TANQUES DE PENSAMIENTO
PROYECTO FINAL FUNDAMENTOS DE PROGRAMACIÓN

PRIMERA NOCION DE LOS CONCEPTOS
DE UNIVERSO, MUNDO, BIG BANG

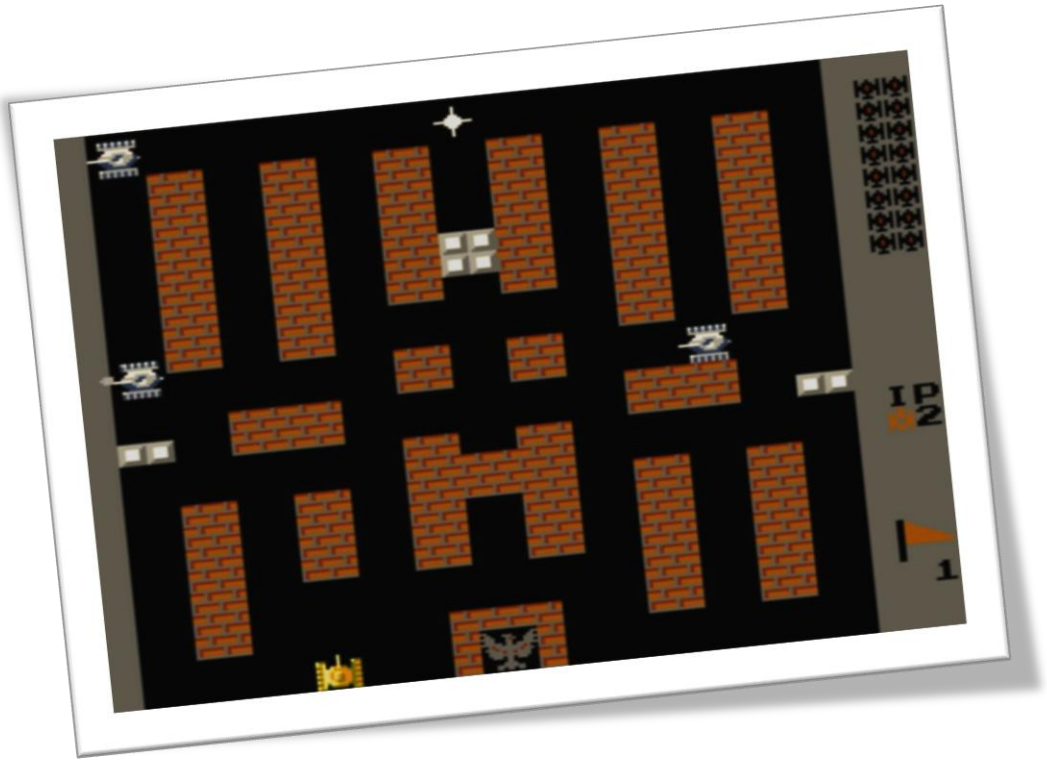
PROFESOR: VÍCTOR BUCHELI
(VICTOR.BUCHELI@CORREOUNIVALLE.EDU.CO)
MONITOR: AURELIO VIVAS
(AURELIO.VIVAS@CORREOUNIVALLE.EDU.CO)

ESCUELA DE INGENIERÍA DE SISTEMAS Y COMPUTACIÓN
FUNDAMENTOS DE PROGRAMACIÓN
UNIVERSIDAD DEL VALLE

CONTENIDO

ADVERTENCIA

La mayor parte de la información de este texto siguen los libros relacionados en la bibliografía. El juego descrito en este documento está basado en el juego de consola PolyStation Battle City, y muestra una versión inicial de sucesivos cambios que buscan dejar la metáfora bélica y llevar el juego hacia el concepto de tanques de pensamiento.



