**Министерство образования науки Российской федерации**

**Федеральное государственное бюджетное образовательное**

**учреждение высшего образования**

**«Рязанский Государственный Радиотехнический Университет»**

**Кафедра вычислительной и прикладной математики**

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

к курсовому проекту по дисциплине:

«Графические интерфейсы информационных систем»

на тему:

«Разработка графического интерфейса приложения: «Порядковый номер».

**Выполнили:**

**Студенты группы:** 4041

Бирюков П.Ю и Бирюков Н.Ю.

(Ф.И.О.)

**Проверила:**

**Доцент кафедры ВПМ, к.т.н.:**

Маркова Вера Валерьевна

(Ф.И.О.)

**Оценка:**

**Рязань 2017**

**ЗАДАНИЕ**

на выполнение курсового проекта по дисциплине:

**«Графические интерфейсы информационных систем»**

Студенты группы: 4041 Бирюков П.Ю. и Бирюков Н.Ю.

Тема проекта

**«Разработка графического интерфейса приложения:**

«Порядковый номер»

(название предметной области)

**Обязательные разделы пояснительной записки**

**Задание**

**Содержание**

**Постановка задачи**

**Описание проектного решения**

1. разработка сценария работы приложения в табличной форме (идентификатор формы, назначение формы, элементы управления, содержащиеся на форме, взаимодействие с другими формами);
2. внешнее и внутреннее представление элементов управления (идентификатор элемента, визуальное представление, возможные значения);
3. структура и описание пунктов меню.

**Разработка программы**

1. описание процедур и функций (заголовок, назначение, описание параметров процедуры или функции);
2. состав проекта: перечисляются файлы проекта. Для каждого файла указываются: 1) имя; 2) функциональное назначение; 3) местоположение (необходимое или рекомендуемое расположение файла на внешнем носителе).

**Разработка документации**

Раздел должен включать «Руководство пользователя», содержащее следующие подразделы:

* назначение системы и выполняемые функции;
* условия применения системы;
* сценарий взаимодействия с пользователем.

Подраздел «Назначение системы и выполняемые функции» должен содержать общую характеристику системы и перечень обслуживаемых запросов. Подраздел «Условия применения системы» должен содержать требования к техническим характеристикам ЭВМ (типу процессора, объему оперативной и внешней памяти, периферийному оборудованию), операционной системе и вспомогательному программному обеспечению. Сценарий взаимодействия с пользователем включает следующие описания:

* структура основного окна программы;
* описание последовательности действий пользователя при работе с элементами управления графического интерфейса окна приложения (визуальное представление, описание, возможные значения элемента управления, действия пользователя после выбора каждого из значений);
* перечень сообщений пользователю.

**Тестирование программной системы**

В разделе, посвященном описанию процесса тестирования разработанной системы, следует указать план тестирования, результаты тестирования и их анализ.

**Приложения**

В одном из приложений обязательно должны приводиться распечатки наиболее важных фрагментов программы. Объем распечаток не должен превышать 25 страниц. Приводимые тексты программ должны быть тщательно прокомментированы.

**Правила сдачи проекта руководителю проекта**

1. Предъявляются следующие материалы:
2. пояснительная записка в электронном виде;
3. исходные коды проекта.
4. Проверяется соответствие пояснительной записки предъявляемым требованиям.
5. Как правило, осуществляются:
6. компиляция и сборка проекта;
7. решение тестовых задач.

**Инструментальные требования**

ЭВМ – ПЭВМ, совместимая с IBM PC.

Инструментальная среда - MS Windows, среда разработки - Microsoft Visual Studio 6.0. Средства проектирования — Win 32 API.

Задание выдано « » 2017 года Доцент каф. ВПМ Маркова В.В.

Содержание

[Введение 5](#_Toc483780687)

[Постановка задачи 7](#_Toc483780688)

[Описание проектного решения 8](#_Toc483780689)

[Сценарий работы приложения 8](#_Toc483780690)

[Внешнее и внутреннее представление элементов 9](#_Toc483780691)

[Разработка программы 10](#_Toc483780692)

[Описание процедур и функции 10](#_Toc483780693)

[Состав проекта 11](#_Toc483780694)

[Другие стандартные файлы 12](#_Toc483780695)

[Разработка документации 13](#_Toc483780696)

[Руководство пользователя 13](#_Toc483780697)

[Назначение системы и выполняемые функции 13](#_Toc483780698)

[Минимальные системные требования 14](#_Toc483780699)

[Сценарий взаимодействия с пользователем 15](#_Toc483780700)

[Тестирование программной системы 16](#_Toc483780701)

[Список литературы 17](#_Toc483780702)

[Приложение 18](#_Toc483780703)

[Текст программы 18](#_Toc483780704)

[Main.cpp 18](#_Toc483780705)

[Resource.rc 22](#_Toc483780706)

[Resource.h 23](#_Toc483780707)

# Введение

В настоящее время электронная техника прочно вошла в нашу жизнь. Она проникла во все сферы: банковское дело, экономику, промышленность, торговлю. Повышение уровня образования так же тесно связано с использованием компьютеров. Сейчас уже трудно представить жизнь без электронных гаджетов. Мы каждый день сталкиваемся, с какими либо, автоматизированными процессами, будь то оплата общественного транспорта или покупка в интернет магазине.

Графический интерфейс пользователя (Graphical User Interface, GUI) это система средств для взаимодействия пользователя с компьютером, основанная на представлении всех доступных пользователю системных объектов и функций в виде графических компонентов экрана (окон, значков, кнопок, полей ввода, списков и т.п.). При работе с GUI пользователь имеет произвольный доступ с помощью устройства ввода ко всем видимым экранным объектам – элементам интерфейса.

