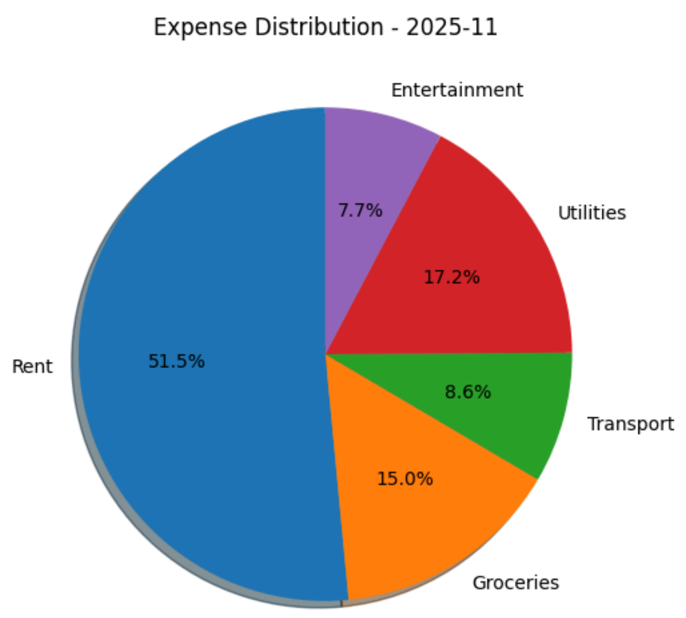
```
import datetime
import matplotlib.pyplot as plt

class MonthlyExpenseTracker:
    def __init__(self):
        self.transactions = []

    def add_transaction(self, amount, category, trans_type, date=None):
        if trans_type not in ['income', 'expense']:
            raise ValueError("Transaction type must be 'income' or 'expense'")
        if date is None:
            date = datetime.date.today()
        self.transactions.append({
            'amount': amount,
            'category': category,
            'type': trans_type,
            'date': date
        })

    def plot_monthly_expenses(self, year, month):
        category_expenses = {}
        for t in self.transactions:
            if t['type'] == 'expense' and t['date'].year == year and t['date'].month == month:
                category_expenses[t['category']] = category_expenses.get(t['category'], 0) + t['amount']

        if not category_expenses:
            print("No expenses recorded for this month.")
            return

        # Pie chart
        plt.figure(figsize=(6,6))
        plt.pie(category_expenses.values(), labels=category_expenses.keys(), autopct='%1.1f%%', startangle=90, shadow=True)
        plt.title(f"Expense Distribution - {year}-{month:02d}")
        plt.show()

        # Bar chart
        plt.figure(figsize=(8,5))
        plt.bar(category_expenses.keys(), category_expenses.values(), color='skyblue', edgecolor='black')
        plt.title(f"Expenses by Category - {year}-{month:02d}")
