

CC1001-7 Auxiliar Extra – Control 2

Daniel Calderón – Sebastián Fehlandt

29 de Abril de 2009

P1) CC1001-Racer (Clases y objetos, Herencia, Strings, Archivos, Ventanas, Dibujos)

(a) Escriba la clase abstracta Movil con las siguientes características:

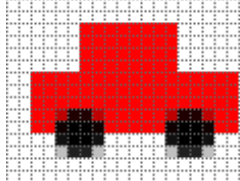
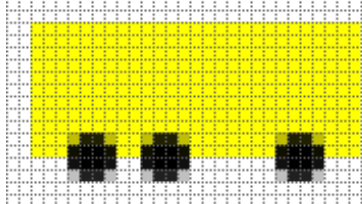
<code>protected int x,v;</code>	posición y velocidad.
<code>protected int n;</code>	Lugar de llegada (se utiliza mas adelante). $n=-1$ si no ha llegado
<code>protected String tipo,color;</code>	
<code>public Movil(int v,String tipo,String color)</code>	crea un auto en la posición $x=0$ y $n=-1$.
<code>public void avanzar(int t)</code>	mueve al auto durante un tiempo t ($v=ctte$).
<code>public Boolean pintar(String color)</code>	true, si el color es valido. false, en caso contrario.*
<code>abstract public acelerar()</code>	no implementar. Modifica la velocidad.
<code>public void detener(int n)</code>	detiene al movil ($v=0$) en la posición $x=350$. Asigna además el lugar de llegada n .
<code>public int getN(),getX(),getV()</code>	Retornan n , x y v respectivamente
<code>public String toString()</code>	Retorna un String con el formato **
<code>abstract public void dibujar(Graphics g,int x,int y)</code>	no implementar. Dibuja el Movil centrado en la posición (x,y) .

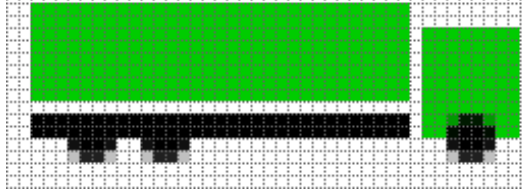
*Los colores válidos son: "rojo", "verde", "azul" y "blanco".

**formato que utiliza toString():

- Caracteres 0 al 9: tipo de móvil (auto, camion o micro).
- Caracteres 10 al 19: color del móvil (solo colores validos de la parte a).
- Caracter 20: lugar de llegada(1, 2 o 3).

(b) Implemente las clases Auto, Micro y Camion que extiendan a Movil y puedan ser acelerados (variación en la velocidad) y dibujados como se muestra en la siguiente tabla:

Clase	Velocidad	Acelerar	Dibujo (respetar tamaño en pixeles)
Auto	10	Aleatorio entre $v-2$ y $v+5$	
Micro	12	Aleatorio entre $v-1$ y $v+4$	

Camión	8	Aleatorio entre v-3 y v+4	
--------	---	---------------------------	--

El móvil debe pintarse de su color correspondiente.

(c) Implemente la clase Carrera donde puedan correr 3 móviles.

<code>private Movil a,b,c;</code>	Móviles en competencia
<code>private int n;</code>	Móviles que han llegado a la meta
<code>public Carrera(Movil a,Movil b, Movil c);</code>	Constructor.
<code>public void next(int t)</code>	Acelera y avanza a todos los móviles durante t. si un auto llega a la meta se guarda su lugar de llegada y se detiene.
<code>public void dibujar(Graphics g)</code>	Dibuja la carrera, esto es, pinta el fondo plomo, dibuja las líneas de inicio y término, y dibuja los autos en sus respectivas posiciones. (Ver cuadro siguiente).*
<code>public void guardar(String file)</code>	Guarda los móviles en el archivo de nombre file utilizando toString() de cada móvil.
<code>public boolean termino()</code>	true si es que todos llegaron a la meta.
<code>public Movil getWinner()</code>	Retorna el móvil ganador, null si aun no ha terminado.

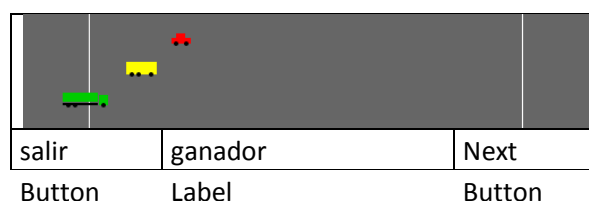
*dibujar(...) debe pintar algo similar a.



La línea de partida está en el pixel 100, la de llegada en el 400. Las dimensiones totales son 500x100 pixeles.

(d) Implemente la función de encabezado `public static Carrera startRacer()` que retorne una Carrera donde los 3 moviles sean leídos del archivo "cars.1001", donde solo hay 3 moviles guardados en el formato especificado por toString() de Movil.

(e) Implemente un programa que muestre la siguiente ventana:



Donde al presionar “next” los móviles avancen el equivalente a 1 unidad de tiempo y vuelvan a dibujarse. Si un auto llega a la meta debe quedarse en dicha posición y registrar su lugar de llegada. Cuando todos los móviles finalicen la carrera, debe guardarse en el archivo “resultados.1001” y en el label ganador se escribe el tipo y color del primer móvil en llegar a la meta.

P2) Codificando y decodificando en Morse (String, Archivos y Ventanas) [Propuesto]

1. Escriba los métodos **String morse(String alfabetico)** que retorne el texto en clave morse, y **String alfabetico (String morse)** que retorne el texto ingresado en clave morse como texto “normal”. Para esto use el archivo morse.code en el cual en cada línea se encuentra un carácter seguido de un espacio en blanco y luego su representación en morse. Para separar caracteres morse use *, y para separar palabras en morse utilice **
2. Implemente la interfaz de la figura

Label	Texto alfabético	Texto en morse	Label
TextField	Ingrese texto aquí	Ingrese texto aquí	TextField
Button	Pasar a morse	Pasar a alfabético	Button

que permita al usuario ingresar un texto alfabético o en morse y que al presionar el botón respectivo o enter (estando dentro del TextField respectivo), escriba en el otro TextField el mensaje pero en la otra codificación. En cada línea del archivo “morse.code” se encuentra un carácter seguido de un espacio en blanco y luego su representación en morse. **Hint:** meta los métodos creados anteriormente dentro de la clase que contiene la interfaz gráfica.