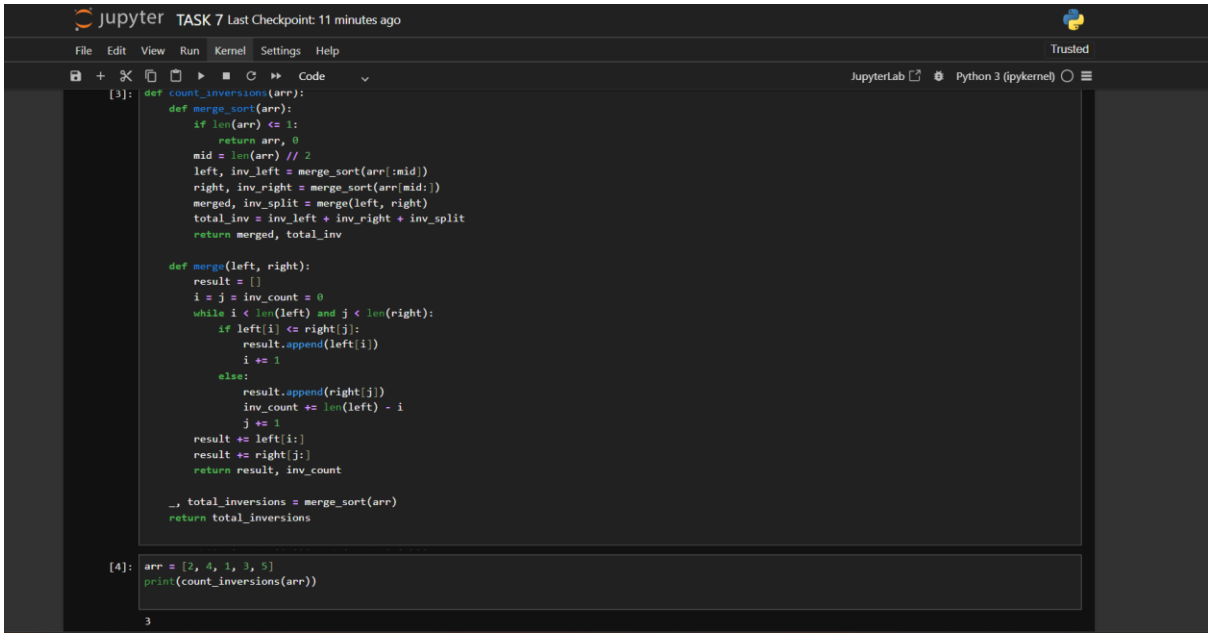
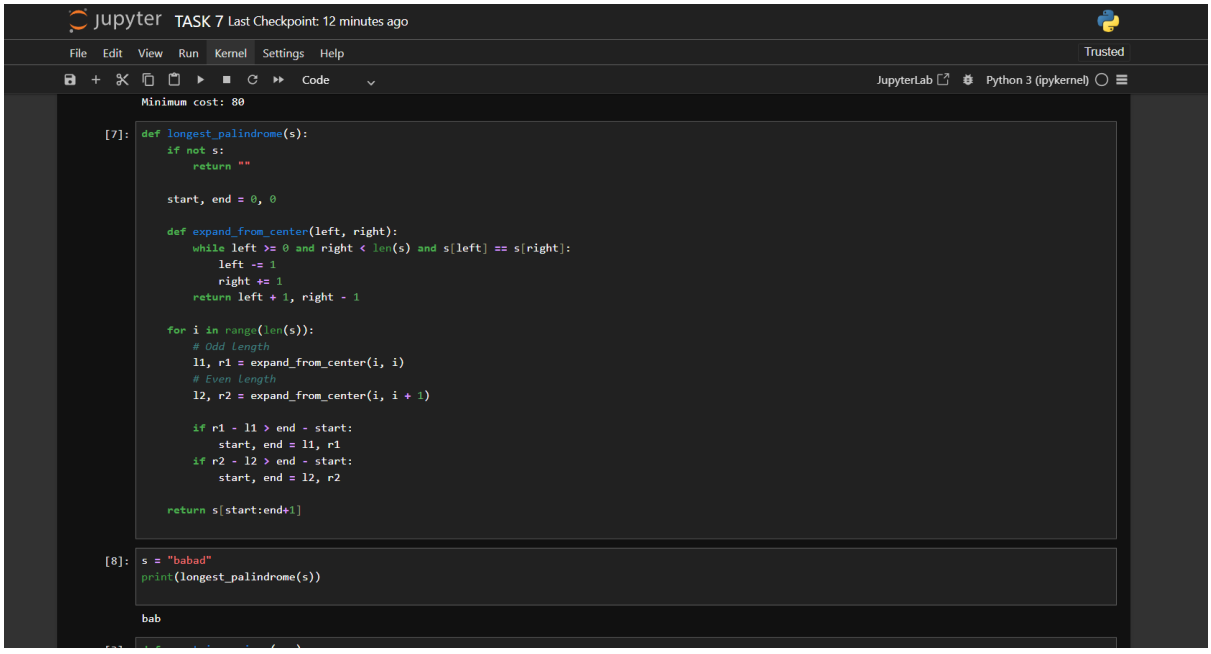

jupyter TASK 7 Last Checkpoint: 11 minutes ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)
[3]:
def count_inversions(arr):
def merge_sort(arr):
if len(arr) <= 1:
return arr, 0
mid = len(arr) // 2
left, inv_left = merge_sort(arr[:mid])
right, inv_right = merge_sort(arr[mid:])
merged, inv_split = merge(left, right)
total_inv = inv_left + inv_right + inv_split
return merged, total_inv
def merge(left, right):
result = []
i = j = inv_count = 0
while i < len(left) and j < len(right):
if left[i] <= right[j]:
result.append(left[i])
i += 1
else:
result.append(right[j])
inv_count += len(left) - i
j += 1
result += left[i:]
result += right[j:]
return result, inv_count
_, total_inversions = merge_sort(arr)
return total_inversions
[4]:
arr = [2, 4, 1, 3, 5]
print(count_inversions(arr))
3


