

Instituto Tecnológico y de Estudios Superiores de Monterrey.

Proyecto Integrador.

Situación Problema: Juegos Cognitivos (Battleship).

Santiago Ramírez Niño - A01665906.

Chiara Bombardieri Balanzá - A01659462.

Gabriel Gutiérrez Guerra - A01660505.

Grupo 416.

Pensamiento Computacional para Ingeniería.

Profesor: Luis Alejandro Arce Sáenz.

Lunes 16 de octubre del 2023.

INTRODUCCIÓN

La informática y computación son ciencias tecnológicas que se encargan del estudio de procesos para almacenar información de manera digital, por medio de máquinas compuestas de hardware y software que facilitan el manejo de datos y sistemas computacionales. Para el desarrollo de nuevos softwares, se emplea la programación; la cual es la rama de la informática que se enfoca en la elaboración de algoritmos y secuencias de código que sigue una serie de instrucciones y pasos ordenados y sistematizados para ejecutar una tarea específica.

Los códigos se elaboran a partir de un lenguaje de programación, que no son más que sistematizaciones para convertir una serie de instrucciones de lenguaje natural a lenguaje máquina, de manera que la computadora reciba instrucciones, las interprete y las ejecute en sus sistemas informáticos.

Una problemática actual que vivimos los seres humanos, es que conforme pasan los años, las personas padecen una reducción de sus capacidades cognitivas, intelectuales y de sus habilidades mentales; el cerebro sufre un desgaste debido a la edad, e incluso puede desarrollar enfermedades crónico-degenerativas como el Alzheimer (caracterizado por la pérdida progresiva de la memoria). No obstante, está científicamente comprobado por especialistas en neurociencia, que los juegos de destreza o de mesa ayudan a ejercitar nuestra memoria y a mantener la mente activa en situaciones que nos hagan pensar y analizar posibles estrategias y dar soluciones a un reto en específico (como lo es un juego cualquiera). En el caso de este proyecto, se aplicaron nuestros conocimientos de programación en el lenguaje de Python, para desarrollar un programa simulador de un juego de mesa convencional de nombre “Battleship”.

Éste es un juego de tipo adivinanza con estrategias, el cual está diseñado para jugarse entre dos usuarios. Tiene como objetivo encontrar los barcos del usuario oponente dentro de un tablero y hundirlos, el usuario que encuentre todos los barcos primero, será el ganador.

OBJETIVO

Para fines de este proyecto, se pretende construir un código que le permita al usuario interactuar con la computadora para jugar Battleship. El juego consiste en un tablero, en el que la computadora coloca barcos en posiciones aleatorias del tablero, y el usuario deberá ingresar coordenadas para verificar si un barco se encuentra ahí, o no. Ésto con el propósito de ejercitar la mente mediante juegos de destreza o memoria, y potencializar nuestras habilidades y capacidades intelectuales con el uso de tecnología.

METODOLOGÍA

En las indicaciones del profesor para la realización de este proyecto se nos pidió que incluyamos ciertos elementos dentro de nuestro código, además de que para que el funcionamiento del programa sea correcto, este tiene ciertos requisitos, los cuales serán explicados a continuación:

- Al inicio del juego se despliega un Menú para que el usuario decida si quiere jugar o no.
- Contar con una *matriz* la cual cree el tablero del usuario, en esta se encontrarán los barcos en posiciones aleatorias.
- Como se mencionó en el inciso anterior, el programa cuenta con una función, la cual coloca los barcos en *posiciones aleatorias*, para que el usuario pueda jugar.

- Además de que los barcos deben de estar en posiciones aleatorias, al inicio de la partida estos deben de estar *escondidos*, para que el usuario pueda encontrarlos uno a uno.
- Al inicio del juego, el programa indica la cantidad de barcos que hay y la cantidad de intentos que tiene el usuario para encontrar los barcos.
- Se le debe de dar la oportunidad al usuario de escoger los cuadrantes en lo que cree que se encuentran los barcos, para esto también se debe de señalar la cantidad de *filas* y *columnas* que existen en el tablero.
- Se debe indicar al usuario la cantidad de intentos que le quedan para encontrar los barcos.
- El juego debe de terminar tanto como cuando el jugador acaba sus turnos, como si encuentra todos los barcos.
- Al final del juego, se le pregunta al usuario si quiere volver a jugar.

