

**INFORME PRIMER PARCIAL (PROYECTO SUDOKU)**



**PROGRAMACIÓN (G02)**

**DEYTON RIASCOS ORTIZ**

**2246208**

**DANA ISABELLA MOSQUERA MORQUERA**

**2245975**

**SAMUEL IZQUIERDO BONILLA**

**2246993**

**UNIVERSIDAD AUTÓNOMA DE OCCIDENTE**

**FACULTAD DE INGENIERÍA**

**PROGRAMA INGENIERÍA DE DATOS E INTELIGENCIA ARTIFICIAL**

**SANTIAGO DE CALI**

**2025**

## INFORME PROYECTO-1

### Objetivos

- Familiarizar al estudiante con las fases de desarrollo de una solución software a un problema planteado: análisis, pseudocódigo, codificación, pruebas, depuración y documentación.
- Comprender el uso y la sintaxis de funciones, estructuras de decisión y repetitivas en lenguaje de programación Python.

### Metodología

- El proyecto debe ser desarrollado en grupos de hasta cuatro personas (mínimo tres personas).
- Para cada problema los entregables son:
  1. Análisis del problema.
  2. Algoritmo en pseudocódigo.
  3. Implementación funcional en Python.
  4. Evidencias de ejecución del programa mediante capturas de pantalla.

### Entregables

- (30%) Se debe entregar un informe en PDF con los puntos 1,2 y 4 de la metodología.
- (40%) Código en Python debidamente comentado (notebook de Jupyter) y subido a GitHub.
- (30%) Sustentación: jueves 06 de marzo de 2025 en el horario de clase.

### Problema 3.

Diseñe e implemente una aplicación que permita jugar una versión 'Pythonica' del juego batalla Sudoku, con las siguientes especificaciones:

- Se deben tener almacenados al menos 5 tableros.
- El nivel de los tableros debe ser fácil.
- Al inicio del juego se debe escoger de forma aleatoria el tablero a resolver.
- El usuario debe ingresar el número y las coordenadas donde lo quiere ubicar. También, se debe indicar si esa casilla ya está ocupada.
- Al final, se debe indicar si ganó.
- Ustedes son libres de indicar al jugador si el número ubicado es correcto o no o de brindar algún tipo de ayuda o sugerencia.

La aplicación debe estar construida usando funciones.

Ustedes son libres de escoger los elementos gráficos y de jugabilidad que consideren adecuados. También son libres de elegir las soluciones que no se encuentren definidas dentro de las especificaciones.

## **Análisis del problema**

### **Descripción del problema:**

El problema consiste en desarrollar un juego de Sudoku en la terminal donde el usuario pueda jugar resolviendo un tablero generado aleatoriamente. Se debe garantizar que el tablero sea válido, es decir, que cumpla con las reglas del Sudoku.

### **Entradas:**

- Elección de la dificultad del juego.
- Coordenadas (fila, columna) donde el usuario quiere ingresar un número.
- Número que el usuario quiere colocar en la celda seleccionada.

### **Procesos:**

1. **Generación del tablero completo:**
  - Crear un tablero vacío (9x9).
  - Llenarlo siguiendo las reglas del Sudoku usando **backtracking**.
2. **Eliminación de números para crear el desafío:**
  - Dependiendo de la dificultad elegida, eliminar un porcentaje de números.
3. **Interacción con el usuario:**
  - Mostrar el tablero en pantalla con el formato adecuado.
  - Pedir coordenadas y número al usuario.
  - Validar si el número ingresado es correcto según las reglas del Sudoku.
  - Actualizar el tablero con la nueva entrada.
  - Verificar si el Sudoku está completo.
4. **Finalización del juego:**
  - Si el usuario completa el tablero correctamente, mostrar un mensaje de felicitación.

### **Salidas:**

- Tablero de Sudoku con el formato adecuado.
- Mensajes de error si el usuario ingresa un número inválido.
- Mensaje de éxito cuando el usuario complete el Sudoku.

