



UFRJ

UNIVERSIDADE FEDERAL
DO RIO DE JANEIRO

Politécnica
UFRJ

Trabalho final da disciplina Computação I

Tema: Campo Minado em Python

Aluna: Luísa Oliveira Gonçalves
luisagoncalves.20242@poli.ufrj.br

Professores: Fernanda Duarte Vilela Reis de Oliveira
Fábio Marujo da Silva







- ❖ **Funções Principais do código do jogo**
 - **função jogada, criar campo, mostrar tabuleiro, contar bombas, revelar células, verificar vitória**
- ❖ **Funcionalidades adicionais**
 - **salvar e carregar jogo, registrar e mostrar recordes**
- ❖ **Demonstração do jogo no terminal**
 - **opções do menu, novo jogo, dificuldades, salvamento, jogos salvos, recordes e sair**





Função criar_campo

- cria uma matriz preenchida com símbolos de células vazias “” dependendo da dificuldade escolhida
- coloca bombas aleatoriamente na matriz

```
#Cria o campo baseado na dificuldade escolhida
def criar_campo(dificuldade):
    if dificuldade == "fácil": #Cria uma matriz 5x5, esconde 5 bombas
        tamanho = 5
        bombas = 5
    elif dificuldade == "normal": #Cria uma matriz 8x8, esconde 12 bombas
        tamanho = 8
        bombas = 12
    elif dificuldade == "difícil": #Cria uma matriz 10x10, esconde 20 bombas
        tamanho = 10
        bombas = 20
    else:
        raise ValueError("Dificuldade inválida")

    jogo = [["" for i in range(tamanho)] for i in range(tamanho)]

    #Coloca bombas aleatoriamente na matriz
    bombas_colocadas = 0 #Contador

    while bombas_colocadas < bombas:
        linha = random.randint(0, tamanho - 1)
        coluna = random.randint(0, tamanho - 1)
        if jogo[linha][coluna] != "":
            jogo[linha][coluna] = "
            bombas_colocadas += 1
    return jogo, tamanho
```

Função criar_campo

Funções Principais do código



```
jogo = [{" " } for i in range(tamanho)] for i  
in range(tamanho)]
```

- Cria uma matriz bidimensional (lista de listas) utilizando compreensão de listas (list comprehension)
- O primeiro `for i in range(tamanho)` cria as linhas da matriz
- O segundo `for i in range(tamanho)` preenche cada linha com (tamanho) de células vazias

Frames

Objects

Global frame

criar_campo

function

criar_campo(dificuldade)

criar_campo

dificuldade "fácil"

tamanho 5

bombas 5

<listcomp>

.0

tamanho 5

i 4

Return
value

range_iterator instance

list

0

1

2

3

4

list

0

1

2

3

4

list

0

1

2

3

4

list

0

1

2

3

4

list

0

1

2

3

4

list

0

1

2

3

4



#Imprime o tabuleiro e esconde as minas

```
def mostra_tabuleiro(jogo, tamanho, revelar_minas=False, nome_jogador=""):  
    print(f"Jogador: {nome_jogador}")
```

#Cabeçalho

```
print(" ", end=" ")
```

```
for i in range(1, tamanho + 1):
```

```
    print(f"{chr(64 + i):>2} ", end="") #Alinha as letras das colunas
```

```
print(" ")
```

```
for i in range(tamanho):
```

```
    print(f"{i + 1:2} ", end="") #Alinha os números das linhas
```

```
    for j in range(tamanho):
```

```
        #Garantir que todos os símbolos ocupem o mesmo espaço
```

```
        if jogo[i][j] == "●":
```

```
            if revelar_minas:
```

```
                print(f"'●':>2} ", end="")
```

```
            else:
```

```
                print(f"'□':>2} ", end="")
```

```
        else:
```

```
            #Para os números e células vazias, também garantir o alinhamento
```

```
            print(f"{jogo[i][j]:>2} ", end="")
```

```
print("")
```

Função mostra_tabuleiro

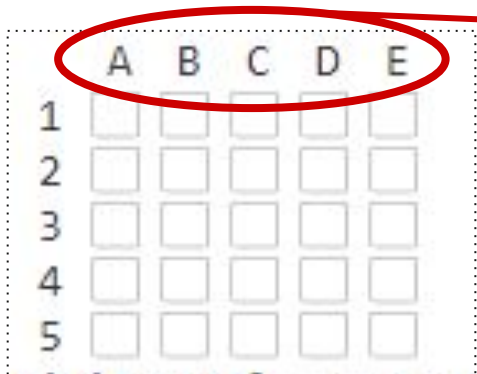
- **imprime o tabuleiro do jogo com o cabeçalho alfabético para as colunas e numérico para as linhas**
- **opcionalmente revela ou esconde as minas no argumento do tipo booleano "revelar_minas"**

