***Trabajo Práctico Nro. 4 (GRUPAL):***

1. ***Formato de entrega:*** ver reglamento de entregas
2. ***Fecha de entrega:*** La fecha de presentación de los ejercicios está pactada en cada uno de los distintos grupos que componen el presente trabajo práctico. No se aceptará una entrega posterior sin tener entregadas todas las anteriores.

# ***Evaluación:*** Luego de entregado el trabajo práctico los ayudantes procederán a evaluar los ejercicios resueltos, en caso de encontrar errores se documentará en la carátula del TP que será devuelta al grupo con la evaluación final del TP y en caso de ser necesaria una fecha de reentrega (en caso de no cumplir con dicha fecha de reentrega el práctico será desaprobado). Cada ayudante podrá determinar si un grupo debe o no rendir coloquio sobre el trabajo práctico presentado. Es importante indicar que dentro de la evaluación se tendrá en cuenta si se cumple o no con las peticiones de tener makefiles, la buena codificación y documentación del código fuente.

1. ***Fecha de entrega:*** El **19/11/2012 ó 20/11/2012,** dependiendo del día que se curse la materia.
2. ***Fecha tardía de entrega:*** El **19/11/2012 ó 20/11/2012,** dependiendo del día que se curse la materia. **(NO ES ERROR, ES EL MISMO RANGO DE DÍAS)**.
3. **Tipo de desarrollo*:*** El objetivo del presente trabajo práctico es el de programar un juego en formato gráfico para múltiples jugadores tanto en una como en varias máquinas al mismo tiempo. Es obligatorio para el desarrollo del TP el uso de Procesos pesados, memoria compartida, Threads, SDL, y sockets para la implementación del juego. No está permitido el uso de Threads de SDL. Además se debe poder ejecutar un cliente local y remoto contra el mismo proceso servidor.
4. **Objetivo:** Programar un juego basado en el juego **Frogger** denominado “**D-Frogger Tournament”** para ser ejecutado en múltiples máquinas y multijugador
5. **Arquitectura del juego:** el juego será un torneo **todos contra todos** de **Frogger** que se podrá jugar en múltiples máquinas y por múltiples jugadores (todos los que se hayan conectado en un período de tiempo dado). Estará compuesto por un servidor principal (Servidor del torneo) que se encargará de llevar toda la administración del torneo, recibir las conexiones de los procesos clientes (Jugadores) y generar hijos (Servidores de la partidas) que se encargarán de administrar cada una de las partidas que se efectúan dentro del torneo.

Adicionalmente a los procesos anteriormente mencionados existirá un proceso cliente (Jugador) que se encargará de manejar la comunicación con el proceso servidor (en un principio con el servidor del torneo para inicializar la comunicación y luego con el servidor de partida que le fuera asignado), graficar el juego y aceptar los comandos del usuario que esté jugando la partida. Deberá existir un proceso cliente por cada jugador del torneo y podrán estar en la misma máquina que se ejecutó el servidor, o en máquinas distintas.

**Servidor del torneo**

**Servidor de la partida 3**

**Servidor de la partida 2**

**Servidor de la partida 1**

**Jugador 1**

**Jugador 4**

**Jugador 5**

**Jugador 3**

**Jugador 2**

1. **Descripción detallada de los componentes:**

**Servidor del torneo:** Será el proceso principal del juego. Sus funciones serán las siguientes:

* Recibir las conexiones de los nuevos clientes durante un tiempo que se indicará en el archivo de parámetros. Luego de pasado el tiempo cerrará el ingreso al torneo y comenzará a lanzar las partidas. Es importante mencionar que como cada partida involucra a dos jugadores, si al momento de cerrar el ingreso quedan jugadores impares el último jugador que se conectó deberá ser eliminado (informándolo)
* Asignar los clientes a los servidores de las partidas (ver detalles más adelante)
* Mostrar los resúmenes del torneo en la pantalla donde se ejecuta. Es importante mencionar que la información que se mostrará deberá estar en formato gráfico, informando al menos los siguientes puntos:
  + Cantidad de partidas jugadas
  + Cantidad de partidas faltantes para terminar el todos contra todos
  + Cantidad de clientes conectados
  + Posiciones actualizadas a cada momento
* Determinar y publicar las posiciones del torneo. La posición de cada jugador en el torneo deberá ser informada gráficamente en cada pantalla de juego.

