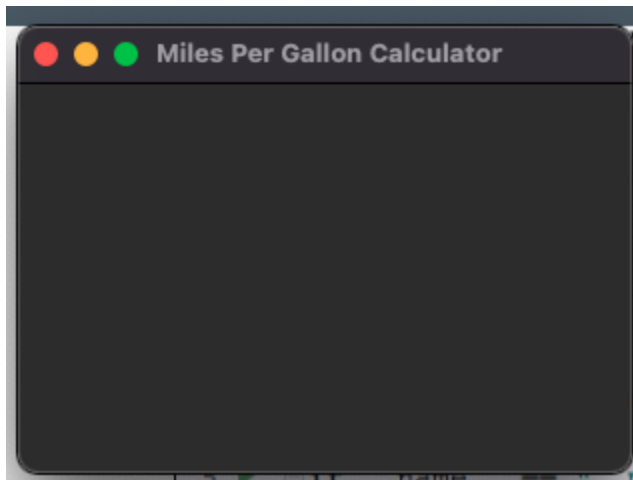


Screen Capture #1

```
1 #!/usr/bin/env python3
2
3 import tkinter as tk
4
5 if __name__ == "__main__":
6     # Create the root window
7     root = tk.Tk()
8
9     # Add a title to the root
10    root.title("Miles Per Gallon Calculator")
11
12    # Display the root Window
13    root.mainloop()
14
```

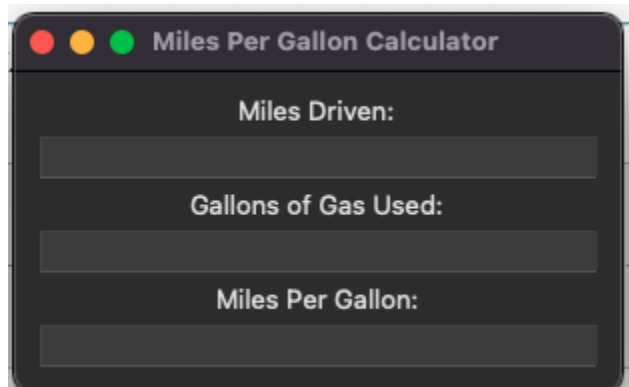
Screen Capture #2



Screen Capture #3

```
1  #!/usr/bin/env python3
2
3  import tkinter as tk
4  from tkinter import ttk
5
6  class MPGFrame(ttk.Frame):
7      def __init__(self, parent):
8          ttk.Frame.__init__(self, parent, padding="10 10 10 10")
9          self.pack()
10
11         # Define string variables for text entry fields
12         self.milesDriven = tk.StringVar()
13         self.gallonsUsed = tk.StringVar()
14         self.milesPerGallon = tk.StringVar()
15
16         # Display the components
17         ttk.Label(self, text="Miles Driven:").pack()
18         ttk.Entry(self, width=30, textvariable=self.milesDriven).pack()
19
20         ttk.Label(self, text="Gallons of Gas Used:").pack()
21         ttk.Entry(self, width=30, textvariable=self.gallonsUsed).pack()
22
23         ttk.Label(self, text="Miles Per Gallon:").pack()
24         ttk.Entry(self, width=30, textvariable=self.milesPerGallon, state="readonly").pack()
25
26  if __name__ == "__main__":
27      # Create the root window
28      root = tk.Tk()
29
30      # Add a title to the root
31      root.title("Miles Per Gallon Calculator")
32
33      MPGFrame(root)
34
35      # Display the root window
36      root.mainloop()
37
```

Screen Capture #4

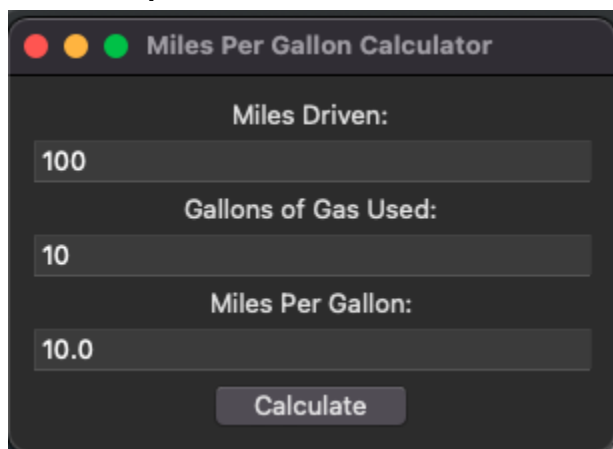


Screen Capture #5

```
21     ttk.Entry(self, width=30, textvariable=self.gallonsUsed).pack()
22
23     ttk.Label(self, text="Miles Per Gallon:").pack()
24     ttk.Entry(self, width=30, textvariable=self.milesPerGallon, state="readonly").pack()
25
26     ttk.Button(self, text="Calculate",
27               command=self.calculate).pack()
28
29     def calculate(self):
30         # Get numbers from the first two text entry fields
31         milesDriven = float(self.milesDriven.get())
32         gallonsUsed = float(self.gallonsUsed.get())
33
34         # Calc the miles per gallon (mpg)
35         mpg = milesDriven / gallonsUsed
36         mpg = round(mpg, 2)
37
38         # Display the mpg in the third text field
39         self.milesPerGallon.set(mpg)
```

NORMAL main assignment 17 01.py utf-8 | unix | python 51%

Screen Capture #6



The screenshot shows a macOS-style window titled "Miles Per Gallon Calculator". It contains three text input fields and a "Calculate" button. The first field, labeled "Miles Driven:", contains the value "100". The second field, labeled "Gallons of Gas Used:", contains the value "10". The third field, labeled "Miles Per Gallon:", contains the calculated value "10.0".

