Максимовская М.А.

Информатика - 11

Москва, 2019

Школа №109

Перечень тем 9-10 класса для повторения (в форме конспекта):

1. Перевод чисел в различных системах счисления. Арифметические действия с числами в различных системах счисления;
2. Измерение информации. Различные подходы к измерению информации. Формула Хартли для измерения информации;
3. Кодирование различных видов информации;
4. Основы логики. Логические основы компьютера;

Арифметические действия в различных позиционных системах счисления

Рассмотрим основные арифметические действия (сложение, вычитание, умножение и деление) в двоичной, восьмеричной и шестнадцатеричной системах счисления. Для этого удобно пользоваться специальными таблицами: сложения и умножения.

Таблица сложения:

Двоичная СС

+	0	1
0	0	1
1	1	10

Восьмеричная СС

+	0	1	2	3	4	5	6	7
0	0	1	2	3	4	5	6	7
1	1	2	3	4	5	6	7	10
2	2	3	4	5	6	7	10	11
3	3	4	5	6	7	10	11	12
4	4	5	6	7	10	11	12	13
5	5	6	7	10	11	12	13	14
6	6	7	10	11	12	13	14	15
7	7	10	11	12	13	14	15	16

Шестнадцатеричная СС

+	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
1	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	10
2	2	3	4	5	6	7	8	9	A	B	C	D	E	F	10	11
3	3	4	5	6	7	8	9	A	B	C	D	E	F	10	11	12
4	4	5	6	7	8	9	A	B	C	D	E	F	10	11	12	13
5	5	6	7	8	9	A	B	C	D	E	F	10	11	12	13	14
6	6	7	8	9	A	B	C	D	E	F	10	11	12	13	14	15
7	7	8	9	A	B	C	D	E	F	10	11	12	13	14	15	16
8	8	9	A	B	C	D	E	F	10	11	12	13	14	15	16	17
9	9	A	B	C	D	E	F	10	11	12	13	14	15	16	17	18
A	A	B	C	D	E	F	10	11	12	13	14	15	16	17	18	19
B	B	C	D	E	F	10	11	12	13	14	15	16	17	18	19	1A
C	C	D	E	F	10	11	12	13	14	15	16	17	18	19	1A	1B
D	D	E	F	10	11	12	13	14	15	16	17	18	19	1A	1B	1C
E	E	F	10	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D
F	F	10	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D	1E

Пример 1. Сложим числа 141,5 и 59,75 в различных СС.

$141{,}5_{10} + 59{,}75_{10}$:	$10001101{,}1_2 + 111011{,}11_2$:	$215{,}4_8 + 73{,}6_8$:	$8D{,}8_{16} + 3B{,}C_{16}$:
141,5 59,75 201,25	10001101,1 111011,11 11001001,01	215,4 73,6 311,2	8D,8 3B,C C9,4

Пример 2. Вычтем число 59,75 из числа 201,25 в различных СС. Сравните результат с Примером 1.

$201{,}25_{10} - 59{,}75_{10}$:	$11001001{,}01_2 - 111011{,}11_2$:	$311{,}2_8 - 73{,}6_8$:	$C9{,}4_{16} - 3B{,}C_{16}$:
201,25 59,75	11001001,01 111011,11	311,2 73,6	C9,4 3B,C

Таблица умножения:

Двоичная СС	Восьмеричная СС

*	0	1
0	0	0
1	0	1

*	0	1	2	3	4	5	6	7
0	0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6	7
2	0	2	4	6	10	12	14	16
3	0	3	6	11	14	17	22	25
4	0	4	10	14	20	24	30	34
5	0	5	12	17	24	31	36	43
6	0	6	14	22	30	36	44	52
7	0	7	16	25	34	43	52	61

Пример 3. Перемножим числа 115 и 51 в различных СС (умножение выполняется как в десятичной системе, но с учётом соответствующих таблиц умножения, сложение – с использованием таблиц сложения):

$115_{10} \cdot 51_{10}$:	$1110011_2 \cdot 110011_2$:	$163_8 \cdot 63_8$:
115 51 115 575 5865	1110011 110011 1110011 1110011 1110011 1110011 1011011101001	163 63 531 1262 13351

Пример 4. Перемножим числа 23,5 и 3,25 в различных СС. (умножение выполняется как в десятичной системе, запятой в результате отделяется столько знаков, сколько в обоих множителях вместе).

$23,5_{10} \cdot 3,25_{10}$	$10011,1_2 \cdot 11,01_2$	$27,4_8 \cdot 3,2_8$
23,5 3,25 1175 470 705 76,375	10011,1 11,01 100111 100111 100111 111111,011	27,4 3,2 570 1064 114,30

Примеры рассмотрены в презентации, представленной на сайте http://matem-109.ru, Информатика, Материалы, Презентации 8 – 11 класс.

Упражнения.