Função mostra_tabuleiro

Funções Principais do código



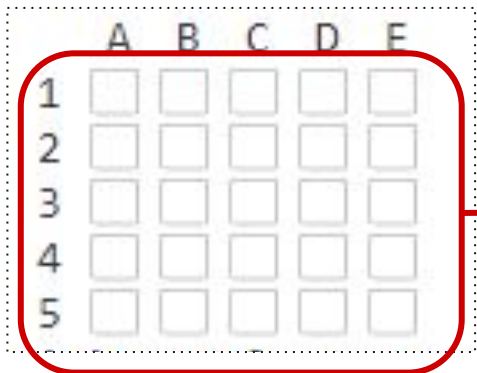
Politécnica
UFRJ



	A	B	C	D	E
1					
2					
3					
4					
5					

```
for i in range(1, tamanho + 1):  
    print(f"{chr(64 + i):>2} ", end="")  
print("")
```

→ Caso revelar_minas = False

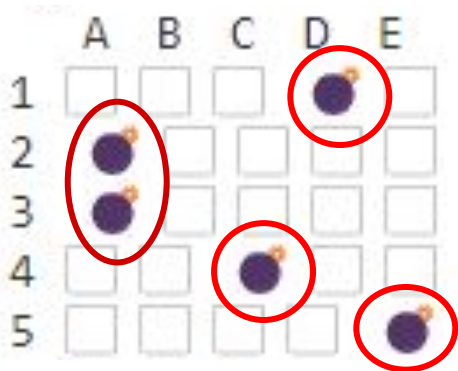


	A	B	C	D	E
1					
2					
3					
4					
5					

```
for i in range(tamanho):  
    print(f"{i + 1:2} ", end="")  
    for j in range(tamanho):  
        if jogo[i][j] == "💣":  
            if revelar_minas:  
                print(f"'💣':>2} ", end="")  
            else:  
                print(f"'⬜':>2} ", end="")  
        else:  
            print(f"{jogo[i][j]:>2} ", end="")  
    print("")
```



Caso revelar_minas = **True**



- Sem o **Booleano**, seria necessário escrever duas funções separadas para cada situação, o que levaria a um código menos flexível.
- Utilizando o argumento do tipo **Booleano**, é fácil controlar o comportamento da função com uma simples alteração de **True** ou **False**. Sem precisar reestruturar o código inteiro.
- Isso torna o código mais simples e menos propenso a erros, pois a mesma lógica principal é aplicada, com apenas um pequeno ajuste.

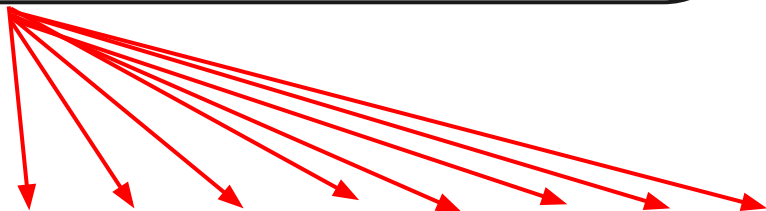


- ❖ **Função conta_bombas**
 - **conta quantas bombas existem nas adjacências de uma célula específica, a contagem é feita utilizando o laço de repetição **for**, que itera sobre uma lista de tuplas.**

```
#Conta as bombas que tem nas adjacências
def conta_bombas(jogo, linha, coluna, tamanho):
    direções = [(-1, -1), (-1, 0), (-1, 1), (0, -1), (0, 1), (1, -1), (1, 0), (1, 1)]
    bombas_ao_redor = 0
    for d in direções:
        nova_linha, nova_coluna = linha + d[0], coluna + d[1]
        if 0 <= nova_linha < tamanho and 0 <= nova_coluna < tamanho:
            if jogo[nova_linha][nova_coluna] == "💣":
                bombas_ao_redor += 1
    return bombas_ao_redor
```


Função conta_bombas

```
bombas_ao_redor = 0
for d in direções:
    nova_linha, nova_coluna = linha + d[0], coluna + d[1]
    if 0 <= nova_linha < tamanho and 0 <= nova_coluna <
tamanho:
        if jogo[nova_linha][nova_coluna] == "💣":
            bombas_ao_redor += 1
return bombas_ao_redor
```



```
direções = [(-1, -1), (-1, 0), (-1, 1), (0, -1), (0, 1), (1, -1), (1, 0), (1, 1)]
```

Funções Principais do código



- Argumentos:
 - ◆ jogo: a matriz do campo
 - ◆ linha: o número da linha digitada pelo jogador
 - ◆ coluna: a letra da coluna digitada pelo jogador
 - ◆ tamanho: o tamanho da matriz
- O primeiro **if** garante que a posição esteja dentro do limite da matriz
- O segundo **if** verifica se há uma bomba na coordenada, caso sim, será adicionado +1 na contagem de bombas
- Retorna o total de bombas ao redor