Одним из требований к хорошему графическому интерфейсу программной системы является концепция «делай то, что я имею ввиду» или DWIM (Do What I Mean). DWIM требует, чтобы система работала предсказуемо, чтобы пользователь заранее интуитивно понимал, какое действие выполнит программа после получения его команды.

Почти все современные графические интерфейсы общего назначения строятся по модели WIMP – Window, Icon, Menu, Pointer (окно, иконка, меню, указатель). Внутри окон рисуются элементы графического интерфейса, которые для краткости будут называться виджетами (widget - штучка). Меню могут располагаться в различных частях окна, но их поведение достаточно однотипно: они служат для выбора действия из набора предопределенных действий. Пользователь графического интерфейса «объясняет» компьютерной программе требуемые действия с помощью указателя. Обычно указателем служит курсор мыши или джойстика, однако есть и другие «указательные» устройства. С помощью иконок графический интерфейс приобретает независимость от языка и в некоторых случаях позволяет быстрее ориентироваться в интерфейсе.

Основной задачей графического интерфейса является упрощение коммуникации между пользователем и компьютером. Об этом следует постоянно помнить при проектировании интерфейса. Применение имеющихся в наличии у программиста (или дизайнера) средств при создании графического интерфейса нужно свести до минимума, выбирая наиболее удобные пользователю виджеты в каждом конкретном случае. Кроме того, полезно следовать принципу наименьшего удивления: из формы интерфейса должно быть понятно его поведение. Плохо продуманный интерфейс портит ощущения пользователя от программы, даже если за фасадом интерфейса скрывается эффективный алгоритм. Интерфейс должен быть удобен для типичных действий пользователя. Для многих приложений такие действия выделены в отдельные серии экранов, называемые «мастерами» (wizards). Однако если приложение - скорее конструктор, из которого пользователь может строить нужные ему решения, типичным действием является именно построение решения. Определить типичные действия не всегда легко, поэтому компромиссом может быть гибрид, в котором есть "мастера" и хорошие возможности для собственных построений. Тем не менее, графический интерфейс не является самым эффективным интерфейсом во всех случаях. Для многих предметных областей решение проще выразить с помощью деклараций на некотором формальном языке или алгоритма на сценарном языке.

# Постановка задачи

Целью выполнения данной работы была разработка графического интерфейса для приложения «Порядковый номер». В качестве интерактивной среды разработки приложения было выбрано Microsoft Visual Studio 2017 на языке C++.

Помимо основной функции разбитие экрана на N частей, программа должна обладать следующим функционалом:

1. Меню «File» с пунктами «Settings…», «Exit»;
2. Color – Синий, зеленый, пурпурный, красный, малиновый, желтый, белый;
3. N – Число задающие насколько частей нужно разделить окно программы.

# Описание проектного решения

## Сценарий работы приложения

|  |  |  |  |
| --- | --- | --- | --- |
| Название  формы | Назначение | Элементы управления | Содержание формы,  взаимодействие с другими формами |
| main | Отображение разбитого экрана на N частей | Главное меню | Форма настроек |
| settings | Изменение цвета, количества на которое нужно разбить экран | Радиогруппа для выбора цвета.  Поле для ввода количества частей.  Кнопки «ОК» «Cancel» | Главное окно |

## Внешнее и внутреннее представление элементов

|  |  |  |
| --- | --- | --- |
| Индетефикатор элекмента | Визуальное представление | Возможные значения |
| IDM\_SETTINGS | DIALOG | – |
| IDM\_EXIT | Exiting the program | – |
| IDD\_PAINT | D:\Загрузки\Без имени-2.jpg | – |
| IDD\_BLUE | D:\Загрузки\Без имени-3.jpg | Checked/Unchecked |
| IDD\_GREEN | D:\Загрузки\Без имени-3.jpg | Checked/Unchecked |
| IDD\_CYAN | D:\Загрузки\Без имени-3.jpg | Checked/Unchecked |
| IDD\_RED | D:\Загрузки\Без имени-3.jpg | Checked/Unchecked |
| IDD\_MAGENTA | D:\Загрузки\Без имени-3.jpg | Checked/Unchecked |
| IDD\_YELLOW | D:\Загрузки\Без имени-3.jpg | Checked/Unchecked |
| IDD\_WHITE | D:\Загрузки\Без имени-3.jpg | Checked/Unchecked |
| IDD\_N | D:\Загрузки\Без имени-3.jpg | CHAR |
| IDOK | D:\Загрузки\Без имени-3.jpg | – |
| IDCANCEL | D:\Загрузки\Без имени-3.jpg | – |

# Разработка программы

## Описание процедур и функции

WinMain**(**HINSTANCE hInstance**,** HINSTANCE hPrevInstance**,** PSTR szCmdLine**,** int iCmdShow**)** – функция обработки главного окна приложения;

WndProc**(**HWND hwnd**,** UINT iMsg**,** WPARAM wParam**,** LPARAM lParam**)** – функция обработки окна «File»;

SettingsDlgProc**(**HWND hDlg**,** UINT iMsg**,** WPARAM wParam**,** LPARAM lParam**)** – функция обработки окна настроек;

PaintTheBlock**(**HWND hCtrl**,** int iColor**,** int iN**)** – процедура обработки разбития окна на N частей.

PaintWindow**(**HWND hwnd**,** int iColor**,** int iN**)** – процедура обработки разбития окна на N частей.

