National University of San Agustin
System Engineer School

TO - Lab 01 - Introduction to OOP

Christian E. Portugal-Zambrano

August 29, 2018

1 Introduction

Welcome to Object Oriented Technology, this is and advanced course for Object Oriented Programming, we need to remember some basic object oriented concepts to get up to date in the course. In this Laboratory you will learn about some concepts and tools that will help you through the course.

2 Pre-requisites

You need some OOP experience, but here you will review and learn more concepts related to the area, throughout the course we will be using C++ as primary language but after a few weeks Java or Python could be selected. Some knowledge of data structures, loops, functions, recursion and some math libraries would be required to this course.

3 Programming environment

Don't worry much about this, as your instructor I will not force you to choose a specific one, contrary to this I encourage you to select from these or another one you are already familiarized:

- Eclipse - https://www.eclipse.org
- Netbeans - https://netbeans.org

- Anaconda - https://www.anaconda.com
- Visual Studio (Code) - https://visualstudio.microsoft.com/es/
- Jetbrain IDE's - https://www.jetbrains.com
- Atom - https://ide.atom.io
- Sublime - https://www.sublimetext.com

All IDE's I've mentioned above are multi-platform and GUI based, but if you are a console user probably you will feel comfortable with VIM or EMACS. For this course Eclipse, Netbeans, Visual Studio and CLion from Jetbrains are good for starting C++.

4 OOP: A REVIEW

So, we need to review your OOP concepts, but first as we are working with LaTeX, presentation of code into your reports will be easy, you just need to add this at the header of your document:

```
\usepackage{listing}
```

You can find more information about Listing package from CTAN:Listing Let's get started!

4.1 CONCEPTS

We need to review some concepts, so please ask this questions:

- ¿What is a fundamental data type?
- ¿Is a fundamental data type the same that a class?
- ¿What do you understand about a class?
- ¿What would be the difference between a structured type and a class?
- ¿What would be the difference between an object and a class?
- ¿Is there some difference between a package and a namespace?
- ¿A variable is the same that a attribute?
- ¿A function is the same that a method?

Now, is necessary to make some research for fill this table:

Language	Paradigm	Inheritance	Package	Concurrency	Garbage C.	Typed
C						
C++						
Java						
Python						
Javascript						
MatLab						
R						
FORTRAN						
GO						
LUA						
C#						
SmallTalk						
ADA						
Cobol						

5 Warm up

This is the class for a Linked List with an OOP programming approach (here you must learn or review templates, function and operators overloading):

```
template <class T>
class List{
public:
   List(); // constructor
   ~List(); // destructor
   void insert(T _data); //insert an element
   void erase(T _data);     //erase and element if exist
   void clear();  //clear all the list
   void remove(int _pos); //remove element at _pos
   void reverse(); //reverse the entire list
   T at(int _pos); //get element at _pos
   bool isEmpty(); //true if is empty
   bool save(std::string); // save to a file
   bool load(std::string); // load from file
   unsigned int size(); //size of the list
   void show(); //show the list to console
   T operator[](int); //overload []
   void operator<<(T); // overload <<
private:
   Node<T>* proot;
   int Size;
};
```

And its node class, here arises one questions, ¿What would have been the correct order of design?:

```
template <class T>
class Node{
private:
    T data;
    Node<T>* next;

public:
    Node(T _data):data(_data),next(NULL){}
    ~Node(){}

    T getData(){
        return data;
    }

    Node<T>* getNext(){
        return next;
    }

    void setNext(Node<T>* _next){
        next = _next;
    }
};
```

The objective of this is implement the class and try to run some examples, to run your class you must implement a main.cpp file with the next code:

```
#include "node.h"
#include "list.h"

int main(){
    List<int> myfirstList;
    //some operations

    return 0;
}
```

6 To do

After you implement the class using the tools of your preference, you must include your implementation into the report and try to ask this questions:

- ¿How many constructors should be there?
- ¿Is necessary to implement all the constructors?
- ¿How much memory you are using?
- ¿After the program finished, the memory was released?
- ¿A constructor can be private?

- ¿If I make a change into List, would it affect node implementation?
- ¿When you implemented operator overload, have you noticed if it is really necessary?
- OK, those were all the question for now, but prepare yourself for the next lab because we will implement Double Linked List,Stacks, Queue, Trees, Dense Matrix and Sparse Matrix.

7 Deep inside

This section is just for anyone who wants to make a deep inside into the theory and like challenges[1], you will have to prepare a presentation of just 5 minutes to explain and run the code, then you will have to defend yourself another five minutes of questions. I want that all students benefit from your presentation, remember that we are here to learn. If you want to do this please email to cportugalz@unsa.edu.pe

8 Deadline

For this course we have three groups, according to this the deadline for each group is six days after the lab group scheduled. Remember that plagiarism must be avoided and if it is detected the grade will be zero and reincidence informed to superior authorities. A pdf unnamed and named report must be sent to cportugalz@unsa.edu.pe before deadline, after deadline you will be qualified under the minimum grade obtained with a maximum of thirteen and penalized with 1 point per day late. All question and doubts must be done to the same email.

[1] This must not be included into the report