#Revela se a posição escolhida não possui bomba, conta e exibe o número de bombas ao redor da célula

```
def revela_celulas(jogo, linha, coluna, tamanho):  
    if jogo[linha][coluna] != "□":  
        return  
    bombas_ao_redor = conta_bombas(jogo, linha, coluna, tamanho)  
    if bombas_ao_redor == 0:  
        jogo[linha][coluna] = "X"  
        direções = [(-1, -1), (-1, 0), (-1, 1), (0, -1), (0, 1), (1, -1), (1, 0), (1, 1)]  
        for d in direções:  
            nova_linha, nova_coluna = linha + d[0], coluna + d[1]  
            if 0 <= nova_linha < tamanho and 0 <= nova_coluna < tamanho:  
                revela_celulas(jogo, nova_linha, nova_coluna, tamanho)  
    else:  
        jogo[linha][coluna] = f"{bombas_ao_redor}"
```

- **Revela as células ao redor caso não haja nenhuma bomba ao redor**
- **Caso uma das células reveladas também não tenha bombas ao redor, também revelará as células ao redor recursivamente**
- **A função revela o número de bombas ao redor das células reveladas**



```
if bombas_ao_redor == 0:  
    jogo[linha][coluna] = "X"  
    direções = [(-1, -1), (-1, 0), (-1, 1), (0, -1), (0, 1), (1, -1), (1, 0), (1, 1)]  
    for d in direções:  
        nova_linha, nova_coluna = linha + d[0], coluna + d[1]  
        if 0 <= nova_linha < tamanho and 0 <= nova_coluna <  
tamanho:  
            revela_celulas(jogo, nova_linha, nova_coluna, tamanho)
```



Jogador: Luisa

	A	B	C	D	E
1	<input type="checkbox"/>	1	X	X	X
2	<input type="checkbox"/>	2	1	X	X
3	<input type="checkbox"/>	<input type="checkbox"/>	2	2	1
4	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
5	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

- Se o bombas_ao_redor for igual a 0:
- ◆ Define a célula atual como "X" para indicar que foi revelada e não tem bombas ao redor
 - ◆ O "for" itera sobre as direções ao redor da célula para revelar recursivamente as células adjacentes



```
def verifica_vitoria(jogo, tamanho):  
    for i in range(tamanho):  
        for j in range(tamanho):  
            if jogo[i][j] == "□":  
                return False  
    return True
```

- Utiliza dois loops aninhados para iterar sobre cada célula do tabuleiro, onde o externo (i) itera sobre as linhas e o interno (j) itera sobre as colunas
- Dentro do loop, verifica se a célula atual está fechada com "■", o que indica que ela ainda não foi aberta e não contém uma bomba
- Se encontrar uma célula fechada, retorna **False**, o que indica que o jogador ainda não ganhou o jogo
- Se o loop completar sem encontrar nenhuma célula fechada, retorna **True**, indicando que o jogador ganhou



Funções Principais do código

```
#Lógica principal para o campo minado, que determina a jogada do usuário
def jogada(jogo, tamanho, nome_jogador, dificuldade, tempo_inicial):
    while True:
        try:
            print('Digite Ctrl + C para sair')
            x = int(input(f'Digite o número da linha (De 1 a {tamanho}): ')) - 1
            y_letra = input(f'Digite a letra da coluna (De A a {chr(64 + tamanho)}): ').strip().upper()
            if len(y_letra) != 1 or not ('A' <= y_letra <= chr(64 + tamanho)):
                raise ValueError("Coluna inválida")
            y = ord(y_letra) - 65

            if 0 <= x < tamanho and 0 <= y < tamanho:
                linha = x
                coluna = y
                if jogo[linha][coluna] == "💣":
                    mostra_tabuleiro(jogo, tamanho, revelar_minas=True, nome_jogador=nome_jogador)
                    print("\n💣💣💣 Você explodiu! 💣💣💣")
                    try:
                        os.remove(f"salvo_{nome_jogador}.txt") #Tenta excluir o arquivo de salvamento
                    except FileNotFoundError:
                        print("Nenhum jogo salvo para excluir.")
                    jogo_novo = input("Jogar novamente? (s/n): ").upper()
                    if jogo_novo == "S":
                        jogo, tamanho = criar_campo(dificuldade)
                        tempo_inicial = time.time() #Reinicia o temporizador
                        limpar_terminal()
                        mostra_tabuleiro(jogo, tamanho, nome_jogador=nome_jogador)
                    else:
                        print("Obrigado por jogar")
                        main()
                        return
            else:
                print("Posição inválida")
        except KeyboardInterrupt:
            print("\nPrograma encerrado.")
            return
```