1. Сложите числа, а затем проверьте результаты, выполнив соответствующие десятичные сложения:

а) 1011101_2 и 1110111_2;	б) $1011,101_2$ и $101,011_2$;	в) 1011_2, 11_2 и $111,1_2$;	г) 1011_2 , $11,1_2$ и 111_2;

д) 37_8 и 75_8;	е) 165_8 и 37_8;	ж) $7{,}5_8$ и $14{,}6_8$;	з) 6_8, 17_8 и 7_8;
и) A_{16} и F_{16};	к) 19_{16} и C_{16};	л) $A{,}B_{16}$ и $E{,}F_{16}$;	м) E_{16}, 9_{16} и F_{16}.

2. Вычтите:

а) 111_2 из 10100_2;	б) $10{,}11_2$ из $100{,}1_2$;	в) $111{,}1_2$ из 10010_2;	г) 10001_2 из $1110{,}11_2$;
д) 15_8 из 20_8;	е) 47_8 из 102_8;	ж) $56{,}7_8$ из 101_8;	з) $16{,}54_8$ из $30{,}01_8$;
и) $1A_{16}$ из 31_{16};	к) $F9E_{16}$ из $2A30_{16}$;	л) $D{,}1_{16}$ из $B{,}92_{16}$;	м) ABC_{16} из 5678_{16}.

3. Перемножьте числа, а затем проверьте результаты, выполнив соответствующие десятичные умножения:

а) 101101_2 и 101_2;	б) 111101_2 и $11{,}01_2$;	в) $1011{,}11_2$ и $101{,}1_2$;	г) 101_2 и $1111{,}001_2$;
д) 37_8 и 4_8;	ж) $7{,}5_8$ и $1{,}6_8$;	е) 16_8 и 7_8;	з) $6{,}25_8$ и $7{,}12_8$.

4. Вычислите значения выражений:

а) $256_8 + 10110{,}1_2 \cdot (60_8 + 12_{10}) - 1F_{16}$;	б) $1010_{10} + (106_{16} - 11011101_2)\ 12_8$;

5. Расположите следующие числа в порядке возрастания:

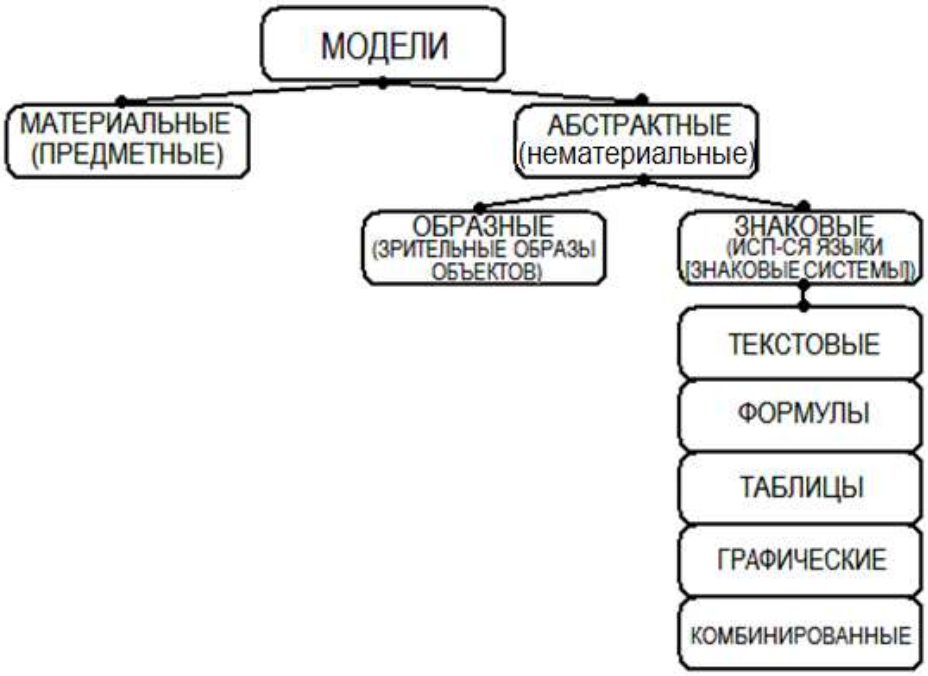
а) 74_8, 110010_2, 70_{10}, 38_{16};	б) $6E_{16}$, 142_8, 1101001_2, 100_{10};
в) 777_8, 101111111_2, $2FF_{16}$, 500_{10};	г) 100_{10}, 1100000_2, 60_{16}, 141_8.

Моделирование как метод познания окружающего мира

Моделирование – метод познания, состоящий в создании и исследовании моделей.

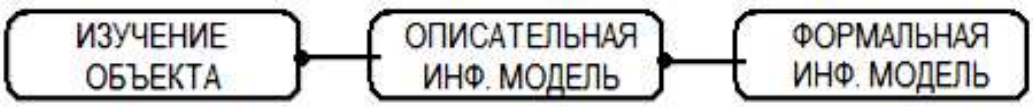
Модель – новый объект, отражающий существенные особенности изучаемого объекта, явления или процесса.

Один и тот же объект может иметь множество моделей, а разные объекты могут описываться одной моделью.



Формы представления моделей:

Процесс познания окружающего мира:



Формализация – процесс построения информационных моделей с помощью формальных языков.

Информационные модели системы:

Система – совокупность взаимосвязанных между собой объектов, функционирующая как единый объект.