**Servidor de la partida:** Serán hijos del servidor del torneo (proceso pesado en forma obligatoria), que se comunicará con el proceso padre a través de memoria compartida (obligatoriamente) y que tendrá como objetivo administrar una partida entre los dos jugadores que el servidor del torneo le asignó. Su funcionalidad deberá ser la siguiente:

* Comunicación con los clientes que participan de la partida
* Administración del estado de la pantalla
* Llevar el resultado de la partida actual, compartirla con los clientes y actualizarla al servidor del torneo cada vez que hay un cambio en ésta
* Este proceso deberá estar programado con threads que se encargarán de cada una de las funciones que tiene asignadas

**Jugador:** serán los procesos clientes y deberán estar programado con threads, teniendo que estar al menos divididas las siguientes funcionalidades:

* Comunicación con los servidores
* Dibujo de pantallas
* Lectura del teclado / mouse

**Reglas del torneo:**

* El servidor del torneo habilitará la inscripción durante un tiempo finito y determinado. Luego de este tiempo dejará de aceptar clientes y comenzará el torneo. Durante este tiempo en la pantalla del servidor se deberá ver la siguiente información:
  + Tiempo faltante para el cierre de inscripciones
  + Lista de jugadores anotados hasta el momento
* En caso que la cantidad de jugadores inscripta sea impar, se deberá echar al último cliente jugador conectado.
* El torneo deberá ser obligatoriamente todos contra todos y se deberá llevar contabilizada la cantidad de partidas ganadas y perdidas por cada jugador (esta información debe estar en la pantalla del servidor constantemente)
* Se declarará ganador de una partida a aquel jugador que logre llevar al menos cuatro ranas a su nido o que se quede sin contrincante (ya que perdió todas sus vidas).
* Las dos ranas (una de cada jugador) saldrán de la misma posición y con teclas o el mouse deberán cruzar hasta sus nidos, la que primero llegue ganará ese lugar apareciendo una nueva rana en la zona de partida
* En caso de que la rana sea pisada por un auto, se caiga al agua, salga de los límites de la pantalla o se choque contra algún objeto fijo, perderá la vida el jugador y su rana aparecerá en la zona de partida nuevamente
* Existirá un tiempo máximo en el que una rana debe si o sí llegar a sus nidos. En caso de que ninguno de los dos jugadores lleguen a la zona de nidos en ese tiempo ambos jugadores perderán una vida.

Este tiempo se reiniciará cada vez que un jugador llegue a un nido.

* Dos ranas no podrán ocupar el mismo lugar al mismo tiempo en ninguna zona del tablero. Es parte del grupo decidir como implementará esto, si al momento de una rana chocar contra la otra desplaza de su lugar a la rana o si no podrá ir para el lado donde la otra rana se encuentra.
* Si durante la partida un jugador sale, o se muere el servidor del torneo deberá asignarle la partida como ganada a todos los jugadores que no hayan jugado con él y al que estaba jugando.

Pantalla de juego de computadora

Descripción generada automáticamente con confianza media

**Información del jugador 2**

**Información del jugador**

**Medidor del tiempo restante**

**Zona de partida**

**Nido de la rana**

**Parámetros obligatorios del servidor:**

En esta sección se detallan todos los parámetros que obligatoriamente deberá tener el servidor. Es obligatorio generar un archivo de configuración que contenga todos estos datos (no se tomará como válido pasar los valores como parámetros del binario).

* **Puerto del socket:** Indicará en qué puerto esperará el servidor por las conexiones desde los procesos cliente
* **Tiempo de espera para aceptar jugadores:** Será el tiempo que se esperará hasta que se conecten los jugadores
* **Tiempo máximo para llegar a los nidos:** Será el tiempo máximo que tendrán los jugadores para llegar al nido
* **Cantidad de vidas por jugador:** Será la cantidad de vidas que tendrá un jugador en cada partida.

**Parámetros obligatorios de los clientes:**

En esta sección se detallan todos los parámetros que obligatoriamente deberán tener los procesos clientes. Se recomienda generar un archivo de configuración que contenga todos estos datos.

