Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Петрозаводский государственный университет»

Физико-технический институт

Кафедра информационно-измерительных систем и физической электроники

«ПЕКАРНЯ»

ИНДИВИДУАЛЬНЫЙ ОТЧЁТ

Командный проект по дисциплине «Технология программирования»

Выполнил:

студент группы 21312

А. И. Шаркевич

Научный руководитель:

канд. физ.-мат. наук, доцент КИИСиФЭ А. В. Бульба

Петрозаводск 2023

История коммитов:

1. **Expense.h, ExpenseInputScreen.h, ExpenseList.h** (ШАИ)  – В заголовочные файлы были добавлены прототипы класса Expense, ExpenseInputScreen и ExpenseList.
2. **Исправил и дополнил методы Expense.h и** **ExpenseList.h** (ШАИ)  – Были внесены изменения в прототипы классов Expense и ExpenseList.
3. **Добавлены get-теры в заголовочный файл ExpenseList.h, расширены поля ExpenseInputScreen.h** (ШАИ)  – Были добавлены методы Get в прототип класса Expense, а так же добавлены новые поля в прототип метода ExpenseList.
4. **Добавлены get-теры в заголовочный файл ExpenseList.h, расширены поля ExpenseInputScreen.h. Подправлены комментарии** (ШАИ)  –  Предыдущий коммит, только в этом были изменены комментарии к строкам кода.
5. **Merge branch 'develop-Senya'  into develop-general** (ШАИ)  –  Залил все наработки со своей ветки в основную ветку разработки.
6. **Merge branch 'develop-general' into develop-Senya** (ШАИ)  – Залил все наработки с 'develop-general' ветки в свою  ветку разработки.
7. **Merge branch 'develop-general' into develop-Senya** (ШАИ)  – Залил все наработки с 'develop-general' ветки в свою  ветку разработки.
8. **Реализованы классы Expense, ExpenseList и ExpenseInputScreen** (ШАИ)  – Была написана реализация классов Expense, ExpenseList и ExpenseInputScreen.
9. **Подправлены комментарии к методам** (ШАИ)  – Были исправлены неточности в комментариях к методам.
10. **Merge branch 'develop-Senya' into develop-general** (ШАИ) – Залил все наработки с ветки 'develop-Senya'в основную ветку разработки.

**EXPENSE.h**

#ifndef EXPENSE

#define EXPENSE

#include "libs.h"

//Класс затрат

class Expense

{

private:

string category; // категория расходов (газ, свет, ремонт и тд)

string name; //название расхода

string date; // дата уплаты расходов

float amount; // сколько платим

public:

Expense(string c, string n, string d, float a):

category(c), name(n), date(d), amount(a)

{ }

~Expense();

//геттеры:

string GetCategory();

string GetName();

string GetDate();

float GetAmount();

};

#endif // EXPENSE

**EXPENSEINPUTSCREEN.h**

#ifndef EXPENSEINPUTSCREEN

#define EXPENSEINPUTSCREEN

#include "ExpenseList.h"

//Экран для ввода расходов

class ExpenseInputScreen

{

private:

ExpenseList\* ptrExpenseList; // запись о расходах

string ExpenseCategory; // категория расходов (газ, свет, ремонт и тд)

string ExpenseName; //название расхода

string ExpenseDate; // дата уплаты расходов

float ExpenseAmount; // сколько платим

public:

ExpenseInputScreen(ExpenseList\* ptrExp): ptrExpenseList(ptrExp)

{ }

void SetExpense(); //внести расход

};

#endif // EXPENSEINPUTSCREEN

**EXPENSELIST.h**

#ifndef EXPENSELIST

#define EXPENSELIST

#include "Expense.h"

//Класс записей о расходах

class ExpenseList

{

private:

vector<Expense\*> vectPtrsExpenses; //вектор указателей на расходы

vector<Expense\*>::iterator iter;

public:

~ExpenseList();

void InsertExpense(Expense\*); //внести расход

float DisplaySummExpenses(); //вывести сумму всех расходов

void DisplayExp(); //вывести строку расходов

//геттеры:

//string GetCategoryExpense(string);

//string GetDateExpense(string);

//float GetAmountExpense(string);

};

#endif // EXPENSELIST

**Expense.cpp**

#include "Expense.h"

Expense::~Expense() // деструктор

{ }

string Expense::GetCategory()

{

return category;

}

string Expense::GetName()

{

return name;

}

string Expense::GetDate()

{

return date;

}

float Expense::GetAmount()

{

return amount;

}

**ExpenseInputScreen.cpp**

#include "ExpenseInputScreen.h"

void ExpenseInputScreen::SetExpense() // добавить данные о расходе

{

cout << "Input the category of expense: " << endl;

getaLine(ExpenseCategory);

cout << "Input the name of expense: " << endl;

getaLine(ExpenseName);

cout << "Input the date of expense: " << endl;

getaLine(ExpenseDate);

cout << "Input the amount of expense: " << endl;

cin >> ExpenseAmount;

cin.ignore(80, '\n');

Expense\* ptrExpense = new Expense (ExpenseCategory, ExpenseName, ExpenseDate, ExpenseAmount); // создать расход

ptrExpenseList->InsertExpense(ptrExpense); // занести в список расходов

}

**ExpenseList.cpp**

#include "ExpenseList.h"

ExpenseList::~ExpenseList() // деструктор

{ // удалить объекты expense

// удалить указатели на вектор

while (!vectPtrsExpenses.empty())

{

iter = vectPtrsExpenses.begin();

delete \*iter;

vectPtrsExpenses.erase(iter);

}

}

void ExpenseList::InsertExpense(Expense\* ptrExp)

{

vectPtrsExpenses.push\_back(ptrExp);

}

void ExpenseList::DisplayExp() // отобразить все строки расходов

{

cout << endl;

cout << setw(15) << "Category" << setw(15) << "Name"

<< setw(15) << "Date" << setw(15) << "Amount";

cout << endl << "----------------------------------------------------------------------" << endl;

if (vectPtrsExpenses.empty()) // если список расходов пуст

cout << "\*\*\*There's nothing here\*\*\*\n" << endl; // выводим запись, что он пуст)

else

{

iter = vectPtrsExpenses.begin();

while (iter != vectPtrsExpenses.end()) // выводим все расходы

{

cout << setw(15) << (\*iter)->GetCategory()

<< setw(15) << (\*iter)->GetName()

<< setw(15) << (\*iter)->GetDate()

<< setw(15) << (\*iter)->GetAmount() << endl;

\*iter++;

}

}

}

float ExpenseList::DisplaySummExpenses() // сумма всех платежей

{

float summExpenses = 0.0;

iter = vectPtrsExpenses.begin();

while (iter != vectPtrsExpenses.end())

{ // плюсуем суммы всех платежей жильцов за все время

summExpenses += (\*iter)->GetAmount();

iter++;

}

return summExpenses;

}