UVG
Redes
17 de agosto, 2020

Integrantes: Oliver Mazariegos, José Pablo Viana, Estuardo Ureta

Documento resumen: Proyecto 1

Juego de cartas: Old Maid

Explicación e historia:

Old Maid es el nombre de un juego de cartas simple que requiere al menos dos jugadores pero puede llegar hasta 8 jugadores.
La traducción lógica para Old Maid es <Solterona>.
En esencia una solterona es una mujer que no está casada. Es una frase despectiva que se utiliza para describir a una mujer que se considera "vieja" y que no tiene marido.
La frase <solterona> ha existido al menos desde la época victoriana (aunque podria ser desde antes). En aquel entonces, la idea de que una mujer permaneciera soltera era impactante. El juego ha existido desde dicha época.
En resumen, el juego requiere que los jugadores intenten evitar quedarse atrapados con la tarjeta Old Maid. Mientras bajan parejas por turnos para que acabe el juego solo con dicha tarjeta.
El juego reforzó las reglas sociales de la época. Implica que las mujeres deben casarse cuando sean jóvenes para no terminar siendo una solterona. También implicaba que los hombres debían casarse con una mujer joven para evitar quedarse atrapados con una solterona.
Old Maid se juega con una baraja estándar de 52 cartas. Se consideraba que la reina negra era la vieja solterona. Una de las otras tres reinas se eliminó del mazo para asegurarse de que la carta de Old Maid no coincidiera.

Descripción:

Detalles de protocolo:

En escencia se desarrollo un protocolo sobre TCP basado en un 3-way-handshake. Ya que se buscaba implementar la funcion de multi jugadores en nuestro juego de cartas, era necesario que se compartieran estados y que existiera uno general.
Con esto en mente primero definimos que el intercambio de información de mensajes (sean estos: texto, mano, username etc.) cuentan con una sección llamada HEADER. En esta es donde se encuentra la longitud del mensaje y en la segunda parte el mensaje. Esta segunda basicamente es un diccionario. Estos mensajes tienen una estructura basica que contaba con: el tipo, mensaje y usuario. Por lo tanto esta modalidad se implemento con toda la informacion que se debia transmitir para el funcionamiento del multijugador.

Link a documento con diccionario y guia de protocolo:
https://docs.google.com/document/d/1DK29HZ7POETEu4KUkKcmrgEaqQT9JHIwpxHBIffnqc0/edit?usp=drivesdk

Instalaciones:

No se requiere la instalación de ninguna librería ya que las utilizadas vienen en el estándar.

Librerias utilizadas:

- import socket
- import select
- import errno
- import sys
- import pickle
- import threading
- import time
- import itertools
- import random
- import datetime

Versiones y para la ejecución:

Local:

Para la ejecución local del programa es importante primero descargarlo en github y clonar. En la ubicación de la carpeta correr en la terminal el archivo server.py para inicializarlo.

Online:

Se ha implementado el uso de Amazon Web Services (aws). Esto para montar el servidor en la nube y se incluyen todos los datos de conexión.

Corriendo cliente.py seguir instrucciones para el usuario.

Retos y dificultades:

A continuacion se listan algunos retos y proyecto que se presentaron durante el progreso del proyecto acompañadas con una breve descripción:

- La primera complicación fue la comunicación con el servidor:
    - Ya que el servidor debe aceptar nuevas conexiones de los clientes. Se buscó alguna forma de identificar a nuestros usuarios como únicos. Aunque bien se podria mostrar a los usuarios por dirección IP, la mayoría de las personas tienden a crear algún tipo de nombre de usuario. Por lo que nuestro servidor primero permite a los clientes conectarse y elegir un nombre de usuario. Más allá de esto, el servidor recopilará los mensajes entrantes y luego los distribuirá al resto de los clientes conectados.

- La segunda y tercero fue la creacion de los Rooms y como asignar personas a un room:
    - En client.py, tras ingresar usuario, se depliega un menú y cuando el jugador selecciona la opcion de jugar se crean los Rooms para que el server ya lo pueda operar. No estabamos haciendo que el server reconociera si es un Room nuevo o si ya estaba en el Room.

- Cuarto como hacer que todos compartan un solo Deck u OldMaid por asi decirlo:
    - Se comenzo instanciando el OldMaid individualmente en cada cliente y notanos que asi no funcionaba. Llegamos a la conclusión, realizando pruebas, que el server debia de encargarse de esto y por lo tanto se corrio al mismo solucionando el problema.