Министерство образования и науки Российской Федерации  
Федеральное‌ ‌государственное‌ ‌бюджетное‌ ‌образовательное‌ ‌учреждение‌

высшего‌ ‌образования‌

**«Пермский национальный исследовательский  
политехнический университет»**

Кафедра «Информационные технологии и автоматизированные системы»

**О Т Ч Ё Т**

**по творческому заданию**

Тема: Разработка калькулятора

Выполнил работу

студент группы РИС-20-2б

Катаев М.М.

Проверила

Доцент кафедры ИТАС

к.т.н ПоляковаО.А.

Пермь, 2021

**Постановка задачи**

1. Разработать алгоритм калькулятора в соответствии с заданием.
2. Реализовать алгоритм в виде программы на алгоритмическом языке С++.
3. Разработать интерфейс средствами Qt.
4. Использование анимации, автоматизации, креативной визуализации повышает рейтинг автора проекта и влияет на экзаменационную оценку.

Вариант 17

|  |  |  |
| --- | --- | --- |
| 17 | Хранение капиталов в банке. | Банк предлагает своим клиентам условия:   1. Положить деньги на год под p - процентов годовых с начислением процентов по окончанию года 2. Положить деньги на год под p1 – процентов с начислением процентов ежемесячно 3. Положить деньги на год с ежедневным начислением p2 – процентов на каждую полную тысячу вклада 4. Положить деньги на год с ежедневным начислением p3 – процентов на сумму вклада   Банк может изменять уговорённые проценты. Калькулятор получает на вход величину вклада в рублях и копейках или в долларах и центах, а также вариант условий размещения капитала с вычислением величины вклада к концу срока.  Калькулятор должен иметь окна для исходных данных и результатов, для ввода данных, операций и служебных клавиш, клавиши изменения вида валюты. Должен позволять корректировать последнее введённое число и выдавать сообщения об ошибках. |

**Анализ задачи**

1. Для решения задачи необходимо:

1.1. Создать пять классов форм Qt Designer, для 5 окон работы калькулятора соответственно (экран приветствия, меню, окно калькулятора, дополнительная информация для вычислений, о приложении):