## Состав проекта

**GraphicalInterfaces.vcxproj** – Это основной файл проекта VC++,' создаваемый с помощью мастера приложений. Он содержит данные о версии языка Visual C++, использованной для создания файла, а также сведения о платформах, конфигурациях и функциях проекта, выбранных с помощью мастера приложений.

**GraphicalInterfaces.vcxproj.filters** – Это файл фильтров для проектов VC++, созданный с помощью мастера приложений. Он содержит сведения о сопоставлениях между файлами в вашем проекте и фильтрами. Эти сопоставления используются в среде IDE для группировки файлов с одинаковыми расширениями в одном узле.

**Main.срр** – Это основной исходный файл приложения.

**Resource.rc** – Этот файл представляет собой список всех ресурсов Microsoft Windows, используемых программой. В него включены значки, точечные рисунки и курсоры, хранящиеся в подкаталоге RES. Этот файл можно редактировать непосредственно в Microsoft Visual C++.

**Resource.h** – Это стандартный файл заголовка, в котором определяются новые идентификаторы ресурсов. Такой файл можно прочитать и обновить в Microsoft Visual C++.

## Другие стандартные файлы

**StdAfx.h, StdAfx.cpp** – Эти файлы используются для построения файла предкомпилированного заголовка (РСН) с именем CourseWorkGI.pch и файла предкомпилированных типов с именем StdAfx.obj.

# Разработка документации

## Руководство пользователя

### Назначение системы и выполняемые функции

Система предназначена для работы с графикой в интерактивном режиме. Она позволяет разбивать, рабочую область формы, на требуемое количество частей, а также задавать цвет для них.

### Минимальные системные требования

* Система – Windows Seven;
* Процессор – INTEL с частотой 1ГГц или эквивалентный;
* Видеокарта - Графическая карта Local bus SVGA (совместимая с DirectX);
* ОЗУ– 128 МБ;
* Жесткий диск – 256 МВ свободного места на жестком диске;
* Устройство ввода – Клавиатура/Мышь;

### Сценарий взаимодействия с пользователем

Основное окно программы представляет из себя форму разделенную на несколько частей с пунктом меню (Рис. 1).

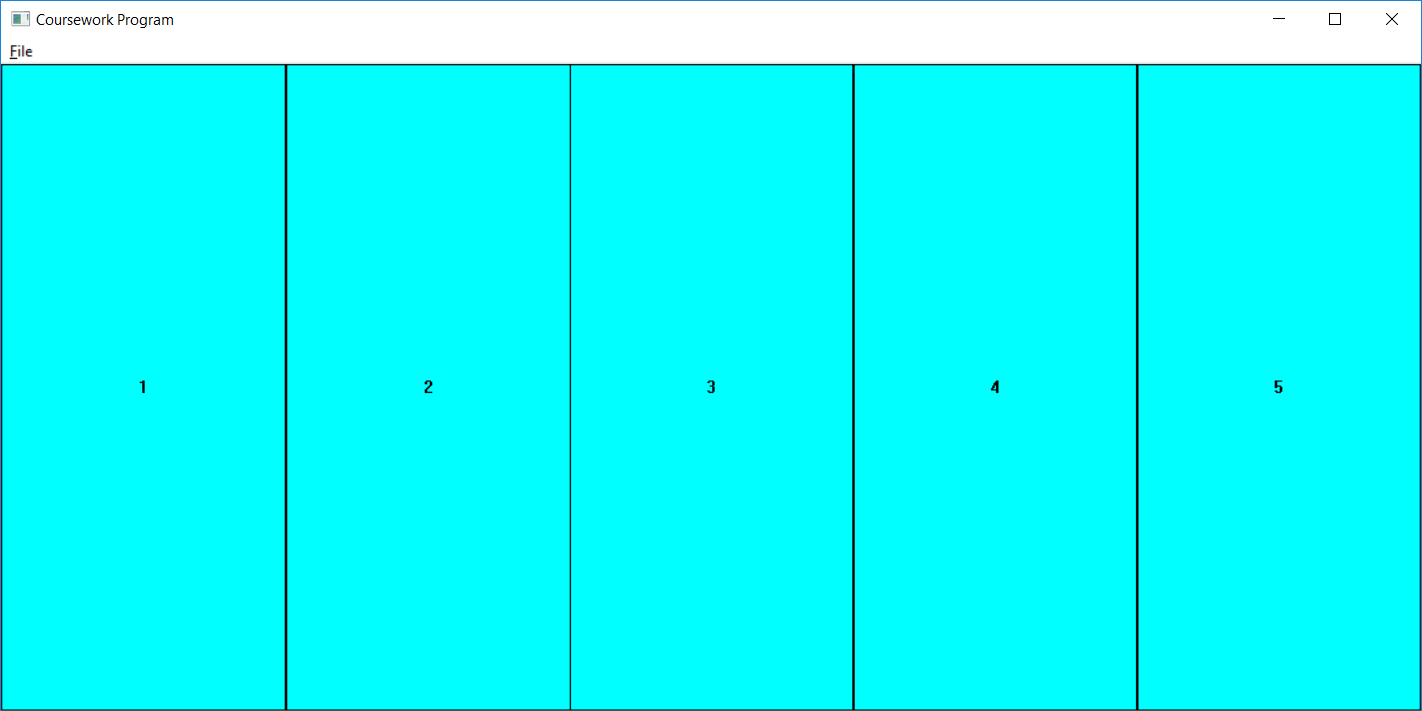


Рисунок 1. Основное окно программы

Пункт Файл «File» содержит 2 пункта: – Настройки «Settings…» в нем представлены все действия, осуществляемые в программе, а именно выбор цвета, и количество частей на которое нужно разбить форму, «Exit» – осуществляет немедленный выход в систему (Рис. 2).