- ❖ **Função jogada**
 - **Controla o jogo, lidando com a entrada do usuário**
 - **Gerencia o fluxo do jogo**
 - **A função é um loop que continua até que o jogo termine ou o jogador interrompa manualmente**

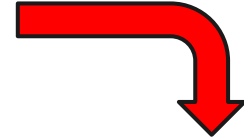


```
elif jogo[linha][coluna] == "□":
    revela_celulas(jogo, linha, coluna, tamanho)
    limpar_terminal()
    mostra_tabuleiro(jogo, tamanho, nome_jogador=nome_jogador)
    if verifica_vitoria(jogo, tamanho):
        mostra_tabuleiro(jogo, tamanho, revelar_minas=True, nome_jogador=nome_jogador)
        tempo_final = time.time()
        tempo_total = tempo_final - tempo_inicial
        print(f"\n Parabéns, {nome_jogador}! Você venceu!")
        print(f"Tempo total: {tempo_total:.2f} segundos")
        registrar_recorde(nome_jogador, tempo_total, dificuldade)
        try:
            os.remove(f"salvo_{nome_jogador}.txt") #Tenta excluir o arquivo de salvamento
        except FileNotFoundError:
            print("-")
        main() #Voltar ao menu principal após vencer
        return
    else:
        print("Coordenadas fora do tabuleiro! Tente novamente.")
except ValueError:
    print("Erro! ⚠ Verifique o valor digitado!")
except KeyboardInterrupt:
    salvar_jogo(jogo, tamanho, nome_jogador, dificuldade, tempo_inicial)
    main()
    break
```



→ Condição de derrota

```
if jogo[linha][coluna] == "💣":  
    mostra_tabuleiro(jogo, tamanho, revelar_minas=True, nome_jogador=nome_jogador)  
    print("\n💣💣💣 Você explodiu! 💣💣💣")  
    try:  
        os.remove(f"salvo_{nome_jogador}.txt")  
    except FileNotFoundError:  
        print("Nenhum jogo salvo para excluir.")  
    jogo_novo = input("Jogar novamente? (s/n): ").upper()  
    if jogo_novo == "S":  
        jogo, tamanho = criar_campo(dificuldade)  
        tempo_inicial = time.time()  
        limpar_terminal()  
        mostra_tabuleiro(jogo, tamanho, nome_jogador=nome_jogador)  
    else:  
        print("Obrigado por jogar")  
        main()  
        return
```



Jogador: Luisa

	A	B	C	D	E
1	1	1	X	X	X
2	💣	2	1	X	X
3	□	💣	2	2	1
4	💣	□	💣	□	💣
5	□	□	□	□	□

💣💣💣 Você explodiu! 💣💣💣

Jogar novamente? (s/n): |



→ Condição de vitória

```
elif jogo[linha][coluna] == " ":
    revela_celulas(jogo, linha, coluna, tamanho)
    limpar_terminal()
    mostra_tabuleiro(jogo, tamanho, nome_jogador=nome_jogador)
    if verifica_vitoria(jogo, tamanho):
        mostra_tabuleiro(jogo, tamanho, revelar_minas=True, nome_jogador=nome_jogador)
        tempo_final = time.time()
        tempo_total = tempo_final - tempo_inicial
        print(f"\n Parabéns, {nome_jogador}! Você venceu!")
        print(f"Tempo total: {tempo_total:.2f} segundos")
        registrar_recorde(nome_jogador, tempo_total, dificuldade)
    try:
        os.remove(f"salvo_{nome_jogador}.txt")
    except FileNotFoundError:
        print("-")
main()
return
```



Jogador: Luisa

	A	B	C	D	E
1	1	1	X	1	1
2	💣	1	X	1	💣
3	1	1	1	2	2
4	1	2	2	💣	1
5	💣	2	💣	2	1

Parabéns, Luisa! Você venceu!

Tempo total: 46.20 segundos



```
#Função para salvar o jogo, onde os dados serão armazenados em um arquivo txt
def salvar_jogo(jogo, tamanho, nome_jogador, dificuldade, tempo_inicial):
    tempo_total = time.time() - tempo_inicial #Calcula o tempo decorrido
    with open(f"salvo_{nome_jogador}.txt", "w", encoding="utf-8") as arquivo:
        arquivo.write(f"{nome_jogador}\n")
        arquivo.write(f"{dificuldade}\n")
        arquivo.write(f"{tamanho}\n")
        arquivo.write(f"{tempo_total}\n") #Salva o tempo decorrido
    for linha in jogo:
        arquivo.write(" ".join(linha) + "\n") #Usando espaços para separar as células
    print("\nProgresso salvo! Você pode continuar da próxima vez pela opção 'Continuar'.")
```