mainwindow.h/ mainwindow.cpp/ mainwindow.ui

calc.h/ calc.cpp/ calc.ui

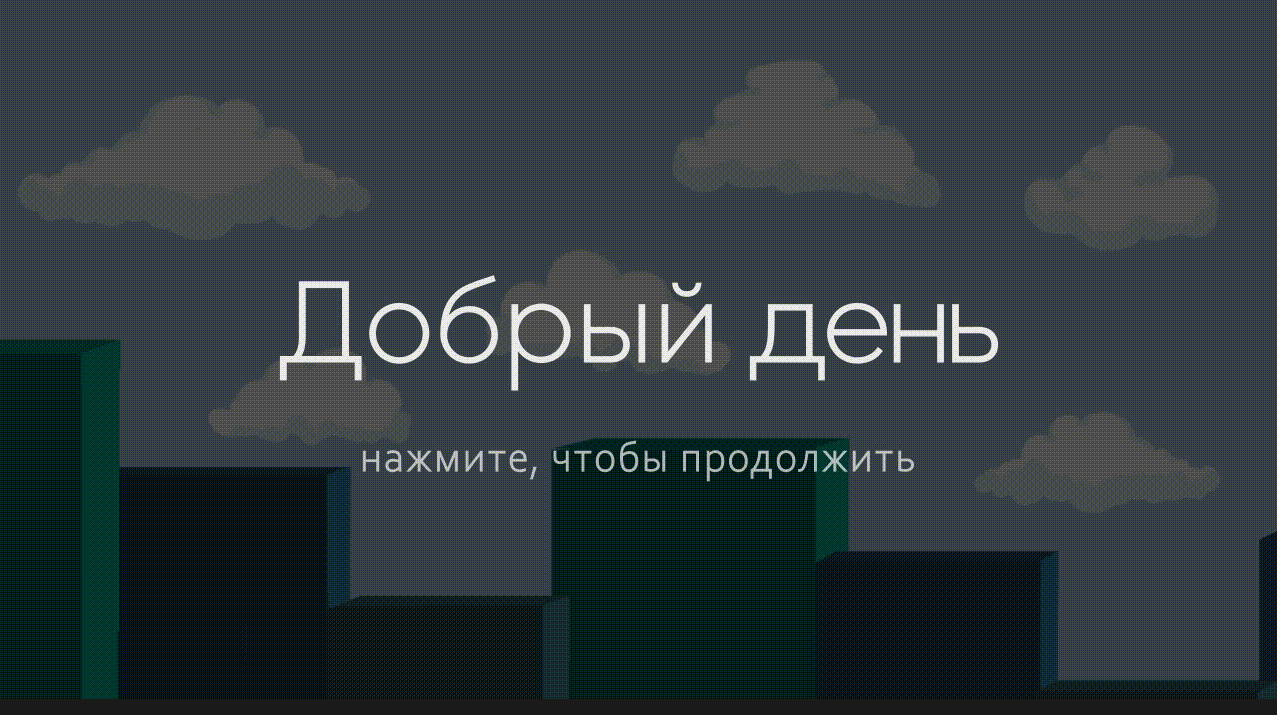
info.h/ info.cpp/info.ui

menu.h/ menu.cpp/ menu.ui

usloviya.h/usloviya.cpp/usloviya.ui

1.2. Для каждой формы разработать дизайн, прописать для всех активных модулей Qt Designer (QButton, QLineEdits, etc) сигналы и их функции.

1) mainwindow (Экран приветствия)



В данной форме активна одна невидимая кнопка, размер которой соответствует размеру окна. При нажатии на нее, появится окно с меню. Сигналом для данной операции является нажатие в любую часть экрана.

void MainWindow::on\_pushButton\_clicked()

{

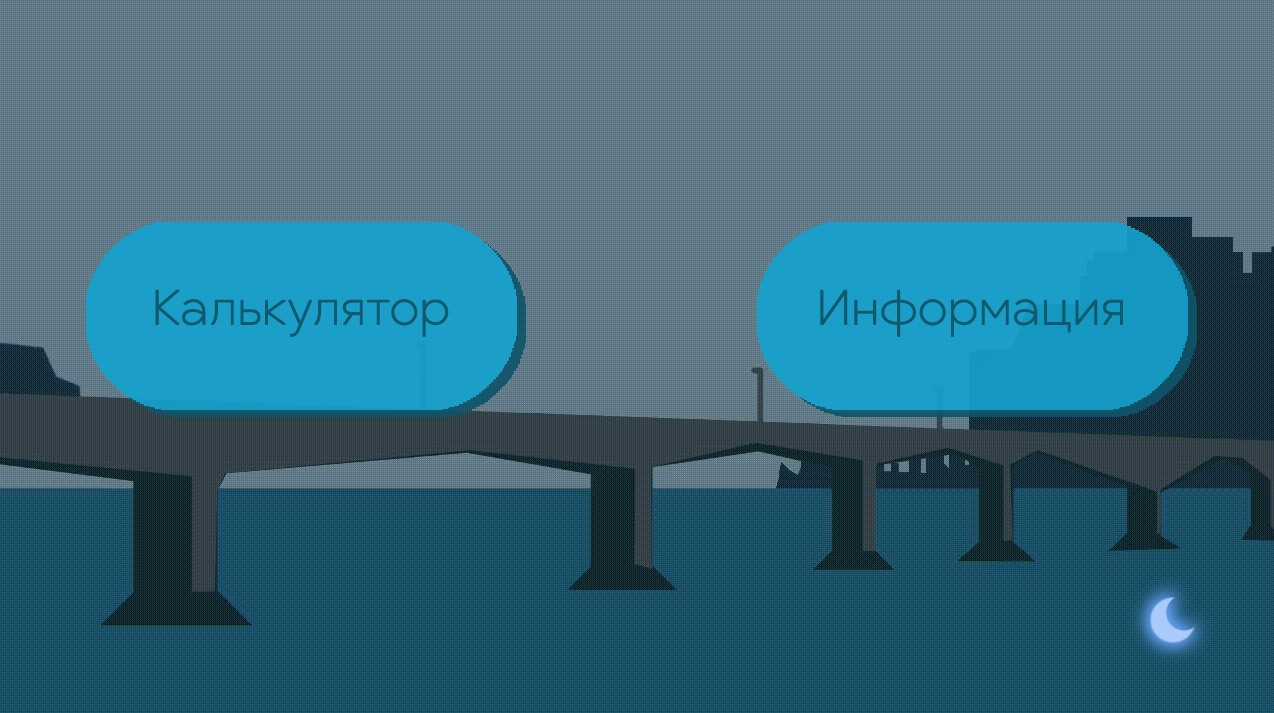
hide();

m=new Menu(this);

m->show();