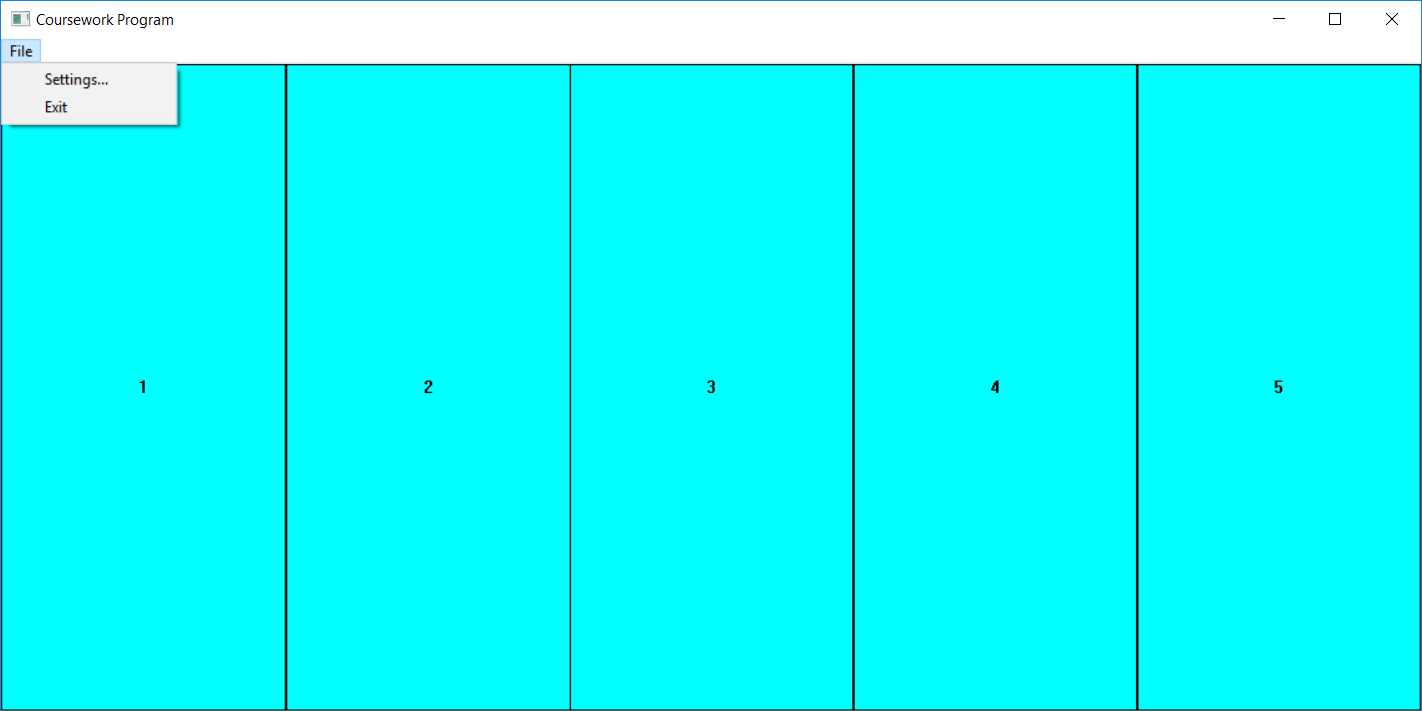


Рисунок 2. Меню «Мели»

При выборе пункта меню «Settings», откроется диалоговое окно «Sittings» в котором мы можем задавать цвет и выбрать, насколько частей необходимо разбить форму (Рис. 3).

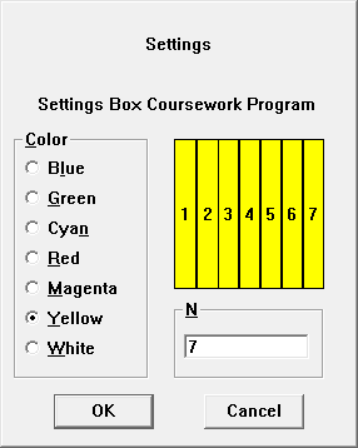


Рисунок 3. Диалоговое окно для выбора с настройками

После выбранных выше параметров и нажатия на кнопку «ОК» форма разделиться на указанное количество частей и зальется указанным цветом (Рис. 4).