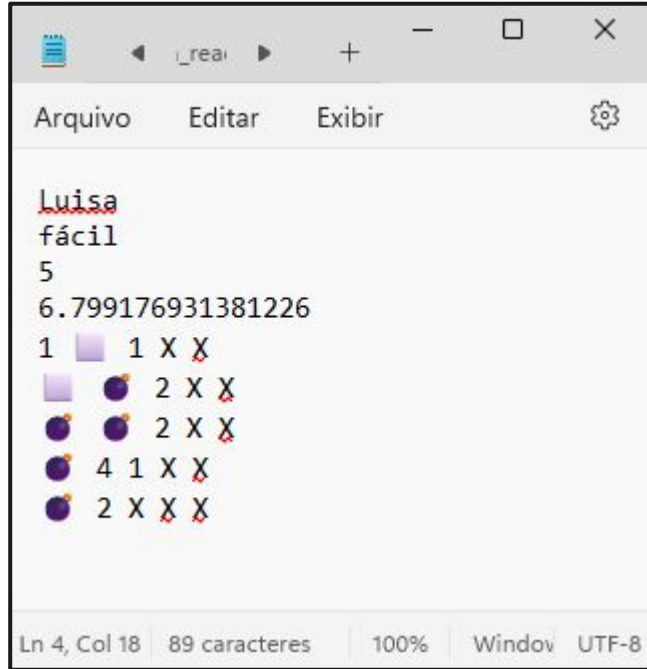
- **Salva o estado atual do jogo em um arquivo de texto**
- **Calcula o tempo decorrido desde o início do jogo até o momento do salvamento**
- **A função é chamada após o jogador gerar o erro
KeyboardInterrupt**

Função salvar_jogo

Funções Adicionais do código



➤ Arquivo de texto criado após o salvamento



```
with open(f"salvo_{nome_jogador}.txt", "w",  
encoding="utf-8") as arquivo:  
    arquivo.write(f"{nome_jogador}\n")  
    arquivo.write(f"{dificuldade}\n")  
    arquivo.write(f"{tamanho}\n")  
    arquivo.write(f"{tempo_total}\n")  
    for linha in jogo:  
        arquivo.write(" ".join(linha) + "\n")
```

Função carregar_jogo

```
#Função para carregar jogo salvo
def carregar_jogo(nome_jogador):
    try:
        with open(f"salvo_{nome_jogador}.txt", "r", encoding="utf-8") as arquivo:
            nome_jogador = arquivo.readline().strip()
            dificuldade = arquivo.readline().strip()
            tamanho = int(arquivo.readline().strip())
            tempo_total = float(arquivo.readline().strip()) #Lê o tempo decorrido
            jogo = []
            for _ in range(tamanho):
                jogo.append(arquivo.readline().strip().split(" "))
            return jogo, tamanho, nome_jogador, dificuldade, tempo_total
    except FileNotFoundError:
        print("Arquivo de salvamento não encontrado.")
        return None
```



```
Jogos salvos:
1- Luisa
Escolha o número do jogo que deseja continuar: █
```

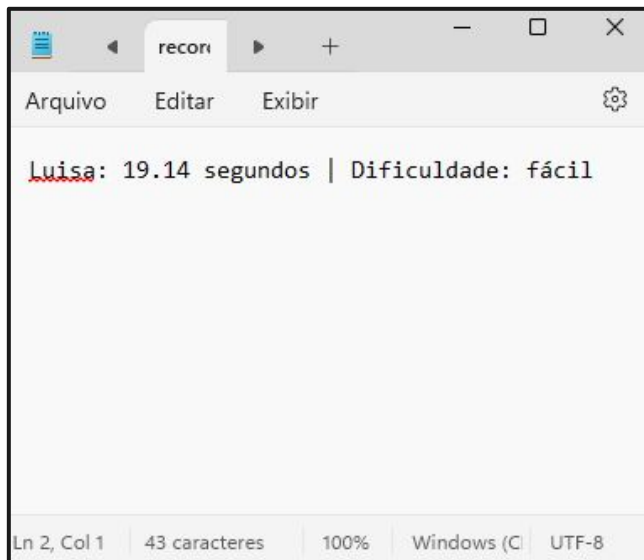
Funções Adicionais do código



- **Tenta abrir o arquivo específico do progresso salvo para o jogador**
- **Lê e processa as informações do arquivo**
- **Retorna essas informações para que o jogo possa ser restaurado**



```
#Função para registrar o recorde incluindo a dificuldade
def registrar_recorde(nome_jogador, tempo, dificuldade):
    with open("recordes.txt", "a", encoding="utf-8") as arquivo:
        arquivo.write(f"{nome_jogador}: {tempo:.2f} segundos | Dificuldade: {dificuldade}\n")
```



- **Registra em um arquivo de texto o jogador vencedor, juntamente com o tempo de conclusão do jogo e a dificuldade escolhida**
- **A função armazena os dados que serão mostrados na tabela de recordes que fica no menu do jogo**