INTRODUCCIÓN

Battle City es un videojuego de tanques producido y publicado por Namco como una adaptación del clásico árcade Tank Battalion. El juego consiste en controlar un tanque sobre un escenario plagado de tanques enemigos. Su misión consiste en evitar que destruyan su base militar. El nivel es completado cuando se destruyen todos los tanques enemigos. El juego terminara si el enemigo destruye la base o el jugador gasta todas sus vidas. [1] Para esta versión del proyecto la dinámica del juego es igual y se cambia el entorno gráfico buscando desarrollar la idea de tanques de pensamiento que solucionan un problema de acuerdo a preguntas, respuestas e ideas.

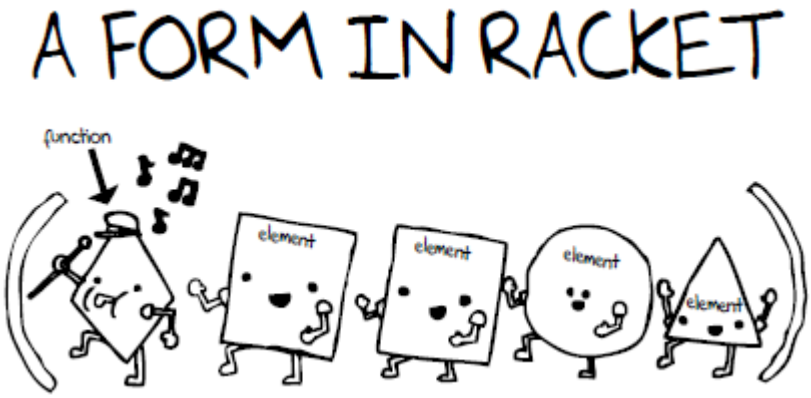
Los Think tank es una expresión inglesa literalmente depósito de pensamiento y que se ha intentado adaptar al idioma español con distintas expresiones, lo podríamos entender como un laboratorio de ideas o aún almacén de ideas [2]. La idea final del juego seria tener un banco de preguntas y respuestas donde para cada disparo, la efectividad de dicho disparo se da sólo si conoces la respuesta, en principio podría ser preguntas de falso y verdadero.

El lenguaje de programación Racket en su versión actual 6.1.1, La librería universe.rkt que provee funciones para la creación de programas gráficos interactivos. En muchos casos nos referimos a este tipo de programas como mundos. [*The universe.rkt teachpack implements and provides the functionality for creating interactive, graphical programs that consist of plain mathematical functions. We refer to such programs as world programs. In addition, world programs can also become a part of a universe, a collection of worlds that can exchange messages.*] [3].

El objetivo de este proyecto es recrear un versión similar del juego Battle City usando del entorno de desarrollo Drracket y el lenguaje de programación Racket versión 6.1.1, donde se evidencia como se modela funcionalmente cada variante del juego y se evaluara como un juego pasa de un estado a otro y que factores intervienen en particular en la lógica del mismo. En el transcurso de la creación de este juego se aplicaran conceptos vistos en el curso como: abstracción, recursión, manejos de tipos de datos, etc.

ACERCA DE RACKET

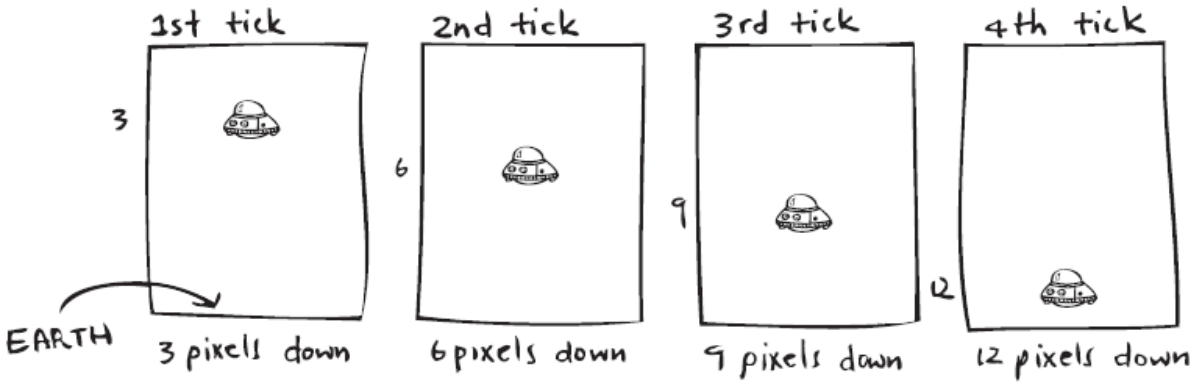
Racket es un amplio lenguaje de programación proveniente de la familia de Lisp y Scheme. Es multiparadigma así como de propósito general. Uno de sus principales objetivos tras su diseñ o es posibilitar la creación de nuevos lenguajes o dialectos, es comúnmente usado para enseñ ar materias como lenguajes de programación y muy aplicado afines investigativos.



