Programación II. I.T.I. de Gestión

Ejercicios de Programación Orientada a Objetos

Curso 2009/10

Ejercicio 1

Completar la clase `PUNTO` implementando las operaciones que faltan y utilizar esa clase en una aplicación simple que explote todas las características de `PUNTO`. Añadir métodos `to_string_cartesianas` y `to_string_polares` que devuelvan las coordenadas en cartesianas y polares respectivamente representadas en una cadena de caracteres, entre paréntesis y separadas por una coma.

Nota: Las funciones seno, coseno, etc. son aplicables a variables de clase `DOUBLE`. Aplicar `short` a `double.e` nos permite conocer cómo se manejan.

Ejercicio 2

Programar un conjunto de clases que representen figuras geométricas sencillas como triángulo, rectángulo, polígono en general, basándose en la clase `PUNTO` y si es necesario en clases predefinidas como arrays o listas. Debe ser posible desplazar la figura, imprimir sus vértices, calcular el perímetro, etc. `ARRAY` y `LINK_LIST` se encuentran en la biblioteca standard de SmallEiffel.

Ejercicio 3

Elaborar una clase `RACIONAL` que modele los números racionales implementando al menos las operaciones de suma, resta, opuesto e inverso de un número racional a imitación de la suma o resta de los números reales o enteros.

Ejercicio 4

Elaborar una clase `COMPLEJO` que modele los números complejos implementando al menos las operaciones de suma, resta y módulo de un número complejo a imitación de la suma o resta de los números reales o enteros.

Ejercicio 5

Elaborar una clase `POLINOMIO` que modele los polinomios de grado dos implementando al menos las operaciones de suma (de un polinomio con `Current` para obtener un tercer polinomio) y producto por un número y el cálculo de las raíces reales del polinomio, si es que existen.

Ejercicio 6

Escribir una clase `RELOJ` que simule el comportamiento de un cronómetro digital (con las características `puesta_a_cero`, `incremento`, etc.). Cuando el contador llegue a `23:59:59` y reciba el mensaje de `incremento` deberá pasar a `00:00:00`.

Ejercicio 7

Elaborar una clase Eiffel que modele una fecha. La clase deberá disponer de características que devuelvan el día, el mes y el año, además de métodos que que devuelvan un *STRING* con la fecha en forma abreviada (16/02/2000) y extendida (16 de febrero de 2000) y de una función *incremento*, con un parámetro entero, que fabrique una nueva fecha, resultado de incrementar la original en ese número de días.

Nota 1: Son años bisiestos los múltiplos de cuatro que no lo son de cien, salvo que lo sean de cuatrocientos, en cuyo caso si son bisiestos.

Nota 2: Para la solución de este problema puede ser útil definir un método *incrementa_un_dia*.

Nota 3: El ejercicio es mucho más divertido si no se utiliza la clase *TIME* de la biblioteca estandar de SmallEiffel

Ejercicio 8

Elaborar una clase Eiffel que implemente de forma sencilla una pila genérica, utilizando para el almacenamiento un *ARRAY*. Elaborar una segunda implementación del mismo estilo, pero utilizando para el almacenamiento una *LINKED_LIST*.

Ejercicio 9

Dentro de una biblioteca Eiffel en funcionamiento disponemos de una clase empleado definida del siguiente modo:

```
class EMPLEADO
creation {ANY}
  make
feature {ANY}
  nombre: STRING;
  edad: INTEGER;
  nif: STRING;
  make(elnombre: STRING; laedad: INTEGER; elnif: STRING) is
    do
      nombre := clone(elnombre);
      edad := laedad;
      nif := clone(elnif);
    end; -- make
  muestra is
    do
      io.put_string("Nombre: ");io.put_string(nombre);io.put_string("%N");
      io.put_string("edad: ");io.put_integer(edad);io.put_string("%N");
      io.put_string("NIF: ");io.put_string(nif);io.put_string("%N");
    end; -- muestra
end -- class EMPLEADO
```

Al añadir nuevas capacidades a la biblioteca descubrimos que necesitamos modelar nuevos tipos de empleados:

- Empleado temporal, del que nos interesa saber la fecha de alta y de baja en la empresa.
- Empleado por horas. Nos interesa el precio de la hora trabajada, y el número de horas que ha trabajado este mes. El primero es un dato fijo, mientras el segundo varía todos los meses.
- Empleado fijo. Debemos añadir a la información que almacenamos sobre él el año de alta en la empresa.