        plt.xlabel("Category")
        plt.ylabel("Amount Spent")
        plt.show()

# Example Usage
tracker = MonthlyExpenseTracker()
tracker.add_transaction(1200, 'Rent', 'expense', datetime.date(2025, 11, 2))
tracker.add_transaction(350, 'Groceries', 'expense', datetime.date(2025, 11, 10))
tracker.add_transaction(200, 'Transport', 'expense', datetime.date(2025, 11, 15))
tracker.add_transaction(400, 'Utilities', 'expense', datetime.date(2025, 11, 18))
tracker.add_transaction(180, 'Entertainment', 'expense', datetime.date(2025, 11, 21))

tracker.plot_monthly_expenses(2025, 11)
```

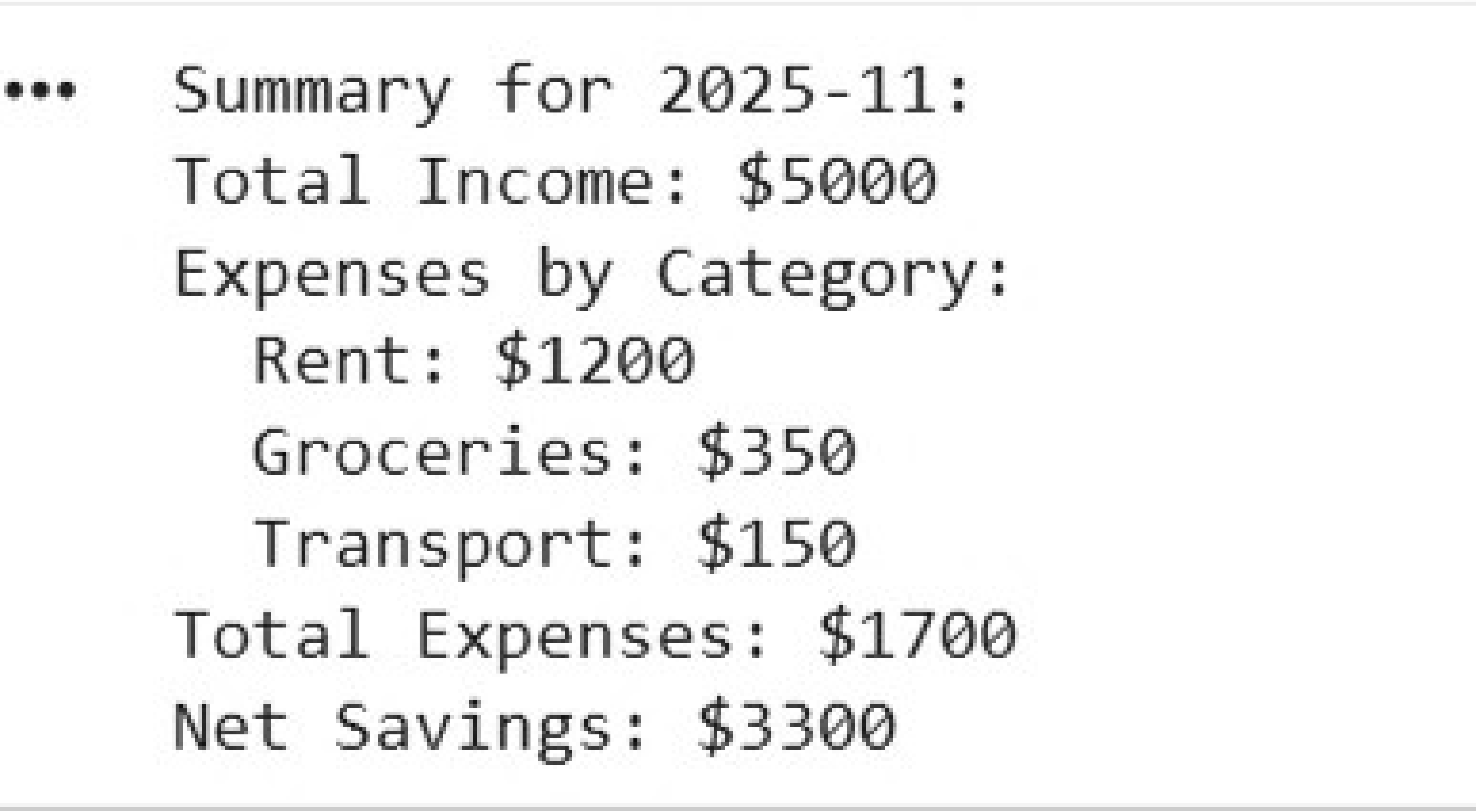


```
import datetime

class WeeklyExpenseTracker:
    def __init__(self):
        self.transactions = []

    def add_transaction(self, amount, category, trans_type, date=None):
        if trans_type not in ['income', 'expense']:
            raise ValueError("Transaction type must be 'income' or 'expense'")
        if date is None:
            date = datetime.date.today()
        self.transactions.append({
            'amount': amount,
            'category': category,
            'type': trans_type,
            'date': date
        })

    def week_summary(self, year, week_number):
        total_income = 0
        total_expense = 0
        category_expense = {}

        for t in self.transactions:
            tx_year, tx_week, _ = t['date'].isocalendar()
            if tx_year == year and tx_week == week_number:
                if t['type'] == 'income':