jupyter TASK 7 Last Checkpoint: 12 minutes ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)
Minimum cost: 80
[7]:
def longest_palindrome(s):
if not s:
return ""
start, end = 0, 0
def expand_from_center(left, right):
while left >= 0 and right < len(s) and s[left] == s[right]:
left -= 1
right += 1
return left + 1, right - 1
for i in range(len(s)):
# Odd Length
l1, r1 = expand_from_center(i, i)
Even Length
l2, r2 = expand_from_center(i, i + 1)
if r1 - l1 > end - start:
start, end = l1, r1
if r2 - l2 > end - start:
start, end = l2, r2
return s[start:end+1]
[8]:
s = "babad"
print(longest_palindrome(s))
bab

jupyter TASK 7 Last Checkpoint: 13 minutes ago

File Edit View Run Kernel Settings Help

Trusted

Code

JupyterLab Python 3 (ipykernel)

[11]:
```
import sys

def tsp(dist):
    n = len(dist)
    ALL_VISITED = (1 << n) - 1
    memo = [[None] * n for _ in range(1 << n)]

    def dp(mask, pos):
        if mask == ALL_VISITED:
            return dist[pos][0]

        if memo[mask][pos] is not None:
            return memo[mask][pos]

        min_cost = float('inf')
        for city in range(n):
            if mask & (1 << city) == 0:
                new_cost = dist[pos][city] + dp(mask | (1 << city), city)
                min_cost = min(min_cost, new_cost)

        memo[mask][pos] = min_cost
        return min_cost
    return dp(1, 0)
```

[10]:
```
distances = [
    [0, 10, 15, 20],
    [10, 0, 35, 25],
    [15, 35, 0, 30],
```

jupyter TASK 7 Last Checkpoint: 13 minutes ago

File Edit View Run Kernel Settings Help

Trusted

Code

JupyterLab Python 3 (ipykernel)

```
        if memo[mask][pos] is not None:
            return memo[mask][pos]

        min_cost = float('inf')
        for city in range(n):
            if mask & (1 << city) == 0:
                new_cost = dist[pos][city] + dp(mask | (1 << city), city)
                min_cost = min(min_cost, new_cost)

        memo[mask][pos] = min_cost
        return min_cost
    return dp(1, 0)
```

[10]:
```
distances = [
    [0, 10, 15, 20],
    [10, 0, 35, 25],
    [15, 35, 0, 30],
    [20, 25, 30, 0]
]

print("Minimum cost:", tsp(distances))  # Output: 80
```

```
Minimum cost: 80
```