Элемент системы – каждый из объектов, входящих в систему.

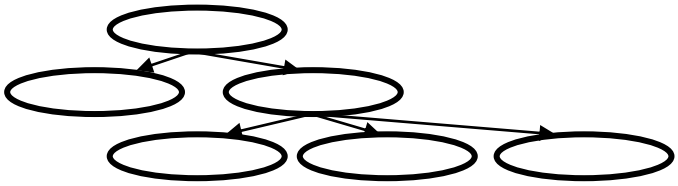
Статическая информационная модель – описывает состояние системы в определённый момент времени.

Динамическая информационная модель – описывает процесс изменения и развития системы.

Типы информационных моделей

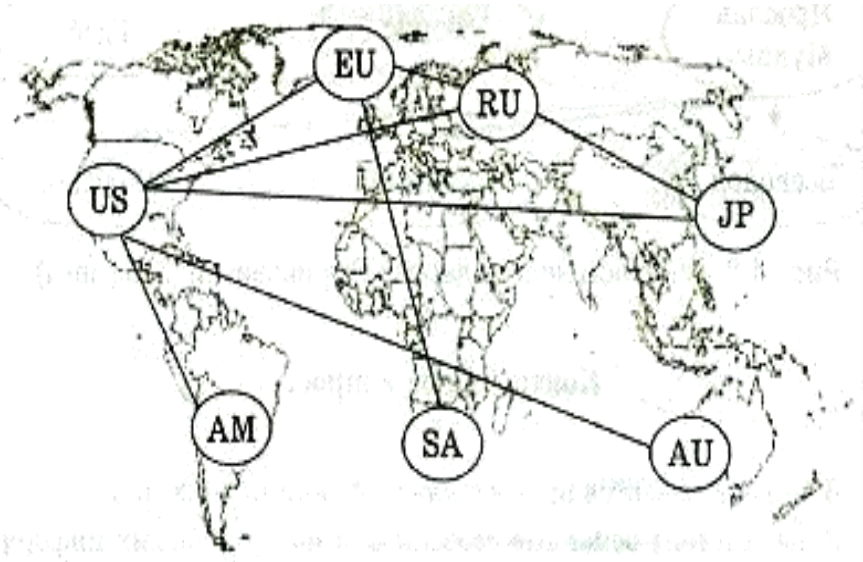
Табличные. Прямоугольная таблица, состоящая из столбцов и строк. Объекты и их свойства представлены в виде списка, значения свойств размещаются в ячейках таблицы. **Перечень однотипных объектов** размещён **в первом столбце (строке)** таблицы; **значения их свойств** – в **следующих столбцах (строках)** таблицы. Чаще всего используется для отображения статических информационных моделей.

Иерархические. Объекты распределены по уровням. Каждый элемент более высокого уровня состоит из элементов нижнего уровня. **Элемент нижнего уровня** может входить в состав **только одного элемента более высокого уровня**. Отображается с помощью графа: элементы системы – вершины графа (овалы, прямоугольники и т.д.), связи между элементами системы – направленная линия (стрелка). Такой граф называется **ориентированным**:



Используется как для создания статических, так и динамических информационных моделей.

Сетевые. Также отображаются с помощью графа, но **связи** между вершинами носят **двусторонний характер** (отображаются ненаправленными линиями), а также **возможна связь элементов нижнего уровня не только с одним элементом верхнего уровня, а также с элементами одного уровня**. Такой граф называется **неориентированным**. Например, сетевая структура глобальной сети Интернет:



Используется как для создания статических, так и динамических информационных моделей.

Система управления базами данных MS Access

База данных (БД) – информационная модель, позволяющая упорядоченно хранить данные о группе объектов, обладающих одинаковым набором свойств. Существуют **табличные**, **иерархические** и **сетевые** базы данных.

Табличная база данных – содержит перечень объектов, имеющих одинаковый набор свойств. Представляется в виде двумерной таблицы.

- **поле базы данных**: столбец таблицы, содержащий значения определённого свойства;

- **ключевое поле**: поле, значения которого однозначно определяют каждую запись в таблице;

- **запись базы данных**: строка таблицы, содержащая набор значений свойств одного объекта, размещённый в полях базы данных.

Основные типы полей:

Счётчик – последовательность целых чисел, которые задаются автоматически при вводе записи (значения не могут быть изменены пользователем). Чаще всего используется в ключевом поле;

Текстовый – может содержать до 255 символов;

Числовой – содержит числа;

Дата/время – содержит дату или время в заданном формате;

Денежный – содержит числа в денежной форме;

Логический – содержит значения Истина (Да) или Ложь (Нет);

Гиперссылка – содержит ссылку на информационный ресурс в Интернете;

Наиболее важные свойства полей:

Размер поля – определяет максимальную длину текстового или числового поля;

Формат поля – устанавливает формат данных;

Обязательное поле – указывает на то, что данное поле обязательно для заполнения

Иерархическая база данных – состоит из объектов различного уровня. Верхний уровень – один объект, на втором уровне – объекты второго уровня. Между объектами существуют связи, каждый объект более верхнего уровня (**предок**) может включать в себя несколько объектов более низкого уровня (**потомки**). «Потомок» не может иметь более одного «предка». Объекты-потомки, имеющие одного общего предка, называются **близнецами**.

