Прикладное программирование

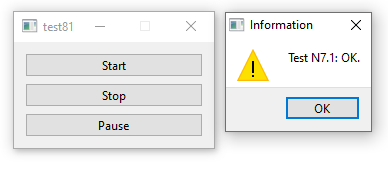
Практика №8

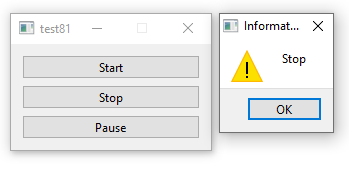
Дата 20.02.2023

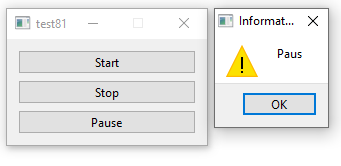
1. Примените инкрементный метод конструирования программы для решения следующей задачи. По примеру 10 из лекции создайте приложение с графическим интерфейсом с применением библиотеки Qt, в котором выполняется вывод текста при щелчке по кнопке. В окне должны быть 2 кнопки Start и Stop. Добавьте 3-ю кнопку Pause.

Оценка: 1) 1 балл за каждую версию-инкремент.

Скриншот:







Исходный код:

//widget.cpp

#include "widget.h"

#include <QMessageBox>

Widget::Widget(QWidget \*parent):QWidget(parent) {

this->setFixedWidth(200);

start = new QPushButton("Start");

stop = new QPushButton("Stop");

pause = new QPushButton("Pause");

l = new QVBoxLayout(this);

l->addWidget(start);

l->addWidget(stop);

l->addWidget(pause);

connect(start, &QPushButton::clicked, this, &Widget::play);

connect(stop, &QPushButton::clicked, this, &Widget::stoping);

connect(pause, &QPushButton::clicked, this, &Widget::pausing);

}

Widget::~Widget() {}

void Widget::play() {

QMessageBox::warning(this, "Information", "Test N7.1: OK.");

}

void Widget::stoping() {

QMessageBox::warning(this, "Information", "Stop");

}

void Widget::pausing() {

QMessageBox::warning(this, "Information", "Paus");

}

//widget.h

#ifndef WIDGET\_H

#define WIDGET\_H

#include <QWidget>

#include <QPushButton>

#include <QVBoxLayout>

class Widget : public QWidget {

Q\_OBJECT

public:

Widget(QWidget \*parent = nullptr);

~Widget();

private:

QPushButton \*start;

QPushButton \*stop;

QPushButton \*pause;

QVBoxLayout \*l;

void play();

void stoping();

void pausing();

};

#endif

//main.cpp

#include "widget.h"

#include <QApplication>

int main(int argc, char \*argv[]) {

QApplication app(argc, argv);

Widget win;

win.show();

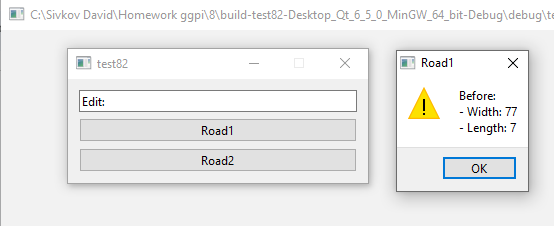
return app.exec();

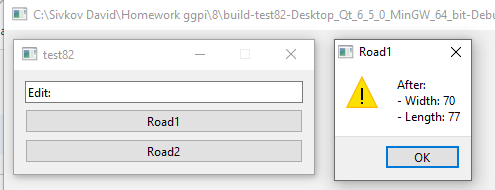
}

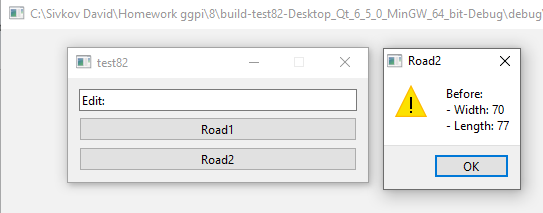
Вывод: Применил инкрементный метод конструирования программы для решения следующей задачи. По примеру 10 из лекции создал приложение с графическим интерфейсом с применением библиотеки Qt, в котором выполняется вывод текста при щелчке по кнопке. В окне 2 кнопки Start и Stop.