}

2) menu (Меню)



В данном окне пользователю предоставляется нажать на одну из двух кнопок, тем самым выбрать окно, к которому он хочет перейти. Переход осуществляется при помощи сигналов.

void Menu::on\_calcButton\_clicked()

{

hide();

c=new Calc(this);

c->show();

}

void Menu::on\_infoButton\_clicked()

{

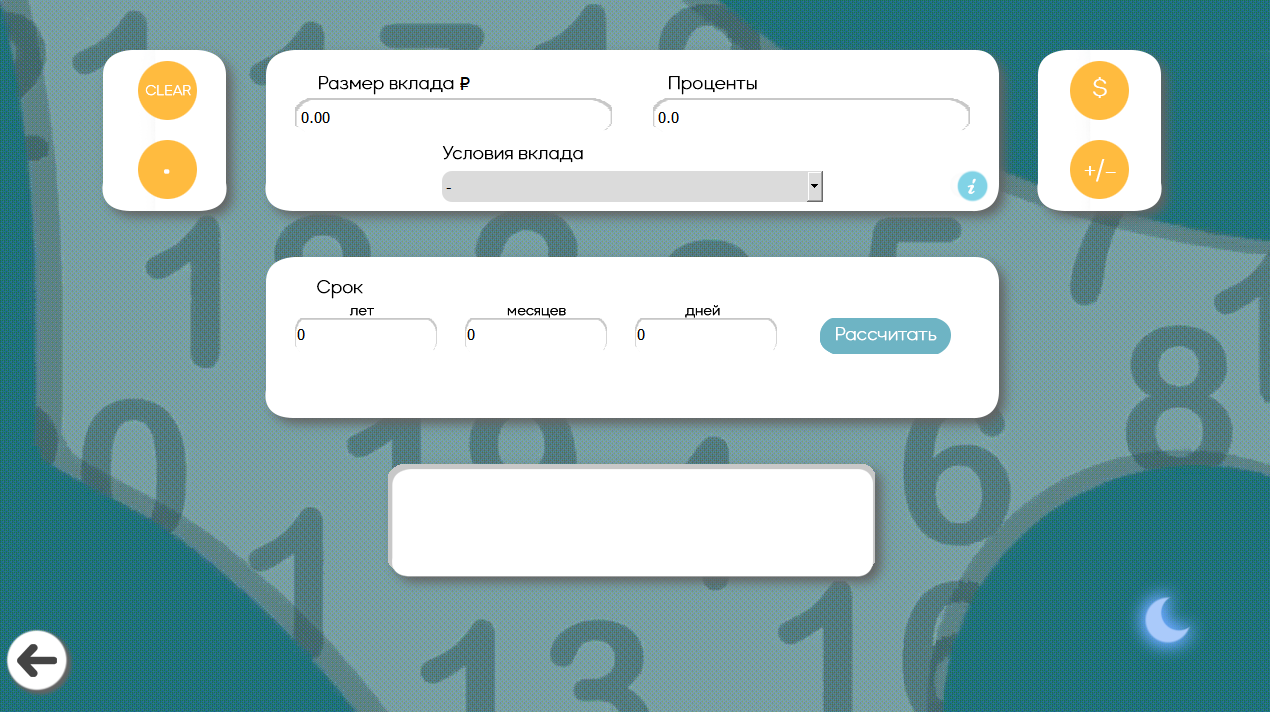
hide();

i=new info(this);

i->show();

