UNIVERSIDAD INTERNACIONAL DE DESARROLLO

LOGICA DE PROGRAMACIÓN

NOMBRE: ROJAS ESCOBAR JAIME SEBASTIAN

TAREA: TRABAJO AUTÓNOMO 1

FECHA DE ENTREGA: 25 DE MAYO DEL 2025

1. NOMBRE DEL SOFTWARE A DESARROLLAR.

Piedra papel o tijera.

2. INVESTIGACIÓN Y SELECCIÓN DEL DIAGRAMA.

El diagrama de casos de uso es un diagrama UML debido a que permite visualizar las funciones que tiene el juego y como los usuarios interactúa. Es útil para entender los requisitos del sistema, comunicar el funcionamiento y orientar el desarrollo y diseño del software.

Se eligió el diagrama de arquitectura por capas, puesto a que permite que se represente el programa de forma ordenada, debido a que separa la parte en la que interactúa el jugador de la parte que decide cómo funcionará el juego, lo que hace que este sea fácil de entender.

3. DESCRIPCION FUNCIONAL DEL JUEGO.

El juego “Piedra, papel o tijera” está diseñado para al menos dos jugadores.

Al iniciar el sistema solicitara el nombre de ambos participantes, en cada ronda los jugadores deben seleccionar cualquier opción: piedra, papel o tijera.

Condiciones:

- Papel gana a Piedra
- Tijera gana a Papel
- Piedra gana a Tijera
- Si se elige el mismo, es empate.

El sistema lleva un registro automático de las victorias de cada jugador. Al final del número de rondas jugadas, se muestra un resumen indicando cuántas rondas ganó cada uno y quién fue el ganador general.

Si la respuesta es "sí", se reinicia el juego desde la primera ronda.

Si la respuesta es "no", el programa mostrará un mensaje de despedida con felicitaciones y cerrará automáticamente.

4. DIAGRAMAS

Diagrama de caso de uso (UML)

Se muestra el diagrama que representa las funcionalidades del sistema y la interacción de los actores con el juego.

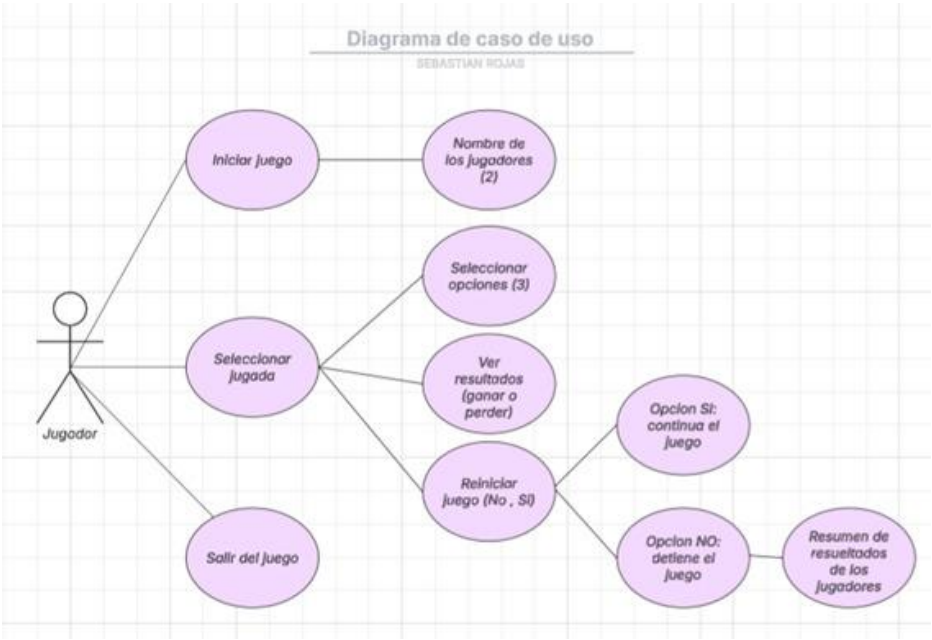

Diagrama de Arquitectura (CAPAS)

Se muestra, a nivel macro, cómo se estructura el software y cómo interactúan sus diferentes componentes.

5. POSIBLE PSEDUCÓDIGO (PSEINT)

```
Algoritmo PiedraPapelTijera
	Definir nombre1, nombre2, opcion1, opcion2, seguir Como Caracter
	Definir gana1, gana2, empates Como Entero
	gana1 <- 0
	gana2 <- 0
	empates <- 0
	Escribir 'Ingrese el nombre del jugador 1:'
	Leer nombre1
	Escribir 'Ingrese el nombre del jugador 2:'
	Leer nombre2
	Repetir
		Escribir nombre1, ', elige: Piedra, Papel o Tijera'
		Leer opcion1
		opcion1 <- Minusculas(opcion1)
		Mientras opcion1<>'piedra' Y opcion1<>'papel' Y opcion1<>'tijera' Hacer
			Escribir 'Opción inválida. Intenta de nuevo:'
			Leer opcion1
			opcion1 <- Minusculas(opcion1)
		FinMientras
		Escribir nombre2, ', elige: Piedra, Papel o Tijera'
		Leer opcion2
		opcion2 <- Minusculas(opcion2)
		Mientras opcion2<>'piedra' Y opcion2<>'papel' Y opcion2<>'tijera' Hacer
			Escribir 'Opción inválida. Intenta de nuevo:'
			Leer opcion2
			opcion2 <- Minusculas(opcion2)
		FinMientras
		Si opcion1=opcion2 Entonces
			Escribir 'Empate!'
			empates <- empates+1
```

```
                SiNo
                    Si (opcion1='piedra' Y opcion2='tijera') O (opcion1='papel' Y opcion2='piedra') O (opcion1='tijera' Y opcion2='papel') Entonces
                        Escribir nombre1, ' gana esta ronda!'
                        gana1 <- gana1+1
                    SiNo
                        Escribir nombre2, ' gana esta ronda!'
                        gana2 <- gana2+1
                    FinSi
                FinSi
                Escribir '¿Quieren seguir jugando? (si/no)'
                Leer seguir
                seguir <- Minusculas(seguir)
            Hasta Que seguir='no'
            Escribir 'Resumen de resultados:'
            Escribir '------------------------------------'
            Escribir nombre1, ' ganó ', gana1, ' veces.'
            Escribir nombre2, ' ganó ', gana2, ' veces.'
            Escribir 'Empates: ', empates
            Escribir '------------------------------------'
            Si gana1>gana2 Entonces
                Escribir nombre1, ', ganaste, ¡felicidades!'
            SiNo
                Si gana2>gana1 Entonces
                    Escribir nombre2, ', ganaste, ¡felicidades!'
                SiNo
                    Escribir 'Hubo un empate general entre ', nombre1, ' y ', nombre2
                FinSi
            FinSi
FinAlgoritmo
```

6. REFERENCIA BIBLIOGRÁFICA

Jiménez de Parga, C. (5 de Enero de 2021). *UML. Arquitectura de aplicaciones en Java, C++ y Python. 2ª Edición*. Obtenido de UML. Arquitectura de aplicaciones en Java, C++ y Python. 2ª Edición: https://www.ra-ma.es/libro/uml-arquitectura-de-aplicaciones-en-java-c++-y-python-2a-edicion_118845/

Vega Fajardo, A. (28 de Febrero de 2022). *Método basado en la programación por capas para generar código automático desde el diagrama de clases*. Obtenido de Método basado en la programación por capas para generar código automático desde el diagrama de clases: https://revistasinvestigacion.unmsm.edu.pe/index.php/rpcsis/article/view/17015