# Certamen 2, Programación II

Prof. Rodrigo Olivares

Diciembre 01, 2016

**Instrucciones**:

- El puntaje máximo del certamen es 100 %, siendo el 60 % el mínimo requerido para aprobar.
- Responda la primera y segunda parte de certamen, en hoja indicada, agregando su nombre.
- Responda la tercera parte de certamen, según se indica en el aula virtual.
- El certamen es **individual**. Cualquier intento de copia, será sancionado con nota **1,0**.

1. 30pts. De las siguentes afirmaciones, encierre en un círculo la o las alternativas correctas.

i. **Una clase abstracta**:
(a) Posee métodos abstractos implementados.
(b) Posee atributos abstractos definidos.
(c) Instancia objetos abstractos.
(d) Permite extender una interfaz.
(e) Permite implementar una clase padre.

ii. **Respecto a las interfaces**:
(a) Su constructor es creado en compilación.
(b) Sus métodos definidos deben ser void.
(c) Sus métodos siempre deben abstract.
(d) Sus métodos implementados son abstract.
(e) Se extiende.

iii. **Sobre la herencia**:
(a) En java se realiza con la palabra reservada extends.
(b) En java se realiza con la palabra reservada implements.
(c) Todas las clases heredan de la clase Object.
(d) La clase padre hereda el comportamiento de la clase hija.
(e) La clase hija hereda el comportamiento de la clase padre.

iv. **Sobre la herencia múltiple**:
(a) Permite heredar diverso compartimiento.
(b) Apoya el principio ocultamiento.
(c) Apoya el principio de encapsulamiento.
(d) En Java se desarrolla implementado clases abstractas.
(e) En Java se desarrolla implementado interfaces.

v. **Un TDA Bean**:
(a) Sólo tiene atributos.
(b) Sólo tiene métodos.
(c) No debe incluir lógica.
(d) El constructor debe ser siempre incluido.
(e) Es posible agregar métodos como equals() y toString().

vi. **Un TDA Lista**:
(a) Se describe como una colección de nodos de objetos.
(b) Tiene un tamanño variable.
(c) Se almacena de forma contigua en memoria.
(d) La inserción de elementos es al final de la lista.
(e) La eliminación de elementos es al final de la lista.

vii. **La clase Vector**:
(a) Es un List.
(b) Es un ArrayList.
(c) Es synchronized.
(d) Es no synchronized.
(e) Es un nodo.

viii. **La clase ArrayList**:
(a) Es un List.
(b) Es un Vector.
(c) Es synchronized.
(d) Es no synchronized.
(e) Es un nodo.

ix. **En relación a la manipulación de archivos**.
(a) Se lee un archivo con la instancia FileWriter.
(b) Se lee un archivo con la instancia FileReader.
(c) Se lee un archivo con la instancia Scanner.
(d) Se leen sólo archivos con delimitar.
(e) StringTokenizer se usa para archivos con delimitador.

x. **En relación a la manipulación de archivos**.
(a) Se escribe un archivo con la instancia FileWriter.
(b) Se escribe un archivo con la instancia FileReader.
(c) Se escribe un archivo con la instancia StreamReader.
(d) No es factible agregar contenido a un archivo existente.
(e) FileWriter("f.txt", false) agrega el contenido al final.

2. 10pts. De acuerdo a las siguientes clases/interfaces, describa la salida.

```
public abstract class ClaseHijaMayor extends ClaseAbuelo implements ClaseMadre,
    ClasePadre {
  private String titulo;
  private String mensaje;

  @Override
  public void saludar() {
    titulo = "Saludo";
    mensaje = "Saludando desde ClaseHermanoMayor";
    saludar(mensaje);
  }

  private void saludar(String mensage) {
    System.out.println(mensaje);
  }

  private void saludar(String titulo, String mensage) {
    System.out.println(titulo);
    System.out.println(mensaje);
  }
}

public abstract class ClaseAbuelo {
  public abstract void saludar();
}

public final class ClaseHijaMenor extends ClaseHijaMayor {
  public ClaseHijaMenor() {
    super.saludar();
  }

  @Override
  public void saludar() {
    System.out.println("Saludando desde ClaseHijaMenor");
  }

  public static void main(String[] args) {
    ClaseHijaMenor cHija = new ClaseHijaMenor();
    cHija.saludar();
  }
}

interface ClasePadre {
  public abstract void saludar();
}

interface ClaseMadre {
  public abstract void saludar();
}
```

| Salida |
| --- |
| `Saludando desde ClaseHermanoMayor`<br>`Saludando desde ClaseHijaMenor` |