}

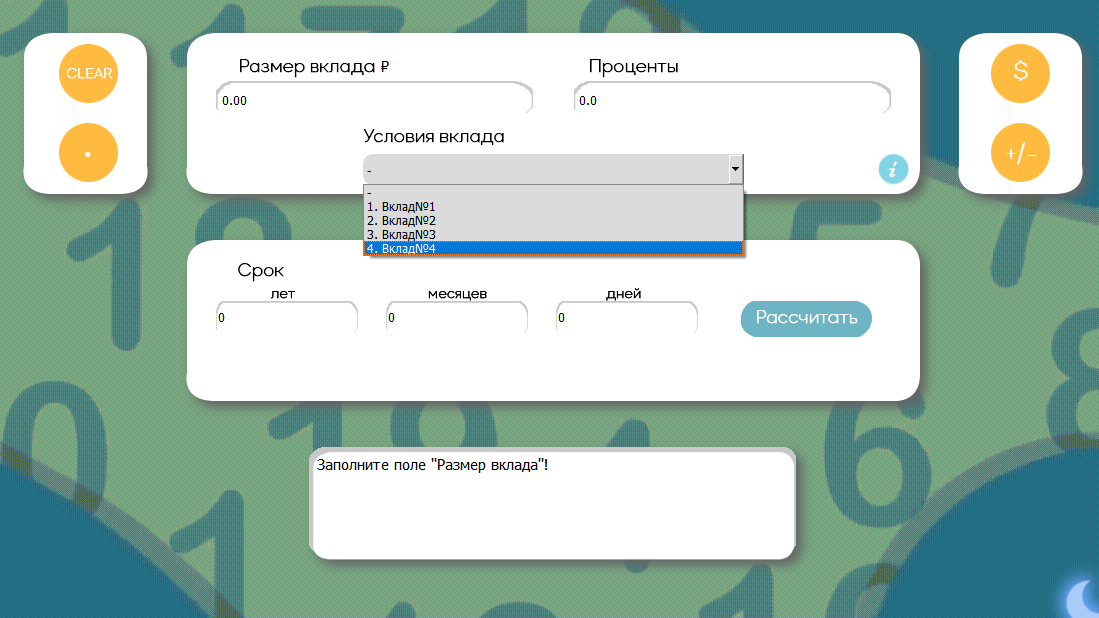
3) cacl(Окно с калькулятором)



В данном окне присутствуют такие активные модули как:

* 1. В данном поле пользователь вводит размер вклада, который он хочет положить в банк. Реализовано данное поле с помощью виджета QLineEdit.

При вводе переменная преобразуется из QString в double. Поля с процентами и срокам реализованы с помощью аналогичного виджета. При выборе условий вклада все поля должны быть заполнены, иначе в поле с результатом вылезет сообщение об ошибке.



void Calc::**on\_pick\_activated**(int index)

{

procent=ui->procent->text().toDouble();

value=ui->value->text().toDouble();

timey=ui->years->text().toInt();

timem=ui->month->text().toInt();

timed=ui->days->text().toInt();

times=(timey\*12+timem)\*30+timed;

*if*(value<0)

{

ui->textBrowser->setText("Размер вклада должен быть положительным числом!");

ui->pick->setCurrentIndex(0);

}

*else* *if*(value==0)

{

ui->textBrowser->setText("Заполните поле ''Размер вклада''!");

ui->pick->setCurrentIndex(0);

}

*else* *if*(procent<0)

{

ui->textBrowser->setText("Процент должен быть положительным числом!");

ui->pick->setCurrentIndex(0);

}

*else* *if*(procent==0)

{

ui->textBrowser->setText("Заполните поле ''Процент''!");

}

*else* *if*(timey<0||timem<0||timed<0)

{

ui->textBrowser->setText("Срок не может быть отрицательным!");

ui->pick->setCurrentIndex(0);

}

*else* *if*(timey==0&&timem==0&&timed==0)

{

ui->textBrowser->setText("Заполните поле ''Срок''!");

ui->pick->setCurrentIndex(0);

}

Во всех полях ограничен ввод с помощью валидаторов.

ui->value->setValidator(*new* QRegExpValidator(QRegExp("[1-9][0-9]{1,11}\\.[0-9]{1,2}|[0-9]\\.[0-9]{1,2}")));

ui->procent->setValidator(*new* QRegExpValidator(QRegExp("[1-9][0-9]{1,2}\\.[0-9]")));

