

Examen de Fundamentos de Computadores y Lenguajes

Examen Final. Septiembre 2001

Cuestiones (5 cuestiones, 5 puntos en total)

- 1) Se dispone de la siguiente clase en Java:

```
public class Ecuacion2G {
    float a,b,c;
}
```

Se pide añadir a la clase una operación que retorne el valor de $ax^2 + bx + c$, dado el valor de x que se pasa como parámetro.

- 2) Se pide añadir a la clase de la cuestión 1ª un constructor al que se le pasen como parámetros los valores a , b y c . Si los tres valores son cero, el constructor lanzará la excepción `IllegalArgumentException`. Si no, almacenará los tres valores mencionados.
- 3) Reescribir el siguiente fragmento de código Java utilizando una instrucción `switch` en lugar de instrucciones condicionales de tipo `if`.

```
if (i==1) {
    accion="hacer";
} else if (i==2) {
    accion="cancelar";
} else if (i==3) {
    accion="reintentar";
} else {
    accion="error";
}
```

- 4) Los métodos en Java pueden retornar un dato. ¿Cómo se puede hacer para que un método retorne dos o más datos?
- 5) Dados los valores a , número real, y k , número natural, es fácil diseñar un algoritmo, que mediante un bucle que usa no más de k iteraciones calcula el valor de la potencia a^k . Sin embargo, este método no es el más eficaz posible. Para obtener una solución más rápida del orden de magnitud $O(\log_2 k)$, basta tener en cuenta la siguiente idea: Si k es impar entonces podemos escribir $a^k = a^{k-1} \times a$, ahora bien, si k es par entonces $a^k = (a^2)^{k/2}$. Suponiendo que se desee calcular el valor de a^k en la variable p , la observación anterior lleva al siguiente algoritmo: Se inicializa la variable p en el valor 1.0 (correspondiente al valor de a^0) y se considera un bucle cuya condición de terminación es $k=0$. El cuerpo del bucle se construye mediante una alternativa en la que la primera rama se ejecuta si k es impar, actualizando el valor de la variable p multiplicándola por a y decrementando en una unidad el valor de k ; y la segunda rama se ejecuta si k es par y actualiza los valores de a y k multiplicando a por si mismo y dividiendo k por dos.

Se pide especificar y diseñar un algoritmo que traslade a pseudocódigo la idea descrita anteriormente.

Examen de Fundamentos de Computadores y Lenguajes

Examen final. Septiembre 2001

Problema (5 puntos)

Se desea implementar parte del sistema electrónico portable para ayuda a un jugador de golf. El sistema se basa en un receptor de GPS capaz de detectar la posición del sistema en el globo terráqueo, y un receptor de radio que capta una señal procedente del club de golf que le indica la velocidad y dirección del viento. Basándose en estos datos y en las coordenadas de las banderas y puntos de salida (“tee”) de cada hoyo, el sistema deberá calcular la distancia a la bandera, las correcciones debidas al viento, y el palo de golf recomendado para el siguiente tiro.

El software del subsistema de GPS está ya implementado en una clase cuya especificación es:

```
public class Gps {  
  
    // retorna la distancia entre p1 y p2 en metros  
    public static double distancia (double p1Lat, double p1Long,  
                                   double p2Lat, double p2Long) {...}  
  
    // retorna el rumbo a seguir para ir de p1 a p2, en metros  
    public static double rumbo (double p1Lat, double p1Long,  
                                double p2Lat, double p2Long) {...}  
  
    // retorna la latitud GPS del punto actual, en grados  
    public static double latActual() {...}  
  
    // retorna la longitud GPS del punto actual, en grados  
    public static double longActual() {...}  
  
    // retorna la velocidad del viento actual, en m/s  
    public static double velViento() {...}  
  
    // retorna la dirección hacia donde va el viento, en grados  
    public static double dirViento() {...}  
}
```

La descripción de las operaciones de esta clase aparece en los respectivos comentarios. En las operaciones distancia y rumbo, p1Lat y p1Long son la latitud y longitud del punto p1, en grados; p2Lat y p2Long son lo mismo para el punto p2. El rumbo y dirección del viento se dan en grados, donde 0° es el Norte, y los grados se incrementan en el sentido de las agujas del reloj.

Se pide implementar una clase con operaciones para realizar los principales cálculos. La clase obedecerá a la siguiente especificación:

```
public class Golf {  
  
    private static double salidaLat, salidaLong,  
                        banderaLat, banderaLong;  
  
    public static void datosSalida (double Lat, double Long) {...}
```

```

public static void datosBandera (double Lat, double Long) {...}

public static double distanciaReal() {...}

public static double distanciaCorregida() {...}

public static double anguloCorregido() {...}

public static String paloRecomendado() {...}
}

```

Los campos privados `salidaLat` y `salidaLong` sirven para almacenar la latitud y la longitud del punto de salida del hoyo actual. Los campos privados `banderaLat` y `banderaLong` sirven para almacenar la latitud y la longitud de la bandera del hoyo actual. Las operaciones deben realizar lo siguiente:

- a) `datosSalida`. Guarda en `salidaLat` y `salidaLong` los datos que se le pasan como parámetros.
- b) `datosBandera`. Guarda en `banderaLat` y `banderaLong` los datos que se le pasan como parámetros.
- c) `distanciaReal`. Retorna la distancia del punto actual (que se leerá de la clase `Gps`) a la bandera del hoyo actual. Para el cálculo de la distancia se usa la función `distancia` de la clase `Gps`.
- d) `distanciaCorregida`. Retorna la distancia que hay hasta la bandera corregida para tener en cuenta el efecto del viento. La fórmula es:

$$Dist_{corregida} = Dist_{real} (1 - Vel_{viento} \cos \alpha / C_v)$$

donde α es la diferencia entre la dirección del viento y el rumbo a seguir desde el punto actual (que se leerá de la clase `Gps`) a la bandera del hoyo actual, y $C_v=97.3$ es el coeficiente de influencia del viento sobre la bola de golf.

- e) `anguloCorregido`. Retorna la corrección de ángulo que el jugador debe observar en el tiro debida al viento lateral. La fórmula es:

$$Angulo_{corregido} = \text{atan2}(Vel_{viento} \sin(\alpha) Dist_{real}, Dist_{real})$$

- f) `paloRecomendado`. Retorna un string con el nombre del palo recomendado. Éste se calcula en función de la distancia corregida, de acuerdo con la siguiente tabla:

Distancia (m)	Desde el "tee" (salida)	Desde la calle
>250	W1	
200-250	W3	

Distancia (m)	Desde el “tee” (salida)	Desde la calle
150-200	W5	
>220		DI
200-220		I3
180-200		I4
160-180		I5
140-160		I6
120-140		I7
100-120		I8
80-100		I9

Se considera que estamos en la salida si estamos a menos de 30 metros del punto de salida del hoyo actual. Si no, se entiende que estamos "en la calle". Para los casos que no estén en la tabla, el palo recomendado es Ninguno.

Nota: Se valorará cada apartado en función de su complejidad.