UNIVERSO, MUNDO, BIG BANG

CONCEPTO MUNDO

Para entender un poco esto de los mundos y el universo, hablaremos un poco sobre lo que sucede cuando un UFO aterriza en un planeta desconocido. Analizaremos paso a paso cada uno de los factores que intervienen en dicho proceso.



Como podemos observar en la imagen anterior, nuestro pequeñ o amigo se encuentra solo en el planeta desconocido. Podemos afirmar que está solo en el mundo y mucho más interesante aun, afirmar que él es el mundo. ¿ Por qué llegamos a esta conclusión? En Racket y específicamente en la creación de juegos interactivos usando la librería *universe,* un mundo es una especie de contenedor que guardara todo lo que sea susceptible a cambios, por dicha razón se puede afirmar que el UFO es el mundo, debido a que este cambiara su posición paulatinamente hasta llegar al suelo. Si queremos ser más específicos también es correcto afirmar que solo la posición del UFO es el mundo o está contenida en el mundo que al final termina siendo lo mismo. Si observamos, la imagen del UFO nunca cambia mientras el desciende, siendo su posición lo que en realidad cambia.

CONCEPTO UNIVERSO

Continuando con la descripción de la travesía de nuestro pequeñ o amigo hasta tocar tierra, hemos encontrado que el mundo es un contenedor de cosas que pueden cambiar. No necesariamente un mundo debe ser un objeto, podemos encontrar en la práctica situaciones que nos lleven a un mundo que contenga muchos objetos cambiantes como lo es el caso del juego de tanques.

Si bien un mundo es un contenedor de objetos cambiantes, un universo es un contenedor de mundos. En lo que llevo de experiencia haciendo juegos en racket, puedo deducir que a eso se le atribuye el nombre a la librería *universe.*

Para entender un poco mejor la relación entre mundo y universo, seguiremos enfocados en la imagen del aterrizaje de nuestro amigo. Diremos que la primera imagen (de izquierda a derecha) es la representación gráfica del mundo que llamaremos M1, la segunda imagen la representación gráfica del mundo M2 y así sucesivamente hasta llegar a M4.
Siguiendo la afirmación que hicimos anteriormente acerca de que el mundo es la posición del UFO, para ser más exactos la posición vertical, es correcto decir que M1 = 3, M2 = 6, M3 = 9, M4 = 12.

Esto quiere decir que debido a algún suceso o evento, en este caso que el UFO sea movido por un piloto desconocido, la posición del UFO pasa de ser el mundo M1 a ser el mundo M2, solo con sumar 3 unidades a M1.

El conjunto de mundos creados debido a que es afectado o modificado un mundo se le conoce como universo. Para nuestro ejemplo el universo estaría compuesto por M1, M2, M3 y M4. Donde M1 es el estado inicial del mundo y M4 el estado final.

Algo que es importante mencionar, es que cada mundo creado a causa de un evento o suceso que obliga al mundo a cambiar de estado, está relacionado con el mundo anterior, esto es correcto debido a que si no tengo la información de la posición anterior del UFO no podría saber cuál es la siguiente, siendo que la siguiente posición siempre será la suma de la posición anterior más 3 unidades hacia abajo.