## Algoritmo en pseudocódigo

### Inicio

Limpiar pantalla

Mostrar título del juego

Definir un tablero vacío de 9x9

Llenar el tablero usando backtracking

Mostrar opciones de dificultad

Leer dificultad seleccionada

Eliminar números del tablero según la dificultad

Mientras el tablero no esté completo hacer

    Limpiar pantalla

    Mostrar reglas del Sudoku

    Mostrar título

    Imprimir el tablero

    Pedir al usuario la fila donde quiere ingresar un número

    Pedir al usuario la columna donde quiere ingresar un número

    Si la celda ya tiene un número entonces

        Mostrar mensaje de error

        Continuar con la siguiente iteración

    Pedir al usuario el número a colocar en la celda

    Si el número es válido según las reglas del Sudoku entonces

        Colocar el número en la celda

    Sino

        Mostrar mensaje de error

Limpiar pantalla

Mostrar título

Imprimir tablero final

Mostrar mensaje de éxito

### Fin

# Implementación funcional en Python

```
1 import sys, random
2 import os # Importa el módulo para interactuar con el sistema operativo
3 import random # Importa la librería random para generar números aleatorios
4
5 def limpiar_pantalla():
6     os.system('cls' if os.name == 'nt' else 'clear') # Limpia la pantalla según el sistema operativo
7
8 def mostrar_titulo():
9     """Muestra el título del juego."""
10    print("\n" + 75 * "=") # Muestra una línea de separación
11    print(" " + 30 * " ", "JUEGO DE SUDOKU" + " " + 30 * " ") # Muestra el título del juego centrado
12    print("\n" + 75 * "=") # Muestra otra línea de separación
13
14 def mostrar_reglas():
15     """Muestra las reglas del juego Sudoku en la consola"""
16    print("\n" + 75 * "=") # Muestra una línea de separación
17    print(" " + 20 * " ", "REGLAS DEL SUDOKU" + " " + 20 * " ") # Muestra un encabezado de las reglas con salto de línea inicial
18    print("\n" + 75 * "=") # Muestra otra línea de separación
19    print("1. Se genera un tablero de Sudoku 9x9 con algunos espacios vacíos.") # Explica la primera regla
20    print("2. Debes rellenar los espacios vacíos con números del 1 al 9.") # Explica la segunda regla
21    print("3. No puedes repetir números en la misma fila, columna o subcuadrícula 3x3.") # Explica la tercera regla
22    print("4. Ganas cuando completes el tablero correctamente.") # Explica la cuarta regla
23    print("\n" + 75 * "=") # Muestra una línea divisoria para finalizar la sección de reglas
24
25 def imprimir_tablero(tablero):
26     """Imprime el tablero de Sudoku con formato estructurado"""
27    print("\n F 0 1 2 | 3 4 5 | 6 7 8 ") # Imprime el encabezado de las columnas
28    print("C -----") # Imprime la parte superior del tablero
29
30    for fila in range(9): # Itera sobre cada fila del tablero
31        if fila % 3 == 0: # Imprime un separador horizontal cada 3 filas
32            print("C |-----|") # Separador de bloques
33
34        print(f"fila {fila} ", end=" ") # Imprime el número de la fila y el borde izquierdo del tablero
35        for col in range(9): # Itera sobre cada columna de la fila
36            if col % 3 == 0: # Imprime un separador vertical cada 3 columnas
37                print(" | ", end=" ") # Separador vertical de bloques
38
39            print(tablero[fila][col] if tablero[fila][col] != 0 else ".", end=" ") # Imprime el número o un punto si la celda está vacía
40
41    print("\n") # Imprime el borde derecho del tablero
42
43    print("L -----") # Imprime la parte inferior del tablero
44
45 def generar_subtablero(tablero, fila, col, num):
46     """Verifica si el número puede colocarse en la posición dada"""
47    if num in tablero[fila]: # Verifica si el número ya está en la fila
48        return False
49
50    for i in range(9): # Verifica si el número ya está en la columna
51        if tablero[i][col] == num:
52            return False
53
54    inicio_fila, inicio_col = (fila // 3) * 3, (col // 3) * 3 # Calcula el inicio de la subcuadrícula 3x3
55    for i in range(3): # Itera sobre las filas de la subcuadrícula
56        for j in range(3): # Itera sobre las columnas de la subcuadrícula
57            if tablero[inicio_fila + i][inicio_col + j] == num: # Verifica si el número ya está en la subcuadrícula
58                return False
59
60    return True # Retorna True si el número puede colocarse en la posición dada
61
62 def generar_tablero(tablero):
63     """Genera el tablero con ceros"""
64    for fila in tablero: # Itera sobre cada fila del tablero
65        if 0 in fila: # Verifica si hay alguna celda vacía (con valor 0)
66            return False
67    return True # Retorna True si el tablero está lleno
68
69 def resolver_sudoku():
70     """Resuelve el Sudoku con backtracking"""
71     def resolver_sudoku(tablero):
72         """Resuelve el Sudoku con backtracking"""
73         for fila in range(9): # Itera sobre cada fila del tablero
74             for col in range(9): # Itera sobre cada columna de la fila
75                 if tablero[fila][col] == 0: # Verifica si la celda está vacía
76                     for num in range(1, 10): # Genera números del 1 al 9
77                         if generar_subtablero(tablero, fila, col, num): # Verifica si el número puede colocarse
78                             tablero[fila][col] = num # Coloca el número en la celda
79                             if resolver_sudoku(tablero): # Llama recursivamente para resolver el resto del tablero
80                                 return True
