



II Proyecto Programación Orientada a Objetos

II Semestre 2019

Profesora

Samanta Ramijan

Integrantes

Emmanuel Murillo	2018201030
Jeison Sandi Mena	2016093589
Kevin Ceciliano	2019023828

Tabla de Contenidos

Tabla de Contenidos	1
Especificación del Proyecto	2
Objetivo del juego	2
Terminología y movimientos	3
Diagrama de Clases UML	4
Diseño del Programa:	5
Decisiones de diseño:	5
Análisis de Resultados	14
Objetivos alcanzados:	14
Manual de Usuario	15
Instrucciones de uso	15
Compilacion y ejecucion	16
https://repl.it/@KSCeciliano/ProyectoPOO	16
Conclusiones Personales	16
Emmanuel	16
Jeison	17
Kevin	17
Autoevaluación	18

Especificación del Proyecto

Batalla Naval es un juego tradicional de adivinación que involucra a dos participantes. Los jugadores manejan dos tableros divididos en casillas, cada tablero representa una zona diferente del mar abierto: la propia y la contraria. En uno de los tableros, el jugador coloca sus barcos y registra los «tiros» del oponente; en el otro, se registran los tiros propios, al tiempo que se deduce la posición de los barcos del contrincante (objetivo del juego).

Objetivo del juego

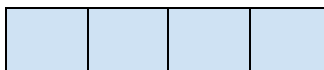
Hundir los 9 barcos del oponente antes que él o ella acabe con los propios.

Cada jugador tiene dos tableros compuestos por 10 filas y 10 columnas.

1. **Tablero de posición:** Es la zona propia, en el se distribuye la propia flota antes de comenzar la partida, durante la partida se podrá observar la posición de los barcos propios, así como los disparos del oponente, pero no se podrán realizar cambios ni disparos en él.
2. **Tablero Principal:** la zona del enemigo, donde tiene desplegada su flota. Será aquí donde se desarrollen los movimientos (disparos) para tratar de hundir los barcos enemigos. Aparecerá en la pantalla del jugador una vez comience la partida y en él quedarán registrados todos los movimientos, reflejando tanto los disparos al agua como los disparos acertados (golpes a barcos enemigos).

Cada jugador tiene una flota de 9 barcos de diferente tamaño, por lo que cada uno ocupará un número determinado de casillas en el tablero: Barcos de la flota.

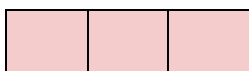
- a) Un portaaviones: cada uno ocupa 4 casillas.



- b) Tres submarinos: cada uno ocupa 3 casillas.



c) Tres destructores: cada uno ocupa 3 casillas.



d) Dos Fragatas: cada una ocupa 1 casilla.