DESARROLLO

El juego se realizó con 5 funciones principales y 2 funciones secundarias y subsecuentes a lo largo del código. En estas funciones, se solicita dar comienzo al juego, muestra un mensaje de bienvenida, crea un tablero con posiciones aleatorias de 5 barcos, se pregunta por las posibles posiciones de los mismos, y así sucesivamente hasta completar 10 intentos. A continuación se presenta bloque por bloque en qué consiste cada función:

- I. Función “main”: Es la primera función a la que se llama. No contiene argumentos y comienza con un ciclo repetitivo While, con la condición “True”, por lo que siempre se repetirá a menos que se le incluya una instrucción “break” para terminarlo. Dentro del ciclo, se le pregunta al usuario si desea jugar (opción 1) o no desea jugar (opción 2) este juego, admitiendo única y exclusivamente estos 2 valores numéricos enteros;

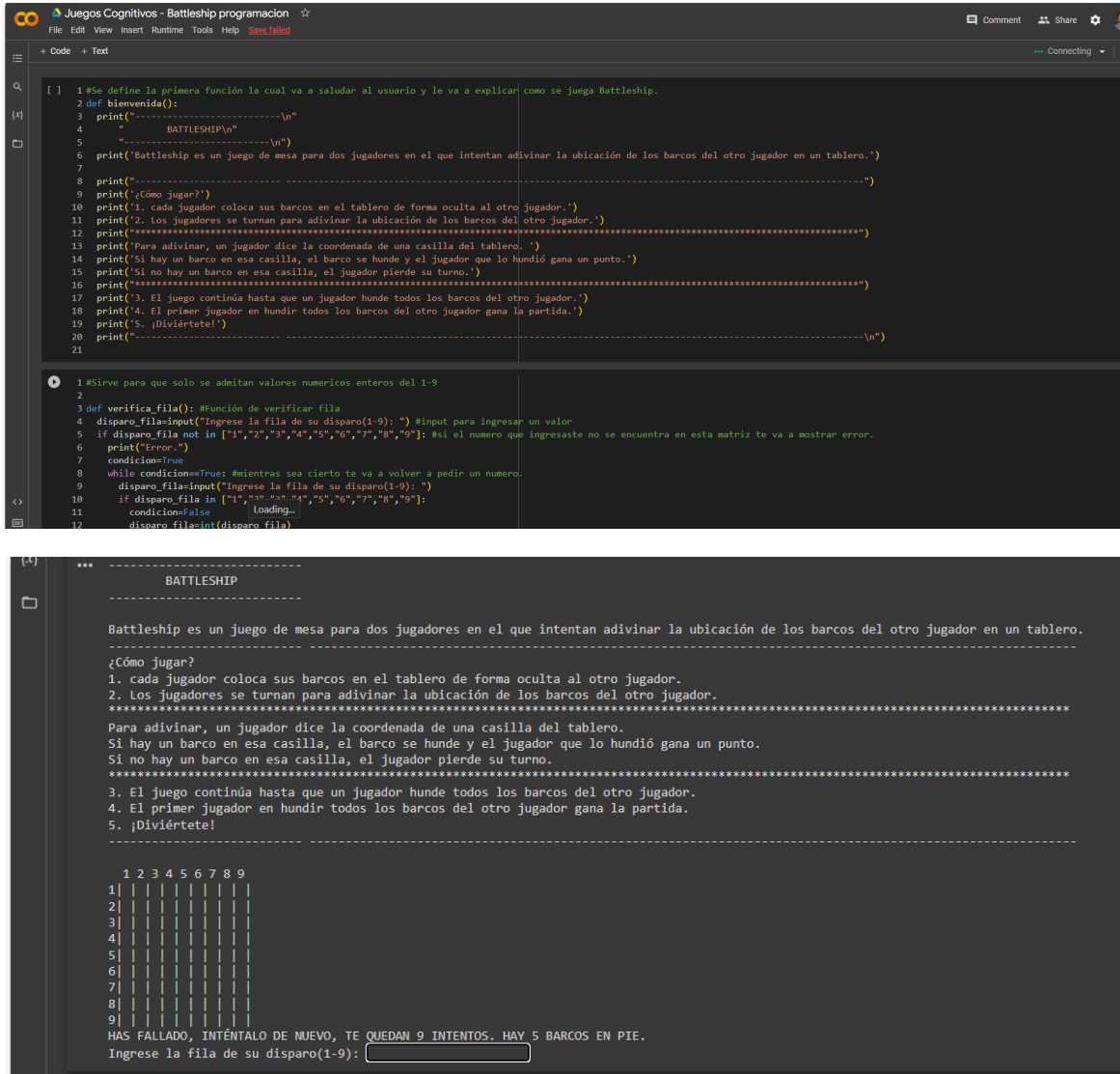
de no ser así, el programa seguirá preguntando hasta que se introduzca alguna de las 2 opciones. Si se introduce el número 2, se proyecta un mensaje de salida del juego y con la instrucción “break” se da fin al ciclo while. Si se introduce el número 1, se llamarán las siguientes 4 funciones y en ese mismo orden.

