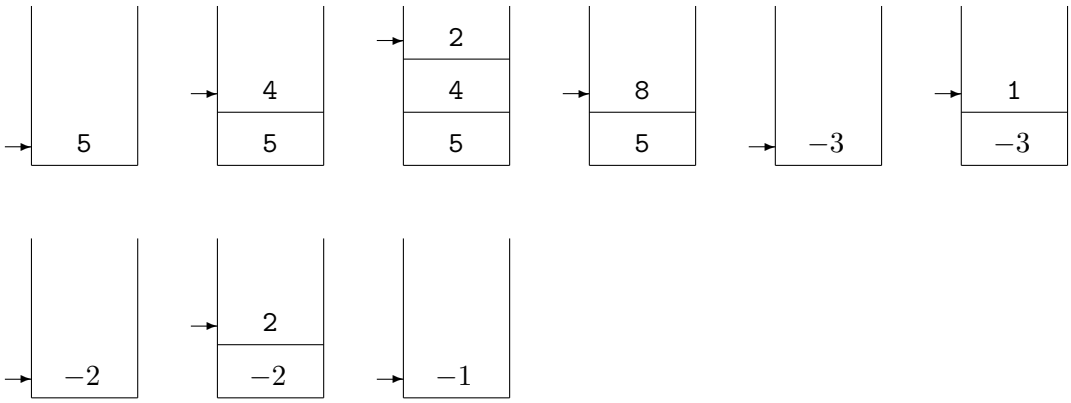
Second step: `last = last->next`.



(b) Code for a method to append a double to the end of a list.

```
void List::append(double item)
{
   if(first == NULL)
   {
      first = new Node(item);
      last = first;
   }
   else if(first == last)
   {
      first->next = new Node(item);
      last = first->next;
   }
   else
   {
      last->next = new Node(item);
      last = last->next;
   }
}
```

5. Evaluate the postfix expression `5 4 2 * - 1 + 2 /` using a stack.
   Draw all intermediate stages of the stack. Write the corresponding infix expression.
   Answer:



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