Sección teórica. 30 %

1. La firma del método *setter* para el atributo tipo *string matriculaAlumno* de la clase ***Alumno*** es:
   a. string setMatricula();
   b. void setMatricula(string);
   c. void setMatricula();
   d. string setMatricula(string);

2. La firma del constructor por omisión de la clase ***Examen*** sería:
   a. void Examen();
   b. void Examen(){};
   c. Examen();
   d. Examen Examen();

3. En el siguiente segmento de código en C++, completa lo que se pide atendiendo la sintaxis correcta del lenguaje.

```
class MiClase {

    int var1;

    public:

            MiClase();
    ____void setVar1(int var1);____ //Coloca aquí la declaración o firma del método setVar1
    ____int getVar1();____ //Coloca aquí la declaración o firma del método getVar1

    …….

};
```

Diseña el contenido del constructor por omisión y del método *setVar1*. Inicializa a *var1* en 101 en el constructor y modifícalo más adelante por el valor 402;

```
// Coloca aquí el código que implementa al constructor por omisión.
MiClase() : var1(101){ }

// Coloca aquí el código que implementa al método setVar1
void setVar1(int var1){
  this -> var1 = var1;
}
```

4. Siguiendo con el ejemplo anterior, completa la siguiente aplicación que utiliza MiClase.

```
#include “Miclase.h”

int main(){
    MiClase miObjeto = MiClase() ; // Declara el objeto miObjeto de tipo MiClase
        miObjeto.setVar1(1500) ; // Cambia por 1500 el valor de var1 de miObjeto
std::cout << miObjeto.getVar1() << std::endl ; // Muestra en pantalla el contenido de var1 de miObjeto
        return 0;
}
```

5. Usando el estándar revisado en clase, diseña el diagrama de clases de **MiClase**.

Sección práctica. 70 %

1. **Diseña** e **implementa** en **C++** la **clase** modelada en el diagrama de la figura 1. Revisa en el apartado Especificaciones algunos requerimientos básicos que deberás considerar.
2. **Diseña** los **casos** de **prueba** que te permitan probar casos generales de tu clase.
3. **Coloca** en el **diagrama** (figura 1) el tipo de relación adecuada entre las clases si en la clase Empleado se utiliza un atributo de clase Fecha.
4. En función de los casos de prueba, **agrega** una **aplicación** en consola que permita implementar los casos de prueba.

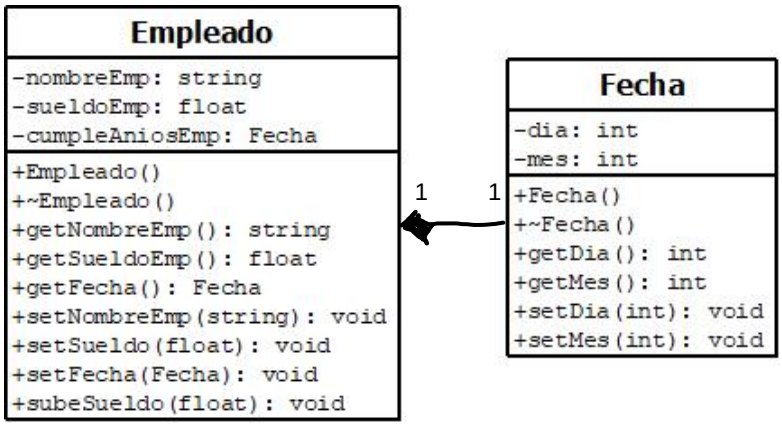

Figura 1. Diagrama de clases. Clase Empleado y clase Fecha.

Especificaciones.

1. Incluye los **constructores** señalados para cada clase.
2. Incluye los métodos ***setters*** y ***getters*** requeridos para ambas clases.
3. ***subeSueldo***. El método recibe como parámetro un porcentaje de incremento entre el 1 y el 10% Si el valor recibido no está dentro de este rango, no se aplicará incremento alguno y se mostrará el mensaje en pantalla "*Porcentaje inválido, no hay incremento salarial*". Si el porcentaje es correcto, al sueldo del empleado deberá aplicarse el incremento.
4. En tu aplicación,
    a. Declara dos empleados, *emp1* y *emp2*. El empleado emp1 se llama "Pepe", gana 10000 y su cumpleaños es mayo 19. El empleado emp2 se llama "Rocio", gana 12000 y cumple años el 1 de enero.
    b. Muestra en pantalla los datos completos de los dos empleados. Acomoda los datos de cada empleado en un mismo renglón.
    c. Intenta subirle el salario a Pepe un 15%.
    d. Muestra en pantalla nombre y sueldo de Pepe.
    e. Muestra en pantalla la fecha del cumpleaños de Rocio.
    f. Intenta subirle el salario a Pepe un 5%.
    g. Muestra en pantalla el nombre y suelde de Pepe.