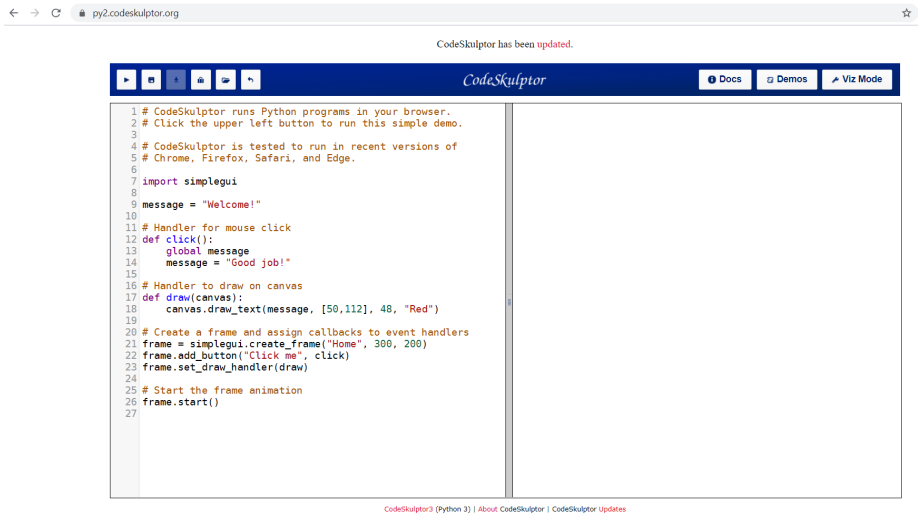
Simulación computacional:
Tarea 1 - Movimiento en 1 dimensión

.

En el archivo movimiento.txt (es un archivo de texto que pueden abrir con cualquier editor de texto básico, por ejemplo el Notepad de Windows o Vim o gedit en Linux), encontrarán un código en lenguaje Python 2 para realizar una mini simulación de un movimiento, que incorpora un paquete para generar una interfaz gráfica (GUI: graphical user interface) para visualizar dicha simulación.

Para ejecutar dicho código, en su navegador, vayan a codeskulptor.org. Allí les aparecerá la página que ven a continuación.



En la parte izquierda va el código y en la parte de la derecha aparecen los outputs que genera y mensajes de error si hubiese algún problema con dicho código.

Borren lo que aparece a la izquierda y copien y peguen allí el contenido del archivo movimiento.txt. Con el botón de arriba a la izquierda (con el mismo símbolo que el botón de "Play" de un reproductor de música), ejecutan el código.

Dentro del código he introducido varias líneas con aclaraciones que están precedidas por el símbolo #. Estas líneas son comentarios, es decir que no son interpretadas como instrucciones del programa. Si uno desea que el programa NO ejecute una cierta línea de código,basta con ponerle un # adelante y listo.

En la primera parte del código verán una parte que dice "Inicialización de las variables globales". Allí están declaradas las variables que pueden ser utilizadas por cualquier sección de nuestro programa. Los lenguajes de programación tienen definidos algunos tipos de variables (int, float, boolean, character, etc., es decir, entero, número con cierta cantidad de decimales, variable lógica (verdadera o falsa), caracter, etc.). En muchos de ellos, en esta primera parte uno debe declarar las variables indicando el tipo de variable del que se trata y su nombre. En Python, al poner

el nombre de la variable igualado a un valor, interpreta automáticamente de qué tipo de variable se trata. Por ejemplo, "radio = 10" interpreta que la variable radio es un número entero. "vel0x = 100.", como tiene el separador decimal, interpreta que se trata de un número con desarrollo decimal.

Cuando uno le asigna un valor a una variable, siempre el nombre de la variable va a la izquierda del = y lo que se va a almacenar en esa variable a la derecha.

En la línea 104 van a ver:

```
103 # Crea un timer para la simulación
104 timer = simplegui.create_timer(1000*dt,tick)
```

este timer hace lo siguiente: cada 1000*dt milisegundos ejecuta el bloque de código bajo el nombre de **tick**, que comienza en la línea 53. Cada vez que lo ejecuta, para nuestra partícula lo que estamos haciendo es avanzarla en una cantidad de tiempo dt.

En muchos lenguajes de programación, los bloques de código van encerrados entre llaves, por ejemplo. En python, están determinados por la llamada *"indentación"*, es decir la sangría. Todas las líneas de un bloque inician 4 espacios a la derecha del comienzo de la línea. Si uno tiene un sub-bloque, otros cuatro espacios a la derecha. Es decir que la cantidad de espacios en blanco al inicio de una línea es crucial para cómo va a ser interpretado nuestro programa.

Actividad:

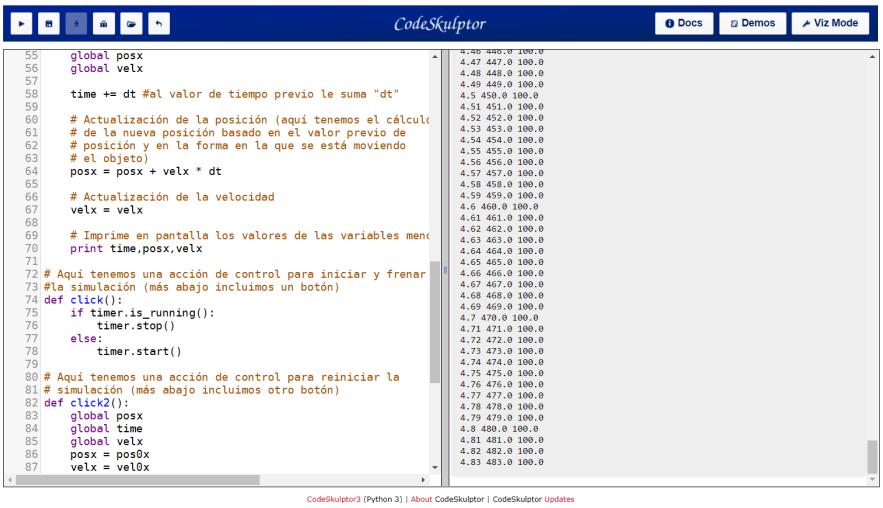
1. Familiarizarse con el código.

2. La posición está dada, al menos por ahora, en pixels y la velocidad en pixels por segundo. Identifiquen el sistema de referencia para este movimiento rectilíneo.

3. Modifiquen el valor de dt: en vez de 0.01, fíjense qué pasa si ponen "dt = 0.1" y luego si ponen "dt = 1" (en la línea 23).

4. Modifiquen los valores de posición inicial (pos0x) y velocidad inicial (vel0x) y vean cómo cambia el movimiento simulado. Van a ver que la línea 70 está comentada. Si borran el # y ejecutan el programa, en la parte derecha les imprimirá una tabla con los valores de las variables que allí figuran (por si les interesa mirar un poco cómo cambian o si los quieren copiar y pegar en otro lado y hacer un gráfico).

```
69     # Imprime en pantalla los valores de las variables men
70     # print time,posx,velx
71
```



5. Modifiquen el código para simular, en lugar de un MRU un MRUV. Piensen qué variable/s nueva/s sería conveniente definir; cómo debería uno actualizar los valores de velocidad y de posición.

6. Una vez implementado el punto anterior, hagan pruebas con distintos valores de las condiciones iniciales y distintos valores de aceleración.