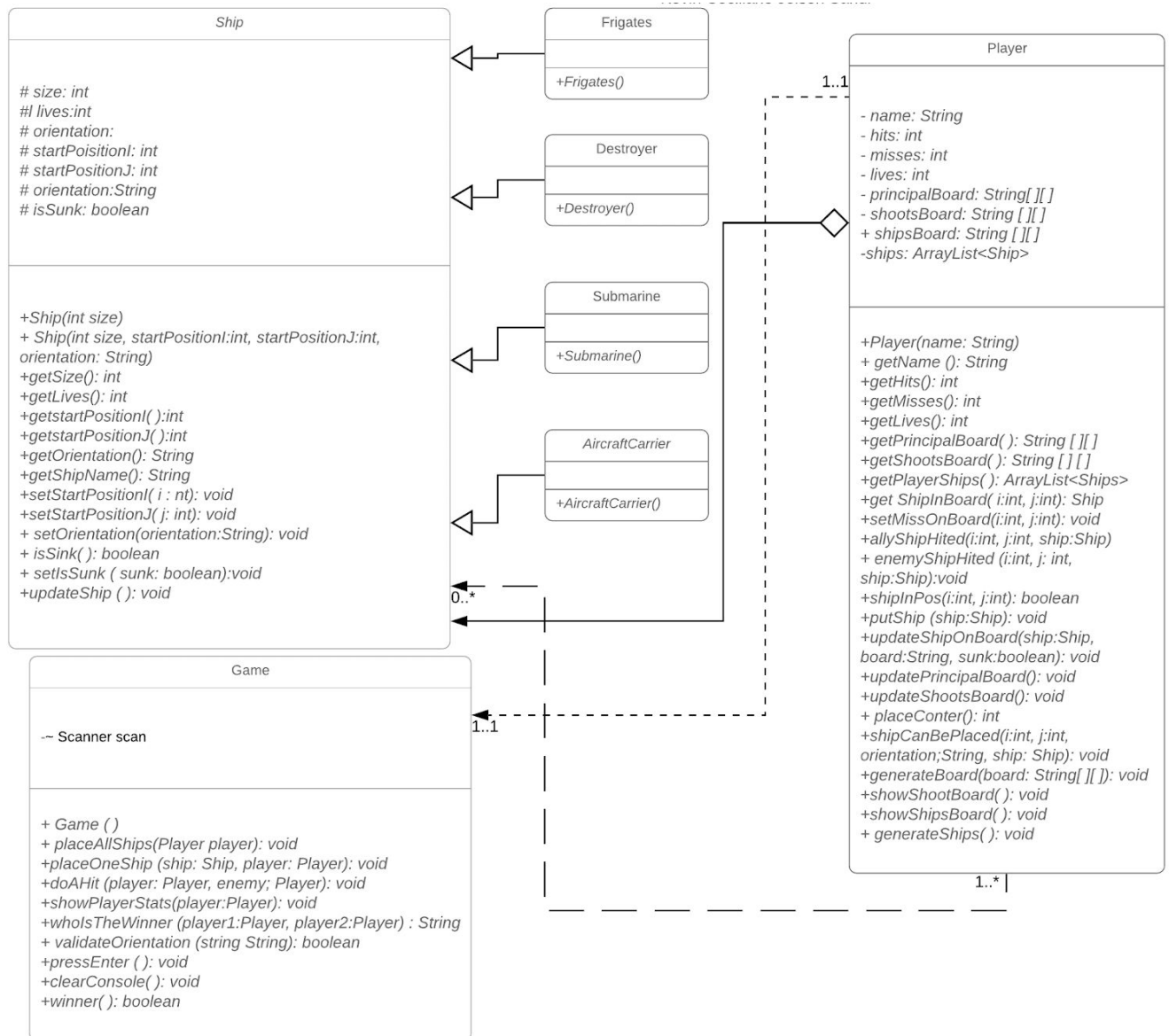
Terminología y movimientos

Para disparar es necesario indicar la casilla, indicando fila y columna.

- 1) **Agua:** cuando se dispara sobre una casilla donde no está colocado ningún barco enemigo, se dispara al agua. En el tablero principal aparecerá una X. Pasa al turno del oponente.
- 2) **Tocado:** cuando se dispara en una casilla en la que está ubicado un barco enemigo que ocupa 2 o más casillas y se destruye solo una parte del barco, es decir que el barco ha sido tocado. En el tablero principal aparece esa parte del barco con fuego.
- 3) **Hundido:** si se dispara en una casilla en la que está ubicada una fragata (1 casilla) u otro barco con el resto de las casillas tocadas, el otro barco se hunde, es decir, se elimina ese barco del juego. Eso deberá ser visible en el tablero.

Se repite el juego hasta que alguna de las dos flotas haya sido hundida por completo.

Diagrama de Clases UML



Diseño del Programa:

Para la toma de decisiones relacionadas con el diseño del programa requerido en la presente entrega, el grupo de trabajo basa en sus decisiones en la búsqueda por el cumplimiento de los requerimientos solicitados a través de la rúbrica y el logro de los objetivos planteados a través de este.

De acuerdo con lo anterior en una primera instancia, se determinó que dicho proyecto sería realizado a través del entorno de desarrollo (IDE) conocido como Apache NetBeans, dadas las distintas facilidades brindadas por el mismo y la experiencia previa de los integrantes del grupo en el uso de esta herramienta. A su vez dicha aplicación será ejecutada en un computador que cuente el sistema operativo Windows con alguna de sus más recientes versiones.