jupyter TASK 7 Last Checkpoint: 15 minutes ago

File Edit View Run Kernel Settings Help

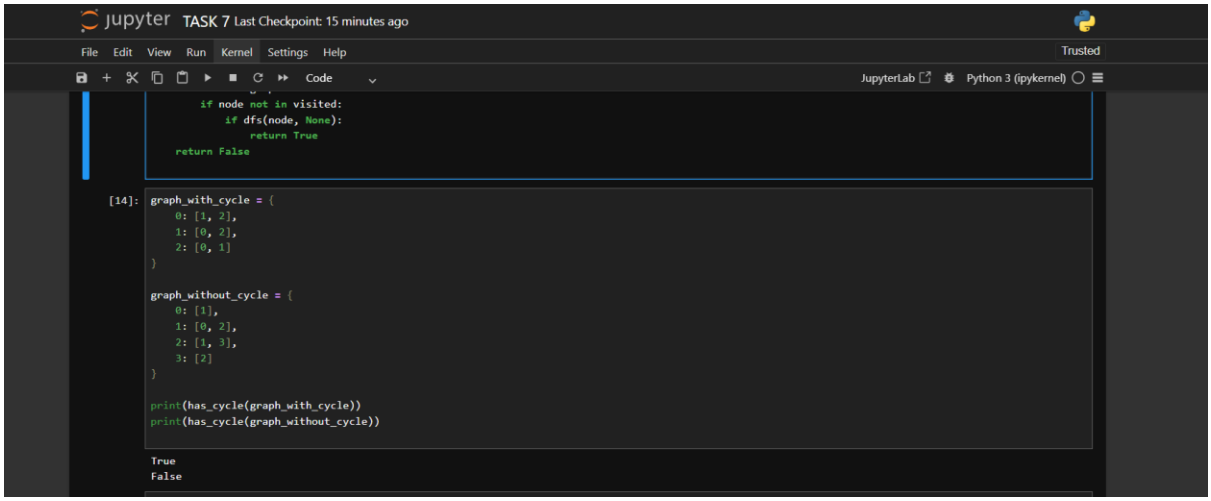
Trusted

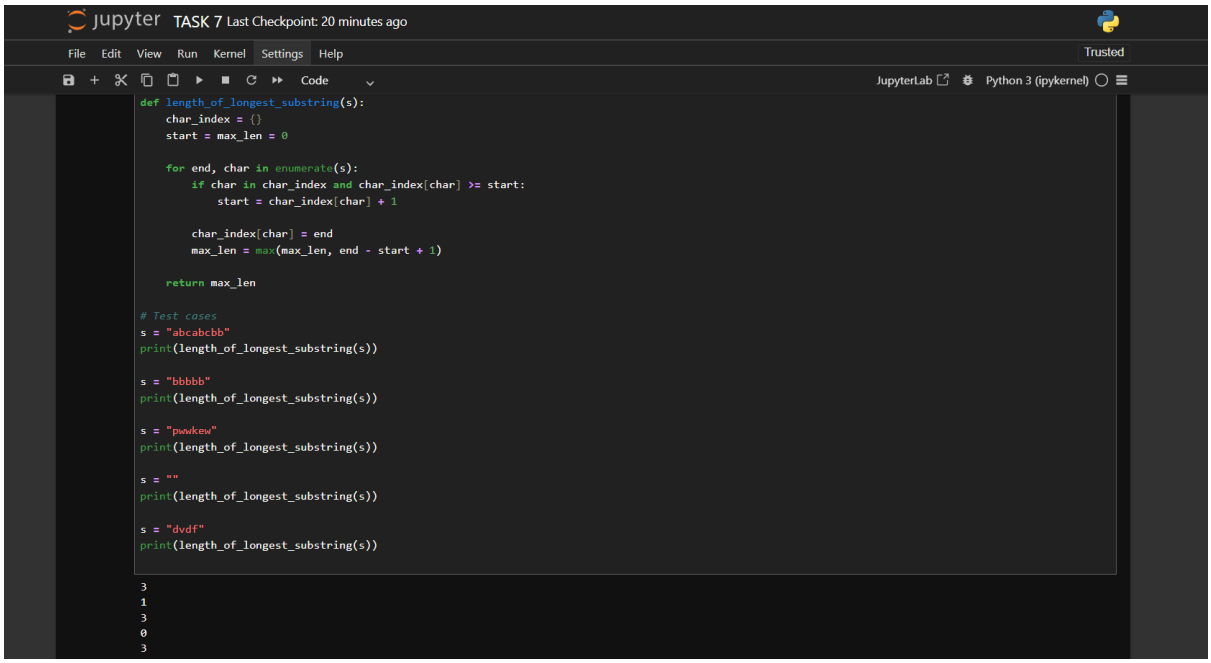
Code

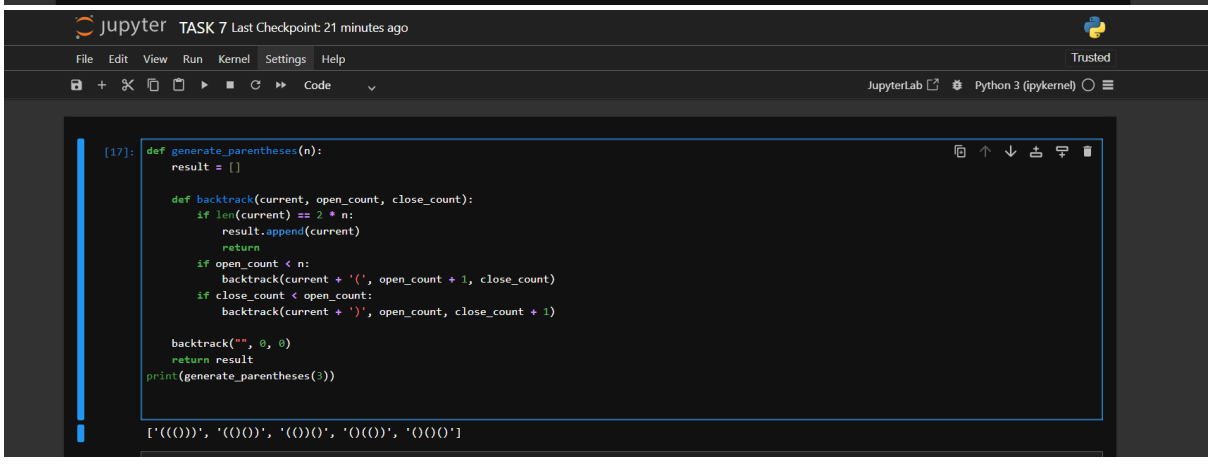
JupyterLab Python 3 (ipykernel)

[12]:
```
def has_cycle(graph):
    visited = set()

    def dfs(node, parent):
        visited.add(node)
        for neighbor in graph[node]:
            if neighbor not in visited:
                if dfs(neighbor, node):
                    return True
            elif neighbor != parent:
                return True
        return False

    for node in graph:
        if node not in visited:
            if dfs(node, None):
                return True
    return False
```

[14]:
```
graph_with_cycle = {
    0: [1, 2],
    1: [0, 2],
    2: [0, 1]
}

graph_without_cycle = {
    0: [1],
    1: [0, 2],
    2: [1, 3],
    3: [2]
```


jupyter TASK 7 Last Checkpoint: 15 minutes ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)
if node not in visited:
if dfs(node, None):
return True
return False
[14]:
graph_with_cycle = {
0: [1, 2],
1: [0, 2],
2: [0, 1]
}
graph_without_cycle = {
0: [1],
1: [0, 2],
2: [1, 3],
3: [2]
}
print(has_cycle(graph_with_cycle))
print(has_cycle(graph_without_cycle))
True
False


jupyter TASK 7 Last Checkpoint: 20 minutes ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)
def length_of_longest_substring(s):
char_index = {}
start = max_len = 0
for end, char in enumerate(s):
if char in char_index and char_index[char] >= start:
start = char_index[char] + 1
char_index[char] = end
max_len = max(max_len, end - start + 1)
return max_len
# Test cases
s = "abcabcbb"
print(length_of_longest_substring(s))
s = "bbbbb"
print(length_of_longest_substring(s))
s = "pwwkew"
print(length_of_longest_substring(s))
s = ""
print(length_of_longest_substring(s))
s = "dvdf"
print(length_of_longest_substring(s))
3
1
3
0
3


jupyter TASK 7 Last Checkpoint: 21 minutes ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)
[17]:
def generate_parentheses(n):
result = []
def backtrack(current, open_count, close_count):
if len(current) == 2 * n:
result.append(current)
return
if open_count < n:
backtrack(current + '(', open_count + 1, close_count)
if close_count < open_count:
backtrack(current + ')', open_count, close_count + 1)
backtrack("", 0, 0)
return result
print(generate_parentheses(3))
['((()))', '(()())', '(())()', '()(())', '()()()']

jupyter TASK 7 Last Checkpoint: 24 minutes ago

File Edit View Run Kernel Settings Help Trusted

Code JupyterLab Python 3 (ipykernel)