81                             tablero[fila][col] = 0 # Si no se puede resolver, vacía la celda y prueba con otro número
82                     return False # Retorna False si no se puede colocar ningún número en la celda
83         return True # Retorna True si el tablero está resuelto
84
85     tablero = [[0 for _ in range(9)] for _ in range(9)] # Inicializa un tablero vacío (9x9) con ceros
86     resolver_sudoku(tablero) # Llama a la función para resolver el Sudoku
87     return tablero # Retorna el tablero resuelto
88
89 def eliminar_numeros(tablero, porcentaje):
90     """Elimina números del tablero según la dificultad"""
91     num_casillas_a_vaciar = int(9 * porcentaje) # Calcula el número de celdas a vaciar según el porcentaje
92     casillas = [(fila, col) for fila in range(9) for col in range(9)] # Crea una lista con todas las posiciones del tablero
93     random.shuffle(casillas) # Mezcla aleatoriamente las posiciones
94
95     for _ in range(num_casillas_a_vaciar): # Itera sobre el número de celdas a vaciar
96         fila, col = casillas.pop() # Toma una posición aleatoria de la lista
97         tablero[fila][col] = 0 # Vacía la celda en la posición seleccionada
98
99 def generar_dificultad():
100     """Permite seleccionar la dificultad"""
101     niveles = {
102         "Fácil": 3/81, # Define el nivel "Fácil" con su porcentaje de celdas vacías
103         "Medio": 0.30, # Define el nivel "Medio" con su porcentaje de celdas vacías
104         "Difícil": 0.65, # Define el nivel "Difícil" con su porcentaje de celdas vacías
105         "Muy Difícil": 0.70, # Define el nivel "Muy Difícil" con su porcentaje de celdas vacías
106     }
107
108     print("\nSelecciona la dificultad:") # Imprime el mensaje para seleccionar la dificultad
109     for clave, (nombre, _) in niveles.items(): # Itera sobre los niveles de dificultad
110         print(f"{clave} ({nombre})") # Imprime cada nivel de dificultad con su clave
111
112     while True: # Bucle infinito para solicitar una opción válida
113         opcion = input("Opción: ") # Solicita la opción al usuario
114         if opcion in niveles: # Verifica si la opción es válida
115             return niveles[opcion][1] # Retorna el porcentaje correspondiente al nivel seleccionado
116         print("Opción no válida, intenta nuevamente.") # Imprime un mensaje de error si la opción no es válida
117
118 def obtener_dificultad(mensaje, min_val, max_val):
119     """Obtiene un número al usuario y maneja errores"""
120     while True: # Bucle infinito para solicitar un número válido
121         try:
122             num = int(input(mensaje)) # Solicita el número al usuario y lo convierte a entero
123             if min_val <= num <= max_val: # Verifica si el número está dentro del rango permitido
124                 return num # Retorna el número si es válido
125             print(f"Error: Ingresa un número entre {min_val} y {max_val}.") # Imprime un mensaje de error si el número está fuera del rango
126         except ValueError:
127             print(f"Error: Debes ingresar un número válido.") # Imprime un mensaje de error si la entrada no es un número válido
128
129 def jugar_sudoku():
130     """Función principal para jugar Sudoku"""
131     limpiar_pantalla() # Limpia la pantalla
132     mostrar_titulo() # Muestra el título del juego
133     mostrar_reglas() # Muestra las reglas del juego
134     imprimir_tablero(tablero) # Imprime el tablero actual
135
136     print("\nIngresa las coordenadas y el número a colocar:") # Solicita al usuario ingresar las coordenadas y el número
137
138     while not generar_tablero(tablero): # Bucle que se ejecuta mientras el tablero no está lleno
139         limpiar_pantalla() # Limpia la pantalla
140         mostrar_reglas() # Muestra las reglas del juego
141         mostrar_titulo() # Muestra el título del juego
142         imprimir_tablero(tablero) # Imprime el tablero actual
143
144         print("\nIngresa las coordenadas y el número a colocar:") # Solicita al usuario ingresar las coordenadas y el número
145
146         fila = obtener_dificultad("Ingresa la fila (0-8): ", 0, 8) # Solicita la fila al usuario
147         col = obtener_dificultad("Ingresa la columna (0-8): ", 0, 8) # Solicita la columna al usuario
148
149         if tablero[fila][col] != 0: # Verifica si la celda ya tiene un número
150             print(f"Error: La celda ya tiene un número. Intenta en otra posición.") # Imprime un mensaje de error
151             continue # Continúa con la siguiente iteración del bucle
152
153         num = obtener_dificultad("Ingresa un número (1-9): ", 1, 9) # Solicita el número al usuario
154
155         if generar_subtablero(tablero, fila, col, num): # Verifica si el número puede colocarse en la posición dada
156             tablero[fila][col] = num # Coloca el número en la celda
157         else:
158             print(f"Error: Número no válido en esta posición. Intenta de nuevo.") # Imprime un mensaje de error
159
160     limpiar_pantalla() # Limpia la pantalla
161     mostrar_titulo() # Muestra el título del juego
162     imprimir_tablero(tablero) # Imprime el tablero completo
163     print("\n¡Felicitaciones! Has completado el Sudoku correctamente.") # Imprime un mensaje de felicitación
164
165 def main():
166     jugar_sudoku() # Llama a la función principal para jugar Sudoku
167
168 if __name__ == "__main__":
169     main() # Ejecuta la función main si el script se ejecuta directamente
```