Função mostrar_recordes

```
#Função para mostrar os recordes
def mostrar_recordes():
    try:
        with open("recordes.txt", "r", encoding="utf-8") as arquivo:
            recordes = []
            #Lê todos os recordes e armazena em uma lista
            for linha in arquivo:
                partes = linha.strip().split(" | ")
                nome_tempo = partes[0].split(": ")
                nome_jogador = nome_tempo[0]
                tempo = float(nome_tempo[1].replace(" segundos", ""))
                dificuldade = partes[1].replace("Dificuldade: ", "")
                recordes.append((nome_jogador, tempo, dificuldade))

        #Ordena a lista de recordes pela coluna de tempo (segundo item da tupla)
        recordes.sort(key=lambda x: x[1])

        #Exibe apenas a tabela de recordes
        print("\nRecordes (ordenados por tempo de conclusão):\n")
        print(f"{'Jogador':<20} {'Tempo (segundos)':<20} {'Dificuldade'}")
        print("-" * 60)
        for jogador, tempo, dificuldade in recordes:
            print(f"{jogador:<20} {tempo:<20.2f} {dificuldade}")

        #Aguardar o usuário pressionar Ctrl + C para voltar ao menu
        print("\nPressione Ctrl + C para voltar ao menu.")
        while True:
            pass #Aguarda Ctrl + C (não faz nada até o erro ser gerado)
    except FileNotFoundError:
        print("Nenhum recorde encontrado.")
    except KeyboardInterrupt:
        print("\nVoltando ao menu principal...")
        limpar_terminal() #Limpa a tela ao voltar ao menu
        return #Retorna ao menu sem encerrar o programa
```

Funções Adicionais do código



- Lê e processa os dados do arquivo de texto de recordes, armazenando-os em uma lista
- Ordena a lista de recordes pelo tempo de conclusão
- Exibe os recordes em formato de tabela

Função mostrar_recordes

Funções Adicionais do código



```
PROBLEMS 31 OUTPUT DEBUG CONSOLE TERMINAL PORTS

Recordes (ordenados por tempo de conclusão):

Jogador          Tempo (segundos)  Dificuldade
-----
Luisa             15.05             fácil
Lulu              29.36             fácil
Mari              64.59             normal
Luisa             133.88            fácil

Pressione Ctrl + C para voltar ao menu.
```

- ❖ O método sort com o argumento key é útil quando a lista contém elementos complexos, como tuplas ou objetos, onde queremos ordenar pela ordem de apenas um dos atributos ou componentes.

- `recordes.sort(key=lambda x: x[1])`
- ◆ O tempo é organizado por esse método de listas em Python
 - ◆ O argumento key do método sort é uma função que extrai uma chave de cada elemento da lista e determina a ordem dos elementos com base na chave extraída
 - ◆ A expressão `lambda x: x[1]` cria uma função anônima, definindo uma função que aceita um argumento x e retorna o segundo elemento de x(x[1])





PROBLEMS

31

OUTPUT

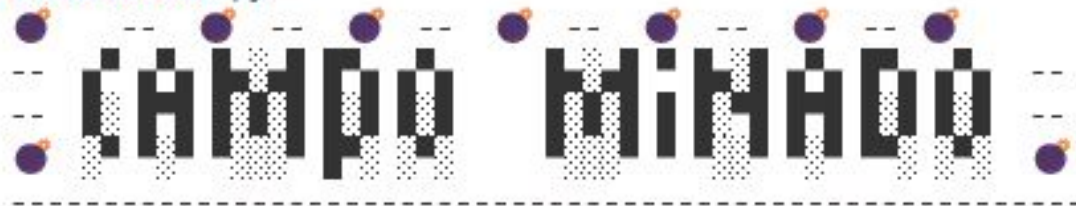
DEBUG CONSOLE

TERMINAL

PORTS

PS D:\computacao1>

```
PS D:\computacao1> d:; cd 'd:\computacao1'; & 'c:\Users\Pichau\AppData\Local\Programs\Python\extensions\ms-python.debugpy-2024.12.0-win32-x64\bundled\libs\debugpy\adapter\..\..\debugpy\finalluisa1.py'
```



