Universidad Técnica Federico Santa María
Campus Santiago San Joaquín
Sistemas Operativos
2do Semestre 2021

UNIVERSIDAD TECNICA
FEDERICO SANTA MARIA

Sistemas Operativos
Tarea 2:

Profesor: Viktor Tapia
Ayudantes De Cátedra: Joaquín Castillo y Juan Pablo Varas
Ayudantes De Tarea: Sofia Mañana y Juan Pablo Varas

8 de octubre de 2021

1. Reglas Generales

Para la siguiente tarea se debe realizar un código programado en lenguaje C. Se exigirá que los archivos se presenten de la forma más limpia y legible posible. Deberá incluir un archivo README con las instrucciones de uso de sus programas junto a cualquier indicación que sea necesaria, y un archivo MAKE para poder ejecutar el programa.

2. Tarea

Se solicita desarrollar un Monopoly, donde el objetivo no es comprar propiedades, solo ganar/perder dinero. Pero como jugar Singleplayer Monopoly es algo aburrido, en esta tarea lo acompañarán 2 jugadores creados por usted con la ayuda de la bien ponderada función *fork()*.

2.1. Tablero

El tablero del juego consiste en un cuadro de 8x8, donde usted y los otros dos jugadores se desplazarán desde la esquina inferior izquierda en sentido horario. A continuación se adjunta una imagen para ilustrar el tablero de mejor manera.

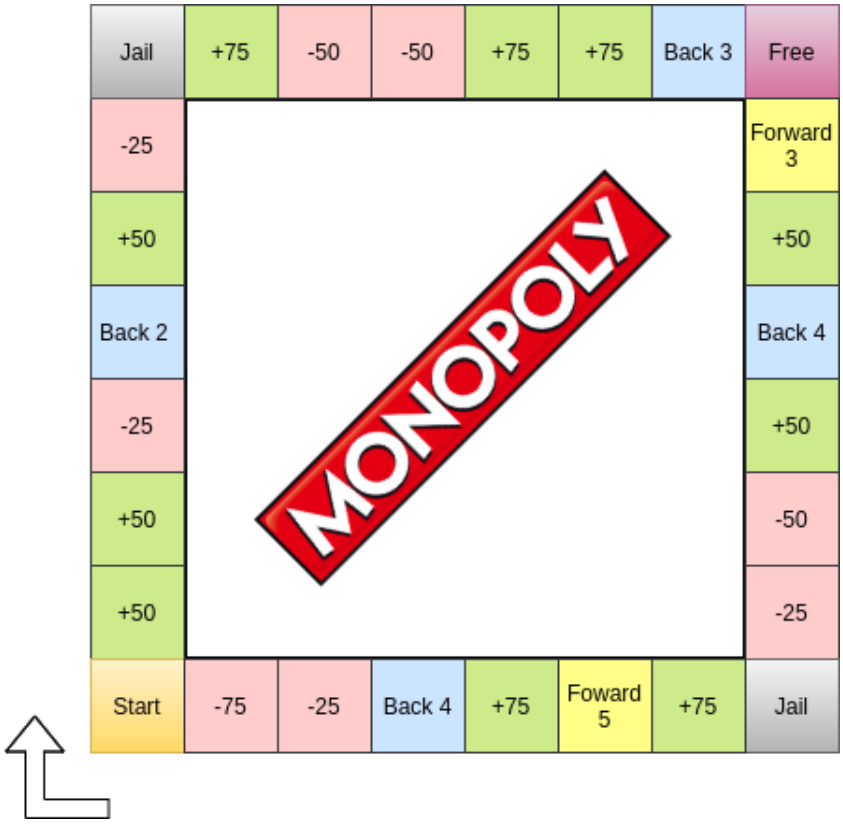

Figura 1: Tablero

El tablero contiene distintas casillas, las cuales explicaremos a continuación:

- **Start**: Es la casilla de inicio del juego. Sin contar la primera vuelta, cada vez que un jugador cruce la casilla **Start** recibirá 100 pesos (Caer justo en la casilla **Start** también se considera como cruzarla).
- **Jail**: La clásica casilla de cárcel. Si un jugador cae en ella, perderá un turno. No se requiere hacer nada para salir, solo esperar el turno perdido, y jugar con normalidad en el siguiente turno.
- **Free**: En esta casilla no pasa absolutamente ningún efecto.
- **+Pesos**: En este tipo de casillas, el jugador gana la cantidad de pesos marcada en ella. Por ejemplo, si cae en una casilla **+50**, el jugador ganará 50 pesos.
- **-Pesos**: En este tipo de casillas, el jugador pierde la cantidad de pesos marcada en ella. Por ejemplo, si cae en una casilla **-50**, el jugador perderá 50 pesos.
- **Forward X**: Si un jugador cae en este tipo de casillas, debe avanzar los X espacios marcados en ella. Por ejemplo, al caer en una casilla **Forward 3**, el jugador avanzará 3 espacios. El efecto de la casilla donde el jugador caiga luego de avanzar esos 3 espacios debe accionarse.

- **Back X**: Si un jugador cae en este tipo de casillas, debe retroceder los X espacios marcados en ella. Por ejemplo, al caer en una casilla **Back 3**, el jugador retrocederá 3 espacios. El efecto de la casilla donde el jugador caiga luego de retroceder esos 3 espacios debe accionarse.

2.2. Jugadores, Procesos y Objetivo del juego

El juego se compone de 3 jugadores (siendo usted uno de ellos). Cada jugador debe ser un proceso distinto, por lo que en total su programa estará ejecutando 4 procesos (un padre y 3 hijos). La única acción que tendrá cada jugador es lanzar un dado de 6 caras (con números de 1 a 6) que simbolizan cuantos espacios avanza cada jugador. Además, considere que usted tendrá el primer turno, luego la primera computadora y finalmente la segunda.

Cada jugador comenzará con 100 pesos. El objetivo de cada jugador es alcanzar la meta de 500 pesos. Cuando esto ocurra, el juego finalizará, y quien haya alcanzado la meta de 500 pesos será el ganador. Recuerde que para finalizar el programa, primero deben finalizar los procesos hijos, y luego el proceso padre.

2.3. Visualización en la Consola

Queda a su creatividad diseñar el formato de visualización del juego. Como mínimo se requiere que se muestren los siguientes elementos:

- **Tablero**: Inicialmente debe imprimirse por pantalla un bosquejo del tablero mostrado anteriormente, junto a algún indicador que nos diga donde se encuentra cada jugador. Cada vez que un jugador se mueva, debe actualizarse de alguna manera el tablero para mostrar el cambio.
- **Turno Actual**: Debe estar indicado por pantalla el jugador que esté jugando. Un pequeño mensaje bastará
- **Número del dado**: Cada vez que se lance el dado, debe indicarse por pantalla el número que este muestre.
- **Dinero actual**: Debe indicarse por pantalla el dinero actual de cada jugador.

3. Presentación Aleatoria

Para cada tarea, se seleccionarán grupos al azar para presentar su tarea frente a ayudantes y eventualmente profesor, recibiendo una ponderación del 75 % y 25 % entre tarea y presentación respectivamente. Si su grupo presentó en una tarea, no volverá a salir nuevamente. Se comunicará días antes que grupos presentarán. Si el grupo se conforma por alumnos de paralelos distintos, nosotros comunicaremos quien debe presentar.

4. README

Debe contener como mínimo:

- Nombre, rol y paralelo de los integrantes.
- Especificación de los nombres de los archivos (cual corresponde a cada sección de la tarea).
- Instrucciones generales de compilación y uso.

5. Consideraciones Generales

- Se deberá trabajar de a pares. Se deberá entregar en Aula a mas tardar el día 29 de Octubre del 2021 a las 23:55 horas. Se descontarán 5 puntos por cada hora o fracción de atraso. Las copias serán evaluadas con nota 0 en el promedio de las tareas.
- La tarea debe ser hecha en el lenguaje C. Se asume que usted sabe programar en este lenguaje, ha tenido vivencias con el, o que aprende con rapidez.
- Pueden crear todas las funciones auxiliares que deseen, siempre y cuando estén debidamente comentadas.
- Las tareas serán ejecutadas en Linux.
- Los archivos deberán ser comprimidos y enviados juntos en un archivo .tar.gz en el formato **TAREA2_ROL1_ROL2**.
- Las preguntas deben ser hechas por Aula.
- Si no se entrega README o MAKE, o si su programa no funciona, la **nota es 0** hasta la recorrección.
- Se descontarán puntos por Warnings y Leaks de memoria.
- Se **descontarán** 50 puntos por:
  - Mala implementación del Makefile.
  - No respetar el formato de entrega.