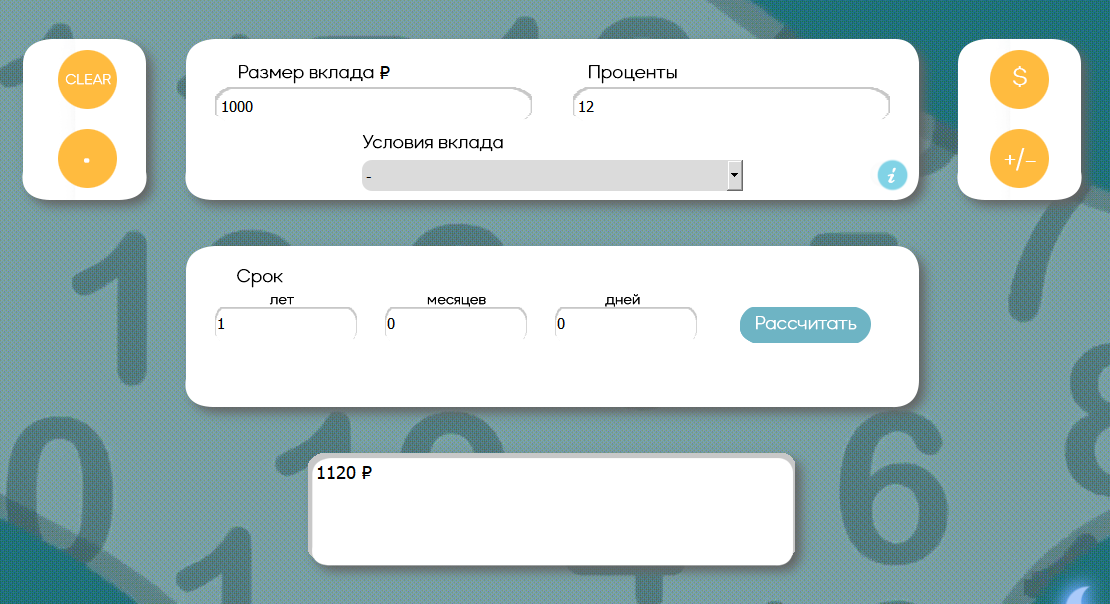
ui->years->setValidator(*new* QRegExpValidator(QRegExp("[1-9][0-9]{1,11}")));

ui->month->setValidator(*new* QRegExpValidator(QRegExp(("[1-9]|[1][0-2]"))));

ui->days->setValidator(*new* QRegExpValidator(QRegExp("[1-9][0-9]")));

Чтобы выбрать условия вклада использовался такой элемент как ComboBox.

После успешного заполнения всех полей и выбора условий вклада, чтобы получить результат, нужно нажать на кнопку “Рассчитать” и в поле, которое находится под всеми полями для ввода, отобразится размер вклада после окончания срока. Для вывода результата использовался виджет textBrowser.



Также по обеим сторонам основных полей для ввода находятся кнопки PushButton.

Первая очищает все поля, вторая ставит точку, третья- переводит в доллары, четвертая- меняет знак.

Также рядом с условиями вклада есть маленькая круглая кнопка, которая перенаправит пользователя в окно, чтобы он смог подробно ознакомится с условиями вклада.

3) usloviya (Окно c условиями) 

В данном окне описаны все возможные условия размещения вклада в банке.

* + 1. При нажатии на кнопку (“Калькулятор”) появиться окно с калькулятором. Сигналом для данной операции является нажатие по иконке кнопки.

void usloviya::**on\_pushButton\_clicked**()

{

hide();

cl=*new* Calc(*this*);

cl->show();

}

При нажатии на кнопку (“Меню”) появиться окно меню.

void usloviya::**on\_pushButton\_2\_clicked**()