## Evidencias de ejecución del programa mediante capturas de pantalla

Inicio Juego:

```
0s MEM: 45% (13/30GB)
python .\sudoku.py_
```

```
JUEGO DE SUDOKU

Selecciona la dificultad:
1. Súper fácil
2. Fácil
3. Medio
4. Difícil
Opción: 3
```

```
REGLAS DEL SUDOKU

1. Se genera un tablero de Sudoku 9x9 con algunos espacios vacíos.
2. Debes rellenar los espacios vacíos con números del 1 al 9.
3. No puedes repetir números en la misma fila, columna o subcuadrícula 3x3.
4. Ganas cuando completas el tablero correctamente.
5. Guía de ingreso de datos C para columnas y F para Filas

JUEGO DE SUDOKU

C 0 1 2 | 3 4 5 | 6 7 8
F
0 | 1 . 3 | 4 . . | . . 9 |
1 | 4 5 6 | . 8 9 | . . 3 |
2 | 7 . 9 | . 2 3 | 4 . . |
3 | . . 4 | . 6 . | . . 7 |
4 | 3 6 5 | 8 . 7 | . 1 . |
5 | 8 9 . | 2 . 4 | . 6 . |
6 | 5 3 . | 6 . 2 | 9 7 8 |
7 | 6 4 2 | 9 7 . | . 3 . |
8 | . . . | 5 3 . | . 4 . |

Ingrese las coordenadas y el número a colocar:
Ingresar la fila (0-8): _
```