[19]:
```python
class TreeNode:
    def __init__(self, val=0, left=None, right=None):
        self.val = val
        self.left = left
        self.right = right

def zigzag_level_order(root):
    if not root:
        return []

    result = []
    current_level = [root]
    left_to_right = True

    while current_level:
        level_values = []
        next_level = []

        for node in reversed(current_level):
            level_values.append(node.val)
            if left_to_right:
                if node.left:
                    next_level.append(node.left)
                if node.right:
                    next_level.append(node.right)
            else:
                if node.right:
                    next_level.append(node.right)
                if node.left:
                    next_level.append(node.left)

        result.append(level_values)
        current_level = next_level
        left_to_right = not left_to_right

    return result

#       1
#      / \
#     2   3
#    / \ / \
#   4  5 6  7

root = TreeNode(1)
root.left = TreeNode(2)
root.right = TreeNode(3)
root.left.left = TreeNode(4)
root.left.right = TreeNode(5)
root.right.left = TreeNode(6)
root.right.right = TreeNode(7)

# Step 4: Run
print(zigzag_level_order(root))
```

```
[[1], [3, 2], [4, 5, 6, 7]]
```

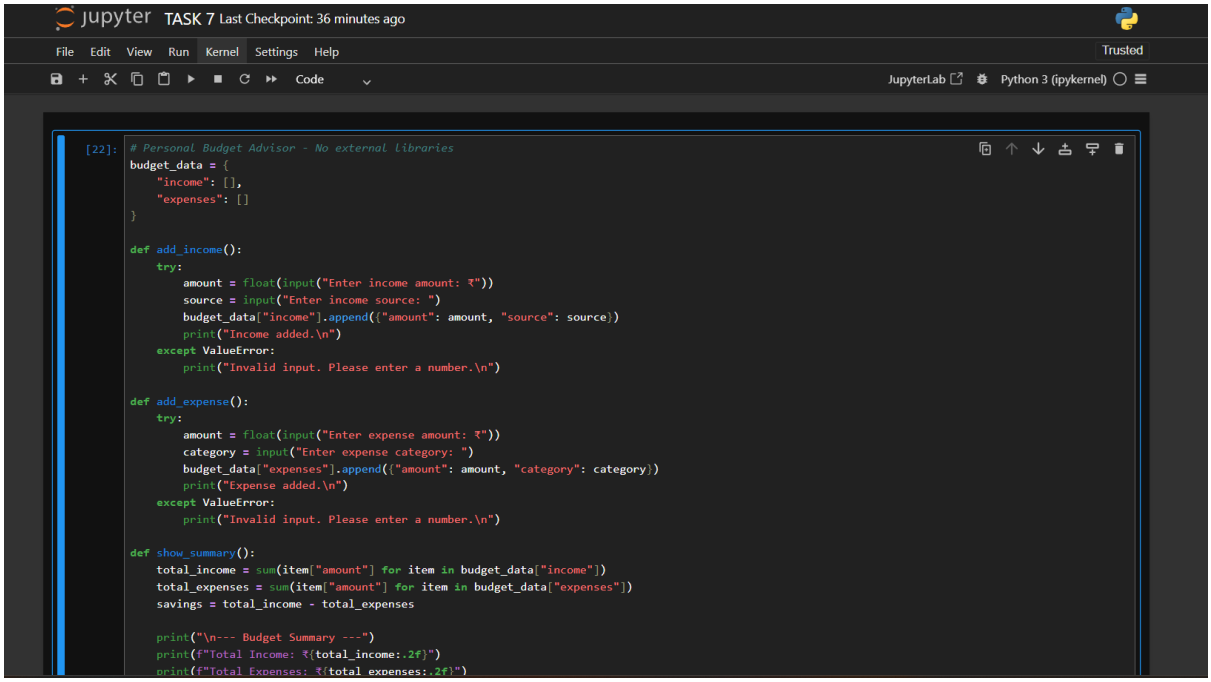
jupyter TASK 7 Last Checkpoint: 25 minutes ago

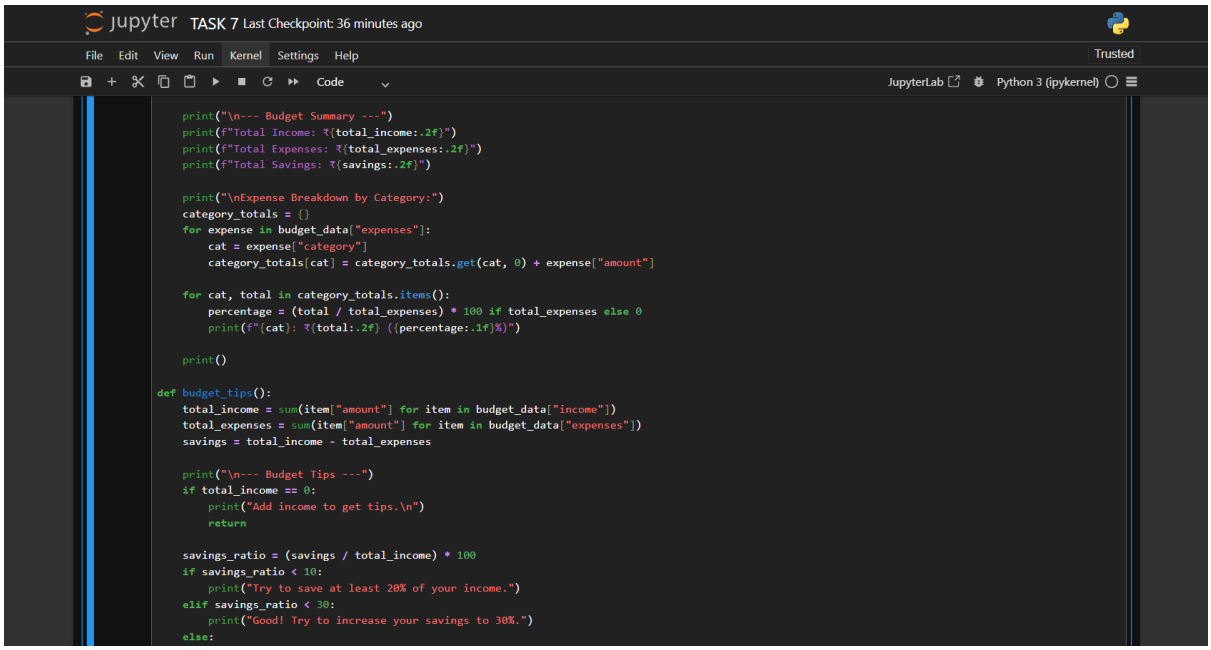
File Edit View Run Kernel Settings Help Trusted