Пример иерархической базы данных – Реестр Windows:

Иерархическая база данных – доменная система имён сети Интернет:

Первый уровень – таблица, содержащая перечень доменов верхнего уровня (всего 264, из них 7 административных, 257 - географические);

Второй уровень – таблица, содержащая перечень доменов второго уровня;

Третий уровень – таблица, содержащая перечень доменов третьего уровня для каждого домена второго уровня и IP-адреса компьютеров, находящихся в домене второго уровня.

№	Домен верхнего уровня	Комментарий
1	com	коммерческие организации
2	ru	Россия
264	zr	Заир

№	Домен второго уровня	Домен первого уровня
1	microsoft	com
2	intel	com
		com
1884466		com

№	Домен второго уровня	Домен первого уровня
1	keldysh	ru
2	mipkro	ru
11147		ru

№	Доменное имя компьютера	IP-адрес компьютера	Комментарий
1	www.mipkro.ru	213.171.37.202	WWW-сервер
2	ftp.mipkro.ru	213.171.37.203	FTP-сервер

Сетевая база данных. На связи элементов сетевой базы данных не накладывается никаких ограничений: «потомки» могут иметь более одного «предка», могут существовать также связи между «близнецами».

Практическая работа.
СУБД Access. Создание и работа с таблицами, формами, запросами и отчётами

1) Система управления базами данных (СУБД) – программа, позволяющая создавать, обрабатывать базы данных, осуществлять поиск данных.

Основные объекты СУБД:

Таблицы: базовый объект БД, в которых хранится вся информация. Каждая строка в таблице – запись БД, каждый столбец – поле БД.

Запросы: главное предназначение – отбор данных на основании заданных условий.

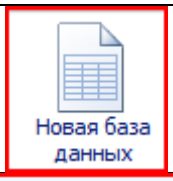
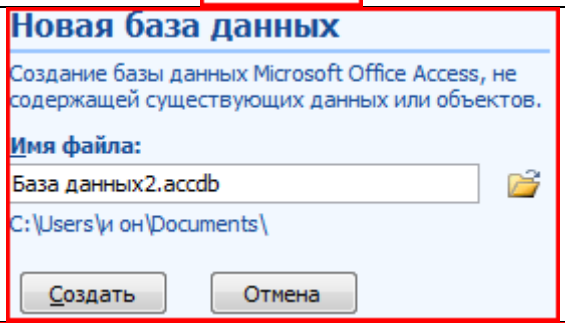
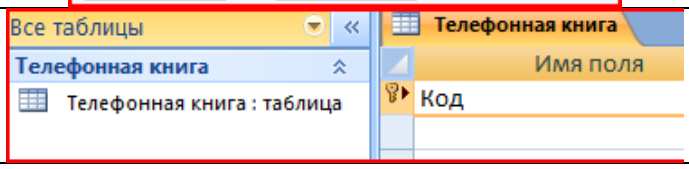
Формы: позволяют отображать данные, содержащиеся в таблицах или запросах в более удобной форме. Через форму можно добавлять в таблицы новые данные, редактировать или удалять существующие. Форма может содержать рисунки, графики и другие внедрённые объекты.