Adicionalmente, dado que la implementación de una interfaz conocida como GUI es opcional, pero no obligatoria, se descartó su implementación, dada la nula experiencia en el uso de dicha herramienta.

A continuación, se detallan algunos aspectos más concretos, en relación con el código o programa desarrollado.

Decisiones de diseño:

Siguiendo la línea de lo dicho anteriormente, en cuanto a las decisiones propias de la elaboración del código, en una segunda etapa se determinó que para la realización de este se implementarían las clases mostradas a continuación con su respectiva información:

- 1) **Game:** Esta clase será la encargada de llevar el flujo principal del juego, por lo que se encargará de conectar las demás clases existentes en el mismo y coordinar su correcto funcionamiento.

Métodos clase: Game	
Nombre	Descripción
Game()	Constructor de la clase en este caso se encuentra vacío.
placeAllShips(Player player)	Método para que el jugador ingresado coloque todos los barcos.
placeOneShip(Ship ship, Player player)	Método para colocar un barco actualizando los atributos correspondientes en player.
doAHit(Player player, Player enemy)	Pide al usuario ingresar los datos(fila, columna) del tiro que desea realizar. Actualiza los atributos de player y enemy conforme al tiro que player efectúa.
showPlayerStats(Player player)	Imprime las estadísticas(Hits,Misses,Lives) del Player recibido.
winner(Player player1, Player player2)	Verifica si uno de los jugadores tiene su contador de vidas en cero, retorna True en caso de ser así.
whoIsTheWinner(Player player1, Player player2)	Retorna el nombre del jugador que ha ganado la partida.

validateOrientation(String string)	Verifica si la orientación ingresada es válida.
pressEnter()	Imprime mensaje a los jugadores de presionar enter para continuar en el juego.
clearConsole()	Imprime saltos de línea para limpiar la consola

Adicionalmente en dicha clase se hace uso de las siguientes librerías para llevar a cabo la ejecución de los métodos planteados anteriormente.

- java.util.ArrayList
- java.util.Scanner

2) Player: Esta clase se encargará de definir y contener todos los atributos necesarios y métodos pertinentes a cada uno de los jugadores que participan durante la ejecución del juego, se destaca que dicha clase contendrá los tableros pertenecientes a cada jugador y un ArrayList de tipo Ship, con los barcos de cada jugador.

Métodos clase: Player	
Nombre	Descripción

Player(String name)	Constructor de la clase, en este caso como puede observarse recibe como parámetro únicamente el nombre correspondiente a un determinado jugador.
getName()	Retorna el nombre del jugador.
getHits()	Retorna los hits logrados por el jugador.
getMisses()	Retorna la cantidad de fallos del jugador.
getLives()	Retorn las vidas restantes del jugador
getPrincipalBoard()	Retorna el tablero principal
getShootsBoard()	Retorna el tablero de disparos del jugador
getPlayerShips()	Retorna el arreglo de barcos del jugador.
getShipInBoard(int i, int j)	Retorna los barcos en tablero.
setMissOnBoard(int i, int j)	Redefine los tiros fallidos en el tablero.
setAllyMissOnBoard(int i, int j)	Marca los fallos en el tablero de disparos del jugador.

setEnemyMissOnBoard(int i, int j)	Marca el fallo enemigo en el tablero principal del jugador
allyShipHited(int i, int j, Ship ship)	Actualiza el estado de los barcos y elimina los tengan todas sus casillas tocadas.
enemyShipHited(int i, int j, Ship ship)	Cuando se acertó un tiro se incrementa el contador de Hits del jugador.
shipInPos(int x, int y)	Es llamado por shipHit, verifica en una matriz alterna si la posición del disparo coincide con la casilla de un barco.
putShip(Ship ship){	Inserta un barco en el tablero principal y en el de verificación de disparos.
updatePrincipalBoard()	Actualiza el tablero principal con los movimientos del adversario
updateShipsOnBoard(Ship ship, String board[][], Boolean sunk)	Actualiza el barcos presentes en el tablero.
updateprincipalBoard()	Actualiza la informacion del tablero principal del jugador.
updateShootsBoard()	Actuzaliza el tablero de disparos del jugador.

shipCanBePlaced(int x,int y,String orientation,Ship ship)	Verifica si el barco puede ser insertado en las posiciones dadas por el jugador.
generateBoard(String[][] board)	Genera los tableros de juego.
showShootBoard()	Muestra la matriz de disparos del juego
showPrincipalBoard()	Metodo que imprime en pantalla principalBoard[][]
generateShips()	Genera los barcos del jugador al iniciar la partida.