Realmente en la implementación de un juego interactivo usando mundos, no es muy claro el concepto de universo, bastaría con tener presente que un mundo puede cambiar su estado o valor y pasar a ser uno nuevo.

CONCEPTO BIG BANG

Realmente no es necesario adentrarnos en el concepto del Big Bang, lo único que puedo decir es que se relaciona estrechamente con la frase "Hágase el universo" que se me acaba de ocurrir y claro no sobra decir que está relacionado con la creación del universo y por consiguiente los mundos.

USO DEL BIG BANG: SINTAXIS Y SEMÁNTICA

En esta parte mostrare las funciones más relevantes para la creación de juegos interactivos con la librería *universe,* especialmente la creación del juego de tanques. Antes que nada, debemos tener en cuenta dos aspectos importantes: el primero es que para un juego tenemos una representación en forma de datos validos en Racket y una presentación visual de los datos que también son asistidos por estructuras y funciones de Racket.

BIG-BANG

Es la función encargada de iniciar el juego y asociarlo a funciones capaces de detectar los eventos que el big-bang este en capacidad de detectar y manejar. Un evento es una acción que se puede generar ya sea por la interacción del usuario con el computador, una acción del mismo computador o en algunos casos cuando ocurre un evento es posible dar pie a la generación de otros eventos. Un manejador o función manejadora es la función que decidirá qué hacer cuando se genera un evento, por lo general cada función que se declara para detectar un evento debe ser especificada con la función que va a manejar ese tipo de evento.

`state-expr`: representa el estado inicial del mundo.
`clause`: representa uno o varios detectores de eventos que se pueden asociar al juego.

```
(big-bang state-expr clause ……)
```

```
(big-bang state-expr (on-key key-expr)
                     (on-tick tick-expr rate-expr)
                     (to-draw draw-expr)
                     (stop-when stop-expr))
```

La función big-ban posee muchas variantes que se pueden definir dependiendo del tipo de situación que se desee representar con mundos.

ON-TICK

Esta función llama a la función manejadora tick-expr cada tick del reloj del sistema, por ejemplo si rate-expr = 1/28, entonces on-tick llamara a la función tick-expr 28 veces cada segundo.

Esta función solicita que su función manejadora tick-expr sea una función que reciba como parámetro un mundo y ratona un valor valido en Racket que sea el resultado de manejar dicho evento.

```
(on-tick tick-expr rate-expr)
```

TO-DRAW

Es la función encargada de llamar a la función manejadora render-expr cada que un evento se genera, con el objetivo de repintar la representación gráfica del último cambio que sufrió el mundo.

Esta función solicita que su función manejadora render-expr reciba como parámetro un mundo o estado del mundo y retorne una scene (imagen o representación gráfica del mundo en su estado actual)

```
(to-draw render-expr)
```

ON-KEY

Es la función encargada de llamar a la función manejadora key-expr cada que se presiona una tecla del teclado.

```
(on-key key-expr)
```

Esta función solicita que su función manejadora key-expr reciba como parámetro un mundo o estado del mundo, la representación de la tecla presionada en Racket y retorne un valor valido en Racket que sea el resultado de manejar dicho evento.