- II. Función “bienvenida”: Es la primera función subsecuente de la principal y tampoco cuenta con argumentos. La única acción que realiza esta función es imprimir un mensaje de introducción al juego, mostrando instrucciones claras al usuario sobre la manera de jugar Battleship.
- III. Función “tablero”: Se imprime una matriz cuadrada de 10*10, con ayuda de ciclos for y con rangos que van del 0 hasta 9 (son 10 posiciones). A pesar de ser 10 posiciones, en realidad el tablero consiste en un cuadrado de 9*9, ya que la primera fila y la primera columna (fila 0 y columna 0) se utilizan para enumerar a las mismas del 1 al 9, y así facilitar el proceso del juego.
- IV. Función “colocar_barcos”: En esta función, se crea un set() con valores que representan las coordenadas de los barcos dentro del tablero. Para obtener una coordenada se utiliza la librería de python “random”. Con ella utilizamos funciones como random.randint, que obtiene un valor aleatorio, en este caso del 1 al 9, para determinar en qué fila y en qué columna están los barcos, siendo en total 5 barcos.
- V. Función “disparos”: En esta función, se le pide al usuario disparar en las coordenadas del tablero. Tiene 10 intentos en total. Para esto, almacenamos dos números enteros que representan la fila y columna del disparo en una variable, se checa si en esas coordenadas existe un barco, y en caso de que si, se derriba un barco. Todo esto se hace usando el if, while, in. Esta función acaba cuando el jugador se queda sin intentos, o cuando todos los barcos son derribados, terminando así el juego.

- a. Función “verifica_fila”: Ésta función no recibe ningún argumento de entrada y se manda a llamar al declarar la variable “disparo_fila”. Su labor principal es solicitar un valor de entrada, pero con la condición que sea un valor numérico entero del 1 al 9. En caso de que se llegara a ingresar un valor de cadena (string), un valor numérico negativo, un cero o un valor mayor a 9 (es decir, si se llega a ingresar algún valor que no sea 1, 2, 3, 4, 5, 6, 7, 8 o 9), se imprime un mensaje de error y vuelve a solicitar un valor para la fila, y lo seguirá haciendo (con ayuda de un ciclo while) hasta que se ingrese un valor válido.
- b. Función “verifica_columna”: La función tampoco recibe ningún argumento de entrada y se manda a llamar al declarar la variable “disparo_columna”, misma a la que se le asignará el valor de retorno de esta función. Su propósito es exactamente el mismo que el de la función “verifica_fila”, y trabaja de la misma manera, pero ahora para ingresar un valor de disparo hacia una columna, y así, junto con la fila, se forma una coordenada (fila, columna) para realizar las comparaciones pertinentes y desarrollar el juego en sí, gracias a las funciones “colocar_barcos” y “disparos” mencionadas previamente.