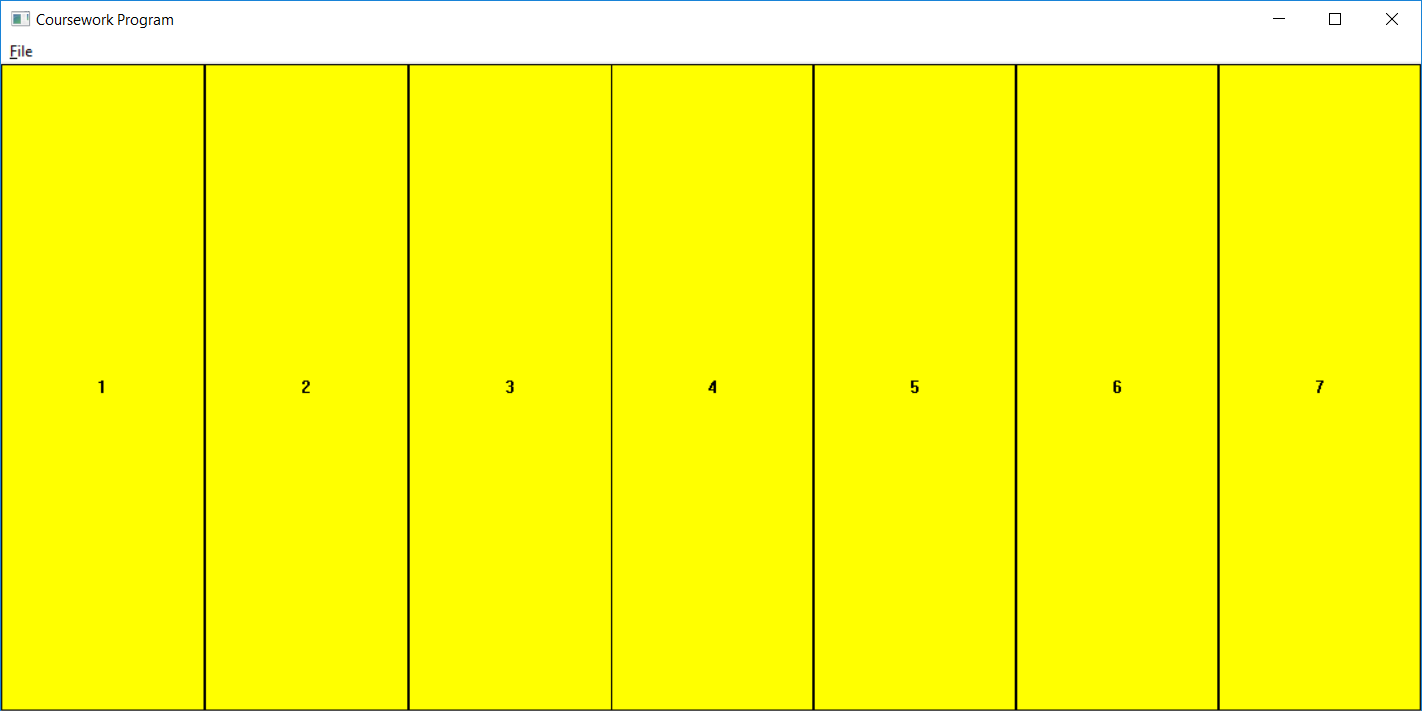


Рисунок 4. Результат работы программы

# Тестирование программной системы

В ходе тестирования системы, были проверены все возможные сценарии работы. Были исправлены возникающие ошибки, а также добавлены обработчики исключительных ситуаций.

Тестирование исключительно ситуации вводе пустого значения в поле.

Для проверки открываем форму «Settings» и оставляем поле для ввода количества частей «N» пустым, нажимаем кнопку «OK». В результате программа берет значение количества частей по умолчанию (Рис. 5).