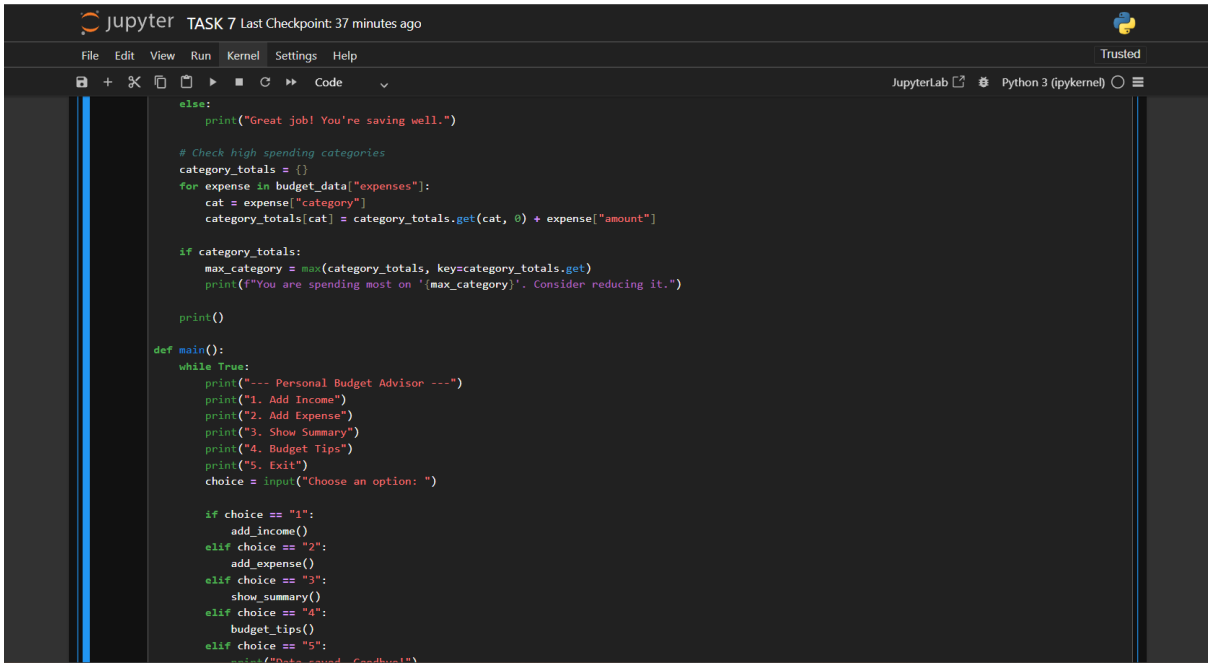
Code JupyterLab Python 3 (ipykernel)

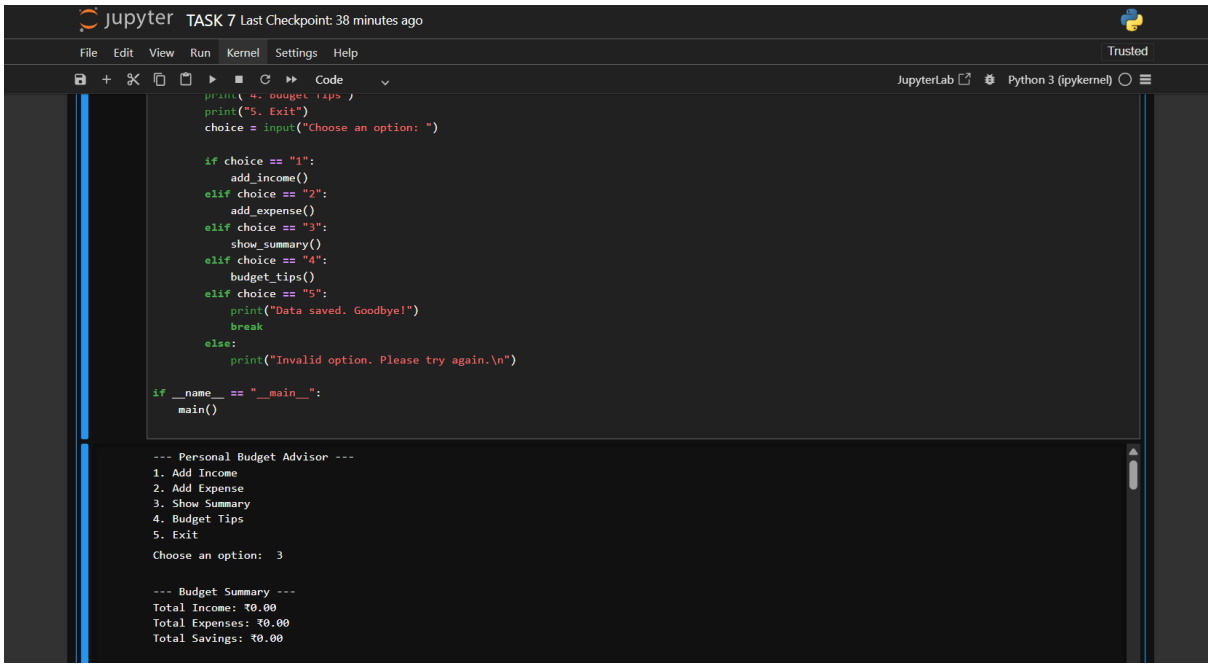
[20]:
```python
def partition(s):
    result = []

    def is_palindrome(sub):
        return sub == sub[::-1]

    def backtrack(start, path):
        if start == len(s):
            result.append(path[:])
            return

        for end in range(start + 1, len(s) + 1):
            substring = s[start:end]
            if is_palindrome(substring):
                path.append(substring)
                backtrack(end, path)
                path.pop()  # backtrack

    backtrack(0, [])
    return result

s = "aab"
print(partition(s))
```