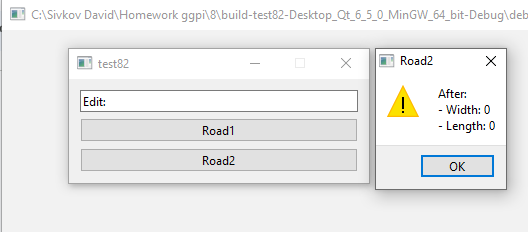
2. К проекту из задания 1 подключите библиотеку Дорога TRoad. Создайте 2 кнопки: Road1 и Road2. При нажатии на каждую кнопку вызывается своя функция, в которой свойствам дороги присваиваются новые значения и выводятся в окне программы. Добавьте возможность изменения свойств 1) при щелчке по новой кнопке присваиванием в функции; 2) с помощью ввода в однострочный редактор QLineEdit.

Скриншот:









Исходный код:

//widget.cpp

#include "widget.h"

#include "troad.h"

#include <string>

#include <QMessageBox>

using namespace std;

TRoad road(7,77);

Widget::Widget(QWidget \*parent):QWidget(parent) {

this->setFixedWidth(300);

Road1 = new QPushButton("Road1");

Road2 = new QPushButton("Road2");

Edit = new QLineEdit("Edit: ");

NUM = new QVBoxLayout(this);

NUM->addWidget(Edit);

NUM->addWidget(Road1);

NUM->addWidget(Road2);

connect(Road1, &QPushButton::clicked, this, &Widget::newFunction);

connect(Road2, &QPushButton::clicked, this, &Widget::editFunction);

}

Widget::~Widget() {}

void Widget::newFunction() {

QString width = QString::fromStdString(to\_string(road.Width));

QString length = QString::fromStdString(to\_string(road.Length));

QMessageBox::warning(this, "Road1", "Before:\n- Width: " + width + "\n- Length: " + length);

road.Width = 70;

road.Length = 77;

width = QString::fromStdString(to\_string(road.Width));

length = QString::fromStdString(to\_string(road.Length));

QMessageBox::warning(this, "Road1", "After:\n- Width: " + width + "\n- Length: " + length);

}

void Widget::editFunction() {

QString edit=Edit->text();

int wwidth = edit.split(" ")[0].toInt();

int llength = edit.split(" ")[1].toInt();

QString width = QString::fromStdString(to\_string(road.Width));

QString length = QString::fromStdString(to\_string(road.Length));

QMessageBox::warning(this, "Road2", "Before:\n- Width: " + width + "\n- Length: " + length);

road.Width = wwidth;

road.Length = llength;

width = QString::fromStdString(to\_string(road.Width));

length = QString::fromStdString(to\_string(road.Length));

QMessageBox::warning(this, "Road2", "After:\n- Width: " + width + "\n- Length: " + length);

}

//widget.h

#ifndef WIDGET\_H

#define WIDGET\_H

#include <QWidget>

#include <QPushButton>

#include <QVBoxLayout>

#include <QLineEdit>

class Widget : public QWidget {

Q\_OBJECT

public:

Widget(QWidget \*parent = nullptr);

~Widget();

private:

QPushButton \*Road1;

QPushButton \*Road2;

QLineEdit \*Edit;

QVBoxLayout \*NUM;

void newFunction();

void editFunction();

};

#endif

//main.cpp

#include "widget.h"

#include <QApplication>

int main(int argc, char \*argv[]) {

QApplication app(argc, argv);

Widget win;

win.show();

return app.exec();

}

//troad.h

#ifndef TROAD\_H

#define TROAD\_H

class TRoad {

public:

int Length;

int Width;

TRoad(int a, int b);

};

#endif

//troad.cpp

#include "troad.h"

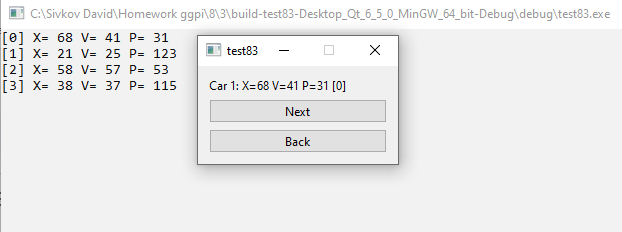
TRoad::TRoad(int a, int b) : Length(a), Width(b) {}

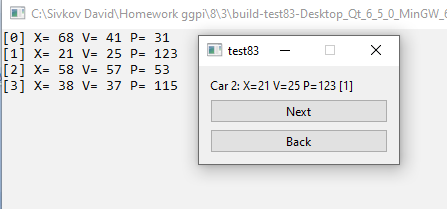
Вывод: Для проекта из задания 1 была подключена библиотека TRoad и созданы две кнопки: Road1 и Road2. При нажатии на каждую кнопку вызывается своя функция, которая присваивает свойствам дороги новые значения и выводит их в окне программы. Также была добавлена возможность изменения свойств дороги при щелчке по новой кнопке или через однострочный редактор QLineEdit.