                    total_income += t['amount']
                else:
                    total_expense += t['amount']
                    category_expense[t['category']] = category_expense.get(t['category'], 0) + t['amount']

        print(f"Summary for Year {year}, Week {week_number}:")
        print(f"Total Income: ${total_income}")
        print("Expenses by Category:")
        for cat, amt in category_expense.items():
            print(f"  {cat}: ${amt}")
        print(f"Total Expenses: ${total_expense}")
        print(f"Net Savings: ${total_income - total_expense}")

# Example Usage
tracker = WeeklyExpenseTracker()
tracker.add_transaction(1200, 'Salary', 'income', datetime.date(2025, 11, 17))
tracker.add_transaction(300, 'Groceries', 'expense', datetime.date(2025, 11, 18))
tracker.add_transaction(100, 'Transport', 'expense', datetime.date(2025, 11, 20))
tracker.add_transaction(50, 'Entertainment', 'expense', datetime.date(2025, 11, 21))

# For week 47 of 2025 (which covers Nov 17 to Nov 23)
tracker.week_summary(2025, 47)
```

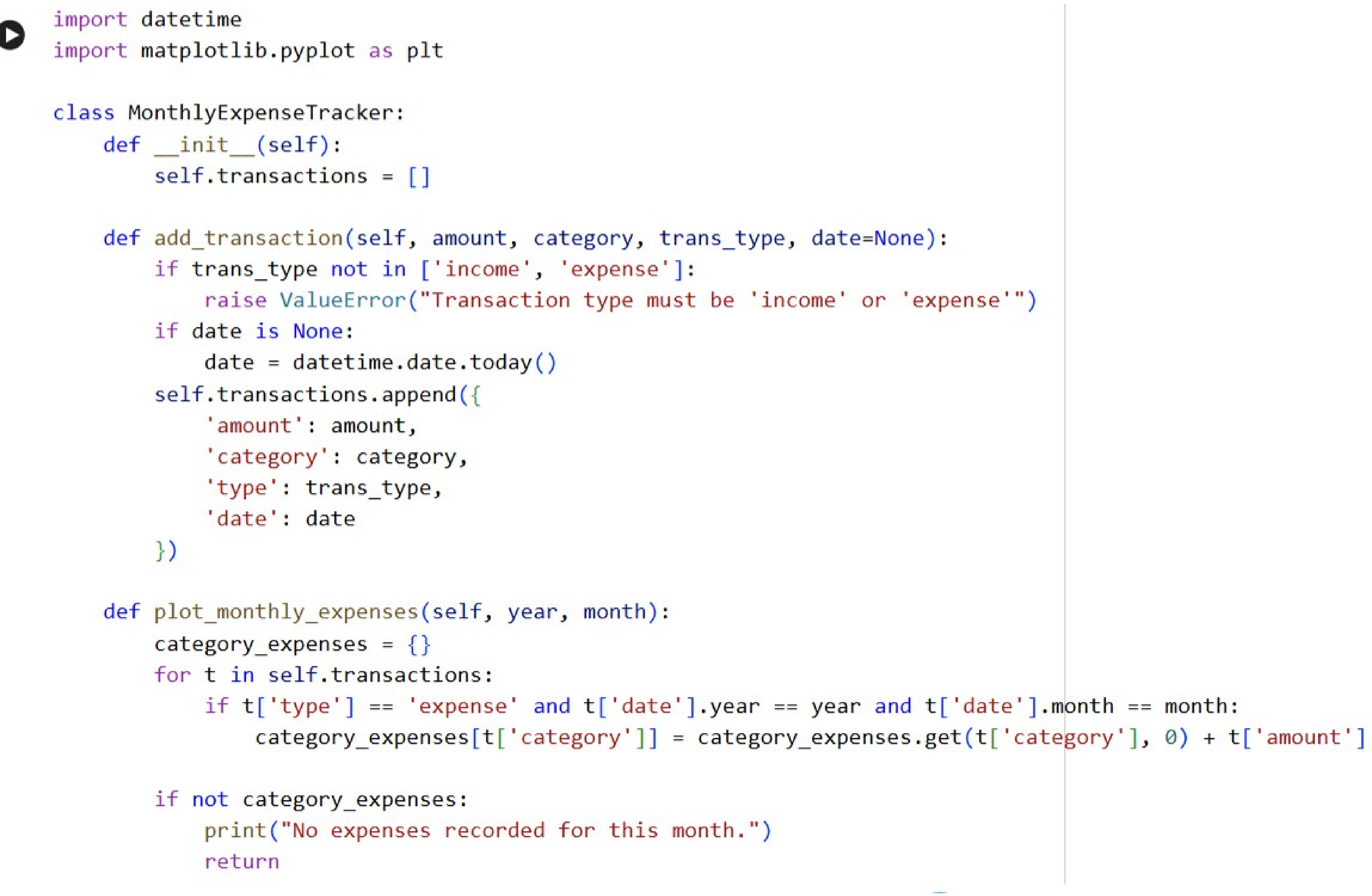
```
Summary for Year 2025, Week 47:
Total Income: $1200
Expenses by Category:
  Groceries: $300
  Transport: $100
  Entertainment: $50
Total Expenses: $450
Net Savings: $750
```

```python
import datetime

class MonthlyExpenseTracker:
    def __init__(self):
        self.transactions = []

    def add_transaction(self, amount, category, trans_type, date=None):
        if trans_type not in ['income', 'expense']:
            raise ValueError("Transaction type must be 'income' or 'expense'")
        if date is None:
            date = datetime.date.today()
        self.transactions.append({
            'amount': amount,
            'category': category,
            'type': trans_type,
            'date': date
        })

    def monthly_summary(self, year, month):
        total_income = 0
        total_expense = 0
        category_expense = {}

        for t in self.transactions:
            if t['date'].year == year and t['date'].month == month:
                if t['type'] == 'income':
                    total_income += t['amount']
                else:
                    total_expense += t['amount']
                    category_expense[t['category']] = category_expense.get(t['category'], 0) + t['amount']

        print(f"Summary for {year}-{month:02d}:")
        print(f"Total Income: ${total_income}")
        print("Expenses by Category:")
        for cat, amt in category_expense.items():
            print(f"  {cat}: ${amt}")
        print(f"Total Expenses: ${total_expense}")
        print(f"Net Savings: ${total_income - total_expense}")

# Example Usage
tracker = MonthlyExpenseTracker()
tracker.add_transaction(5000, 'Salary', 'income', datetime.date(2025, 11, 1))
tracker.add_transaction(1200, 'Rent', 'expense', datetime.date(2025, 11, 2))
tracker.add_transaction(350, 'Groceries', 'expense', datetime.date(2025, 11, 10))
tracker.add_transaction(150, 'Transport', 'expense', datetime.date(2025, 11, 15))