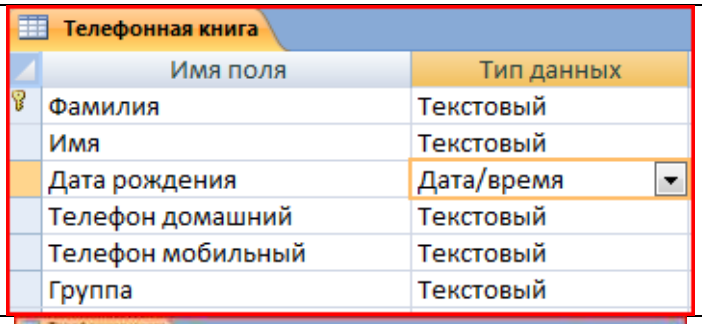
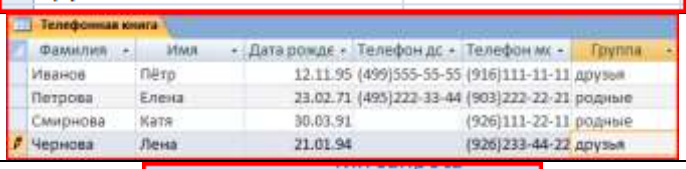
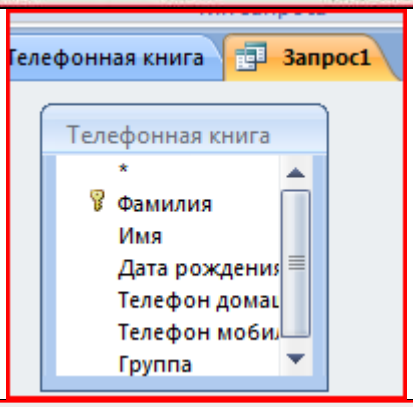
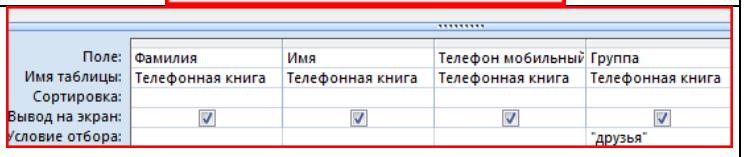
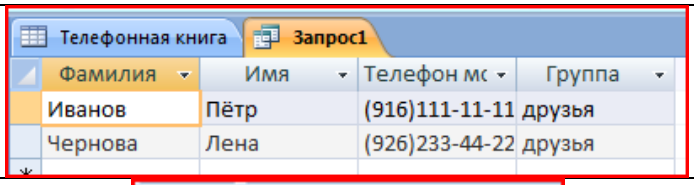
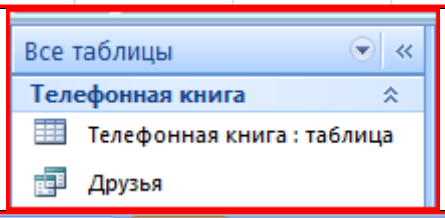
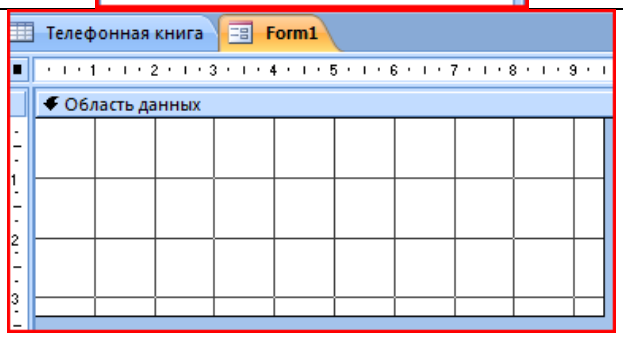
Отчёты: предназначены для печати данных, содержащихся в таблицах и запросах.

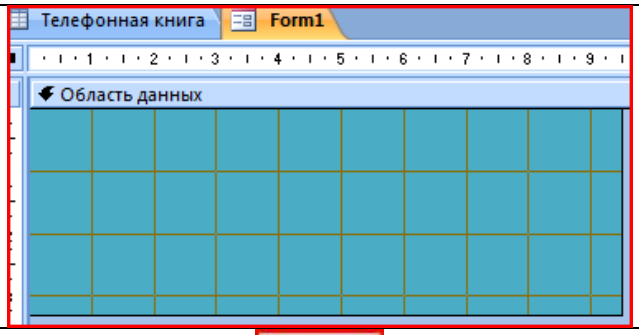
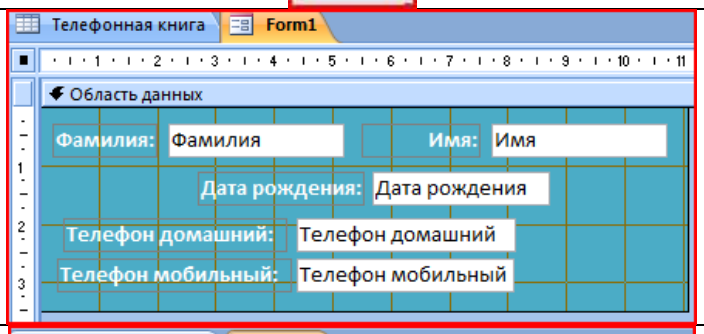
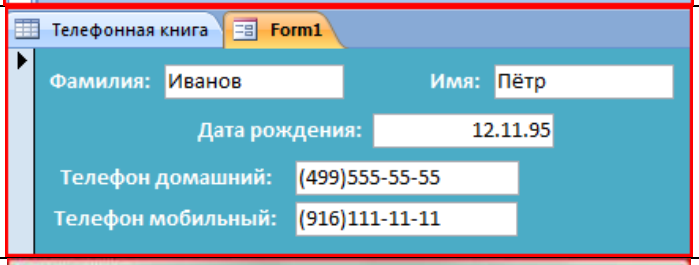
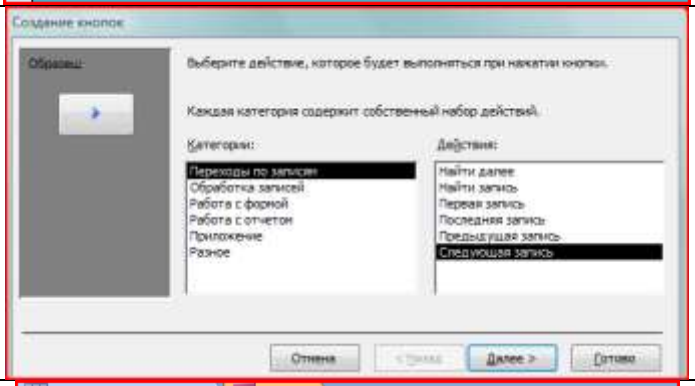
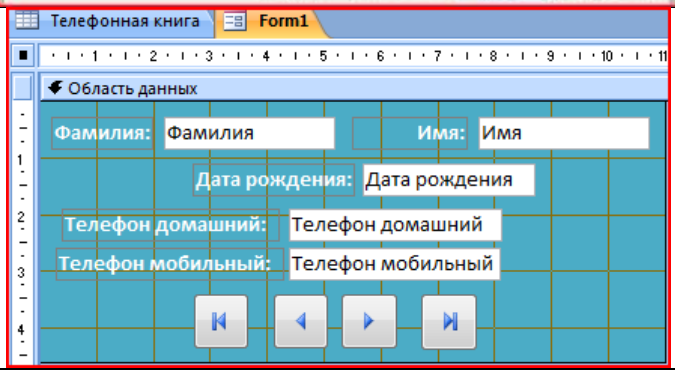
Макросы: служат для автоматизации повторяющихся операций.