Adicionalmente en dicha clase se hace uso de las siguientes librerías para llevar a cabo la ejecución de los métodos planteados anteriormente.

- java.util.ArrayList

3) Ship: Por su parte esta clase, la cual a su vez es importante tener en cuenta que será una superclase o clase padre, contendrá atributos y métodos relacionados con la funcionalidad en general de cada uno de los barcos dentro de la trama del juego o aplicación.

Métodos clase: Ship	
Nombre	Descripción
Ship(int size)	Constructor de la clase, recibe el tamaño del barco
Ship(int size, int startPositionX, int startPositionY, String orientation)	Constructor de la clase, recibe el tamaño del barco, su posición en X y Y, y la orientación hacia la que se desplegará.
getSize()	Retorna el tamaño del barco
getLives()	Retorna las vidas del barco la cual inicialmente es igual al largo del mismo.
getStartPositionI()	Retorna la posición en I de la primer casilla del barco.
getStartPositionJ()	Retorna la posición en J de la primer casilla del barco.
getOrientation()	Retorna la orientación en la que despliega el barco.
getShipName()	Retorna el nombre del barco.
setStartPositionI(int i)	Set para redefinir el atributo de la posición en X del barco.

setStartPositionJ(int j)	Set para redefinir el atributo de la posición en Y del barco.
setOrientation(String orientation)	Set para redefinir el atributo de la posición en Y del barco.
isSunk()	Verifica si el barco se ha quedado sin vidas y establece su estado como hundido mediante un boolean.
updateShip()	Actualiza el estado de las naves.

Para el caso de las clases anteriores no se importó ninguna librería.

Frigate

Destroyer

Submarine

AircraftCarrier

Para el caso de las clases anteriores, es importante tener en cuantas que todas son clases hijas o subclases de la clase Ship, dado que cada una representa un tipo de barco en especial, en general la diferencia entre las mismas, se encuentra en la ejecución del constructor y la cantidad de espacios que ocupa cada barco.

Métodos clase: Frigate	
Nombre	Descripción

Frigate()	Constructor de la clase, ejecuta el constructor de su superclase Ship, indicando un valor para el parámetro size.
------------------	--

Métodos clase: Destroyer	
Nombre	Descripción
Destroyer()	Constructor de la clase, ejecuta el constructor de su superclase Ship, indicando un valor para el parámetro size.

Métodos clase: Submarine	
Nombre	Descripción
Submarine()	Constructor de la clase, ejecuta el constructor de su superclase Ship, indicando un valor para el parámetro size.

Métodos clase: AircraftCarrier	
Nombre	Descripción

AircraftCarrier()	Constructor de la clase, ejecuta el constructor de su superclase Ship, indicando un valor para el parámetro size.
--------------------------	--

Para el caso de las clases anteriores no se utilizaron librerías.

Análisis de Resultados

Objetivos alcanzados:

En el desarrollo del proyecto logramos alcanzar una serie de objetivos planteados que reforzaron los conocimientos adquiridos en clase. Entre ellos se logró mejorar la comprensión sobre el funcionamiento práctico de la herencia y polimorfismo para mejorar el flujo del proyecto.

Objetivo	Cumplido	No Cumpido
Modelado de Clases UML	Si	
Herencia y Polimorfismo	Si	
Uso de UDE para Java	Si	
Programación en java	Si	
Uso de Objetos	Si	
Uso de clases	Si	
Manejo de modificadores de visibilidad	Si	
Instanciación	Si	

Manual de Usuario

Instrucciones de uso

El programa inicia solicitando los nombre de los jugadores, los cuales luego de escribir su nombre deben presionar la tecla enter.

```
Bienvenidos a Battleship
```

```
Jugador 1 por favor introduce tu nombre: █
```