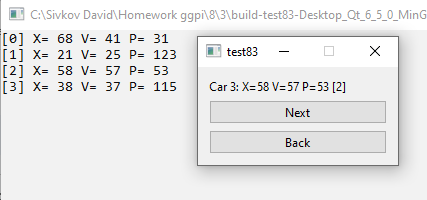
3. К проекту из задания 1 подключите библиотеку с классом Машина. Создайте массив из 4 объектов. Создайте 2 кнопки: Back и Next. При нажатии на кнопки происходит переход к предыдущему или следующему объекту из массива, для которого выводятся значения свойств класса, каждое - в отдельный компонент QLabel: xCarLabel, vCarLabel, pCarLabel и номер объекта в массиве.

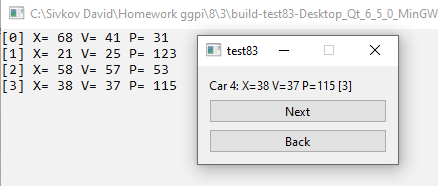
Решение:

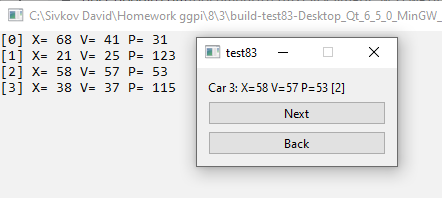
Скриншоты:

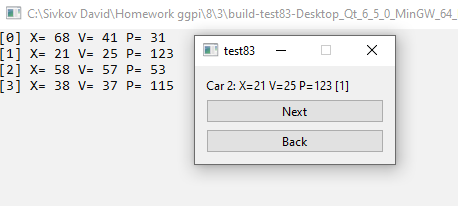


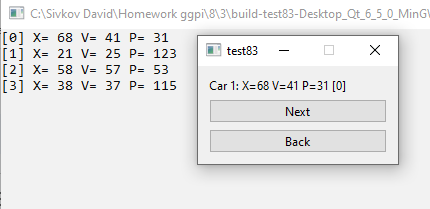




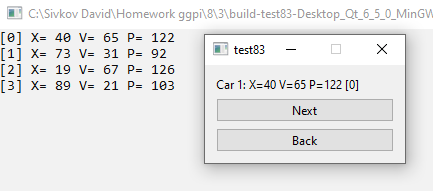








Если перезайти мы увидим уже другие числа, так как в программе массив из рандомных чисел



Исходный код:

//test83.pro

CONFIG+=console

QT+=widgets

SOURCES += \

machine.cpp \

main.cpp \

widget.cpp

HEADERS += \

machine.h \

widget.h

//machine.h

#ifndef MACHINE\_H

#define MACHINE\_H

class Machine {

public:

int X;

int P;

int V;

Machine();

};

#endif

//widget.h

#ifndef WIDGET\_H

#define WIDGET\_H

#include <QWidget>

#include <QPushButton>

#include <QVBoxLayout>

#include <QLineEdit>

#include <QLabel>

class Widget : public QWidget {

Q\_OBJECT

public:

Widget(QWidget \*parent = nullptr);

~Widget();

private:

QPushButton \*next;

QPushButton \*back;

QLabel \*car;

QVBoxLayout \*lay;

void nextCar();

void backCar();

};

#endif

//main.cpp

#include "widget.h"

#include <QApplication>

int main(int argc, char \*argv[]) {

QApplication app(argc, argv);

Widget win;

win.show();

return app.exec();

}

//widget.cpp

#include "widget.h"

#include "machine.h"

#include <string>

#include <QMessageBox>

#include <stdlib.h>

#include <time.h>

#include <iostream>

using namespace std;

int i=0;

Machine cars[4];