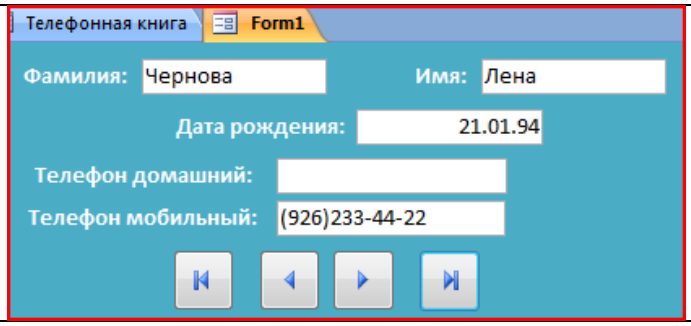
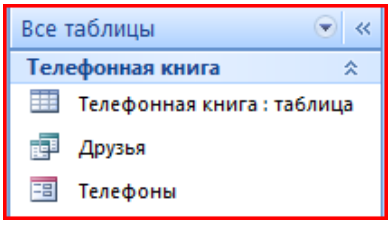
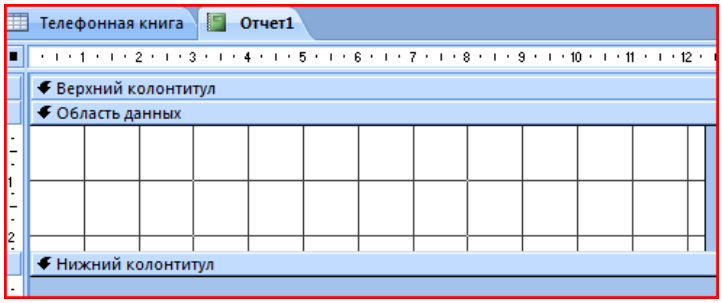
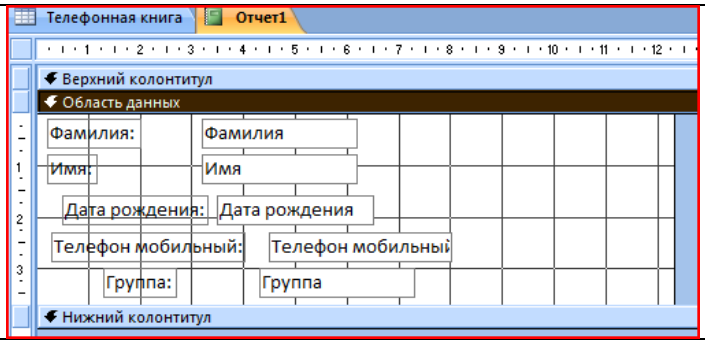
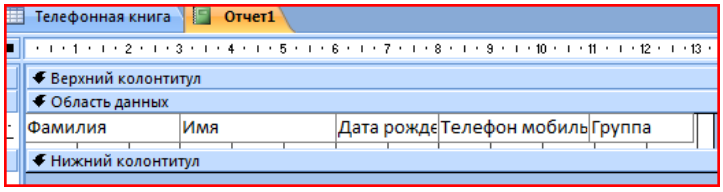
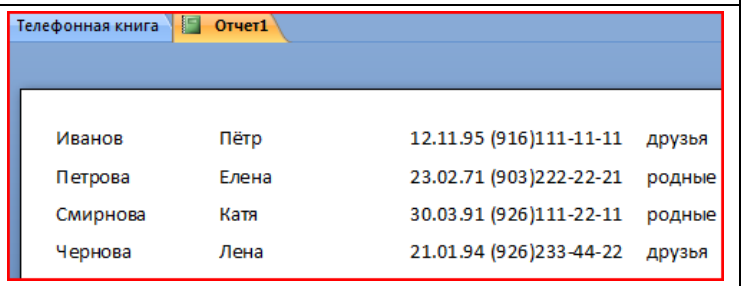
Модули (процедуры обработки событий): служат для автоматизации работы с БД. Пишутся на языке VBA.