Una vez ingresado los nombre de los jugadores, se procede a colocar los barcos de cada jugador en el tablero. Cada jugador tendrá un total de 9 barcos al iniciar la partida, para colocarlos en el tablero el jugador debe ingresar el número de fila y de columna donde se ubicará la primer casilla del barco, posteriormente deberá ingresar la orientacion hacia la cual se desplegaran las demás casillas.

```
Barcos por colocar: 8
```

```
Este es tu tablero de barcos, Jeison
```

```
  0  1  2  3  4  5  6  7  8  9
0 | ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ | 0
1 | ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ | 1
2 | ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ | 2
3 | ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ | 3
4 | ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ | 4
5 | ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ | 5
6 | ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ | 6
7 | ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ | 7
8 | ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ | 8
9 | ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ | 9
  0  1  2  3  4  5  6  7  8  9
```

```
Vas a colocar un barco Submarine
Tiene un tamaño de 3
```

```
Inserte el número de fila: 2
```

```
Inserte el número de columna: 6
```

```
Ingrese la orientación (UP, DOWN, LEFT, RIGHT): LEFT █
```

Una vez que todas los barcos han sido colocados, se desplegaran los tableros del jugador uno con sus barcos y sus disparos, este deberá ingresar la posición a la que quiere disparar ingresando la fila y la columna y presionar la tecla enter. Recibirá información sobre si su disparo acertó o no y pasará al turno del jugador 2 que deberá seguir los mismos pasos.

Cuando uno de los jugadores se quede sin vidas el juego termina.

Compilacion y ejecucion


<https://repl.it/@KSCeciliano/ProyectoPOO>

Conclusiones Personales

Emmanuel

A nivel personal la experiencia adquirida a través de la realización del presente trabajo, ha sido muy enriquecedora desde distintos aspectos, en primer lugar, el llevar a la practica muchos de los conceptos adquiridos por medio de los contenidos del curso sin duda me ha permitido corroborar el aprendizaje adquirido y enfrentar algunas de las distintas situaciones que son propias de la ejecución de un proyecto en un entorno real.

Por otra parte y concretamente en relación con el tipo de proyecto realizado, entre los puntos altos que a nivel personal me han parecido importantes, destaco el análisis que individualmente como estudiantes de la carrera de ingeniería en computación cada uno hemos tenido que realizar, haciendo uso de las distintas herramientas, aprendidas tanto por medio del curso, como de los demás cursos aprobados o por fuentes externas, lo cual ha derivado resultados y posiciones distintas en relación con la metodología y requerimientos necesarios para la ejecución del proyecto. Lo cual puntualmente a nivel grupal pudo ser percibido, al tratar de definir o entrar la manera más eficiente o mejor de realizar el juego.



Finalmente y como otro de los aspectos altos del presente proyecto, no cabe duda de que el aporte del trabajo en equipo representa un aprendizaje extra, dado que como se ha evidenciado en lo dicho anteriormente, cada persona tiene diferentes formas de percibir el conocimiento y llevarlo a la práctica. Lo anterior puede observarse a través de este tipo de proyectos, que asemejan las experiencias a las que como profesionales estamos expuestos en la cotidianidad. En relación con lo anterior, con seguridad la tarea de unificar las diferentes ideas, formas de pensamiento y distintos puntos de vista han generado un aporte muy valioso para los distintos integrantes del grupo. Adicionalmente el éxito en la elaboración del trabajo ha generado una sensación positiva y satisfacción de que el trabajo realizado ha sido valioso para todos.

Jeison

Durante el desarrollo del proyecto se alcanzó una mayor comprensión y manejo de conceptos fundamentales de java como la herencia, las propiedades y alcance que pueden tener los objetos en la simplificación del diseño de un algoritmo. Implementar estos elementos en el programa me permite tener una perspectiva más sólida sobre el funcionamiento del paradigma de Orientación a Objetos y del lenguaje de programación Java que permitirá un desarrollo más ágil en futuros proyectos, todo esto sumándole también la experiencia transmitida por mis compañeros que fue fundamental para cumplir con los requerimientos solicitados.