Widget::Widget(QWidget \*parent):QWidget(parent) {

srand(time(NULL));

for (int i = 0; i < 4; i++) {

cars[i].X=10 + rand() % (80);

cars[i].V=20 + rand() % (70);

cars[i].P=30 + rand() % (100);

cout << "[" << i << "]"<< " X= " << cars[i].X<< " V= " << cars[i].V<< " P= " << cars[i].P<< endl;

}

this->setFixedWidth(200);

next = new QPushButton("Next");

back = new QPushButton("Back");

car = new QLabel("Car " + QString::fromStdString(to\_string(i+1)) + ": X="

+ QString::fromStdString(to\_string(cars[i].X))

+ " V=" + QString::fromStdString(to\_string(cars[i].V))

+ " P=" + QString::fromStdString(to\_string(cars[i].P))

+ " [" + QString::fromStdString(to\_string(i)) + "]");

lay = new QVBoxLayout(this);

lay->addWidget(car);

lay->addWidget(next);

lay->addWidget(back);

connect(next, &QPushButton::clicked, this, &Widget::nextCar);

connect(back, &QPushButton::clicked, this, &Widget::backCar);

}

Widget::~Widget() {}

void Widget::nextCar() {

if(i+1>3) return;

i++;

car->setText("Car " + QString::fromStdString(std::to\_string(i+1)) + ": X="

+ QString::fromStdString(std::to\_string(cars[i].X))

+ " V=" + QString::fromStdString(to\_string(cars[i].V))

+ " P=" + QString::fromStdString(to\_string(cars[i].P))

+ " [" + QString::fromStdString(to\_string(i)) + "]");

}

void Widget::backCar() {

if(i-1<0) return;

i--;

car->setText("Car " + QString::fromStdString(std::to\_string(i+1)) + ": X="

+ QString::fromStdString(std::to\_string(cars[i].X))

+ " V=" + QString::fromStdString(to\_string(cars[i].V))

+ " P=" + QString::fromStdString(to\_string(cars[i].P))

+ " [" + QString::fromStdString(to\_string(i)) + "]");

}

//machine.cpp

#include "machine.h"

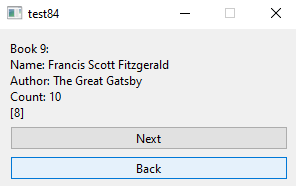
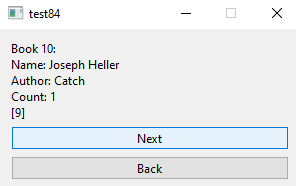
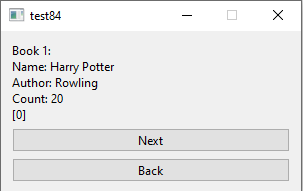
Machine::Machine() : X(70), P(7), V(700) {}

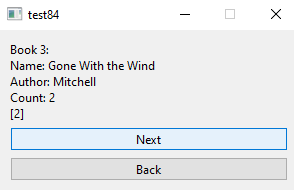
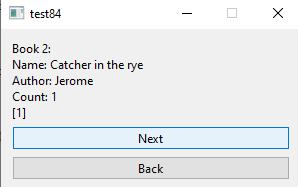
Вывод: В проекте из задания 1 была подключена библиотека с классом Машина, создан массив из 4 объектов и две кнопки - Back и Next. При нажатии на эти кнопки происходит переход к предыдущему или следующему объекту из массива. Значения свойств класса каждого объекта выводятся в отдельные компоненты QLabel: xCarLabel, vCarLabel, pCarLabel и номер объекта в массиве.

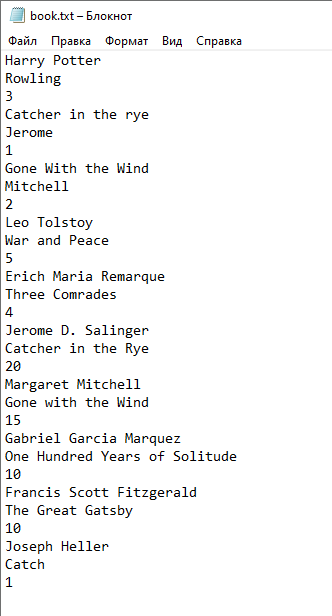
4. Подключите к проекту библиотеку TBook. Создайте текстовый файл, в котором хранятся сведения о 10 книгах. Из этого файла свойства должны загружаться в массив объектов TBook. Создайте 2 кнопки с надписями Back и Next. Эти кнопки служат для навигации по базе данных Книги, хранящейся в массиве. При нажатии на кнопку выводится номер записи и сведения о книге. Кнопка Next - переход по списку вперед, кнопка Back - назад.