tracker.monthly_summary(2025, 11)
```

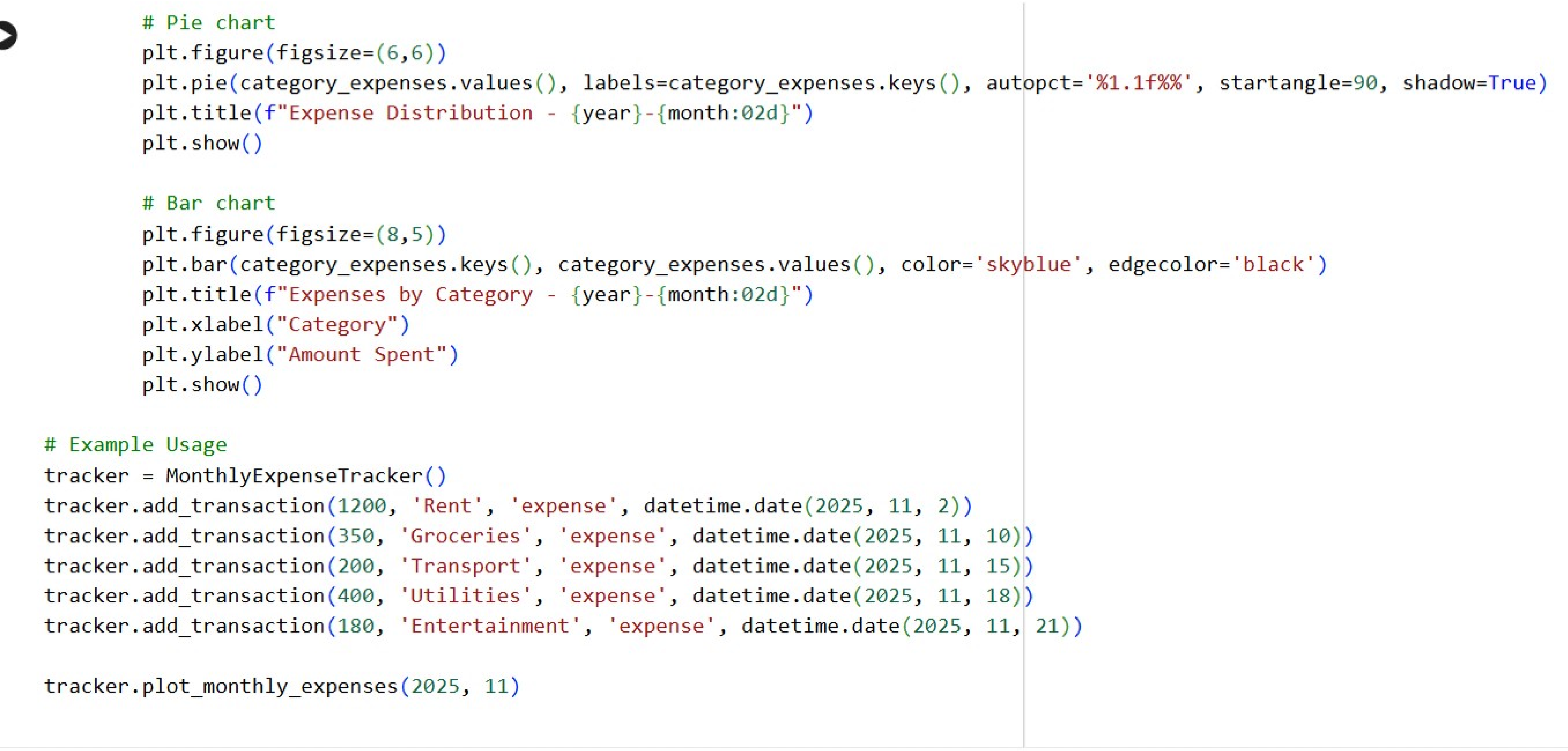
```
Summary for 2025-11:
Total Income: $5000
Expenses by Category:
  Rent: $1200
  Groceries: $350
  Transport: $150
Total Expenses: $1700
Net Savings: $3300
```



```
Summary for 2025-11:
Total Income: $5000
Expenses by Category:
  Rent: $1200
  Groceries: $350
  Transport: $150
Total Expenses: $1700
Net Savings: $3300
```

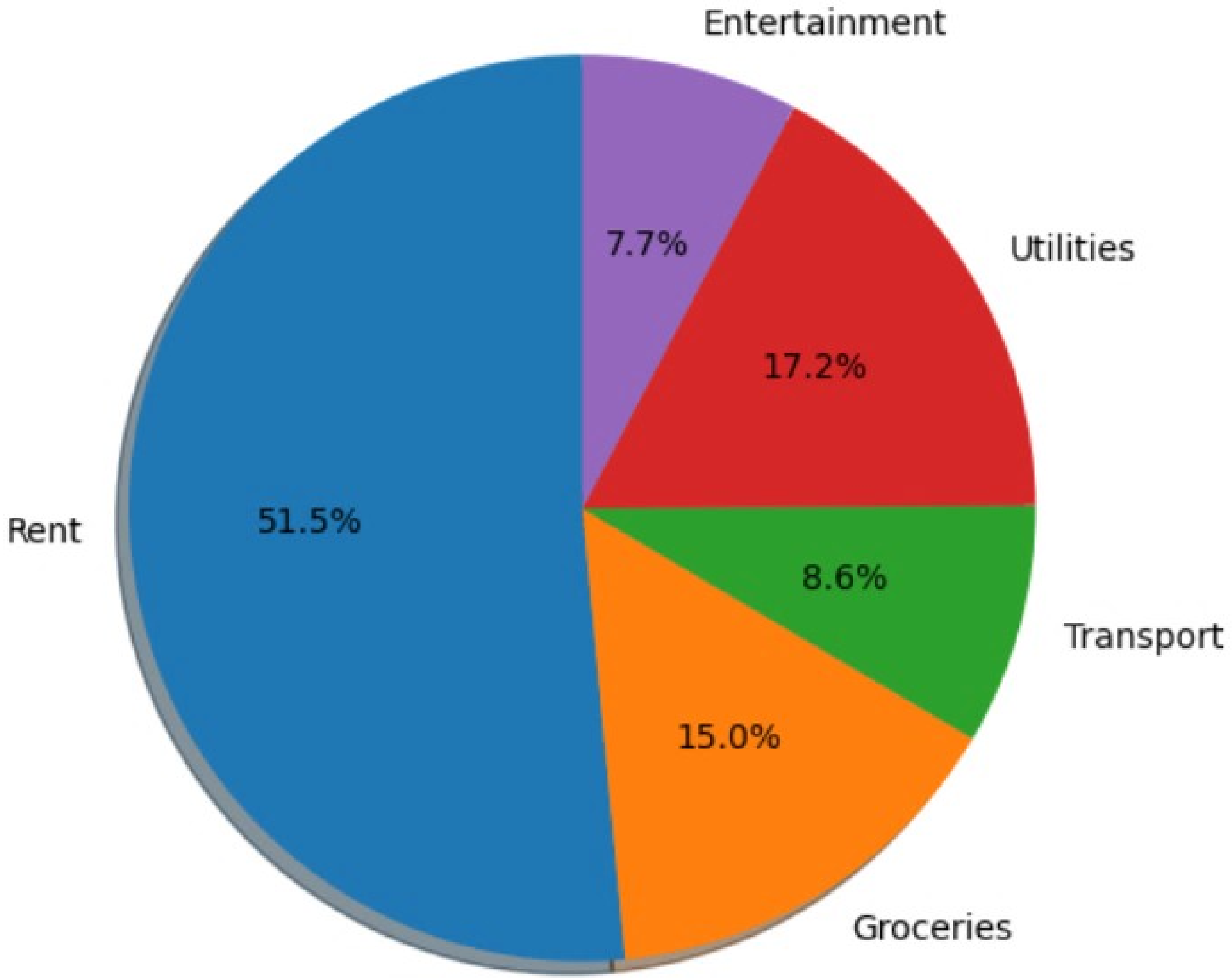


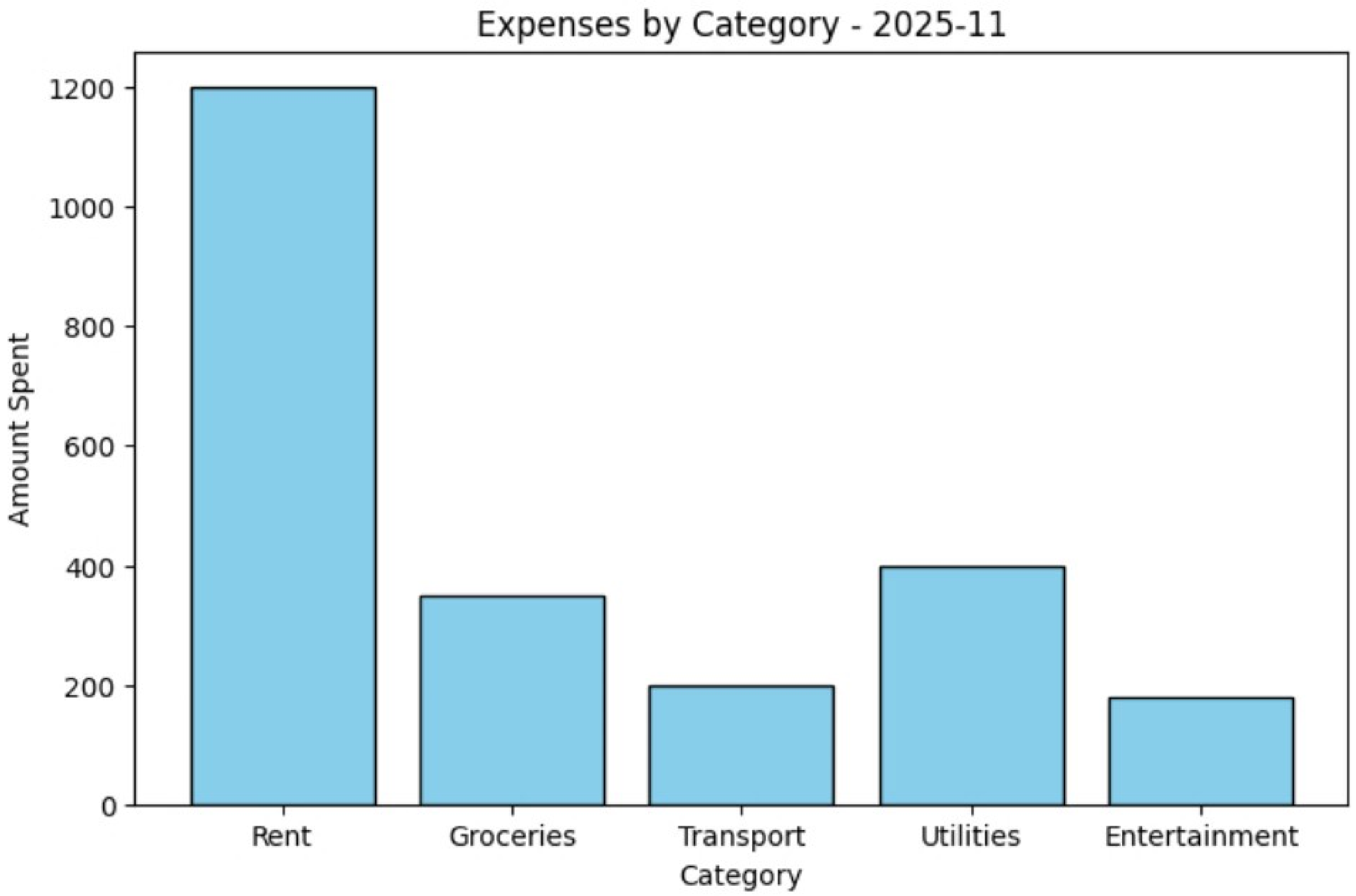
```python
import datetime
import matplotlib.pyplot as plt

class MonthlyExpenseTracker:
    def __init__(self):
        self.transactions = []

    def add_transaction(self, amount, category, trans_type, date=None):
        if trans_type not in ['income', 'expense']:
            raise ValueError("Transaction type must be 'income' or 'expense'")
        if date is None:
            date = datetime.date.today()
        self.transactions.append({
            'amount': amount,
            'category': category,
            'type': trans_type,
            'date': date
        })

    def plot_monthly_expenses(self, year, month):
        category_expenses = {}
        for t in self.transactions:
            if t['type'] == 'expense' and t['date'].year == year and t['date'].month == month:
                category_expenses[t['category']] = category_expenses.get(t['category'], 0) + t['amount']

        if not category_expenses:
            print("No expenses recorded for this month.")