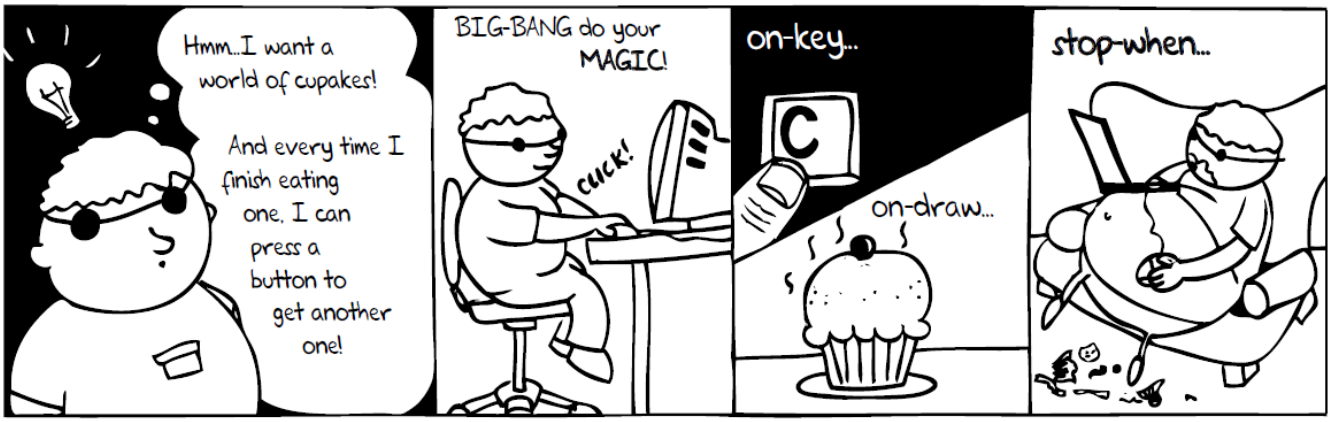
```
(define (change w a-key)
  (cond
    [(key=? a-key "left")  (world-go w -DELTA)]
    [(key=? a-key "right") (world-go w +DELTA)]
    [(= (string-length a-key) 1) w] ; order-free checking
    [(key=? a-key "up")    (world-go w -DELTA)]
    [(key=? a-key "down")  (world-go w +DELTA)]
    [else w]))
```

STOP-WHEN

Esta función se encarga de llamar a la función manejadora last-world? Cada que se genera un evento, verificando si el mundo tiene algún estado que le indique al programa completo terminar su ejecución.

Esta función solicita que su función manejadora reciba como parámetro un mundo o estado del mundo y retorne un booleano. El juego interactivo únicamente se congela si la función retorna true, indicando que ha terminado.

```
(stop-when last-world?)
```

