**2. СТРУКТУРНОЕ ПРОЕКТИРОВАНИЕ**

**2.1 Общее описание модулей**

Для удобства разработки курсового проекта, а также для последующего его модификация было принято решение конструировать приложение на основе отдельных, связанных между собой методами модулей. Полная структурная схема проекта представлена в ПРИЛОЖЕНИИ Б.

Каждый отдельный модуль представляет собой класс, которые представлены на диаграмме классов (см. ПРИЛОЖЕНИЕ А). Ниже будет приведено краткое описание каждого из модулей.

Модуль работы с датой представлен классом Date. В этом классе представлены методы для работы с атрибутами, которые отражают дату кого-либо дня (\_day, \_month, \_year).

Модуль работы с деньгами представлен классом Money. В классе имеются методами для обработки данных отражающие значения некой суммы денег (\_money).

Модуль работы с текстом представлен классом Text. В классе реализованы методы для обработки данных, отражающие различные строки информации(\_str).

Модуль работы с долгами создан для полного отражения всех манипуляция, которые следует производить с такими понятиями как долги и должники. Модуль представлен классом Debts и является производным от классов Text, Money и Date (см. ПРИЛОЖЕНИЕ А). Сам класс имеет собственные методы для обработки информации, относящийся к понятиям долги и должники.

Модуль работы с расходами и доходами представлен двумя классами: Record, Cost\_or\_Income. Отношения между этими классами представлено N-арной ассоциацией. Класс Cost\_or\_Income является производным от трех классов: Text, Money и Date. Класс содержит методы для предоставления и обработки информации, относящейся к информации о доходах и расходах. Класс Record является производным от класса Date и имеет методы для обработки данных, представляющие информации о множестве расходов(\_cost) и доходов (\_income).

Модуль работы всей программы создан для структурирования и всех основных модулей работы программы. Представлен модуль классом Main. Сам класс связан с классами Record, Cost\_or\_Income, Debts, Money с помощью ассоциации (N-арной с Record, Cost\_or\_Income и Debts и бинарной с Money) Модуль объединяет методы для взаимодействия с данными, которые отражают все классы, что указаны выше. Так же этот модуль предназначен для сохранения всех данных пользователя и последующего их воспроизведения при повторном запуске программы. Класс Main представлен в ПРИЛОЖЕНИИ К.

Модуль пользовательского интерфейса предоставляет пользователю возможность взаимодействовать с программой и со всеми её модулями. Модуль представлен шаблонным классом Menu. Класс Menu представлен на диаграмме классов в ПРИЛОЖЕНИИ А.

Модуль, связывающий модуль пользовательского интерфейса и модуль работы всей программе необходим для взаимодействия команд, которых пользователь отдает в модуле пользовательского интерфейса с модулями, отвечающими за работы всей программы. Класс, отражающий этом модуль, называется MainMenu. Сам класс является производным от класса Main. Класс MainMenu вынесен в ПРИЛОЖЕНИИ Ж.

Более детальное описание модулей программы представлено в разделе 4. Разработка программных модулей.

**2.2 Связь между модулями**

Как уже было сказано выше, модули связаны между собой методами классов. Некоторые методы наследованы (см. ПРИЛОЖЕНИЕ А) от базовых классов, другие написаны специально. Ниже будут перечислены методы, связывающие модули между собой.

Модуль работы с текстом связан с модулем работы с долгами следующими методами:

1. std::string get\_name\_debts()const;
2. void set\_name\_debts(std::string str);

Модуль работы с текстом связан с модулем работы с расходами и доходами следующими методами:

1. std::string get\_name\_cost\_or\_income()const;
2. void change\_name\_cost\_or\_income(std::string str);

Модуль работы с датой связан с модулем работы с долгами следующими методами:

1. void set\_day\_repayment(short int day);
2. void set\_month\_repayment(short int month);
3. void set\_year\_repayment(short int year);
4. short int get\_day\_repayment()const;
5. short int get\_month\_repayment()const;
6. short int get\_year\_repayment()const;

Модуль работы с датой связан с модулем работы с расходами и доходами следующими методами:

1. void change\_data\_cost(int index, short int day, short int month, short int year);
2. void change\_data\_income(int index, short int day, short int month, short int year);

Модуль работы с деньгами связан с модулем работы с расходами и доходами следующими методами:

1. void change\_money\_cost(int index, float money);

2. void change\_money\_income(int index, float money);

Модуль работы с деньгами связан с модулем работы долгами следующими методами:

1. void set\_money\_repayment(float money);
2. void get\_money\_repayment(float money);

Модуль работы с деньгами связан с модулем работы всей программы следующими методами:

1. float get\_cash\_flow()const;
2. void set\_cash\_flow(float money);

Модуль работы всей программы связан с модулем работы с расходами и доходами и модулем работы с долгами методами, которые вынесены в ПРИЛОЖЕНИЕ Е.

Модуль работы всей программы связан с модулем работы*,* связывающего интерфейс с модулем работы всей программы всеми методами, что присутствуют в классе MainMenu (см. ПРИЛОЖЕНИЕ Ж).

То, как связан модуль пользовательского интерфейса с модулем работы*,* связывающего интерфейс с модулем работы всей программы показано в файле

main.cpp (см. ПРИЛОЖЕНИЕ И).

Более детальное описание этих методов приведено в разделе 3. Функциональное проектирование и в разделе 4. Разработка программных модулей.