RESULTADOS

Gracias al aprendizaje que obtuvimos durante este curso, pudimos diseñar un código para que el usuario sea capaz de jugar el juego de mesa Battleship contra su computadora, esto a través de diversas funciones y líneas de código las cuales ya se explicaron previamente en este documento.



```

1 #Se define la primera función la cual va a saludar al usuario y le va a explicar como se juega Battleship.
2 def bienvenida():
3     print("-----\n")
4     print("BATTLESHIP\n")
5     print("-----\n")
6     print("Battleship es un juego de mesa para dos jugadores en el que intentan adivinar la ubicación de los barcos del otro jugador en un tablero.")
7
8     print("-----")
9     print("¿Cómo jugar?")
10    print("1. cada jugador coloca sus barcos en el tablero de forma oculta al otro jugador.")
11    print("2. Los jugadores se turnan para adivinar la ubicación de los barcos del otro jugador.")
12    print("-----")
13    print("Para adivinar, un jugador dice la coordenada de una casilla del tablero.")
14    print("Si hay un barco en esa casilla, el barco se hunde y el jugador que lo hundió gana un punto.")
15    print("Si no hay un barco en esa casilla, el jugador pierde su turno.")
16    print("-----")
17    print("3. El juego continúa hasta que un jugador hunde todos los barcos del otro jugador.")
18    print("4. El primer jugador en hundir todos los barcos del otro jugador gana la partida.")
19    print("5. ¡Diviértete!")
20    print("-----\n")
21
22
23 1 #Sirve para que solo se admitan valores numericos enteros del 1-9
24
25 def verifica_fila(): #Función de verificar fila
26     disparo_fila=input("Ingrese la fila de su disparo(1-9): ") #input para ingresar un valor
27     if disparo_fila not in ["1","2","3","4","5","6","7","8","9"]: #si el numero que ingresaste no se encuentra en esta matriz te va a mostrar error.
28         print("Error.")
29         condicion=True
30         while condicion=True: #mientras sea cierto te va a volver a pedir un numero.
31             disparo_fila=input("Ingrese la fila de su disparo(1-9): ")
32             if disparo_fila in ["1","2","3","4","5","6","7","8","9"]:
33                 condicion=False
34                 loading...
35             disparo_fila=int(disparo_fila)

```

```

...
BATTLESHIP
-----

Battleship es un juego de mesa para dos jugadores en el que intentan adivinar la ubicación de los barcos del otro jugador en un tablero.

¿Cómo jugar?
1. cada jugador coloca sus barcos en el tablero de forma oculta al otro jugador.
2. Los jugadores se turnan para adivinar la ubicación de los barcos del otro jugador.
-----
Para adivinar, un jugador dice la coordenada de una casilla del tablero.
Si hay un barco en esa casilla, el barco se hunde y el jugador que lo hundió gana un punto.
Si no hay un barco en esa casilla, el jugador pierde su turno.
-----
3. El juego continúa hasta que un jugador hunde todos los barcos del otro jugador.
4. El primer jugador en hundir todos los barcos del otro jugador gana la partida.
5. ¡Diviértete!
-----

  1 2 3 4 5 6 7 8 9
1| | | | | | | | |
2| | | | | | | | |
3| | | | | | | | |
4| | | | | | | | |
5| | | | | | | | |
6| | | | | | | | |
7| | | | | | | | |
8| | | | | | | | |
9| | | | | | | | |
HAS FALLADO, INTÉNTALO DE NUEVO, TE QUEDAN 9 INTENTOS. HAY 5 BARCOS EN PIE.
Ingrese la fila de su disparo(1-9): 

