```
#include<iostream>
#include <string>
using namespace std;

struct Expense {
    string name;
    double amount;
    Expense* next;

    Expense(string n, double a) : name(n),
amount(a), next(NULL) {}
};

struct Month {
    string name;
    Expense* expenses;
    Month* next;

    Month(string n) : name(n),
expenses(NULL), next(NULL) {}
};

class Expensetr  {
private:
    Month* head;
```

```
public:
    Expensetr() : head(NULL) {}

    Month* addMonth(string monthName) {
        Month* newMonth = new
Month(monthName);
        if (head == NULL) {
            head = newMonth;
        } else {
            Month* temp = head;
            while (temp->next != NULL) {
                temp = temp->next;
            }
            temp->next = newMonth;
        }
        return newMonth;
    }

    Month* getMonth(string monthName) {
        Month* temp = head;
        while (temp != NULL) {
            if (temp->name == monthName) {
                return temp;
            }
```

```
            temp = temp->next;
        }
        return addMonth(monthName); // Create if not found
    }

    void addExpense(string monthName, string itemName, double amount) {
        Month* month = getMonth(monthName);
        Expense* newExpense = new Expense(itemName, amount);

        newExpense->next = month->expenses;
        month->expenses = newExpense;

        cout <<endl<< "Added pkr" << amount << " for " << itemName << " to " << monthName << endl;
    }

    void showExpenses(string monthName) {
```

```
        Month* month =
getMonth(monthName);
        Expense* temp = month->expenses;

        cout << "\nExpenses for " <<
monthName << ":" << endl;
        while (temp != NULL) {
            cout << "- " << temp->name << ":
PKR" << temp->amount << endl;
            temp = temp->next;
        }
    }

    void showMostExpensive(string
monthName) {
        Month* month =
getMonth(monthName);
        if (month->expenses == NULL) {
            cout << "No expenses in " <<
monthName << endl;
            return;
        }

        Expense* temp = month->expenses;
        Expense* maxExpense = temp;
```

```
        while (temp != NULL) {
            if (temp->amount >
maxExpense->amount) {
                maxExpense = temp;
            }
            temp = temp->next;
        }

        cout << "Most expensive in " <<
monthName << ": "
             << maxExpense->name << "
(PKR" << maxExpense->amount << ")" <<
endl;
    }


    void deleteExpense(string monthName,
string itemName) {
        Month* month =
getMonth(monthName);
        Expense* current =
month->expenses;
        Expense* prev = NULL;

        while (current != NULL) {
            if (current->name == itemName) {
```

```
                if (prev == NULL) {
                    month->expenses =
current->next;
                } else {
                    prev->next = current->next;
                }
                delete current;
                cout << "Deleted " << itemName
<< " from " << monthName << endl;
                return;
            }
            prev = current;
            current = current->next;
        }

        cout << itemName << " not found in "
<< monthName << endl;
    }
};

int main() {
    Expensetr E;

    // Adding expenses
    cout<<"*****************WELLCOME******
* * * ***************"<<endl;
```

```
    E.addExpense("MARCH", "Groceries",
1150.25);
     cout<<"**********************************
* * * *"<<endl;
    E.addExpense("MARCH", "Rent",
12000.45);
     cout<<"**********************************
* * * *"<<endl;
    E.addExpense("APRIL", "Dining",
275.25);
     cout<<"**********************************
* * * *"<<endl;
    // Showing expenses
    E.showExpenses("MARCH");
     cout<<"**********************************
* * * *"<<endl;
    E.showExpenses("APRIL");
     cout<<"**********************************
* * * *"<<endl;
    // Most expensive
    E.showMostExpensive("MARCH");
     cout<<"**********************************
* * * *"<<endl;

    // Deleting an expense
```

```
    E.deleteExpense("MARCH",
"Groceries");
     cout<<"**********************************
* * * *"<<endl;
    E.showExpenses("MARCH");
     cout<<"**********************************
* * * *"<<endl;

    return 0;
}
```