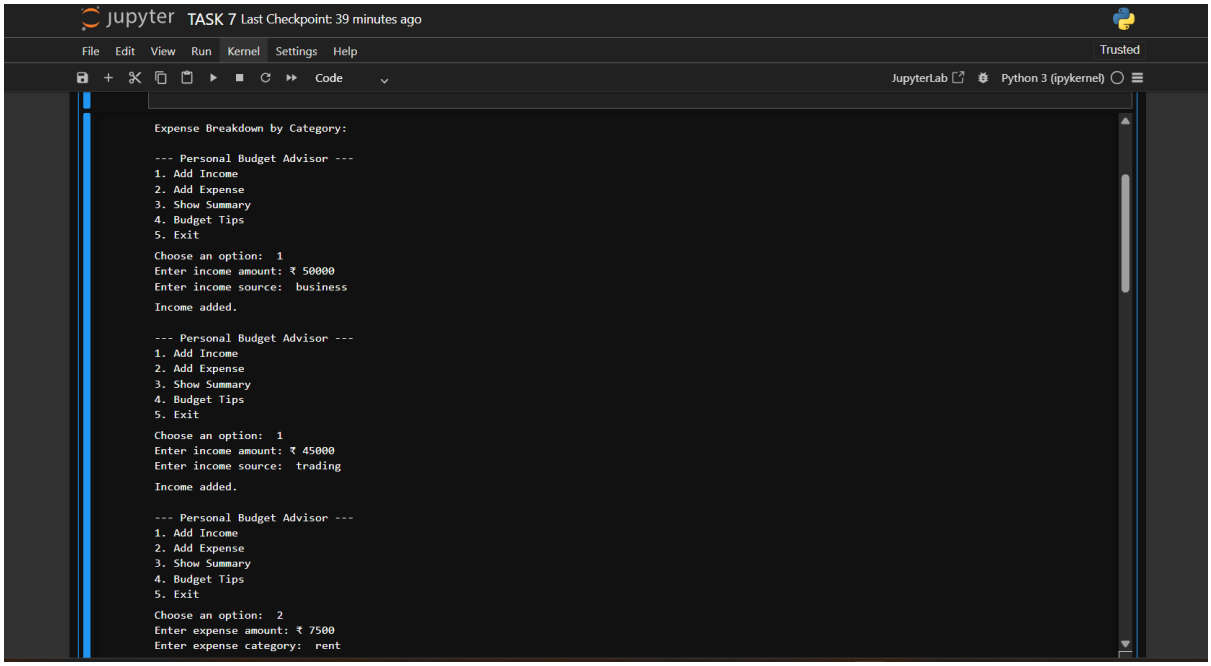
```
[['a', 'a', 'b'], ['aa', 'b']]
```

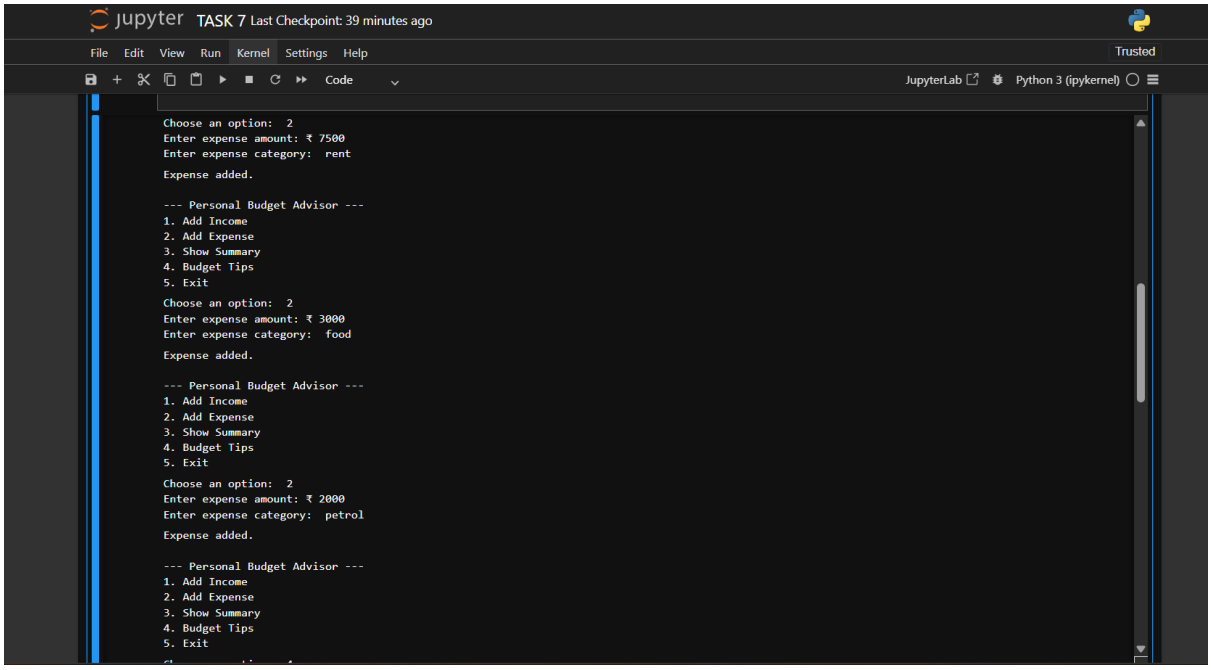

jupyter TASK 7 Last Checkpoint: 36 minutes ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)
[22]:
# Personal Budget Advisor - No external libraries
budget_data = {
"income": [],
"expenses": []
}
def add_income():
try:
amount = float(input("Enter income amount: ₹"))
source = input("Enter income source: ")
budget_data["income"].append({"amount": amount, "source": source})
print("Income added.\n")
except ValueError:
print("Invalid input. Please enter a number.\n")
def add_expense():
try:
amount = float(input("Enter expense amount: ₹"))
category = input("Enter expense category: ")
budget_data["expenses"].append({"amount": amount, "category": category})
print("Expense added.\n")
except ValueError:
print("Invalid input. Please enter a number.\n")
def show_summary():
total_income = sum(item["amount"] for item in budget_data["income"])
total_expenses = sum(item["amount"] for item in budget_data["expenses"])
savings = total_income - total_expenses
print("\n--- Budget Summary ---")
print(f"Total Income: ₹{total_income:.2f}")
print(f"Total Expenses: ₹{total_expenses:.2f}")