            return
```



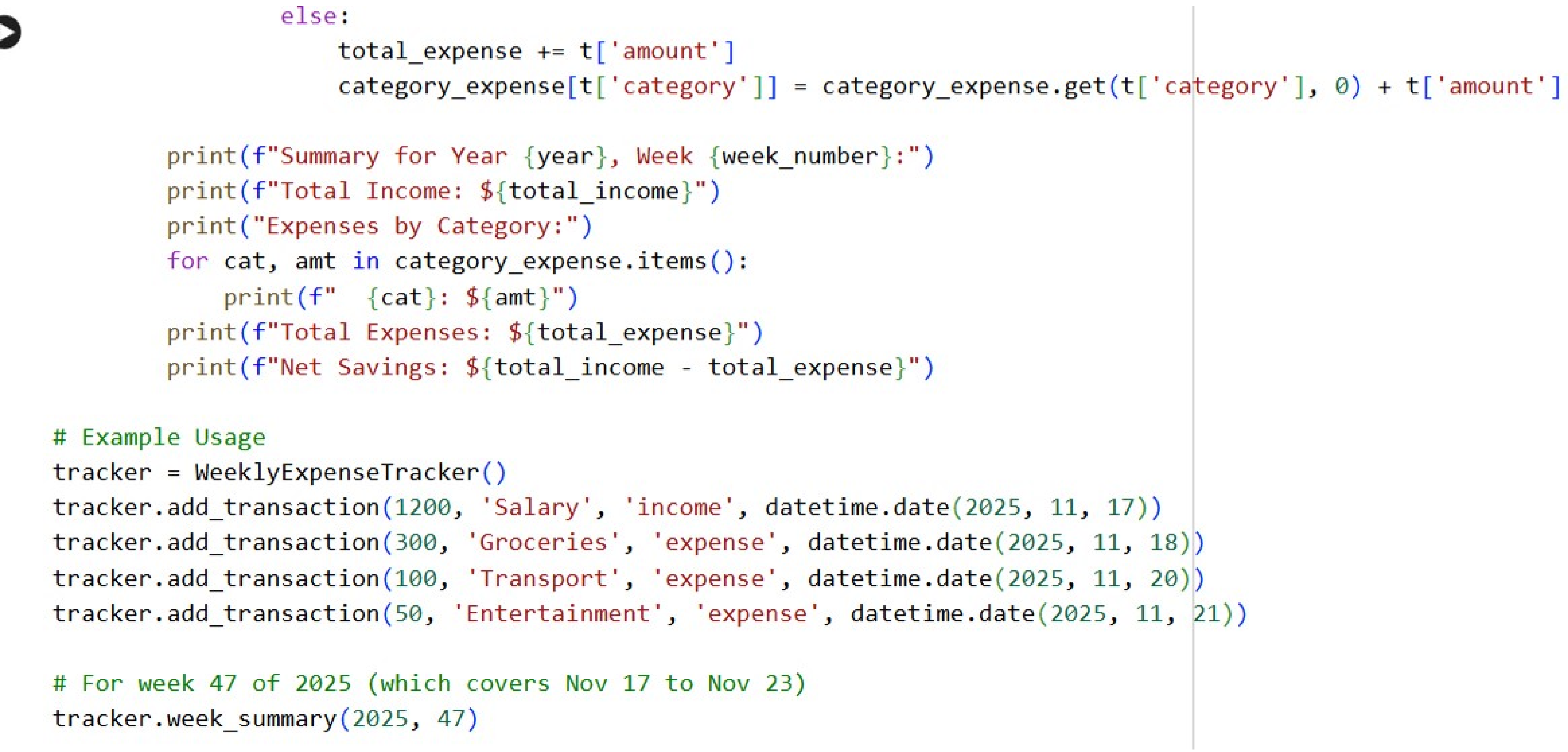
```
        # Pie chart
        plt.figure(figsize=(6,6))
        plt.pie(category_expenses.values(), labels=category_expenses.keys(), autopct='%1.1f%%', startangle=90, shadow=True)
        plt.title(f"Expense Distribution - {year}-{month:02d}")
        plt.show()

        # Bar chart
        plt.figure(figsize=(8,5))
        plt.bar(category_expenses.keys(), category_expenses.values(), color='skyblue', edgecolor='black')
        plt.title(f"Expenses by Category - {year}-{month:02d}")
        plt.xlabel("Category")
        plt.ylabel("Amount Spent")
        plt.show()

# Example Usage
tracker = MonthlyExpenseTracker()
tracker.add_transaction(1200, 'Rent', 'expense', datetime.date(2025, 11, 2))
tracker.add_transaction(350, 'Groceries', 'expense', datetime.date(2025, 11, 10))
tracker.add_transaction(200, 'Transport', 'expense', datetime.date(2025, 11, 15))
tracker.add_transaction(400, 'Utilities', 'expense', datetime.date(2025, 11, 18))
tracker.add_transaction(180, 'Entertainment', 'expense', datetime.date(2025, 11, 21))

tracker.plot_monthly_expenses(2025, 11)
```

Expense Distribution - 2025-11

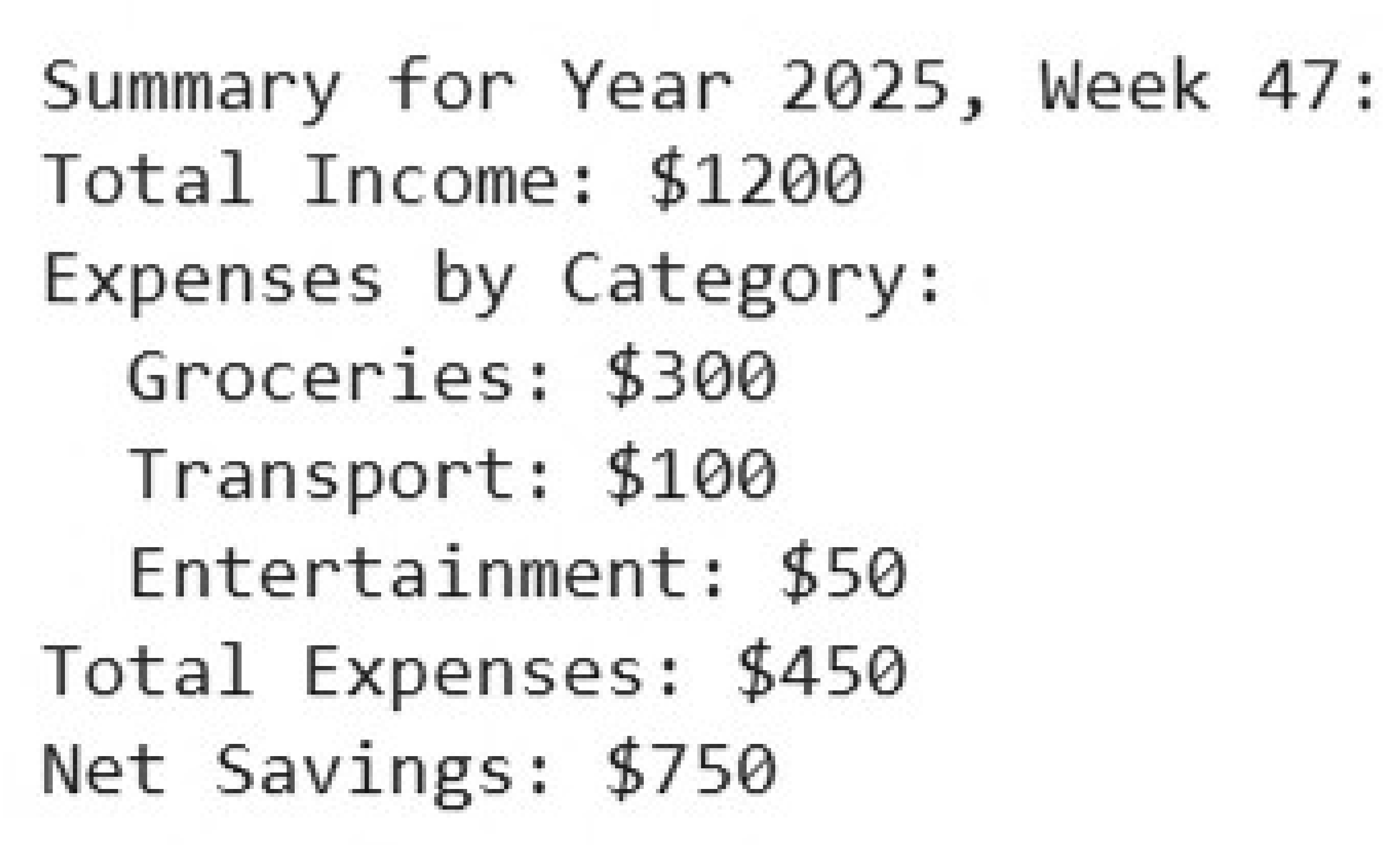



Expenses by Category - 2025-11
Amount Spent
0
200
400
600
800
1000
1200
Rent
Groceries
Transport
Utilities
Entertainment
Category

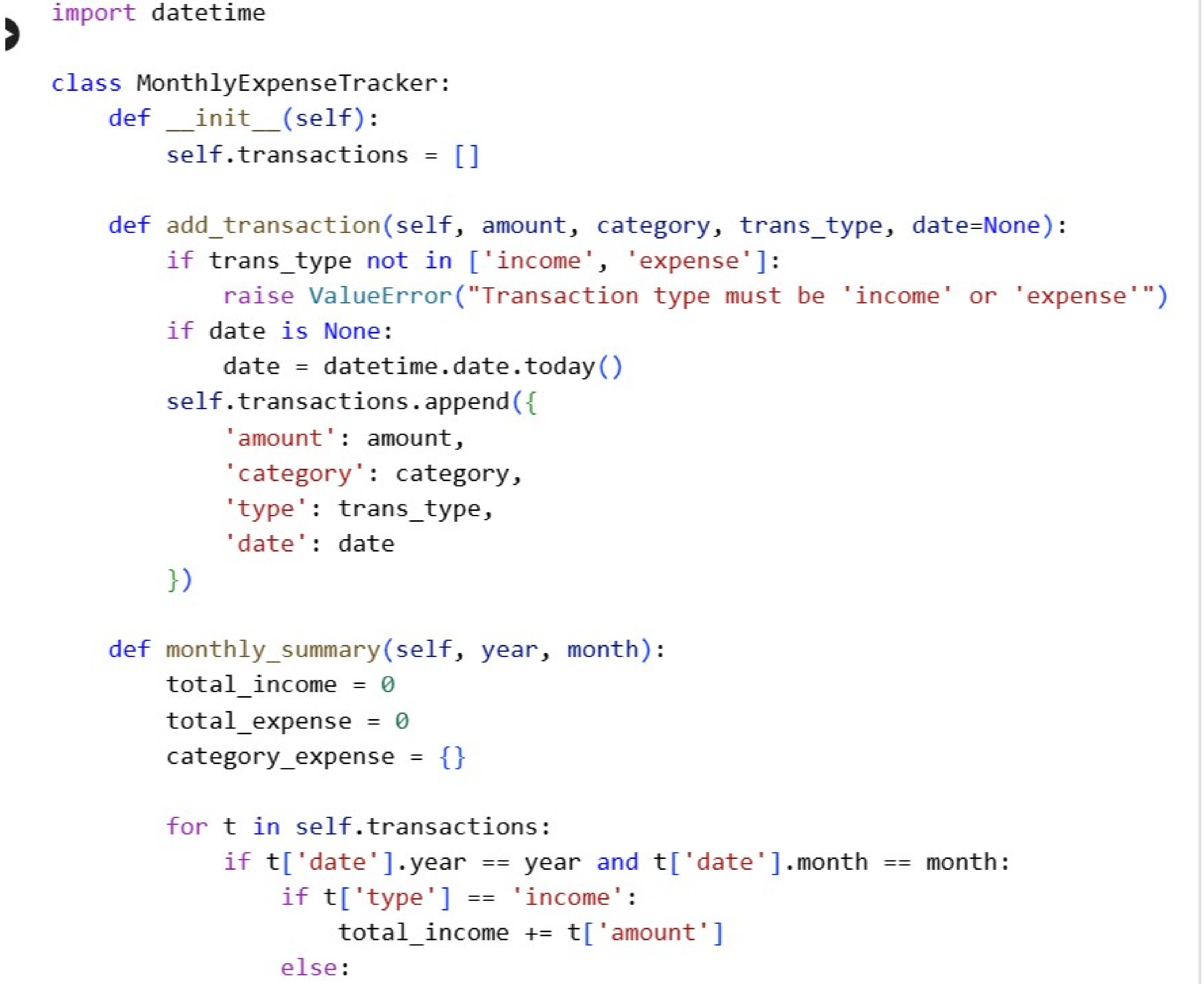
```python
import datetime

class WeeklyExpenseTracker:
    def __init__(self):
        self.transactions = []

    def add_transaction(self, amount, category, trans_type, date=None):
        if trans_type not in ['income', 'expense']:
            raise ValueError("Transaction type must be 'income' or 'expense'")
        if date is None:
            date = datetime.date.today()