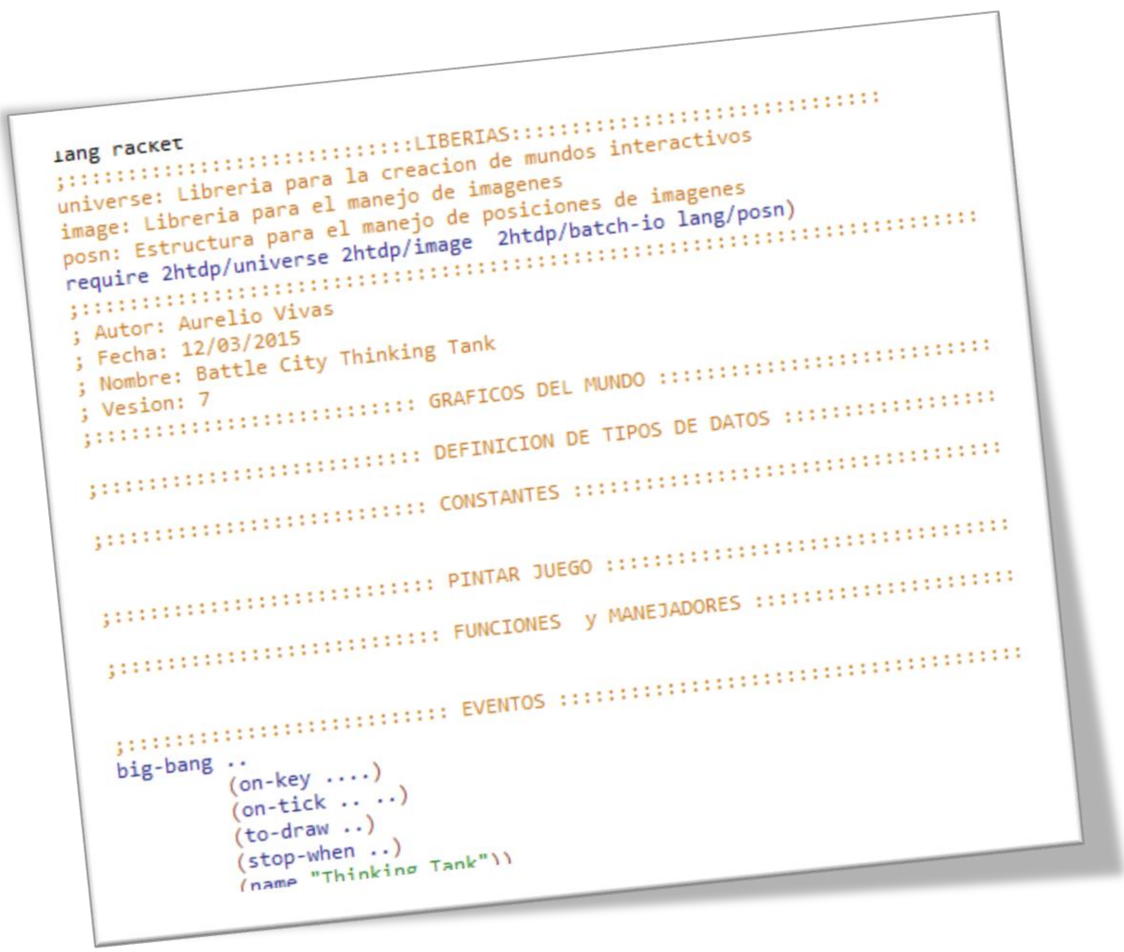


Detalles del proyecto

Especificaciones

El código que usted o su grupo de trabajo desarrollaran debe estar estructurado de la siguiente forma.

- ✓ Librerías: En esta parte se declaran e importan los paquetes o librerías necesarios.
- ✓ Encabezado: En esta parte debe estar el nombre de los integrantes del proyecto con sus respectivos códigos, última fecha de modificación, nombre y versión del proyecto.
- ✓ Gráficos del mundo: En esta parte se define todo lo relacionado con importación de imagines.
- ✓ Definición de tipos de datos: En esta parte se definen todos los tipos de datos que soportaran el funcionamiento del juego.
- ✓ Constantes: En esta parte se definen constantes necesarias para lograr variantes en el juego sin necesidad de modificar todo el código.
- ✓ Pintar juego: En esta parte se logra obtener la representación gráfica del estado del mundo y plasmarla en la pantalla del computador.
- ✓ Funciones y manejadores: En esta parte se desarrolla toda la lógica desarrollada en el juego
- ✓ Eventos: En esta parte se define el Big-Bang y las funciones o eventos necesarios para hacer funcionar la lógica del juego.



```
lang racket
;::::::::::::::::::::::::::::LIBERIAS::::::::::::::::::::::::::::::
universe: Libreria para la creacion de mundos interactivos
image: Libreria para el manejo de imagenes
posn: Estructura para el manejo de posiciones de imagenes
require 2htdp/universe 2htdp/image  2htdp/batch-io lang/posn)
;::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
; Autor: Aurelio Vivas
; Fecha: 12/03/2015
; Nombre: Battle City Thinking Tank
; Vesion: 7
;:::::::::::::::::::::::::::: GRAFICOS DEL MUNDO ::::::::::::::::::::::::::

;:::::::::::::::::::::::::::: DEFINICION DE TIPOS DE DATOS ::::::::::::::::::

;:::::::::::::::::::::::::::: CONSTANTES ::::::::::::::::::::::::::::::::::

;:::::::::::::::::::::::::::: PINTAR JUEGO ::::::::::::::::::::::::::::::::

;:::::::::::::::::::::::::::: FUNCIONES  y MANEJADORES ::::::::::::::::::::

;:::::::::::::::::::::::::::: EVENTOS ::::::::::::::::::::::::::::::::::::
big-bang ..
         (on-key ....)
         (on-tick .. ..)
         (to-draw ..)
         (stop-when ..)
         (name "Thinking Tank"))
```

PRIMERA ENTREGA

Para desarrollar los siguientes puntos puede hacer uso del código guía que le brindara el profesor por el campus virtual o podrá crear desde cero su propia versión del juego teniendo en cuenta las especificaciones dadas anteriormente.