**Comunicación y tiempo de espera:**

* **IP / nombre del server:** Indicará la dirección IP o el nombre del servidor al que debe conectarse para iniciar el juego
* **Puerto del socket:** Indicará en qué puerto escucha el servidor
* **Teclas del juego:** Identificará qué teclas se deberán utilizar para jugar en cada uno de los clientes (se deberá poder jugar con distintas teclas para cada jugador)

**Lista de controles mínimos que se le aplicarán al desarrollo al momento de su entrega para evaluación:**

A continuación se detallan algunos de los controles que se le realizarán al juego al momento de ser sometido a su evaluación y cual es la importancia de cada uno de ellos en la evaluación final.

**Nota:** Esto es sólo una lista orientativa, pudiendo el grupo de docentes de la cátedra agregar nuevas condiciones sin previo aviso.

|  |  |  |
| --- | --- | --- |
| **Tema** | **Control** | **Criticidad** |
| **Entrega** | Documentación Impresa (Manual de usuario / instalación y diseño del juego) | Obligatorio |
|  | CD | Obligatorio |
|  |  |  |
| **Funcionalidad** | Jugabilidad | Obligatorio |
|  | Servidor muestra la información de las partidas en formato gráfico | Obligatorio |
|  | Cumple con las reglas del juego | Obligatorio |
|  | Gana el jugador que llevo más ranas al nido | Obligatorio |
|  | Mueren las ranas atropelladas o que se caen al agua | Obligatorio |
|  | Pierden los dos jugadores cuando pasa el tiempo | Obligatorio |
|  | Muestra los nombres de los jugadores | Obligatorio |
|  | El jugador puede salir en cualquier momento | Obligatorio |
|  | No se permite conectar luego de pasado el tiempo | Obligatorio |
|  | Elimina al último jugador cuando la cantidad es impar | Obligatorio |
|  | Existen componentes no planteados en la descripción | Plus |
|  | Tiene pantalla de presentación | Plus |
|  | Los troncos y autos funcionan a diferentes velocidades relativas entre si | Plus |
|  |  |  |
| **Construcción** | Funciona el cliente local | Obligatorio |
|  | Funciona el cliente remoto | Obligatorio |
|  | Funciona el servidor del torneo | Obligatorio |
|  | Funciona el servidor de partidas | Obligatorio |
|  | Cantidad de Jugadores variable e ilimitada | Obligatorio |
|  | Reacciona bien ante la finalización normal de un jugador | Obligatorio |
|  | Utiliza procesos pesados para los servidores de partidas | Obligatorio |
|  | Utiliza Memoria Compartida entre servidor de partida y del torneo | Obligatorio |
|  | Utiliza Threads | Obligatorio |
|  | Sincronización de los Threads | Obligatorio |
|  | Cantidad de sockets limitada (comando netstat) | Plus |
|  |  |  |
| **Parámetros** | Teclas configurables | Obligatorio si usa teclado |
|  | IP/Puerto | Obligatorio |
|  | Funciona con nombres de máquinas | Plus |
|  | Validar que al cambiar los parámetros funcione | Decrementa puntos si no se hace |
|  |  |  |
| **Robustez** | Reacción ante caídas de clientes o servidores | Obligatorio |
|  | Cerrado de recursos (Conexiones, Memoria, Semáforos) | Obligatorio |
|  | No quedan procesos zombies | Obligatorio |
|  | Cumple con la regla de salida de un jugador | Obligatorio |
|  | Consumo de cpu menor al 60% tanto en servidores como clientes | Obligatorio |
|  | Consumo de cpu menor al 30% tanto en servidores como clientes | Plus |
|  |  |  |
| **Codificación** | Orden del código | Plus |
|  | Código legible (Funciones pequeñas 10/15 líneas) | Plus |
|  | No magic numbers. Se debe utilizar constantes para todo, los números y los strings. | Plus |
|  | No hay Implementación en los .h | Obligatorio |
|  | Separación funciones en .c/.o | Plus |
|  | Programación orientada a objetos | Plus |
|  | Comentarios y código autodocumentado | Plus |
|  | Makefiles | Obligatorio |