        self.transactions.append({
            'amount': amount,
            'category': category,
            'type': trans_type,
            'date': date
        })

    def week_summary(self, year, week_number):
        total_income = 0
        total_expense = 0
        category_expense = {}

        for t in self.transactions:
            tx_year, tx_week, _ = t['date'].isocalendar()
            if tx_year == year and tx_week == week_number:
                if t['type'] == 'income':
                    total_income += t['amount']
```



```python
                else:
                    total_expense += t['amount']
                    category_expense[t['category']] = category_expense.get(t['category'], 0) + t['amount']

        print(f"Summary for Year {year}, Week {week_number}:")
        print(f"Total Income: ${total_income}")
        print("Expenses by Category:")
        for cat, amt in category_expense.items():
            print(f"  {cat}: ${amt}")
        print(f"Total Expenses: ${total_expense}")
        print(f"Net Savings: ${total_income - total_expense}")

# Example Usage
tracker = WeeklyExpenseTracker()
tracker.add_transaction(1200, 'Salary', 'income', datetime.date(2025, 11, 17))
tracker.add_transaction(300, 'Groceries', 'expense', datetime.date(2025, 11, 18))
tracker.add_transaction(100, 'Transport', 'expense', datetime.date(2025, 11, 20))
tracker.add_transaction(50, 'Entertainment', 'expense', datetime.date(2025, 11, 21))

# For week 47 of 2025 (which covers Nov 17 to Nov 23)
tracker.week_summary(2025, 47)
```



```
Summary for Year 2025, Week 47:
Total Income: $1200
Expenses by Category:
  Groceries: $300
  Transport: $100
  Entertainment: $50
Total Expenses: $450
Net Savings: $750
```



```python
import datetime

class MonthlyExpenseTracker:
    def __init__(self):
        self.transactions = []

    def add_transaction(self, amount, category, trans_type, date=None):
        if trans_type not in ['income', 'expense']:
            raise ValueError("Transaction type must be 'income' or 'expense'")
        if date is None:
            date = datetime.date.today()
        self.transactions.append({
            'amount': amount,
            'category': category,
            'type': trans_type,
            'date': date
        })

    def monthly_summary(self, year, month):
        total_income = 0
        total_expense = 0
        category_expense = {}

        for t in self.transactions:
            if t['date'].year == year and t['date'].month == month:
                if t['type'] == 'income':
                    total_income += t['amount']
                else:
```

```python
                else:
                    total_expense += t['amount']
                    category_expense[t['category']] = category_expense.get(t['category'], 0) + t['amount']

        print(f"Summary for {year}-{month:02d}:")
        print(f"Total Income: ${total_income}")
        print("Expenses by Category:")
        for cat, amt in category_expense.items():
            print(f"  {cat}: ${amt}")
        print(f"Total Expenses: ${total_expense}")
        print(f"Net Savings: ${total_income - total_expense}")

# Example Usage
tracker = MonthlyExpenseTracker()
tracker.add_transaction(5000, 'Salary', 'income', datetime.date(2025, 11, 1))
tracker.add_transaction(1200, 'Rent', 'expense', datetime.date(2025, 11, 2))
tracker.add_transaction(350, 'Groceries', 'expense', datetime.date(2025, 11, 10))
tracker.add_transaction(150, 'Transport', 'expense', datetime.date(2025, 11, 15))

tracker.monthly_summary(2025, 11)
```