Question ❓

The program solves the ***Bin Packing Problem using the First Fit heuristic***. The question it addresses is: given 8 items of sizes [6, 5, 4, 3, 2, 5, 4, 6] and bins of capacity 10, how can the items be packed efficiently, and what is the utilization of the bins? According to the algorithm, 4 bins are required to accommodate all items. The total space used by the items is 35, leaving 5 units of wasted space across the bins. On average, each bin holds 8.75 units, which corresponds to a packing efficiency of 87.5%.

The distribution of items among bins—[6,4], [5,3,2], [5,4], [6]—results directly from the First Fit procedure: items are placed sequentially into the first bin that has enough remaining space, and a new bin is opened only when necessary. This explains why some bins are fully filled while others are partially [1]filled, producing the calculated average fill and efficiency. The statistics accurately summarize both the number of bins used and the effectiveness of space utilization, reflecting the behavior of the algorithm on this dataset.

****

1. **Methodology and result**

This code demonstrates the First Fit heuristic for the Bin Packing Problem. It starts with a list of items with given sizes and a fixed bin capacity. For each item, it tries to place it into the first bin that has enough remaining space; if no such bin exists, it opens a new bin. While doing this, the code records the state of bins after placing each item. After all items are placed, it computes statistics such as total bins used, total used and wasted space, average fill per bin, and packing efficiency. The matplotlib animation visualizes this process step by step, showing how items are stacked in bins over time. The final result is an animated visualization of how the First Fit heuristic organizes items into bins, along with efficiency metrics summarizing the packing.

A. Methodology (Visualization)

Bin 1: [6, 4] -> used 10 / 10
Bin 2: [5, 3, 2] -> used 10 / 10
Bin 3: [5, 4] -> used 9 / 10
Bin 4: [6] -> used 6 / 10

Explanation:

1. Bin 1 takes 6, then 4 → full.
2. Bin 2 takes 5, then 3, then 2 → full.
3. Bin 3 takes 5, then 4 → 1 unit left.
4. Bin 4 takes remaining 6 → 4 units wasted.

B. Final output:

[1] YouTube Handle: @csquickrevisionshorts

Total bins used: 4
Total used space: 35
Total wasted space: 5
Average fill per bin: 8.75
Packing efficiency: 87.5%

***

Problem we solved here:

Using the First Fit heuristic, 8 items with sizes [6, 5, 4, 3, 2, 5, 4, 6] are packed into 4 bins of capacity 10. The total used space is 35, leaving 5 units wasted, giving an average fill of 8.75 per bin and a packing efficiency of 87.5%. Items are placed sequentially into the first bin with enough space, so some bins are full while others are partially filled, reflecting the algorithm's sequential allocation strategy.

Statistics:

A. total_bins = len(bins).

B. wasted_space = total_capacity - total_used.

C. efficiency = (total_used / total_capacity) * 100.

***

Colab Code:

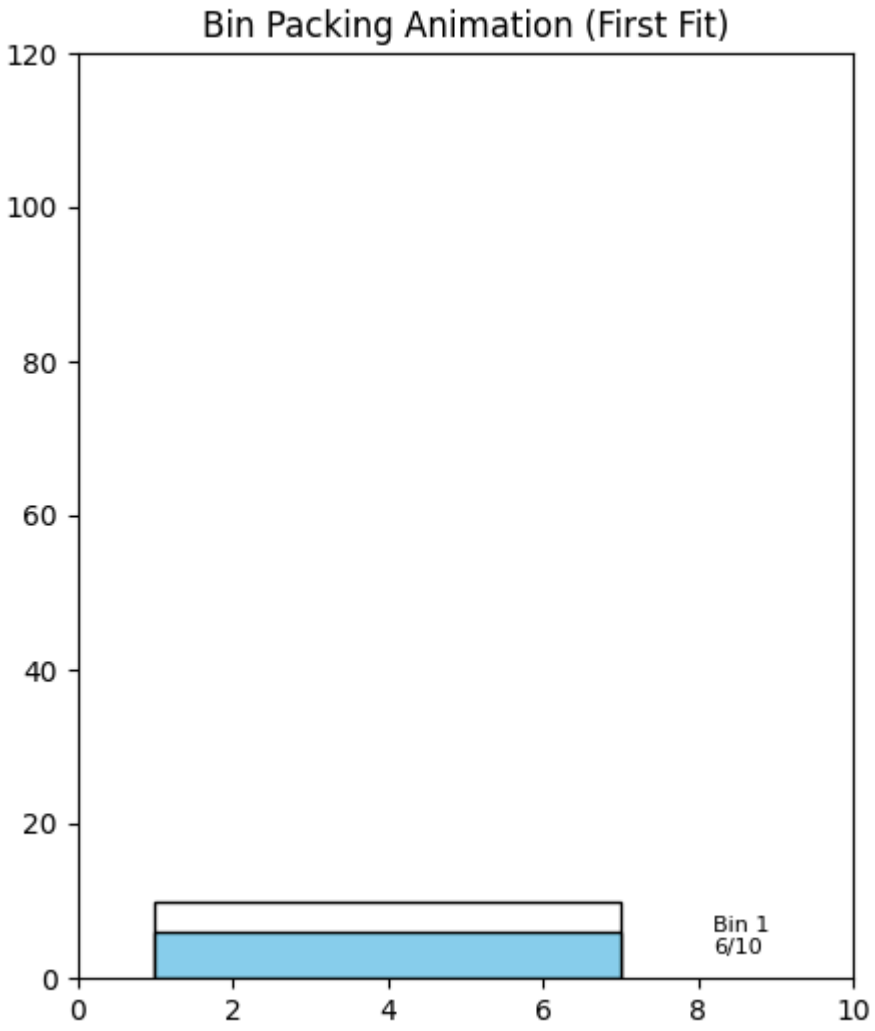
```
import matplotlib.pyplot as plt
import matplotlib.patches as patches
from matplotlib.animation import FuncAnimation
from IPython.display import HTML, display

# --- Data setup ---
bin_capacity = 10
items = [6, 5, 4, 3, 2, 5, 4, 6]
bins, states = [], []

# --- First Fit (recording step by step) ---
for item in items:
    placed = False
    for b in bins:
        if sum(b) + item <= bin_capacity:
            b.append(item)
            placed = True
            break
    if not placed:
        bins.append([item])
    states.append([list(b) for b in bins])  # save current state
```

```
# --- Compute statistics ---
total_bins = len(bins)
total_used = sum(sum(b) for b in bins)
total_capacity = total_bins * bin_capacity
wasted_space = total_capacity - total_used
efficiency = (total_used / total_capacity) * 100
avg_fill = total_used / total_bins

# --- Print statistics ---
print("📊 BIN PACKING STATISTICS")
print("--------------------------")
print(f"Total items: {len(items)}")
print(f"Bin capacity: {bin_capacity}")
print(f"Total bins used: {total_bins}")
print(f"Total used space: {total_used}")
print(f"Total wasted space: {wasted_space}")
print(f"Average fill per bin: {avg_fill:.2f}")
print(f"Packing efficiency: {efficiency:.2f}%")

# --- Visualization setup ---
fig, ax = plt.subplots(figsize=(5, 6))
ax.set_xlim(0, 10)
ax.set_ylim(0, (len(items) + 2) * (bin_capacity + 2))

def draw_state(state):
    ax.clear()
    ax.set_xlim(0, 10)
    ax.set_ylim(0, (len(items) + 2) * (bin_capacity + 2))
    ax.set_title("Bin Packing Animation (First Fit)", fontsize=12)

    for i, b in enumerate(state):
        y = i * (bin_capacity + 2)
        ax.add_patch(patches.Rectangle((1, y), 6, bin_capacity,
                                       fill=False, edgecolor='black'))
        h = 0
        for item in b:
            ax.add_patch(patches.Rectangle((1, y + h), 6, item,
                                           color='skyblue', ec='black'))
            h += item
        ax.text(8.2, y + 3, f"Bin {i+1}\n{sum(b)}/{bin_capacity}", fontsize=8)

def update(frame):
    draw_state(states[frame])

ani = FuncAnimation(fig, update, frames=len(states), interval=1000, repeat=False)

# --- Show animation in Colab ---
```

```
display(HTML(ani.to_jshtml()))

#@=====

# --- Save the animation ---
from matplotlib.animation import PillowWriter

# Save as MP4
ani.save("bin_packing_animation.mp4", writer='ffmpeg', fps=1)

# Save as GIF (alternative)
ani.save("bin_packing_animation.gif", writer=PillowWriter(fps=1))

print("✅ Animation saved as MP4 and GIF")
```

Output:

```
📊 BIN PACKING STATISTICS
------------------------
Total items: 8
Bin capacity: 10
Total bins used: 4
Total used space: 35
Total wasted space: 5
Average fill per bin: 8.75
Packing efficiency: 87.50%
```


Bin Packing Animation (First Fit)
120
100
80
60
40
20
0
0
2
4
6
8
10
Bin 1
6/10