Kevin

Mientras realizamos este proyecto logré adentrarme más en el mundo del trabajo en equipo y a tomar en cuenta las ideas de las personas que conformaban el equipo para así enriquecer de buena forma el proyecto y que su solución fuera mejor. El implementar la solución de este proyecto me llevó a entender de una forma más adentrada el paradigma de la orientación de objetos y también un poco más el lenguaje que utilizamos en este, también el uso de herencia, las matrices, arraylist, entre otras cosas. En general esta fue una experiencia enriquecedora en todo punto y se ganó experiencia en todos los ámbitos que este proyecto abarcó, satisfecho con todo el resultado!

Autoevaluación

Criterio #1

Cumplimiento con todas las tareas en el plazo definido por el grupo, fortaleciendo la comunicación y uso de habilidades blandas para lograr realizar un buen trabajo en equipo.

Evaluación individual			Calificación de compañeros		
Integrante	Porcentaje de participación	Comentario	Emmanuel	Kevin	Jeison
Emmanuel	32%	Estoy satisfecho con la organización alcanzada y la repartición de labores.	N/A	9	9
Kevin	32%	Buena participación y organización.	9	N/A	9
Jeison	36%	Satisfecho con la organización y planificación logrados.	9	9	N/A
Total	100%				

Criterio #2

Participación activa en la elaboración del trabajo en general, aporte de observación o comentarios enriquecedores tanto para el trabajo como para la experiencia de los otros miembros del equipo.

Evaluación individual			Calificación de compañeros		
Integrante	Porcentaje de participación	Comentario	Emmanuel	Kevin	Jeison
Emmanuel	40%	Compartir las experiencias de cada uno, me pareció un punto muy alto.	N/A	10	10
Kevin	30%	Cada uno dió de buena forma su punto de vista con respecto al problema y pensamos en conjunto la solución.	10	N/A	10
Jeison	30%	Comentar sobre las posibles soluciones y formas de implementar la solución ayudó.	10	10	N/A
Total	100%				

Desarrollo adecuado de conceptos vistos a lo largo de curso de Programación Orientada a Objetos, dominio de los principios y procesos asociados a dichos conceptos e implementación correcta en el desarrollo del proyecto.

Evaluación individual			Calificación de compañeros		
Integrante	Porcentaje de participación	Comentario	Emmanuel	Kevin	Jeison
Emmanuel	33%	Llevar a la práctica algunos conceptos fue absolutamente enriquecedor.	N/A	10	10
Kevin	34%	Utilizar los conceptos aprendidos en el curso fue una experiencia excelente.	8	N/A	10
Jeison	33%	Implementar los conceptos ayudó a obtener mayor claridad del funcionamiento de muchos conceptos vistos en clase.	9	10	N/A
Total	100%				

Criterio #4

Disposición para realizar correcciones o aceptar sugerencias con el fin de alcanzar una mejor aplicación de los conceptos implicados en los procesos de modelado de software e implementación de código, necesarios para la elaboración del proyecto.

Evaluación individual			Calificación de compañeros		
Integrante	Porcentaje de participación	Comentario	Emmanuel	Kevin	Jeison
Emmanuel	30%	La buena disposición de todos, facilitó el cumplimiento de este criterio.	N/A	10	10
Kevin	35%	Excelente disposición escuchando a los demás y dando ideas.	10	N/A	10
Jeison	35%	La disposición de todos ayudó a implementar una solución eficiente a la solicitado.	10	10	N/A
Total	100%				

Aporte de tiempo y retroalimentación a través de la búsqueda de información en fuentes externas con la finalidad de ampliar el conocimiento en el tema y por ende poder aplicar mejoras en el proyecto.

Evaluación individual			Calificación de compañeros		
Integrante	Porcentaje de participación	Comentario	Emmanuel	Kevin	Jeison
Emmanuel	32%	Los aportes de cada uno en este ámbito, realmente fueron muy útiles.	N/A	10	10
Kevin	36%	Cada uno se informó más con respecto a la posible solución del problema.	9	N/A	10
Jeison	32%	La colaboración de todos simplificar el planteamiento de la solución.	8	10	N/A
Total	100%				

** La escala de calificación indica 1 como la nota inferior o más baja y 10 como la nota excelente.