{

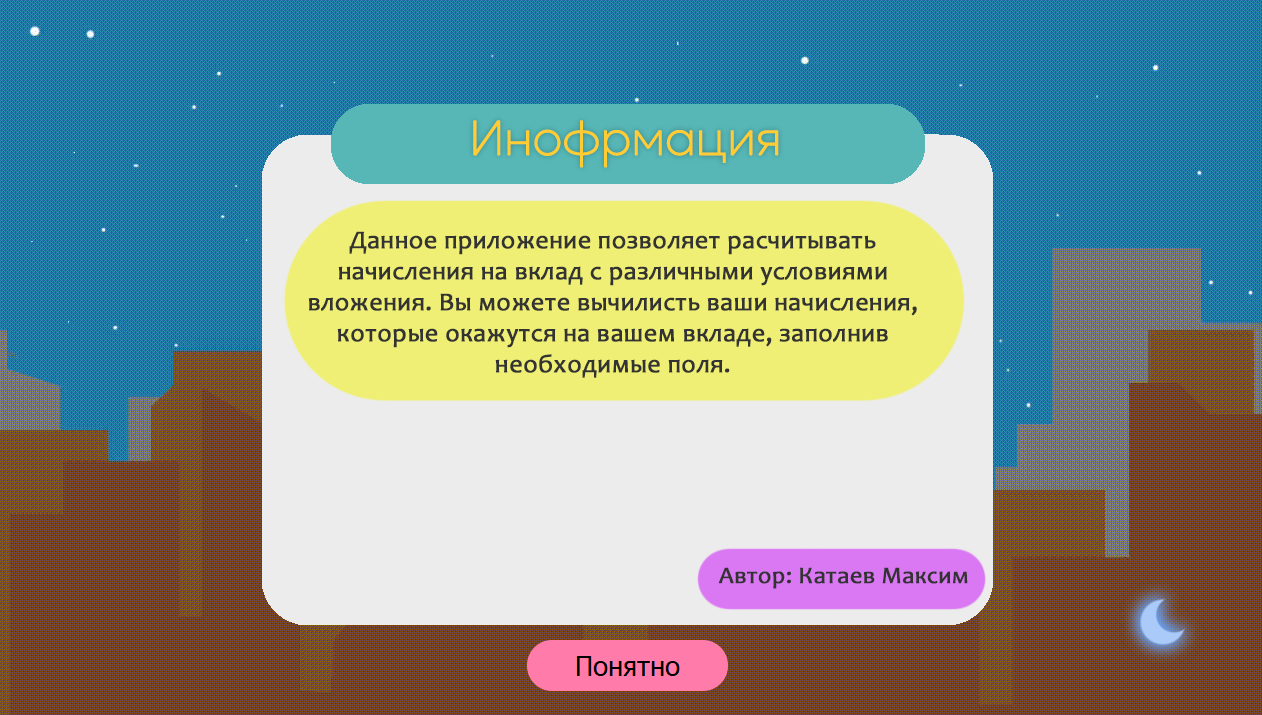
hide();

m3=*new* Menu(*this*);

m3->show();

}

4) info (Окно c информацией о приложении)

****

В данном окне присутствует одна основная кнопка, которая отвечает за перенаправление пользователя назад к окну меню.

5) Во всех окнах в качестве фона использовались GIF-анимации. Это реализовано следующим образом:

QMovie \*movie4 = *new* QMovie(":/Ss/Usloviyaback2.gif");

ui->label\_2->setMovie(movie4);

movie4->start();

QMovie \*moviei = *new* QMovie(":/Ss/Infoback2.gif");

ui->label\_2->setMovie(moviei);

moviei->setSpeed(150);

moviei->start();

Во всех окнах присутствует кнопка переключения темной и светлой темы. Это реализовано с помощью изменения изображения через setStyleSheet, на который наложены прозрачные кнопки.

*if*(light)

{

ui->label->setStyleSheet("background-image: url(:/Ss/usloviyanew2.png);\nbackground-color: rgba(255, 255, 255, 0);\ncolor: rgba(255, 255, 255, 0);");

ui->label\_3->setStyleSheet("background-color: rgba(31, 32, 93, 0);");

ui->dark->setStyleSheet("QPushButton\n{background-color: rgba(255, 255, 255, 0);\nbackground-image: url(:/Ss/switchnight.png);\nborder-radius: 55px;\n}\nQPushButton:hover\n{\n background-color: rgba(128, 214, 250,22);\n}");

}

*else*

{

ui->label->setStyleSheet("background-image: url(:/Ss/usloviyanewdark.png);\nbackground-color: rgba(255, 255, 255, 0);\ncolor: rgba(255, 255, 255, 0);");

ui->label\_3->setStyleSheet("background-color: rgba(0, 0, 0,210);");

ui->dark->setStyleSheet("QPushButton\n{background-color: rgba(255, 255, 255, 0);\nbackground-image: url(:/Ss/switchsun.png);\nborder-radius: 55px;\n}\nQPushButton:hover\n{\n background-color: rgba(128, 214, 250,22);\n}");

}

**2.** В ходе работы были использованы следующие типы данных:

**2.1.** Глобальные переменные типа double для упрощённого применения в функциях-

extern double value,procent,timey,timem,times,timed; ,

bool для изменения темной и светлой темы-.

extern bool light; .