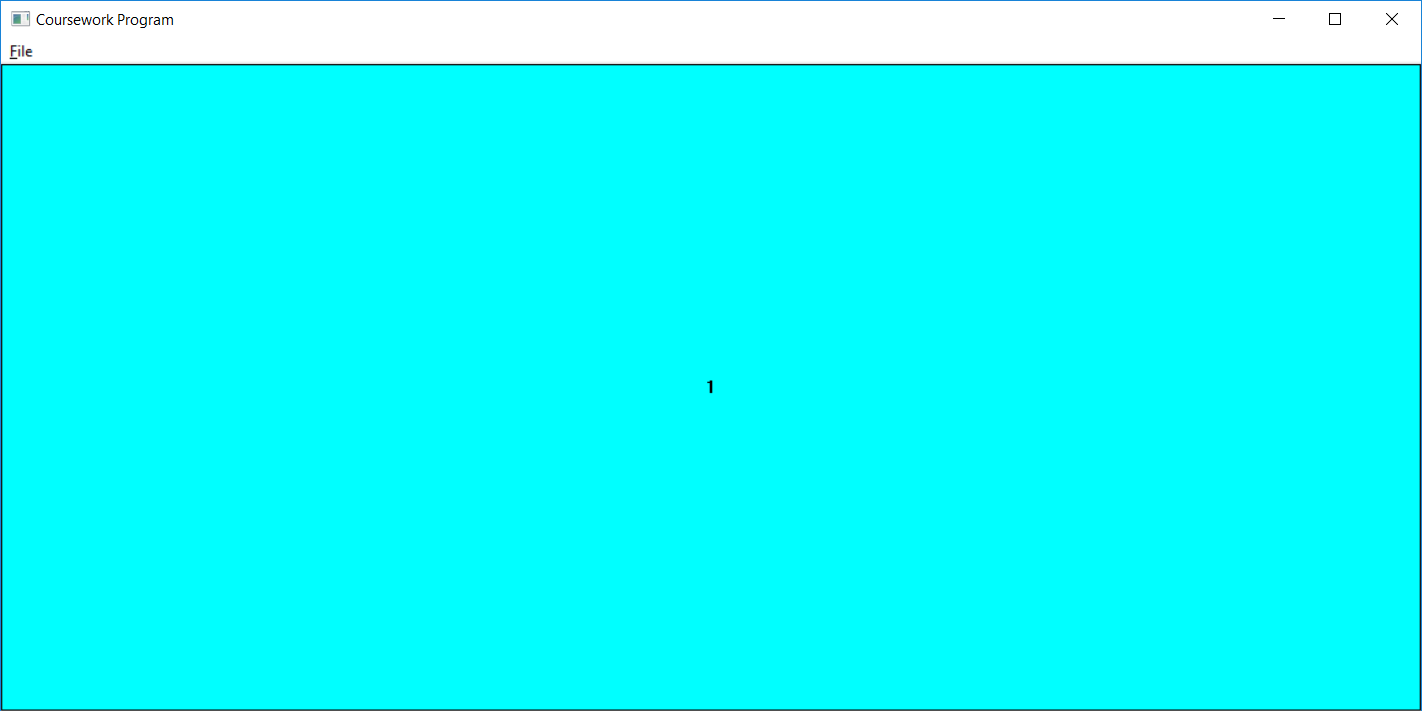


Рисунок 5. Попытка оставить поле пустым

Тестирование исключительности ситуации вводе в поле не числовых символов, например, букв или специальных знаков.

Для проверки открываем форму «Settings» и заполняем поле для ввода количества частей «N» нечисловыми. В результате программа берет значение количества частей по умолчанию (Рис 6).

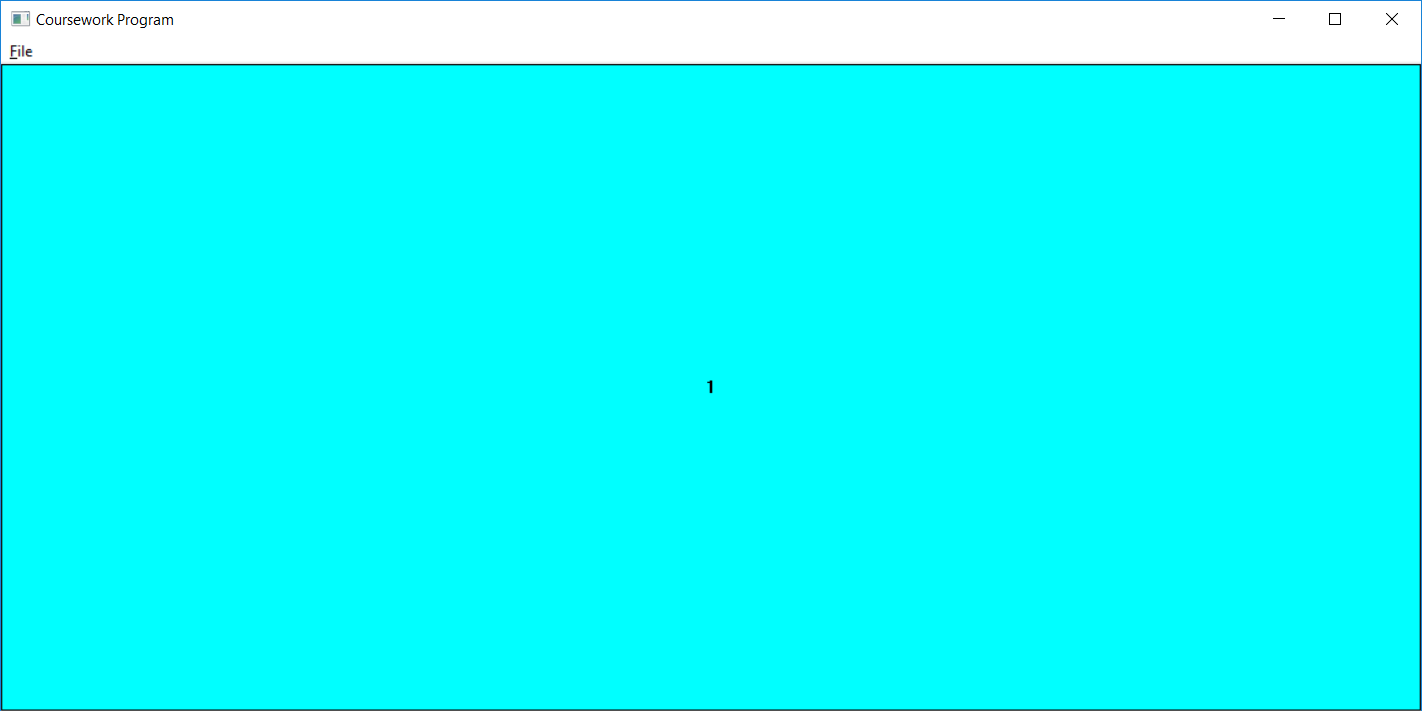


Рисунок 6. Попытка заполнить поле нечисловым значением

# Список литературы

1. Маркова В.В. Графические интерфейсы информационных систем. РГРТУ.
2. Н. А. Литвиненко Технология программирования на C++. Win32 API- приложения., – М. БХВ-Петербург 2010.
3. Юрий Щупак Win32 API - Эффективная разработка приложений., М. Питер, 2007;
4. MSDN - Сеть разработчиков Microsoft (Электронный ресурс). – https://msdn.microsoft.com/ru-ru/ (2016 г.).