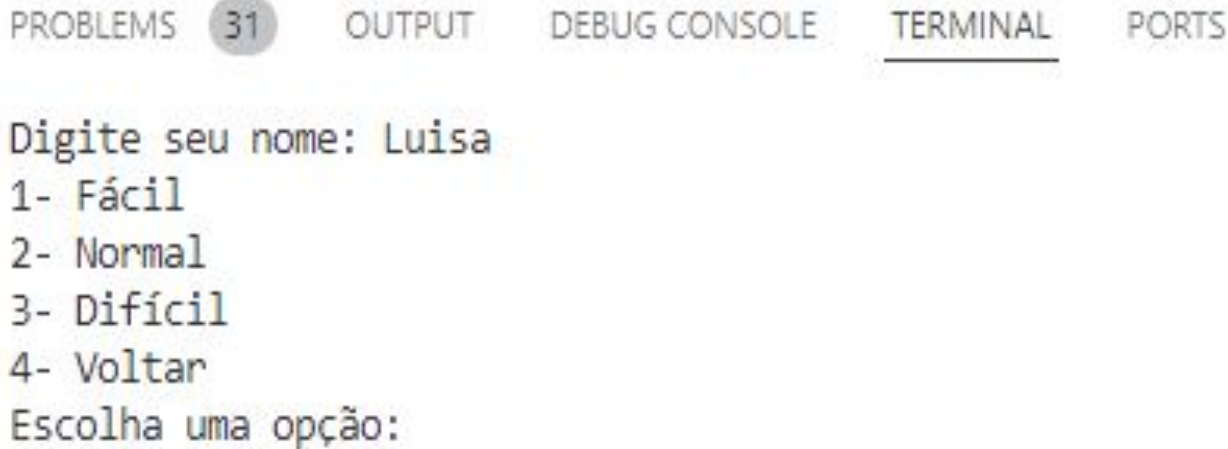
Feito por: Luísa Gonçalves

- 1- Novo Jogo
- 2- Continuar
- 3- Records
- 4- Sair

Escolha uma opção:



- ❖ Após selecionar a opção “Novo Jogo” ao digitar 1:



The screenshot shows a terminal window with a menu. At the top, there are tabs: PROBLEMS (with a count of 31), OUTPUT, DEBUG CONSOLE, TERMINAL (which is selected and underlined), and PORTS. Below the tabs, the text reads: "Digite seu nome: Luisa", followed by a list of options: "1- Fácil", "2- Normal", "3- Difícil", and "4- Voltar". At the bottom, it says "Escolha uma opção:".

- ❖ O programa pedirá o nome do jogador, logo após, será exibido um menu de dificuldades, onde jogador deverá escolher entre fácil, normal ou difícil



- ❖ Dependendo da dificuldade selecionada, o programa vai gerar matrizes de **tamanhos diferentes**

PROBLEMS 31 OUTPUT DEBUG CONSOLE TERMINAL PORTS

Jogador: Luisa

	A	B	C	D	E
1	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
2	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
3	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
4	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
5	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Digite Ctrl + C para sair

Digite o número da linha (De 1 a 5): █

- ❖ Dificuldade escolhida: fácil, matriz 5x5



- ❖ Dependendo da dificuldade selecionada, o programa vai gerar matrizes de tamanhos diferentes

PROBLEMS 31 OUTPUT DEBUG CONSOLE TERMINAL PORTS

Jogador: Lu

	A	B	C	D	E	F	G	H
1	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
2	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
3	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
4	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
5	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
6	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
7	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
8	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Digite Ctrl + C para sair

Digite o número da linha (De 1 a 8): █

- ❖ Dificuldade escolhida: normal, matriz 8x8



- ❖ Dependendo da dificuldade selecionada, o programa vai gerar matrizes de tamanhos diferentes

PROBLEMS 31 OUTPUT DEBUG CONSOLE TERMINAL PORTS

Jogador: Lu

	A	B	C	D	E	F	G	H	I	J
1	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
2	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
3	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
4	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
5	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
6	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
7	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
8	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
9	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
10	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Digite Ctrl + C para sair

Digite o número da linha (De 1 a 10): █

- ❖ Dificuldade escolhida: difícil, matriz 10x10



- ❖ Para selecionar uma determinada célula, o programa pedirá para o jogador digitar o número da linha e a letra da coluna

PROBLEMS 31 OUTPUT DEBUG CONSOLE TERMINAL PORTS

Jogador: Luisa

	A	B	C	D	E
1	X	X	1	<input type="checkbox"/>	<input type="checkbox"/>
2	1	1	2	<input type="checkbox"/>	<input type="checkbox"/>
3	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
4	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
5	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Digite Ctrl + C para sair

Digite o número da linha (De 1 a 5): 5

Digite a letra da coluna (De A a E): e

- ❖ Apertando “Ctrl + C”, o programa gerará o erro KeyboardInterrupt, onde através do tratamento de exceção, retornaremos ao menu principal e o jogo ficará salvo.



- ❖ O progresso ficará salvo quando o jogador retornar ao menu, ele poderá continuar de onde parou ao selecionar “Continuar”

Progresso salvo! Você pode continuar da próxima vez pela opção 'Continuar'.