3. 60pts. Como Ingeniero Civil en Informática, se le ha solicitado desarrollar un sistema de registro de personas pertenecientes a la Escuela. Para eso, usted debe:

   a) Pedir al usuario:

      1) Los datos personales (DNI, Nombre, Apellidos, Edad, Dirección, etc.)
         - Para la dirección, debe considerar que puede tener dirección laboral y personal.
         - La dirección contempla: Calle, Número, Comuna.
         - El sistema debe desplegar el listado de regiones (ordenadas por orden geográfico), Luego de ingresar una región, el sistema debe desplegar las provincias pertenecientes a la región seleccionada. Al ingresar una provincia, el sistema debe desplegar las comunas.
      2) Tipo de persona (Académico, Alumno, Funcionario).
         - Si es Académico, se debe registrar su horario de atención.
         - Si es Alumno, se debe registrar las asignaturas que se encuentra cursando en el semestre (pueden ser más de una).
         - Si es Funcionario, se debe registrar su horario laboral.

   b) La información solicitada, debe ser registrada en un archivo de texto plano, que deberá crearse, si no existe y agregar información al final si ya fue creado.

   c) Si la persona ya existe en el archivo, se debe eliminar su registro e ingresar nuevamente su información (No deben existir dos registros con el mismo DNI).

   → **Recomendación**: Utilice la clase FuenteDatos para la manipulación de los archivos.

   → **Restricción**: Utilice los archivos Region.csv, Provincia.csv y Comuna.csv que se encuentran en el aula virtual. El formato de los archivos es el siguiente:

      - **Region.csv**: IdRegion;NombreRegion;NumeroRomano;OrdenGeograficoRegion
      - **Provincia.csv**: IdProvincia;NombreProvincia;IdRegion
      - **Comuna.csv**: IdComuna;NombreComuna;IdProvincia

   → **Formato de entrega**: Debe subir los archivos fuente ***.java** en un comprimido **.zip** al aula virtual, con el siguiente formato:

      - ApellidoPaternoNombreC2.zip

   → **Fecha de entrega**: Viernes 02 de Diciembre, hasta las 12:00 hrs.

| **¿Cómo seré evaluado en la pregunta 3?** | | | |
|---|---|---|---|
| **Tópico** | **Logrado** | **Medianamente logrado** | **No logrado** |
| Manipulación de archivo. | 10 % Lee correctamente los archivos, los mapea a entidad y escribe en el archivo de salida. | 5 % Realiza dos de las tres acciones del punto anterior. | 0 % No realiza la acciones del punto anterior. |
| TDA Lista. | 40 % Crea la clase TDA Registro e implementa todos los métodos. | 20 % Crea la clase TDA Registro e implementa algunos métodos. | 0 % No crea la clase TDA Registro. |
| TDA Bean / Entidad | 20 % Crea correctamente las clase entidades: Región, Provincia, Comuna, Dirección, Persona, Académico, Alumno, Funcionario. | 10 % La cantidad de clases entidad correctamente es igual o inferior a 4. | 0 % No crea las clases entidad. |
| Clase principal y método main. | 10 % Crea la clase principal en un archivo independiente con el método main. | 5 % Crea el método main en la misma clase. | 0 % No crea el método main. |
| Paradigma Orientación a Objetos | 20 % Resuelve el problema utilizando el POO (herencia, abstacción, interfaces, etc). | 10 % Utiliza parte del POO para resolver el problema. | 0 % No utiliza el POO para dar solución al problema. |
| Total máximo puntaje pregunta 2 | 100 % de 60 pts. | 50 % de 60 pts. | 0 % de 60 pts. |