# Приложение

### Текст программы

### Main.cpp

#include <Windows.h>

#include "resource.h"

LRESULT CALLBACK WndProc**(**HWND**,** UINT**,** WPARAM**,** LPARAM**);**

BOOL CALLBACK SettingsDlgProc**(**HWND**,** UINT**,** WPARAM**,** LPARAM**);**

int iCurrentColor **=** IDD\_CYAN**,**

iCurrentN **=** 5**;**

int WINAPI WinMain**(**HINSTANCE hInstance**,** HINSTANCE hPrevInstance**,**

PSTR szCmdLine**,** int iCmdShow**)**

**{**

static char szAppName**[]** **=** "Settings"**;**

MSG msg**;**

HWND hwnd**;**

WNDCLASSEX wndclass**;**

wndclass**.**cbSize **=** **sizeof(**wndclass**);**

wndclass**.**style **=** CS\_HREDRAW **|** CS\_VREDRAW**;**

wndclass**.**lpfnWndProc **=** WndProc**;**

wndclass**.**cbClsExtra **=** 0**;**

wndclass**.**cbWndExtra **=** 0**;**

wndclass**.**hInstance **=** hInstance**;**

wndclass**.**hIcon **=** LoadIcon**(**hInstance**,** szAppName**);**

wndclass**.**hCursor **=** LoadCursor**(NULL,** IDC\_ARROW**);**

wndclass**.**hbrBackground **=** **(**HBRUSH**)**GetStockObject**(**WHITE\_BRUSH**);**

wndclass**.**lpszMenuName **=** szAppName**;**

wndclass**.**lpszClassName **=** szAppName**;**

wndclass**.**hIconSm **=** LoadIcon**(**hInstance**,** szAppName**);**

RegisterClassEx**(&**wndclass**);**

hwnd **=** CreateWindow**(**szAppName**,** "Coursework Program"**,**

WS\_OVERLAPPEDWINDOW**,**

CW\_USEDEFAULT**,** CW\_USEDEFAULT**,**

CW\_USEDEFAULT**,** CW\_USEDEFAULT**,**

**NULL,** **NULL,** hInstance**,** **NULL);**

ShowWindow**(**hwnd**,** iCmdShow**);**

UpdateWindow**(**hwnd**);**

**while** **(**GetMessage**(&**msg**,** **NULL,** 0**,** 0**))**

**{**

TranslateMessage**(&**msg**);**

DispatchMessage**(&**msg**);**

**}**

**return** msg**.**wParam**;**

**}**

void PaintWindow**(**HWND hwnd**,** int iColor**,** int iN**)**

**{**

static COLORREF crColor**[**8**]** **=** **{** RGB**(**0**,** 0**,** 0**),** RGB**(**0**,** 0**,** 255**),**

RGB**(**0**,** 255**,** 0**),** RGB**(**0**,** 255**,** 255**),**

RGB**(**255**,** 0**,** 0**),** RGB**(**255**,** 0**,** 255**),**

RGB**(**255**,** 255**,** 0**),** RGB**(**255**,** 255**,** 255**)** **};**

HBRUSH hBrush**;**

HDC hdc**;**

RECT rect**;**

long width**;**

hdc **=** GetDC**(**hwnd**);**

GetClientRect**(**hwnd**,** **&**rect**);**

hBrush **=** CreateSolidBrush**(**crColor**[**iColor **-** IDD\_BLACK**]);**

hBrush **=** **(**HBRUSH**)**SelectObject**(**hdc**,** hBrush**);**

width **=** rect**.**right **/** iN **-** rect**.**left**;**

long mod **=** rect**.**right **%** iN**;**

width **=** mod **>** 0 **?** width **+** 1 **:** width**;**

rect**.**right **=** width**;**

**for** **(**int i **=** 1**;** i **<=** iN**;** i**++)**

**{**

char buffer**[**5**];**

\_itoa\_s**(**i**,** buffer**,** 5**,** 10**);**

Rectangle**(**hdc**,** rect**.**left**,** rect**.**top**,** rect**.**right**,** rect**.**bottom**);**

SetBkMode**(**hdc**,** TRANSPARENT**);**

DrawText**(**hdc**,** buffer**,** **-**1**,** **&**rect**,**

DT\_SINGLELINE **|** DT\_CENTER **|** DT\_VCENTER**);**

**if** **(**mod **==** 0**)**

width**--;**

rect**.**left **+=** width**;**

rect**.**right **=** rect**.**left **+** width**;**

mod**--;**

**}**

DeleteObject**(**SelectObject**(**hdc**,** hBrush**));**

ReleaseDC**(**hwnd**,** hdc**);**

**}**

void PaintTheBlock**(**HWND hCtrl**,** int iColor**,** int iN**)**

**{**

InvalidateRect**(**hCtrl**,** **NULL,** TRUE**);**

UpdateWindow**(**hCtrl**);**

PaintWindow**(**hCtrl**,** iColor**,** iN**);**

**}**

LRESULT CALLBACK WndProc**(**HWND hwnd**,** UINT iMsg**,** WPARAM wParam**,** LPARAM lParam**)**

**{**

static HINSTANCE hInstance**;**

PAINTSTRUCT ps**;**

**switch** **(**iMsg**)**

**{**

**case** WM\_CREATE**:**

hInstance **=** **((**LPCREATESTRUCT**)**lParam**)->**hInstance**;**

**return** 0**;**

**case** WM\_COMMAND**:**

**switch** **(**LOWORD**(**wParam**))**

**{**

**case** IDM\_SETTINGS**:**

**if** **(**DialogBox**(**hInstance**,** "SettingsBox"**,** hwnd**,**

SettingsDlgProc**))**

InvalidateRect**(**hwnd**,** **NULL,** TRUE**);**

**return** 0**;**

**case** IDM\_EXIT**:**

exit**(**EXIT\_SUCCESS**);**

**return** 0**;**

**}**

**break;**

**case** WM\_PAINT**:**

BeginPaint**(**hwnd**,** **&**ps**);**

EndPaint**(**hwnd**,** **&**ps**);**

PaintWindow**(**hwnd**,** iCurrentColor**,** iCurrentN**);**

**return** 0**;**

**case** WM\_DESTROY**:**

PostQuitMessage**(**0**);**

**return** 0**;**

**}**

**return** DefWindowProc**(**hwnd**,** iMsg**,** wParam**,** lParam**);**

**}**

BOOL CALLBACK SettingsDlgProc**(**HWND hDlg**,** UINT iMsg**,** WPARAM wParam**,** LPARAM lParam**)**

