Pontificia Universidad Católica de Chile
Escuela de Ingeniería
Departamento de Ingeniería Eléctrica
IEE2463 Sistema electrónicos programables

Tarea 2

Introducción a C - Control de Flujo

---

1. Objetivo

Al finalizar este trabajo el estudiante podrá conocer y aplicar las instrucciones, *If-else, for, Switch, While, do-while, break and goto,* utilizadas para el control de flujo del lenguaje secuencial C. Además debera entender y aplicar el uso de funciones y librerías en el entorno de programación.

2. Battle Royal

En la actualidad los juego battle royal son de las opciones más populares en la industria de los E-Sport. El concepto de juego es bastante simple, un duelo a muerte entre jugadores hasta que solo uno se mantiene en pie, donde cada duelista posee una vida maxima, un daño de ataque y un escudo.

Para esta tarea, se debe implementar un juego de este tipo pero monitoreado por consola. Para esto debe realizar una serie de tareas que lo guiaran hasta el prototipo final. Sin embargo, las instrucciones del juego son las siguientes.

En un comienzo se dispondran de 1000 puntos, los cuales deben ser repartidos a las estadisticas del jugador, no existen restricciones en vida y daño de ataque pero el escudo no puede superar los 100 puntos, antes de comenzar el juego. Una vez se entra al juego, el jugador se debe encontrar en cada ronda con un adversario y pelear a muerte. Si logra ganar, tendrá la posibilidad de obtener un consumible para recuperar parte de su vida base.

El sistema le debe preguntar al jugador en cada ronda si desea continuar o si desea rendirse. En caso de que se rinda, debe avisar que a perdido. En caso contrario, es decir, si logra sobrevivir a todas las rondas el sistema debera avisar que ha ganado.

3. Tareas

3.1. Actividad 1

En esta sección se aplicará el concepto de funciones. Por ende se deberá crear una libreria con todas las funciones listadas a continuación y que serán utiles para el desarrollo del juego (2 pts).

- `Instrucciones`: Esta función debe imprimir en la consola todas las instrucciones del juego necesarias para un jugador nuevo.
- `Actualizar_Jugador`: Esta función debe recibir como entrada las estadisticas del jugador e imprimirlas en un formato adecuado.
- `Nueva_Vida_Maxima`: Esta función debe permitir ingresar la vida actual del jugador, el escudo del jugador y el daño de ataque del oponente. Como salida debe entregar la nueva vida actual, la cual esta dada por la siguiente formula.

  $$\texttt{Nueva\_Vida} = \texttt{Vida\_Actual} + \texttt{Escudo} - (1 + p) \cdot \texttt{Daño}$$

  Donde p es un valor discreto que puede ser 1 o 0. Con una probabilidad de 50 % de ser 1. El jugador no puede ganar vida en ningun caso, por ende si el escudo supera el daño recibido se debe conservar la vida actual.
- `Generar_Consumible`: Esta función debe recibir la vida acutal que posea el jugador y devolver una nueva vida tras realizar la siguiente operación.

  $$\texttt{Nueva\_Vida} = (1 + p/2) \cdot \texttt{Vida\_Actual}$$

  Donde p es un valor discreto que puede ser 1 o 0. Con una probabilidad de 50 % de ser 1. Sin embargo, la vida nueva no puede superar el maximo impuesto en un comienzo (Use if).
- `Adversario`: Esta función debe generar un adversario con valores de vida maxima, daño de ataque y escudo al azar. Este jugador virtual debe ser creado a partir de las reglas para crear un jugador, descritas más abajo.
- `Batalla`: Esta función debe recibir las estadisticas de ambos jugadores y entregar la nueva vida del jugador. Para esto se debe realizar una batalla entre la CPU y el jugador. Para esto debe seguir las siguientes reglas (Use Case). En caso de que la vida de la CPU sea el doble de la vida del jugador, el daño aplicado (sin incluir el golpe critico) debe ser la mitad. En caso de que el jugador se quede sin vida, se le debe otorgar la oportunidad de mantenerse con 1 punto de vida solo una vez. Los ataques desde el jugador al CPU no aplican estas reglas. La etapa de ataques entre jugador y CPU debe durar hasta hasta un maximo de 4 rondas o hasta que alguno de ellos se quede sin vida. Es decir deben realizar 2 ataques cada uno alternando entre jugador y CPU (Use for), teniendo la CPU la prioridad para atacar.

3.2. Actividad 2

En esta etapa se debe programar el flujo de trabajo del programa, a través del uso de lo que se conoce como control de flujo y siguiendo las siguientes instrucciones.

1. Se debe inicializar el sistema, entregando todas las instrucciones al usuario.

2. El usuario comienza con 1000 puntos, los cuales debe repartir entre vida, daño y escudo (La suma de las estadisticas no pueden superar el maximo y los puntos del escudo no pueden superar los 100 puntos).

3. Una vez que se crea el jugador se debe generar un adversario y generar una batalla, durante cada ronda se debe imprimir la vida del jugador. En caso de perder la batalla se debe terminar el juego e imprimir un mensaje que señalice la derrota (2 pts).

4. En caso de no perder se debe llamar la función `Generar_Consumible`, y aplicar la bonificación en caso de que aparezca un consumible, imprimiendo en la consola un aviso que le diga al usuario el uso de esa bonificación. Además se debe volver al punto 3 y realizar el mismo procedimiento (Use while) (2 pts).

5. El sistema debe generar un total de 50 adversarios con solo 200 puntos a repartir. Para que el jugador tenga algo de ventaja.