Решение:

Скриншот:







Исходный код:

//test84.pro

CONFIG+=console

QT+=widgets

HEADERS += \

tbook.h \

widget.h

SOURCES += \

main.cpp \

tbook.cpp \

widget.cpp

//main.cpp

#include "widget.h"

#include <QApplication>

int main(int argc, char \*argv[]) {

QApplication app(argc, argv);

Widget win;

win.show();

return app.exec();

}

//tbook.cpp

#include "tbook.h"

TBook::TBook() : Author("Rowling"), Name("Harry Potter"), Count(20) {}

//tbook.h

#ifndef TBOOK\_H

#define TBOOK\_H

#include <string>

using namespace std;

class TBook {

public:

string Author;

string Name;

int Count;

TBook();

};

#endif

//widget.h

#ifndef WIDGET\_H

#define WIDGET\_H

#include <QWidget>

#include <QPushButton>

#include <QVBoxLayout>

#include <QLineEdit>

#include <QLabel>

class Widget : public QWidget {

Q\_OBJECT

public:

Widget(QWidget \*parent = nullptr);

~Widget();

private:

QPushButton \*next;

QPushButton \*back;

QLabel \*book;

QVBoxLayout \*lay;

void nextBook();

void backBook();

};

#endif

//widget.cpp

#include "widget.h"

#include "tbook.h"

#include <string>

#include <QMessageBox>

#include <iostream>

#include <fstream>

using namespace std;

int i=0;

TBook books[10];

Widget::Widget(QWidget \*parent):QWidget(parent) {

ifstream fin;

fin.open("C:/Sivkov David/Homework ggpi/8/4/test84/book.txt");

if (fin.is\_open()) {

for (int i = 0; i < 10; i++) {

string temp;

getline(fin, books[i].Name);

getline(fin, books[i].Author);

getline(fin, temp);

books[i].Count = stoi(temp);

}

fin.close();

} else cout << "Can't find a file!" << endl;

this->setFixedWidth(300);

next = new QPushButton("Next");

back = new QPushButton("Back");

book = new QLabel("Book " + QString::fromStdString(to\_string(i+1))

+ ":\nName: " + QString::fromStdString(books[i].Name)

+ "\nAuthor: " + QString::fromStdString(books[i].Author)

+ "\nCount: " + QString::fromStdString(to\_string(books[i].Count))

+ "\n[" + QString::fromStdString(to\_string(i)) + "]");

lay = new QVBoxLayout(this);

lay->addWidget(book);

lay->addWidget(next);

lay->addWidget(back);

connect(next, &QPushButton::clicked, this, &Widget::nextBook);

connect(back, &QPushButton::clicked, this, &Widget::backBook);

}

Widget::~Widget() {}

void Widget::nextBook() {

if(i+1>9) return;

i++;

book->setText("Book " + QString::fromStdString(to\_string(i+1))

+ ":\nName: " + QString::fromStdString(books[i].Name)

+ "\nAuthor: " + QString::fromStdString(books[i].Author)

+ "\nCount: " + QString::fromStdString(to\_string(books[i].Count))

+ "\n[" + QString::fromStdString(to\_string(i)) + "]");

}

void Widget::backBook() {

if(i-1<0) return;

i--;

book->setText("Book " + QString::fromStdString(to\_string(i+1))

+ ":\nName: " + QString::fromStdString(books[i].Name)

+ "\nAuthor: " + QString::fromStdString(books[i].Author)

+ "\nCount: " + QString::fromStdString(to\_string(books[i].Count))

+ "\n[" + QString::fromStdString(to\_string(i)) + "]");

}

Вывод: В проекте была подключена библиотека TBook. Текстовый файл содержит информацию о 10 книгах, свойства которых загружаются в массив объектов TBook. Для навигации по базе данных книг были созданы две кнопки: Back и Next. При нажатии на эти кнопки происходит переход по списку вперед и назад соответственно, и выводится номер записи и сведения о книге.