Además debemos añadir a todos los empleados la funcionalidad de cálculo del sueldo con las siguientes consideraciones:

- En los empleados temporales el sueldo mensual es fijo.
- En los empleados fijos el sueldo es el resultado de sumarle a la base un complemento anual fijo multiplicado por el número de años en la empresa.
- En los empleados por horas el sueldo se calcula multiplicando su sueldo por hora por el número de horas de este mes.

Diseñe (y escriba en Eiffel) las clases necesarias y sus relaciones para solucionar las nuevas necesidades detectadas.

Ejercicio 10

En un puerto se alquilan amarres para barcos de distinto tipo. Para cada ALQUILER se guarda el nombre y DNI del cliente, las fechas inicial y final de alquiler, la posición del amarre y el barco que lo ocupará. Un BARCO se caracteriza por su matrícula, su eslora en metros y año de fabricación.

Un alquiler se calcula multiplicando el número de días de ocupación (incluyendo los días inicial y final) por un módulo función de cada barco (obtenido simplemente multiplicando por 10 los metros de eslora) y por un valor fijo (1.200 pesetas en la actualidad).

Sin embargo ahora se pretende diferenciar la información de algunos tipos de barcos:

- número de mástiles para veleros
- potencia en CV para embarcaciones deportivas a motor
- potencia en CV y número de camarotes para yates de lujo.

El módulo de los barcos de un tipo especial se obtiene como el mólulo normal mas:

- el número de mástiles para veleros
- la potencia en CV para embarcaciones deportivas a motor
- la potencia en CV más el número de camarotes para yates de lujo.

Utilizando la herencia de forma apropiada, diseñe el diagrama de clases y sus relaciones, con detalle de atributos y métodos necesarios. Programe en Eiffel los métodos que permitan calcular el alquiler de cualquier tipo de barco.

Ejercicio 11

Dado $n \in \mathbb{N}$ podemos definir $\mathbb{Z}/n\mathbb{Z}$ como el conjunto $\{0, 1, \ldots, n-1\}$ dotado de la operación suma definida por la fórmula $a \widetilde{+} b = (a + b) \bmod n$, en el que cada número tiene un opuesto tal que la suma de los dos es cero y con una resta $a \widetilde{-} b$, definida como la suma de a con el opuesto de b.

Elaborar una clase Eiffel que modele los números de $\mathbb{Z}/187\mathbb{Z}$ proporcionando al menos suma, resta, opuesto y elemento «cero» como servicios.

Nota: Ejercicio de examen del curso 2005/2006

Ejercicio 12

Elaborar una clase que modele ángulos medidos en grados. La clase debe incorporar al menos la suma de ángulos, su resta, el opuesto de un ángulo y el ángulo «cero», quedando bien entendido que el ángulo 0^0 y el ángulo 360^0 son el mismo.

Nota: Ejercicio de examen del curso 2005/2006

Ejercicio 13

Un sistema informático dispone de una clase eiffel `TRABAJO` en cuya forma corta hemos visto una característica `usuario:STRING`.

Elaborar una clase `COLA_DE_TRABAJOS` encargada de almacenar trabajos. Entre sus características deben aparecer obligatoriamente:

- «mete» que almacena un trabajo en la cola
- «primero» que devuelve el trabajo que más tiempo lleva en la cola
- «saca» que elimina de la cola el trabajo que más tiempo lleva en ella
- «trabajos» que devuelve el número de trabajos almacenados en la cola
- «trabajos_de» que devuelve un array con los trabajos almacenados en la cola asociados a un determinado usuario

Nota: Ejercicio de examen del curso 2006/2007

Ejercicio 14

Elaborar una clase que modele ángulos medidos en grados. La clase debe incorporar al menos la suma de ángulos, su resta, el opuesto de un ángulo y el ángulo «cero», quedando bien entendido que el ángulo 0^0 y el ángulo 360^0 son el mismo.

Nota: Ejercicio de examen del curso 2006/2007

Ejercicio 15

Para cierta implementación que no viene al caso, el departamento de diseño ha detectado la necesidad de implementar un nuevo tipo de números a los que ha denominado «números curiosos». Un número curioso se caracteriza por tres coordenadas reales (a, b, c) que verifican $a^2 + b^2 + c^2 = 1$, salvo en el caso del número «cero» cuyas coordenadas son $(0, 0, 0)$. Sobre los números curiosos interesa realizar las siguientes operaciones:

