UNIVERSIDAD NACIONAL DEL LITORAL
FACULTAD DE INGENIERÍA Y CIENCIAS HÍDRICAS
DEPARTAMENTO DE INFORMÁTICA

Ingeniería en Informática

PROGRAMACIÓN ORIENTADA A OBJETOS

Ingeniería en Inteligencia Artificial

PROGRAMACIÓN II

UNIDAD 7
Programación Genérica

Guía de Trabajos Prácticos

EJERCICIOS BÁSICOS

Ejercicio 1

1. Implemente una función templatizada llamada `mayor(...)` que reciba dos valores y devuelva el mayor. Compruebe el correcto funcionamiento de la rutina utilizándola desde un programa cliente con valores de tipo `int`, `float` y `string`
2. Programe una sobrecarga de la función `mayor(...)` que reciba un `std::vector` y retorne la posición del mayor elemento del mismo. Pruebe la función sobrecargada desde un programa cliente con diversos tipos de datos.
3. Responda: ¿Servirán las funciones anteriores con datos del tipo mostrado en el recuadro? Justifique su respuesta. Si responde que no, implemente los cambios necesarios.

```
struct Persona {
    string nombre;
    string apellido;
    int dni;
};
```

Ejercicio 2

1. Implemente una función `clamp(...)` que reciba como parámetros una variable (por referencia) y dos valores indicando los lí mites superior e inferior para el valor de dicha variable. Si el valor de la variable supera su máximo, este debe ajustarse al máximo valor permitido. De la misma forma si el valor es inferior al mínimo.
2. Pruebe la función templatizada desde un programa cliente. Explique el error que ocurre al invocar la función con `float f=0.5; clamp(f, 0, 1);` ¿Cómo lo solucionaría?

Ejercicio 3

Programe una clase templatizada llamada `VectorDinamico` (similar a la de la guía 2). La clase debe poseer:

1. Un constructor que reciba el tamaño inicial del vector, y reserve la memoria necesaria.
2. Un destructor que se encargue de liberar la memoria reservada.
3. Una sobrecarga del operador `[]` que reciba el número de elemento, devuelva su valor, y permita modificarlo.
4. Modifique o sobrecargue el constructor para que permita generar valores aleatorios con los cuales inicializar las posiciones del arreglo que reserva.
5. Utilice la clase desde un programa cliente creando vectores aleatorios con diversos tipos de datos (`int`, `double`, `string`, etc).

Ejercicio 4

Desarrolle una clase templatizada llamada `ManejadorArchivo` que posea métodos y atributos para manipular un archivo binario que contenga registros del tipo de dato especificado por el parámetro (mediante un vector en memoria, no directamente sobre el archivo). La clase debe poseer métodos para:

1. Abrir un archivo binario y cargar los registros en memoria.
2. Obtener el registro en una posición especificada por el usuario.
3. Modificar el registro en una posición determinada.
4. Actualizar la información del archivo con los cambios.

Utilice la clase desde un programa cliente para leer los registros escritos en el archivo binario generado en el ejercicio 6.5.

CUESTIONARIO

1. ¿Qué es la programación genérica? ¿Cómo se implementa en C++?
2. ¿Cuál es la diferencia entre una plantilla de clase y una plantilla de función?
3. ¿A qué se denomina instanciación o especialización de una plantilla?
4. ¿Cómo puede saberse de antemano si la instanciación de una plantilla con un determinado tipo de dato arrojará errores?
5. ¿Por qué la implementación de una clase templatizada no puede separarse en dos archivos .cpp y .h?

EJERCICIOS ADICIONALES

Ejercicio 1

Diseñe e implemente una clase templatizada llamada `Rect` que permita representar un rectángulo. La clase debe poseer métodos para obtener el alto y ancho del rectángulo. Implemente los métodos que considere necesarios para la inicialización o carga de datos. Cree un programa cliente que instancie un rectángulo cuyas coordenadas sean enteros y otro cuyas coordenadas sean de tipo double.

Ejercicio 2

Desarrolle una función `intercambia(...)` que reciba dos variables por referencia e intercambie sus valores. Pruebe la función desde un programa cliente con al menos dos tipos de dato distintos.

Ejercicio 3

Implemente una clase genérica `VectorEstatico` similar a la del ejercicio 2, pero utilizando un vector estático (de tamaño fijo), cuya longitud será el argumento de la plantilla. ¿Qué ventajas tendría el uso de dicha clase sobre el uso de un arreglo estático común?