**3.** Для решения задачи данные были представлены в следующем виде:

**3.1.** Ввод данных производится с помощью полей QLineEdit:

**3.2**. Вывод данных производится с помощью функции:

ui->lineEdit->setText(Qstring);

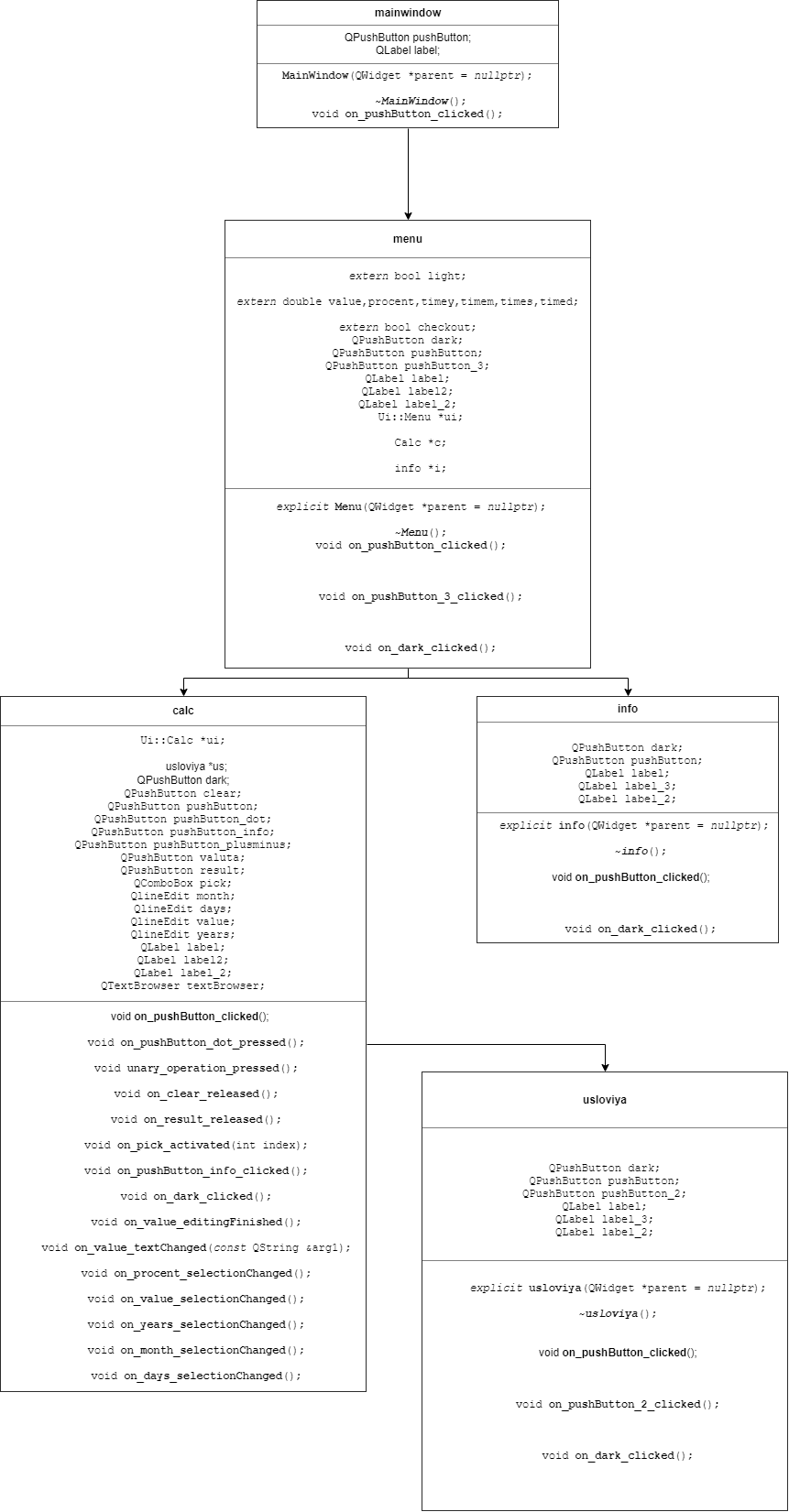
**3.3.** Преобразование типа QString к типу double производится с помощью функции:

QString.toDouble;

**3.4** Выбор условий реализован с помощью текстового меню ComboBox.

**4.** Достоинства программы

К достоинствам проекта можно отнести простоту и удобство интерфейса, многофункциональность программы. Реализация программы в нескольких окнах позволяет одновременно совершать несколько действий над одними числами, что заметно ускоряет работу

**UML-диаграмма**