```

CONCLUSIONES

Nos gustaría concluir este trabajo diciendo que fue una grata experiencia el haber codificado este juego de mesa, ya que nos ayudó en poner en práctica las habilidades y enseñanzas vistas en clase, además de que recalca nuestro punto de que las herramientas vinculadas con la tecnología no solo son indispensables para nuestras carreras, si no que también nos ayudan a tener un mayor conocimiento y comprensión de cómo funciona el mundo a nuestro alrededor. Aún quedan varias cosas que aprender y mejorar, pero fue de suma importancia el aprender las bases de la programación durante estas 10 semanas de curso.

REFLEXIONES

La presente reflexión fue realizada de manera grupal y colaborativa entre los 3 integrantes del equipo. A nuestro parecer, el presente trabajo fue una excelente propuesta de proyecto para integrar todos nuestros conocimientos adquiridos durante el semestre en la materia de Pensamiento Computacional para Ingeniería; conocimientos que sin duda son básicos para nuestra formación como futuros ingenieros. Pudimos demostrar que este código ayuda a resolver una problemática (o al menos a prevenirla), ya que le podemos dar un enfoque social al aplicar este código y orientarlo a una aplicación móvil y que este juego de Battleship pueda ser jugado por personas de la tercera edad, y así mantenerlos mentalmente activos al ejercitar su destreza y memoria. La programación está revolucionando la manera en la que funciona el



mundo en la actualidad y la podemos aplicar a muchas situaciones más: desde el ámbito económico, financiero, educativo, político, legal, científico y hasta lo podríamos orientar a un robot o máquina autónoma que reciba ciertas instrucciones para realizar una tarea cansada,

repetitiva o peligrosa para los seres humanos.

Aún hay muchos aspectos por mejorar, pero sin duda la tecnología va encaminada a ser una herramienta esencial en nuestro día a día (de hecho, el día de hoy ya lo es). Se sigue trabajando en el presente para hacer uso de tecnologías cada vez más seguras, prácticas y eficientes. En este proyecto integrador pudimos recopilar cada una de las lecciones de clase de los últimos 2 periodos, que va desde la declaración de variables, ciclos, condiciones, funciones y hasta listas o matrices. No tenemos la menor duda que con la adquisición de más aprendizajes, podremos elaborar programas mucho más sofisticados que sigan contribuyendo a la solución de diversas situaciones problemáticas en la vida cotidiana.

REFERENCIAS.

- Parzibyte. (2021). Batalla naval en Python – programación de juego. *Parzibyte's blog*.
<https://parzibyte.me/blog/2021/12/21/batalla-naval-python-programacion-juego/#:~:text=Se%20trata%20del%20juego%20Battleship%2C%20juego%20de%20los.c%3%B3mo%20jugarlo%20y%20d%C3%B3nde%20descargar%20el%20c%C3%B3digo%20fuente.>
- Ejemplos de for en Python usando listas, split y join - Convertir una lista en cadena. (2020, June 21). Naps Tecnología y educación. Retrieved October 15, 2023, from
<https://naps.com.mx/blog/ejemplos-de-for-en-python-usando-listas-split/>
- Instrucciones del juego "Battleship". (n.d.). aboutespanol. Retrieved October 15, 2023, from <https://www.aboutespanol.com/battleship-batalla-naval-2077614>
- La importancia de la programación en el mundo digital actual: ¿por qué deberías aprender a programar? (2023, June 13). Corporación Informática. Retrieved October 15, 2023, from
<https://corporacioninformatica.com/la-importancia-de-la-programacion-en-el-mundo-digital-actual-por-que-deberias-aprender-a-programar/>
- Battleship programacion ayuda help - Ayuda de programación. (2020, October 21). The freeCodeCamp Forum. Retrieved October 15, 2023, from
<https://forum.freecodecamp.org/t/battleship-programacion-ayuda-help/426532>