- `suma(otro:CURIOSO):CURIOSO` definida mediante la fórmula:

$$(a_1, b_1, c_1) + (a_2, b_2, c_2) = (\frac{a_1 + a_2}{\sqrt{(a_1 + a_2)^2 + (b_1 + b_2)^2 + (c_1 + c_2)^2}}, \frac{b_1 + b_2}{\sqrt{(a_1 + a_2)^2 + (b_1 + b_2)^2 + (c_1 + c_2)^2}}, \frac{c_1 + c_2}{\sqrt{(a_1 + a_2)^2 + (b_1 + b_2)^2 + (c_1 + c_2)^2}})$$

  cuando $(a_1 + a_2)^2 + (b_1 + b_2)^2 + (c_1 + c_2)^2 \neq 0$ y $(0, 0, 0)$ en caso contrario.

- `resta(otro:CURIOSO):CURIOSO` definida mediante la siguiente fórmula:

$$(a_1, b_1, c_1) - (a_2, b_2, c_2) = (\frac{a_1 - a_2}{\sqrt{(a_1 - a_2)^2 + (b_1 - b_2)^2 + (c_1 - c_2)^2}}, \frac{b_1 - b_2}{\sqrt{(a_1 - a_2)^2 + (b_1 - b_2)^2 + (c_1 - c_2)^2}}, \frac{c_1 - c_2}{\sqrt{(a_1 - a_2)^2 + (b_1 - b_2)^2 + (c_1 - c_2)^2}})$$

  cuando $(a_1 - a_2)^2 + (b_1 - b_2)^2 + (c_1 - c_2)^2 \neq 0$ y $(0, 0, 0)$ en caso contrario.

- `opuesto:CURIOSO` Donde el opuesto del número (a, b, c) es el número $(-a, -b, -c)$.
- `giro:CURIOSO`. El resultado de girar un número curioso (a, b, c) es el número (b, c, a).

Elaborar una clase Eiffel que implemente los números curiosos.
Nota: Ejercicio de examen del curso 2007/2008

Ejercicio 16

Una cola se caracteriza porque el primer elemento en entrar será el primero en salir. Una implementación típica de la cola tiene las siguientes características:

- `primero:G` Devuelve el primer elemento que saldrá de la cola, `Void` si la cola está vacía
- `mete(e:G)` Introduce un elemento en la cola
- `saca` Elimina de la cola el primer elemento, si es que en la cola hay elementos.

Ahora deseamos implementar una «cola con supervisión» en la que la mayoría de los elementos entran y salen de la forma tradicional, pero cada n elementos insertados, el siguiente se almacena de una forma especial para ser inspeccionado. De éste modo además de `primero` disponemos de `primero_supervisado` que es el primer elemento supervisado en la cola, y `saca_supervisado` que elimina el primer elemento supervisado que aparece en la cola.

Implementar esta clase en Eiffel considerando que la cola puede tener varios elementos supervisados en espera, que estos elementos no deben ser extraídos por `saca` ni mostrados por `primero`, y que el valor n se fija al crear la cola y no puede ser cambiado

Nota: Ejercicio de examen del curso 2007/2008

Ejercicio 17

Elaborar una clase Eiffel que permita almacenar elementos en un «orden mezclado». En este orden mezclado el primer elemento insertado se coloca el primero, el siguiente el último, y así alternativamente.

Entre los servicios que la clase debe proporcionar deben aparecer al menos

- Inserción, que añade un elemento a la lista en el orden adecuado
- Elemento en la posición iésima, que devuelve el elemento que se encuentra en esta posición, si es que esta posición está ocupada
- Elementos que indica el número de posiciones ocupadas en este momento
- Poda, que elimina el primer elemento de la lista

Nota: Ejercicio de examen del curso 2008/2009

Ejercicio 18

Una cola se caracteriza porque el primer elemento en entrar será el primero en salir. Una implementación típica de la cola tiene las siguientes características:

- `primero:G` Devuelve el primer elemento que saldrá de la cola, `Void` si la cola está vacía
- `mete(e:G)` Introduce un elemento en la cola
- `saca` Elimina de la cola el primer elemento, si es que en la cola hay elementos.

Ahora deseamos un tipo especial de cola que permita, mediante un método `mezcla` reordenar sus elementos tal y como se hace en la baraja con la mezcla americana, esto es, se divide en dos mitades, y se recolocan de tal modo que la última del primer montón pasa a ser la última, precedida por la última del segundo montón, precedida por la penúltima del primer montón, precedida por la penúltima del segundo montón,...

Elaborar una clase Eiffel que modele el tipo de cola indicado.

Nota: Ejercicio de examen del curso 2008/2009