Feito por: Luísa Gonçalves

- 1- Novo Jogo
- 2- Continuar
- 3- Recordes
- 4- Sair

Escolha uma opção: █



- ❖ Ao selecionar a opção “Continuar” após o jogador digitar 2, será aberto uma lista com os jogos salvos de acordo com o nome do jogador cadastrado

PROBLEMS

31

OUTPUT

DEBUG CONSOLE

TERMINAL

PORTS

Jogos salvos:

1- Luisa

Escolha o número do jogo que deseja continuar: █



- ❖ Depois de selecionar o jogo que deseja continuar, a partida continuará normalmente e o temporizador será despausado

PROBLEMS 31 OUTPUT DEBUG CONSOLE TERMINAL PORTS

Jogador: Luisa

	A	B	C	D	E
1	X	X	1	<input type="checkbox"/>	<input type="checkbox"/>
2	1	1	2	<input type="checkbox"/>	<input type="checkbox"/>
3	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
4	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
5	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	1

Digite Ctrl + C para sair

Digite o número da linha (De 1 a 5):



- ❖ Ao ganhar a partida, o nome do jogador ficará registrado na tabela de Recordes na opção 3, onde o ranking será dado pelo tempo de conclusão

PROBLEMS 31 OUTPUT DEBUG CONSOLE TERMINAL PORTS

	A	B	C	D	E
1	X	X	1		1
2	1	1	2	2	2
3		2	2		2
4	1	2		3	
5	X	1	1	2	1

Parabéns, Luisa! Você venceu!

Tempo total: 133.88 segundos

-- -- -- --
-- CAMPEÃO MIMADO --
-- --

Feito por: Luísa Gonçalves

-
- 1- Novo Jogo
 - 2- Continuar
 - 3- Recordes



- ❖ Ao selecionar a opção de “Recordes” após digitar 3, o programa irá exibir uma tabela com os nomes dos jogadores, o tempo de conclusão de cada um e o nível de dificuldade selecionado por cada jogador.

```
PROBLEMS 31 OUTPUT DEBUG CONSOLE TERMINAL PORTS

Recordes (ordenados por tempo de conclusão):

Jogador          Tempo (segundos)  Dificuldade
-----
Luisa             15.05            fácil
Lulu              29.36            fácil
Mari              64.59            normal
Luisa             133.88           fácil

Pressione Ctrl + C para voltar ao menu.
█
```



- ❖ Ao perder a partida, as minas serão reveladas e o jogador deverá optar entre jogar novamente ou não, caso não, o jogador retornará ao menu principal.

```
PROBLEMS 31 OUTPUT DEBUG CONSOLE TERMINAL PORTS

10 ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐
Digite Ctrl + C para sair
Digite o número da linha (De 1 a 10): 2
Digite a letra da coluna (De A a J): b
Jogador: Lu
  A B C D E F G H I J
1  1 1 1 X 1 1 1 X 1 1
2  2  2  X 1  2  1 3  2
3  3  3  1 2  3  3  3  3
4  4  4  4  4  4  4  4  4
5  5  5  5  5  5  5  5  5
6  6  6  6  6  6  6  6  6
7  7  7  7  7  7  7  7  7
8  8  8  8  8  8  8  8  8
9  9  9  9  9  9  9  9  9
10 10 10 10 10 10 10 10 10

  3 3 3 Você explodiu! 3 3 3
Jogar novamente? (s/n):
```

- ❖ O progresso não ficará salvo e o jogador não será registrado na tabela de recordes.



❖ O programa não encerra caso o usuário digite algo inválido

PROBLEMS 31 OUTPUT DEBUG CONSOLE TERMINAL

Jogador: Luisaaa

	A	B	C	D	E
1	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
2	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
3	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
4	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
5	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Digite Ctrl + C para sair

Digite o número da linha (De 1 a 5): d

Erro! ⚠Verifique o valor digitado!

Digite Ctrl + C para sair

Digite o número da linha (De 1 a 5): oea

Erro! ⚠Verifique o valor digitado!

Digite Ctrl + C para sair

Digite o número da linha (De 1 a 5): 1

Digite a letra da coluna (De A a E): f

Erro! ⚠Verifique o valor digitado!

Digite Ctrl + C para sair

Digite o número da linha (De 1 a 5): █

PROBLEMS 31 OUTPUT DEBUG CONSOLE TERMINAL POF

Jogos salvos:

1- Luisaaa

Escolha o número do jogo que deseja continuar: 3

Número inválido, tente novamente.

Escolha o número do jogo que deseja continuar: aks

Entrada inválida, tente novamente.

Escolha o número do jogo que deseja continuar: █

PROBLEMS 31 OUTPUT DEBUG CONSOLE TERMINAL

Digite seu nome: Oliveira

1- Fácil

2- Normal

3- Difícil

4- Voltar

Escolha uma opção: 7

Opção inválida, escolha novamente:

Escolha uma opção: adk

Opção inválida, escolha novamente:

Escolha uma opção: █



UFRJ
UNIVERSIDADE FEDERAL
DO RIO DE JANEIRO

Politécnica
■ ■ ■ ■
UFRJ

Obrigada!

luisagoncalves.20242@poli.ufrj.br