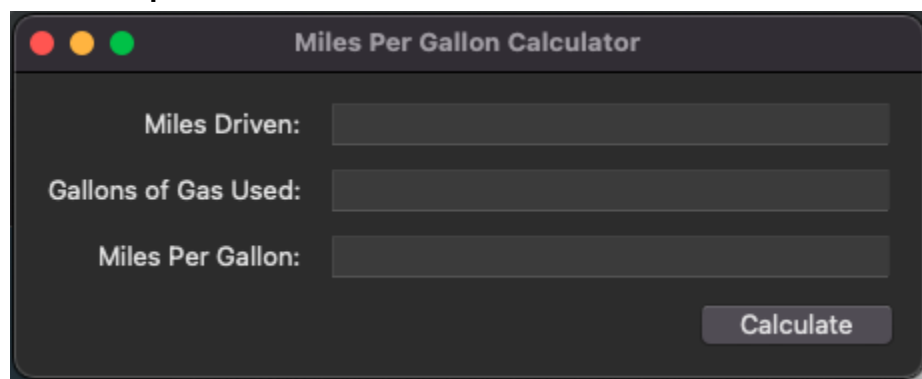
Input Label	Input Value
Miles Driven:	100
Gallons of Gas Used:	10
Miles Per Gallon:	10.0

Calculate

Screen Capture #7

```
6 class MPGFrame(ttk.Frame):
7     def __init__(self, parent):
8         ttk.Frame.__init__(self, parent, padding="10 10 10 10")
9         self.pack()
10
11         # Define string variables for text entry fields
12         self.milesDriven = tk.StringVar()
13         self.gallonsUsed = tk.StringVar()
14         self.milesPerGallon = tk.StringVar()
15
16         # Display the components
17         ttk.Label(self, text="Miles Driven:").grid(column=0, row=0, sticky=tk.E)
18         ttk.Entry(self, width=30, textvariable=self.milesDriven).grid(column=1, row=0)
19
20         ttk.Label(self, text="Gallons of Gas Used:").grid(column=0, row=1, sticky=tk.E)
21         ttk.Entry(self, width=30, textvariable=self.gallonsUsed).grid(column=1, row=1)
22
23         ttk.Label(self, text="Miles Per Gallon:").grid(column=0, row=2, sticky=tk.E)
24         ttk.Entry(
25             self, width=30, textvariable=self.milesPerGallon, state="readonly"
26         ).grid(column=1, row=2)
27
28         ttk.Button(self, text="Calculate", command=self.calculate).grid(
29             column=1, row=3, sticky=tk.E
30         )
31
32         # Add padding to all components
33         for child in self.winfo_children():
34             child.grid_configure(padx=5, pady=3)
```

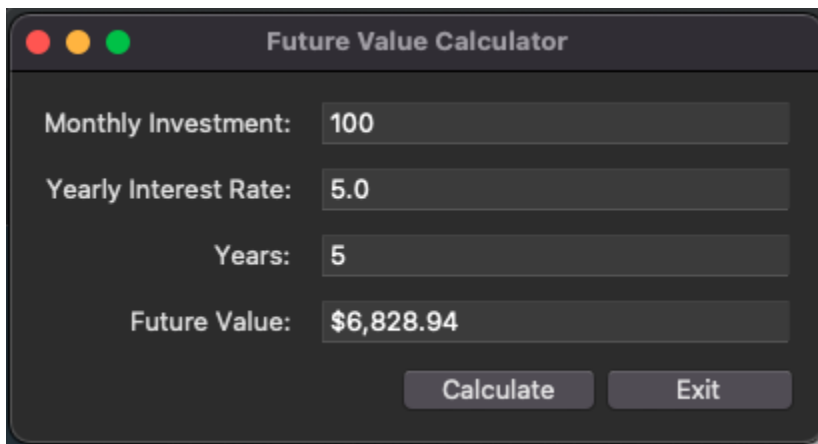
Screen Capture #8



Screen Capture #9

```
1 class Investment:
2     def __init__(self):
3         self.monthly_investment = 0
4         self.yearly_interest_rate = 0
5         self.years = 0
6
7     def calculate_future_value(self):
8         monthly_interest_rate = self.yearly_interest_rate / 12 / 100
9         months = self.years * 12
10
11         future_value = 0
12         for i in range(months):
13             future_value += self.monthly_investment
14             monthly_interest_amount = future_value * monthly_interest_rate
15             future_value += monthly_interest_amount
16
17         return future_value
```

Screen Capture #10



Future Value Calculator

Monthly Investment:	100
Yearly Interest Rate:	5.0
Years:	5
Future Value:	\$6,828.94

Calculate Exit