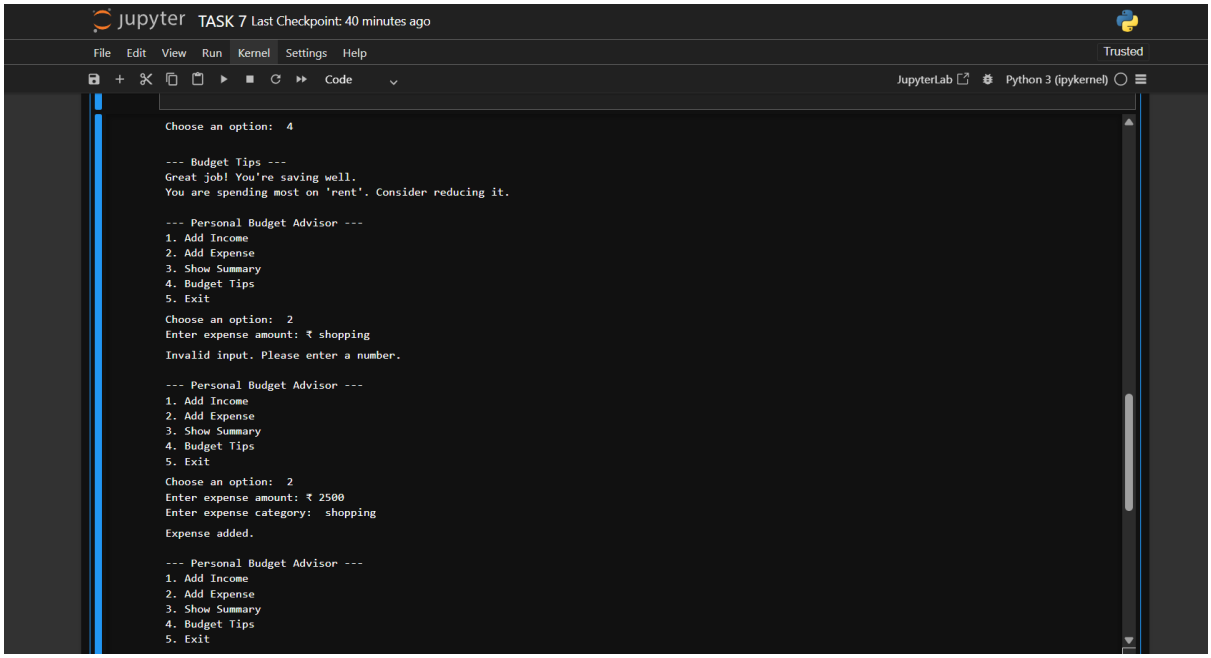

jupyter TASK 7 Last Checkpoint: 36 minutes ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)
print("\n--- Budget Summary ---")
print(f"Total Income: ₹{total_income:.2f}")
print(f"Total Expenses: ₹{total_expenses:.2f}")
print(f"Total Savings: ₹{savings:.2f}")
print("\nExpense Breakdown by Category:")
category_totals = {}
for expense in budget_data["expenses"]:
cat = expense["category"]
category_totals[cat] = category_totals.get(cat, 0) + expense["amount"]
for cat, total in category_totals.items():
percentage = (total / total_expenses) * 100 if total_expenses else 0
print(f"{cat}: ₹{total:.2f} ({percentage:.1f}%)")
print()
def budget_tips():
total_income = sum(item["amount"] for item in budget_data["income"])
total_expenses = sum(item["amount"] for item in budget_data["expenses"])
savings = total_income - total_expenses
print("\n--- Budget Tips ---")
if total_income == 0:
print("Add income to get tips.\n")
return
savings_ratio = (savings / total_income) * 100
if savings_ratio < 10:
print("Try to save at least 20% of your income.")
elif savings_ratio < 30:
print("Good! Try to increase your savings to 30%.")
else:

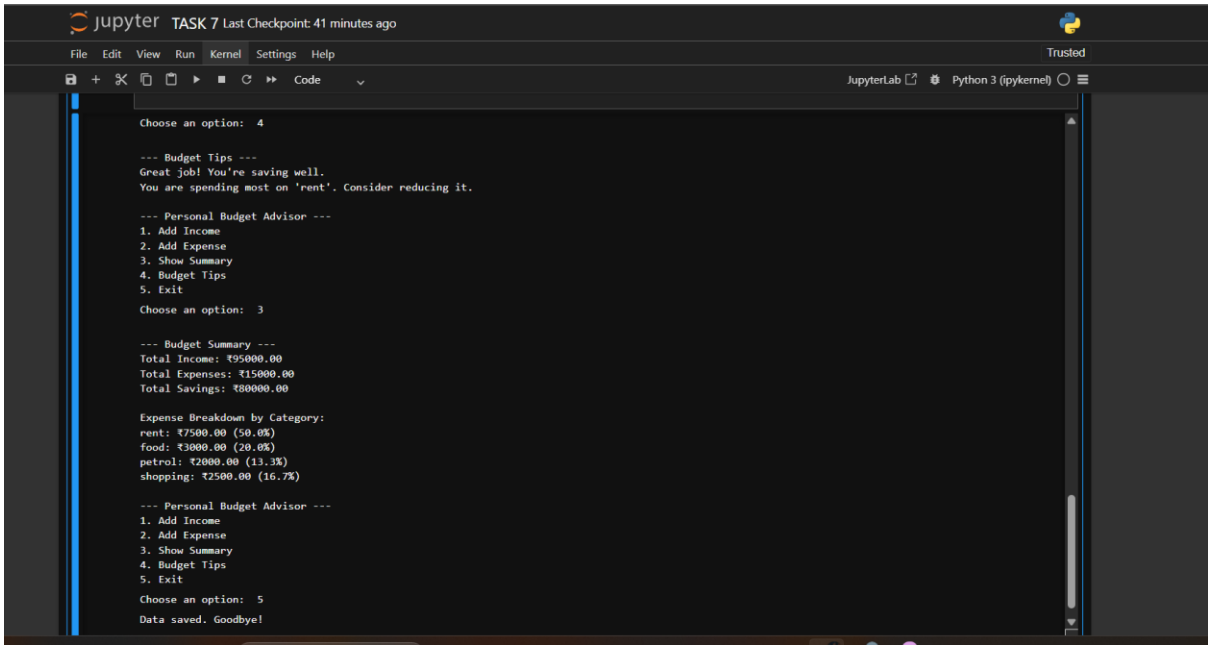

jupyter TASK 7 Last Checkpoint: 37 minutes ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)
else:
print("Great job! You're saving well.")
# Check high spending categories
category_totals = {}
for expense in budget_data["expenses"]:
cat = expense["category"]
category_totals[cat] = category_totals.get(cat, 0) + expense["amount"]
if category_totals:
max_category = max(category_totals, key=category_totals.get)
print(f"You are spending most on '{max_category}'. Consider reducing it.")
print()
def main():
while True:
print("--- Personal Budget Advisor ---")
print("1. Add Income")
print("2. Add Expense")
print("3. Show Summary")
print("4. Budget Tips")
print("5. Exit")
choice = input("Choose an option: ")
if choice == "1":
add_income()
elif choice == "2":
add_expense()
elif choice == "3":
show_summary()
elif choice == "4":
budget_tips()
elif choice == "5":


jupyter TASK 7 Last Checkpoint: 38 minutes ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)
print("5. Exit")
choice = input("Choose an option: ")
if choice == "1":
add_income()
elif choice == "2":
add_expense()
elif choice == "3":
show_summary()
elif choice == "4":
budget_tips()
elif choice == "5":
print("Data saved. Goodbye!")
break
else:
print("Invalid option. Please try again.\n")
if __name__ == "__main__":
main()
--- Personal Budget Advisor ---
1. Add Income
2. Add Expense
3. Show Summary
4. Budget Tips
5. Exit
Choose an option: 3
--- Budget Summary ---
Total Income: ₹0.00
Total Expenses: ₹0.00
Total Savings: ₹0.00


jupyter TASK 7 Last Checkpoint: 39 minutes ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)
Expense Breakdown by Category:
--- Personal Budget Advisor ---
1. Add Income
2. Add Expense
3. Show Summary
4. Budget Tips
5. Exit
Choose an option: 1
Enter income amount: ₹ 50000
Enter income source: business
Income added.
--- Personal Budget Advisor ---
1. Add Income
2. Add Expense
3. Show Summary
4. Budget Tips
5. Exit
Choose an option: 1
Enter income amount: ₹ 45000
Enter income source: trading
Income added.
--- Personal Budget Advisor ---
1. Add Income
2. Add Expense
3. Show Summary
4. Budget Tips
5. Exit
Choose an option: 2
Enter expense amount: ₹ 7500
Enter expense category: rent


jupyter TASK 7 Last Checkpoint: 39 minutes ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)
Choose an option: 2
Enter expense amount: ₹ 7500
Enter expense category: rent
Expense added.
--- Personal Budget Advisor ---
1. Add Income
2. Add Expense
3. Show Summary
4. Budget Tips
5. Exit
Choose an option: 2
Enter expense amount: ₹ 3000
Enter expense category: food
Expense added.
--- Personal Budget Advisor ---
1. Add Income
2. Add Expense
3. Show Summary
4. Budget Tips
5. Exit
Choose an option: 2
Enter expense amount: ₹ 2000
Enter expense category: petrol
Expense added.
--- Personal Budget Advisor ---
1. Add Income
2. Add Expense
3. Show Summary
4. Budget Tips
5. Exit


jupyter TASK 7 Last Checkpoint: 40 minutes ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)
Choose an option: 4
--- Budget Tips ---
Great job! You're saving well.
You are spending most on 'rent'. Consider reducing it.
--- Personal Budget Advisor ---
1. Add Income
2. Add Expense
3. Show Summary
4. Budget Tips
5. Exit
Choose an option: 2
Enter expense amount: ₹ shopping
Invalid input. Please enter a number.
--- Personal Budget Advisor ---
1. Add Income
2. Add Expense
3. Show Summary
4. Budget Tips
5. Exit
Choose an option: 2
Enter expense amount: ₹ 2500
Enter expense category: shopping
Expense added.
--- Personal Budget Advisor ---
1. Add Income
2. Add Expense
3. Show Summary
4. Budget Tips
5. Exit


jupyter TASK 7 Last Checkpoint: 41 minutes ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)
Choose an option: 4
--- Budget Tips ---
Great job! You're saving well.
You are spending most on 'rent'. Consider reducing it.
--- Personal Budget Advisor ---
1. Add Income
2. Add Expense
3. Show Summary
4. Budget Tips
5. Exit
Choose an option: 3
--- Budget Summary ---
Total Income: ₹95000.00
Total Expenses: ₹15000.00
Total Savings: ₹80000.00
Expense Breakdown by Category:
rent: ₹7500.00 (50.0%)
food: ₹3000.00 (20.0%)
petrol: ₹2000.00 (13.3%)
shopping: ₹2500.00 (16.7%)
--- Personal Budget Advisor ---
1. Add Income
2. Add Expense
3. Show Summary
4. Budget Tips
5. Exit
Choose an option: 5
Data saved. Goodbye!