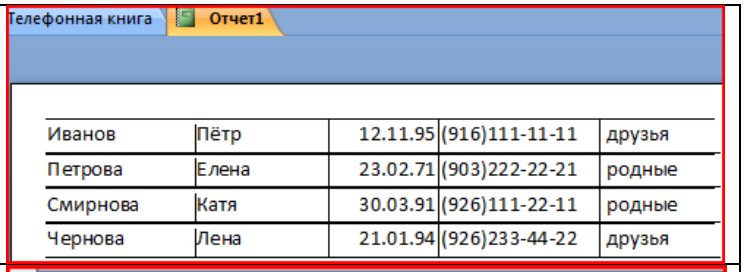
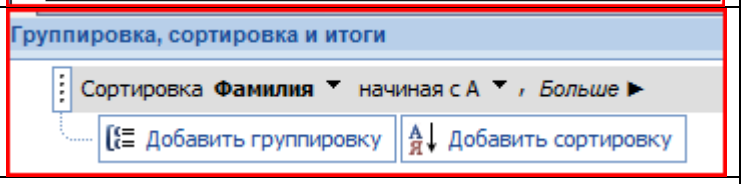
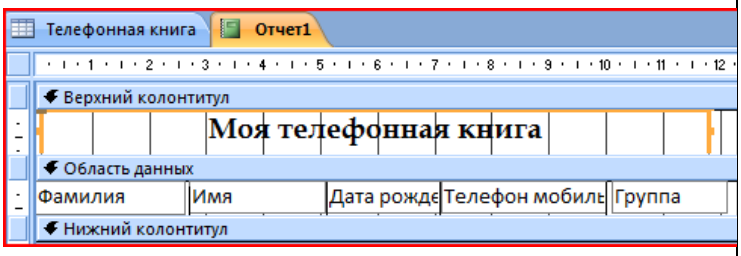
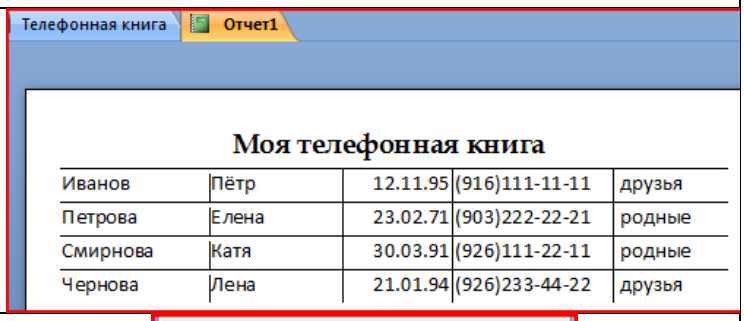
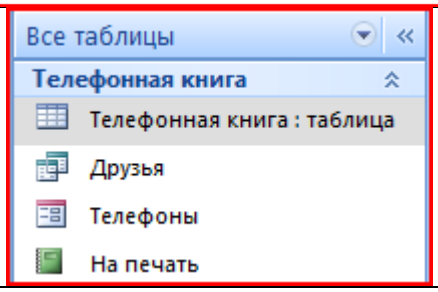
2) **Проектирование Базы данных**

1. Открыть MS Office Access 2007. Выбрать создание Новой базы данных.	
2. Нажмите Создать:	
3. Перейдите в Режим Конструктор и сохраните таблицу с именем Телефонная книга:	

4. Заведите поля таблицы и снимите значок  с поля Фамилия. Для поля Дата рождения установите Тип данных Дата/время, формат Краткий формат данных:	
5. Перейдите в Режим Таблица, сохранив таблицу. Заведите не менее 10 записей в данную таблицу:	
6. Перейдите на вкладку Создание и выберите Конструктор запросов. В появившемся окне Добавление таблицы добавьте таблицу Телефонная книга. Получится:	
7. Перетащите поля Фамилия, Имя, Телефон мобильный, Группа в таблицу Конструктора Запросов и установите Условие отбора «друзья»:	
8. Перейдите в Режим Таблица. Получится:	
9. Закройте и сохраните Запрос1 с именем Друзья. Область Переходов примет вид: Аналогично: создайте Запрос Родные.	
10. На вкладке Создание выберите Конструктор Форм. Появится пустая форма. Уменьшите её размеры. В Области Свойства формы на вкладке Данные установите Источник данных Телефонная книга.	

11. Выделите Область данных (кликнуть). В Окне свойств на вкладке Макет установите цвет фона или фоновый рисунок:	
12. На вкладке Конструктор откройте область Добавить поля:	
13. Из Списка полей перетащите в Форму необходимые поля, отредактируйте их (установите необходимый шрифт, начертание, цвет и т.д., размеры полей):	
14. Перейдите в Режим Формы. Посмотри на результат: Далее, вернитесь в Режим Конструктор.	
15. Добавьте Элемент управления Кнопка. Выберите действие Кнопки – Переход по записям – Следующая запись. Нажмите Готово.	
16. Аналогично: добавьте Кнопку с действием Переход по записям – Предыдущая запись. Добавьте Кнопки с действиями Переход по записям – Первая запись и Последняя запись. Получится:	

17. Перейдите в Режим формы. Попробуйте, как работают поставленные кнопки. Перейдите к последней записи и нажмите Кнопку Следующая запись. В пустые поля введите соответствующие значения.	
18. Закройте и сохраните форму с именем Телефоны. Область перехода примет вид: Создайте свою форму на основе Запроса Друзья или Родные.	
19. На вкладке Создание выберите Конструктор отчётов. В появившемся пустом отчёте уберите Верхний и Нижний колонтитул. Уменьшите область данных. В Окне свойств установите Источник записей Телефонная книга.	
20. В Область данных добавьте необходимые поля:	
22. Удалите наименования полей, оставив сами поля, и выстройте поля в одну строку. Уменьшите Область данных:	
23. Перейдите в Режим Предварительный просмотр. Получится: Так будет выглядеть печатный документ. Можно в Режиме Конструктор внести изменение в расположение полей.	