Para la primera entrega del proyecto su grupo de trabajo debe entregar las siguientes funcionalidades:

- Lograr mover un tanque (jugador) en un escenario.
- Evitar que el tanque (jugador) se salga de los límites del juego.
- Lograr que el tanque (jugador) dispare balas.
- Eliminar las balas cuando lleguen a los límites del juego.
- Colocar ladrillos en el mundo.
- Evitar que el tanque pase por encima de los ladrillos.
- Lograr que el tanque (jugador) destruya ladrillos.
- Eliminar ladrillos que colisionen con balas.
- Eliminar balas que colisionen con ladrillos.
- Colocar tanques (enemigos) en el mundo máximo 6.
- Mover tanques (enemigos) en cualquier dirección en el mundo (Inteligencia Artificial débil).
- Evitar que los tanques enemigos pasen por encima de los ladrillos.
- Evitar que los tanques enemigos salgan de los límites del mundo.
- Lograr que los tanques (enemigos) disparen.

SEGUNDA ENTREGA

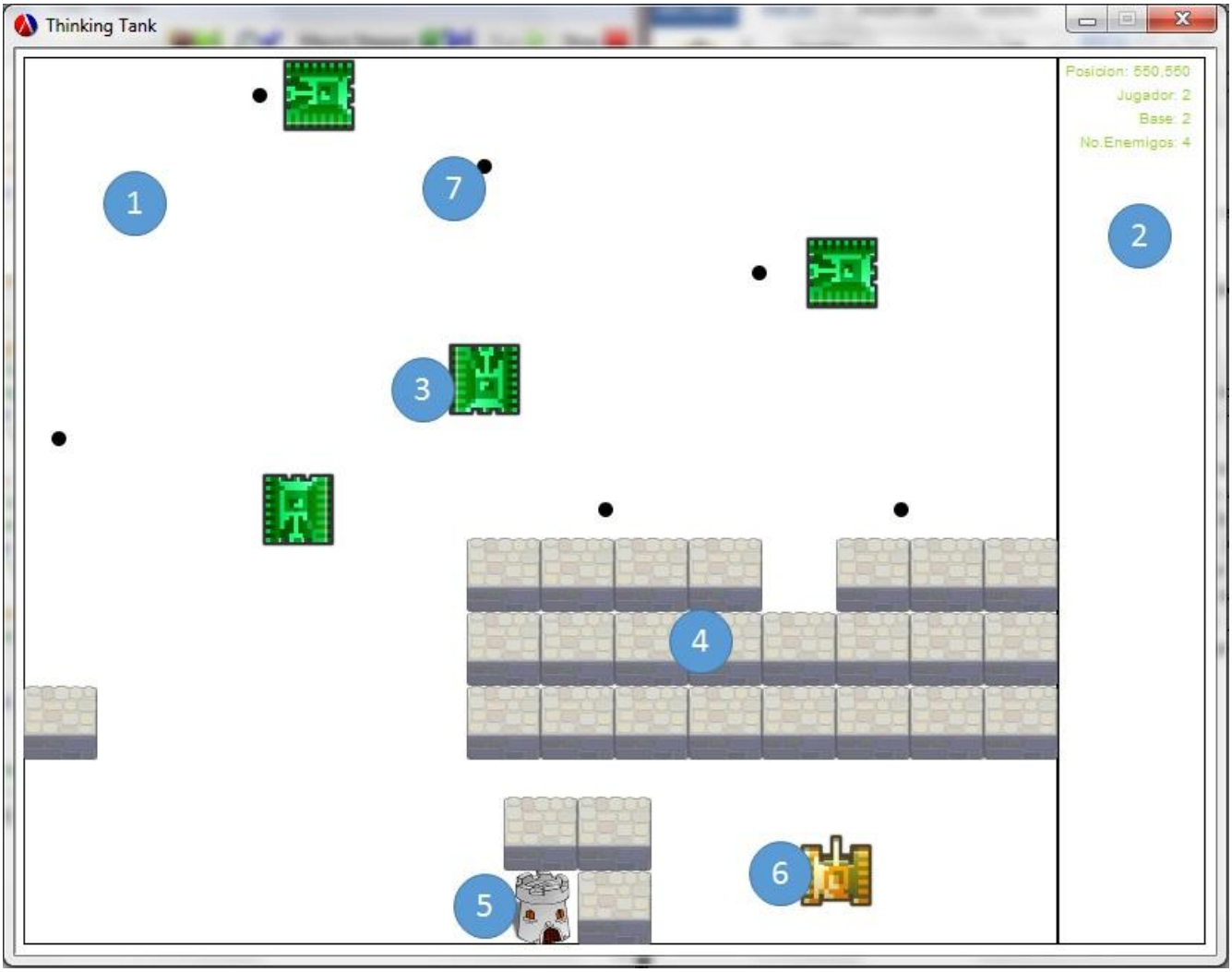
Para la segunda entrega implementaremos la temática del pensamiento para nuestro juego tanques de pensamiento (Thinking Tank).

