Fundação CECIERJ - Consórcio CEDERJ
Curso de Tecnologia em Sistemas de Computação
Disciplina: Programação com Interfaces Gráficas (EAD05030)
AP2 1º semestre de 2023

Gabarito

Resposta Questão 1

O código exibe um retângulo, de 200 pontos de largura por 100 pontos de altura. A cor de fundo do retângulo principal é vermelho. São criados dois retângulos dentro do retângulo principal, um retângulo azul e um retângulo amarelo, contido no retângulo azul. O retângulo azul possui 60% da largura e da altura do retângulo principal, enquanto o retângulo amarelo, 30% da altura e da largura do retângulo principal (valores configurados através das variáveis `box[0]` e `box[1]`). Entre os cantos do retângulo principal e os cantos do retângulo azul são criadas linhas azuis. Dentro do retângulo amarelo, aparece escrito o texto "Python".

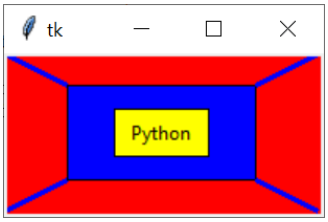

Figura 1: Desenho gerado pelo código da questão 1.

Resposta Questão 2

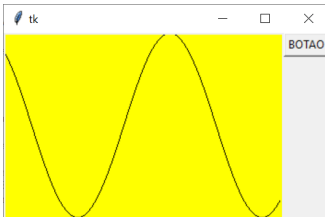

Figura 2: Desenho gerado pelo código da questão 2.

Quando o botão é acionado, é executada a função "`p`". A curva é redesenhada, deslocada a cada vez de 5 unidades no eixo X.

Resposta Questão 3

```
import tkinter as tk
import tkinter.messagebox as messagebox

class Player:
    def __init__(self, label, color):
        self.label = label
        self.color = color

class TicTacToe:
    def __init__(self, parent):
        self.parent = parent
        self.parent.title("Tic Tac Toe")

        self.player1 = Player("X", "blue")
        self.player2 = Player("O", "red")
        self.current_player = self.player1

        # definir grade dos botões
        self.buttons = []
        for row in range(3):
            button_row = []
            for col in range(3):
                button = tk.Button(self.parent, text="", width=10, height=5,
                command=lambda row=row, col=col: self.make_move(row, col))
                button.grid(row=row, column=col)
                button_row.append(button)
            self.buttons.append(button_row)

    def make_move(self, row, col):
        # ao clicar em um botão, marcar com "X" ou "O" dependendo do jogador
        do momento
        # verificar se há ganhador
        # verificar se ocorreu empate
        button = self.buttons[row][col]
        if button["text"] == "":
            button["text"] = self.current_player.label
            button["fg"] = self.current_player.color
            if self.check_winner():
                messagebox.showinfo("Winner!", "Congratulations, " +
                self.current_player.label + " wins!")
                self.parent.destroy()
            elif self.check_tie():
                messagebox.showinfo("Tie!", "It's a tie!")
                self.parent.destroy()
```

```
            else:
                self.switch_player()

    def switch_player(self):
        if self.current_player == self.player1:
            self.current_player = self.player2
        else:
            self.current_player = self.player1

    def check_winner(self):
        # verificar se há ganhador
        for i in range(3):
            if self.buttons[i][0]["text"] == self.buttons[i][1]["text"] ==
            self.buttons[i][2]["text"] != "":
                return True
            if self.buttons[0][i]["text"] == self.buttons[1][i]["text"] ==
            self.buttons[2][i]["text"] != "":
                return True
        if self.buttons[0][0]["text"] == self.buttons[1][1]["text"] ==
        self.buttons[2][2]["text"] != "":
            return True
        if self.buttons[0][2]["text"] == self.buttons[1][1]["text"] ==
        self.buttons[2][0]["text"] != "":
            return True
        return False

    def check_tie(self):
        # verificar se ocorreu empate
        for row in self.buttons:
            for button in row:
                if button["text"] == "":
                    return False
        return True

root = tk.Tk()
game = TicTacToe(root)
root.mainloop()
```

Resposta Questão 4

O comando `print` exibe o resultado da seguinte operação com as listas: `[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]` menos a lista `[5, 6, 7, 8, 9, 10, 11, 12, 13, 14]` mais a lista `[10, 1]`, resultando em `[0, 1, 2, 3, 4, 10]`, multiplicada por 2, resultando em `[0, 2, 4, 6, 8, 20]`. Este é o resultado exibido.

Os métodos `add`, `sub` e `repr` assumem que a variável `x` pode ser transformada em uma lista pela função `list`. Se este não for o caso, por exemplo se `x` for

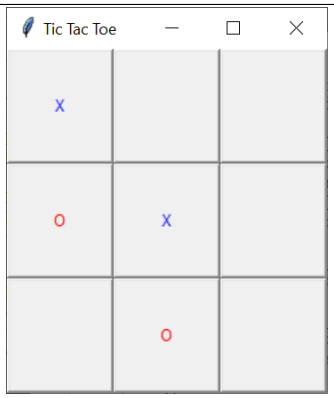

Figura 3: Desenho gerado pelo código da questão 3.

um inteiro, a função `list` falhará. Já o método `mul` assume que o produto será realizado entre uma lista e um valor numérico (inteiro ou float), se não for o caso, falhará. O método funciona, de forma incorreta, mas sem gerar erro de sintaxe, se a operação for realizada entre uma lista e uma tupla ou string.