**{**

static HWND hCtrlBlock**;**

static int iColor**,** iN**;**

char buf**[**5**];**

**switch** **(**iMsg**)**

**{**

**case** WM\_INITDIALOG**:**

iColor **=** iCurrentColor**;**

iN **=** iCurrentN**;**

CheckRadioButton**(**hDlg**,** IDD\_BLACK**,** IDD\_WHITE**,** iColor**);**

\_itoa\_s**(**iN**,** buf**,** 5**,** 10**);**

SetDlgItemText**(**hDlg**,** IDD\_N**,** buf**);**

SetFocus**(**GetDlgItem**(**hDlg**,** IDD\_N**));**

hCtrlBlock **=** GetDlgItem**(**hDlg**,** IDD\_PAINT**);**

**return** FALSE**;**

**case** WM\_COMMAND**:**

**switch** **(**LOWORD**(**wParam**))**

**{**

**case** IDOK**:**

iCurrentColor **=** iColor**;**

iCurrentN **=** iN**;**

EndDialog**(**hDlg**,** TRUE**);**

**return** TRUE**;**

**case** IDCANCEL**:**

EndDialog**(**hDlg**,** FALSE**);**

**return** TRUE**;**

**case** IDD\_BLACK**:**

**case** IDD\_RED**:**

**case** IDD\_GREEN**:**

**case** IDD\_YELLOW**:**

**case** IDD\_BLUE**:**

**case** IDD\_MAGENTA**:**

**case** IDD\_CYAN**:**

**case** IDD\_WHITE**:**

iColor **=** LOWORD**(**wParam**);**

CheckRadioButton**(**hDlg**,** IDD\_BLACK**,** IDD\_WHITE**,**

**(**wParam**));**

PaintTheBlock**(**hCtrlBlock**,** iColor**,** iN**);**

**return** TRUE**;**

**case** IDD\_N**:**

char buf**[**5**];**

GetDlgItemText**(**hDlg**,** IDD\_N**,** buf**,** 5**);**

int value **=** **(**int**)**strtol**(**buf**,** **NULL,** 10**);**

iN **=** value **!=** 0 **?** value **:** 1**;**

PaintTheBlock**(**hCtrlBlock**,** iColor**,** iN**);**

**return** TRUE**;**

**}**

**break;**

**case** WM\_PAINT**:**

PaintTheBlock**(**hCtrlBlock**,** iColor**,** iN**);**

**break;**

**}**

**return** FALSE**;**

**}**

### Resource.rc

#include <Windows.h>

#include "resource.h"

Settings **MENU**

**{**

**POPUP** "&File"

**{**

**MENUITEM** "&Settings..."**,** IDM\_SETTINGS

**MENUITEM** "&Exit"**,** IDM\_EXIT

**}**

**}**

#define TABGRP (WS\_TABSTOP | WS\_GROUP)

SettingsBox **DIALOG** 200**,** 20**,** 140**,** 176

**STYLE** WS\_POPUP **|** WS\_DLGFRAME

**{**

**CTEXT** "Settings" **-**1**,** 0**,** 12**,** 140**,** 8

**CTEXT** "Settings Box Coursework Program" **-**1**,** 4**,** 36**,** 130**,** 8

**CTEXT** "" IDD\_PAINT**,** 68**,** 54**,** 60**,** 60

**GROUPBOX** "&Color" **-**1**,** 4**,** 50**,** 54**,** 100

**RADIOBUTTON** "B&lue" IDD\_BLUE**,** 8**,** 60**,** 40**,** 12

**RADIOBUTTON** "&Green" IDD\_GREEN**,** 8**,** 72**,** 40**,** 12

**RADIOBUTTON** "Cya&n" IDD\_CYAN**,** 8**,** 84**,** 40**,** 12

**RADIOBUTTON** "&Red" IDD\_RED**,** 8**,** 96**,** 40**,** 12

**RADIOBUTTON** "&Magenta" IDD\_MAGENTA**,** 8**,** 108**,** 40**,** 12

**RADIOBUTTON** "&Yellow" IDD\_YELLOW**,** 8**,** 120**,** 40**,** 12

**RADIOBUTTON** "&White" IDD\_WHITE**,** 8**,** 132**,** 40**,** 12

**GROUPBOX** "&N" **-**1**,** 68**,** 118**,** 60**,** 32

**EDITTEXT** IDD\_N**,** 72**,** 132**,** 50**,** 10

**DEFPUSHBUTTON** "OK" IDOK**,** 20**,** 156**,** 40**,** 14**,** WS\_GROUP

**PUSHBUTTON** "Cancel" IDCANCEL**,** 80**,** 156**,** 40**,** 14**,** WS\_GROUP

**}**

### Resource.h

#define IDM\_SETTINGS 1

#define IDM\_EXIT 2

#define IDD\_BLACK 10

#define IDD\_BLUE 11

#define IDD\_GREEN 12

#define IDD\_CYAN 13

#define IDD\_RED 14

#define IDD\_MAGENTA 15

#define IDD\_YELLOW 16

#define IDD\_WHITE 17

#define IDD\_N 20

#define IDD\_PAINT 30