Para esta entrega se tendrá el acumulado de las funcionalidades anteriores más las que relacionaremos a continuación.

- Lograr establecer una base para el jugador.
- Eliminar balas que colisionan entre ellas con objetivos diferentes (jugador -> enemigo, enemigo -> jugador, donde x es jugador y z objetivo, dado x -> z). Es decir balas entre aliados no se afectan.
- Lograr que los tanques enemigos puedan herir o quitar vidas al jugador.
- Lograr que los tanques enemigos puedan quitar vidas a la base del jugador.
- Lograr cargar las preguntas de un archivo de texto.
- Lograr que en el momento en que colisione una bala disparada por el jugador con un tanque enemigo, se muestre una pregunta en la pantalla del juego. por cada colisión se debe mostrar una pregunta diferente a la mostrada anteriormente.
- Lograr quitar vidas al tanque enemigo si, el jugador responde correctamente a la pregunta.
- Lograr pausar el juego cuando se determine la pérdida del mismo.
- Lograr colocar una barra de estado en la pantalla del juego, dicha barra de estado debe mostrar:
    - Posición del jugador en la pantalla del juego.
    - Numero de vidas del jugador.
    - Numero de vidas de la base del jugador.
    - Numero de tanque enemigos en el mundo.
    - Numero de balas en el mundo.
    - Numero de preguntas acertadas.
    - Numero de preguntas no acertadas.

ANEXOS

Aquí presentamos una posible presentación del juego.



1. Área de juego.
2. Barra de estado.
3. Tanque enemigo.
4. Ladrillos.
5. Base del jugador.
6. Tanque del jugador
7. Balas.

REFERENCIAS

[1] «Wikipedia,» [En línea]. Available: http://es.wikipedia.org/wiki/Battle_City. [Último acceso: 12 Marzo 2015].

[2] «Wikipedia,» [En línea]. Available: http://es.wikipedia.org/wiki/Think_tank. [Último acceso: 12 Marzo 2015].

[3] F. Bice, R. DeMaio, S. Florence, F.-Y. Mimi Lin, S. Lindeman, N. Nussbaum, E. Peterson, R. Plessner, D. Van Horn, M. Felleisen y C. Barski, «http://www.realmofracket.com/,» 2013. [En línea]. Available: http://www.realmofracket.com/. [Último acceso: 8 Marzo 2015].