## Validación de datos ingresados:

```
=====
                        REGLAS DEL SUDOKU
=====
1. Se genera un tablero de Sudoku 9x9 con algunos espacios vacíos.
2. Debes rellenar los espacios vacios con números del 1 al 9.
3. No puedes repetir números en la misma fila, columna o subcuadrícula 3x3.
4. Ganas cuando completas el tablero correctamente.
5. Guía de ingreso de datos C para columnas y F para Filas
=====

                        JUEGO DE SUDOKU
=====

  C 0 1 2 | 3 4 5 | 6 7 8
F
0 | 1 . 3 | 4 . . | . . 9 |
1 | 4 5 6 | . 8 9 | . . 3 |
2 | 7 . 9 | . 2 3 | 4 . . |
  +-----+-----+
3 | . . 4 | . 6 . | . . 7 |
4 | 3 6 5 | 8 . 7 | . 1 . |
5 | 8 9 . | 2 . 4 | . 6 . |
  +-----+-----+
6 | 5 3 . | 6 . 2 | 9 7 8 |
7 | 6 4 2 | 9 7 . | . 3 . |
8 | . . . | 5 3 . | . 4 . |

Ingrese las coordenadas y el número a colocar:
Ingresar la fila (0-8): -1
Error: Ingresar un número entre 0 y 8.
Ingresar la fila (0-8): 0
Ingresar la columna (0-8): -1
Error: Ingresar un número entre 0 y 8.
Ingresar la columna (0-8): 1
Ingresar un número (1-9): -1
Error: Ingresar un número entre 1 y 9.
Ingresar un número (1-9): 2_
```

## Ingreso de ingresados:

```
=====
                        REGLAS DEL SUDOKU
=====
1. Se genera un tablero de Sudoku 9x9 con algunos espacios vacíos.
2. Debes rellenar los espacios vacios con números del 1 al 9.
3. No puedes repetir números en la misma fila, columna o subcuadrícula 3x3.
4. Ganas cuando completas el tablero correctamente.
5. Guía de ingreso de datos C para columnas y F para Filas
=====

                        JUEGO DE SUDOKU
=====

  C 0 1 2 | 3 4 5 | 6 7 8
F
0 | 1 2 3 | 4 . . | . . 9 |
1 | 4 5 6 | . 8 9 | . . 3 |
2 | 7 . 9 | . 2 3 | 4 . . |
  +-----+-----+
3 | . . 4 | . 6 . | . . 7 |
4 | 3 6 5 | 8 . 7 | . 1 . |
5 | 8 9 . | 2 . 4 | . 6 . |
  +-----+-----+
6 | 5 3 . | 6 . 2 | 9 7 8 |
7 | 6 4 2 | 9 7 . | . 3 . |
8 | . . . | 5 3 . | . 4 . |

Ingrese las coordenadas y el número a colocar:
Ingresar la fila (0-8): _
```

Finalizar Juego:

JUEGO DE SUDOKU

C	0	1	2	3	4	5	6	7	8
F	1	2	3	4	5	6	7	8	9
0	4	5	6	7	8	9	1	2	3
1	7	8	9	1	2	3	4	5	6
2	2	1	4	3	6	5	8	9	7
3	3	6	5	8	9	7	2	1	4
4	8	9	7	2	1	4	3	6	5
5	5	3	1	6	4	2	9	7	8
6	6	4	2	9	7	8	5	3	1
7	9	7	8	5	3	1	6	4	2
8									

¡Felicidades! Has completado el Sudoku correctamente.