24. В Режиме Конструктор нарисуйте вертикальное и горизонтальное графление. В Режиме Предварительного просмотра это будет выглядеть так:	
25. В Режиме Конструктор нажмите Добавить сортировку, выберите поле Фамилия.	
26. Раскройте Верхний колонтитул и на вкладке Конструктор с помощью инструмента Надпись добавьте заголовок, например, такой:	
26. В Режиме Предварительный просмотр это будет выглядеть так:	
27. Закройте и сохраните Отчет1 с именем На печать. Область переходов примет вид:	

Внимание! Приготовьтесь к тестированию по темам Моделирование и Базы данных.

Темы: "Информационная деятельность человека"; "Информационные основы процесса управления"; "Информатизация общества" и "Компьютерные сети"

Задание: выполнить конспекты (ссылки указаны в журнале). Быть готовыми к собеседованию. Подготовиться к тестированию по этим темам.

Материалы взяты с http://msdn.microsoft.com/ru-ru/library/dd460654 и из руководства по программированию на языке Visual Basic http://msdn.microsoft.com/ru-ru/library/y4wf33f0.

Visual Basic 2010 является развитием языка Visual Basic, который предназначен для эффективного создания типобезопасных и объектно-ориентированных приложений. **Visual Basic позволяет разработчикам создавать приложения Windows, веб-приложения и приложения для мобильных устройств.** Программы, написанные на языке Visual Basic, как и на других языках, предназначенных для Microsoft. NET Framework, отличаются безопасностью и поддержкой взаимодействия.

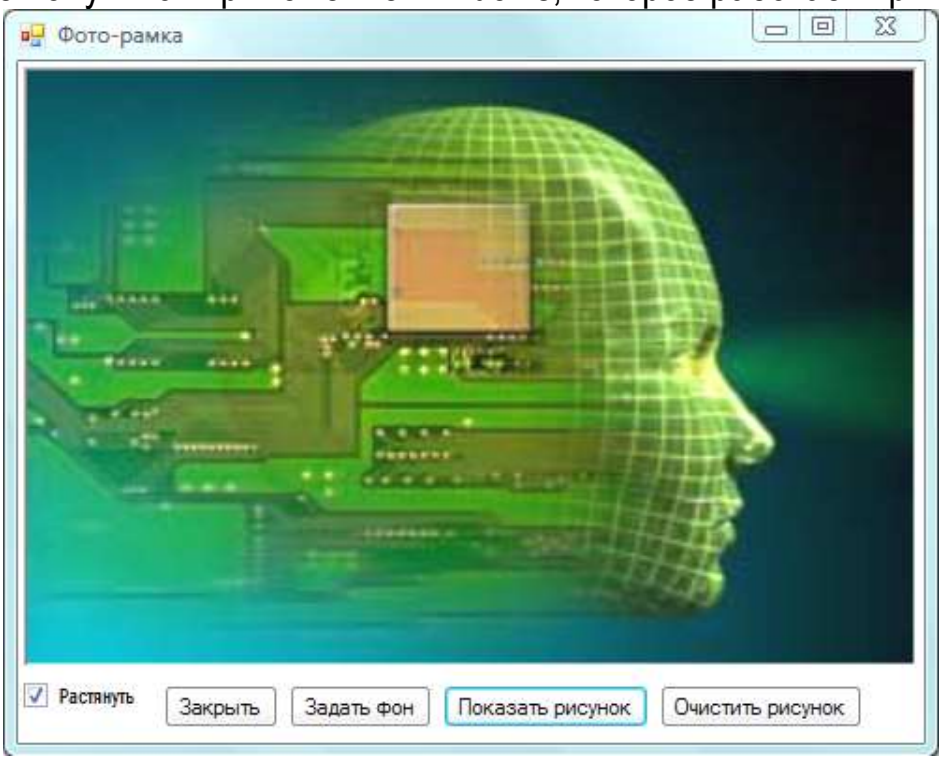
Это поколение Visual Basic продолжает традицию обеспечения возможности быстрого и простого создания приложений на основе .NET Framework.

Изучая Visual Basic мы будем оперировать следующими понятиями: класс (описывают типы объектов) и объект (экземпляр класса). Класс состоит из членов класса, которые обладают свойствами (описывают данные класса), методами (задают поведение класса) и событиями (обеспечивают связь между различными классами и объектами).

Основной объект VBasic — форма. На ней располагаются управляющие элементы (кнопки, бегунки, графические и текстовые окна и т.д.).

Проект 1. Создание программы просмотра изображения («Фото-рамка»)

В результате получится приложение Windows, которое работает примерно так:



Содержание проекта:

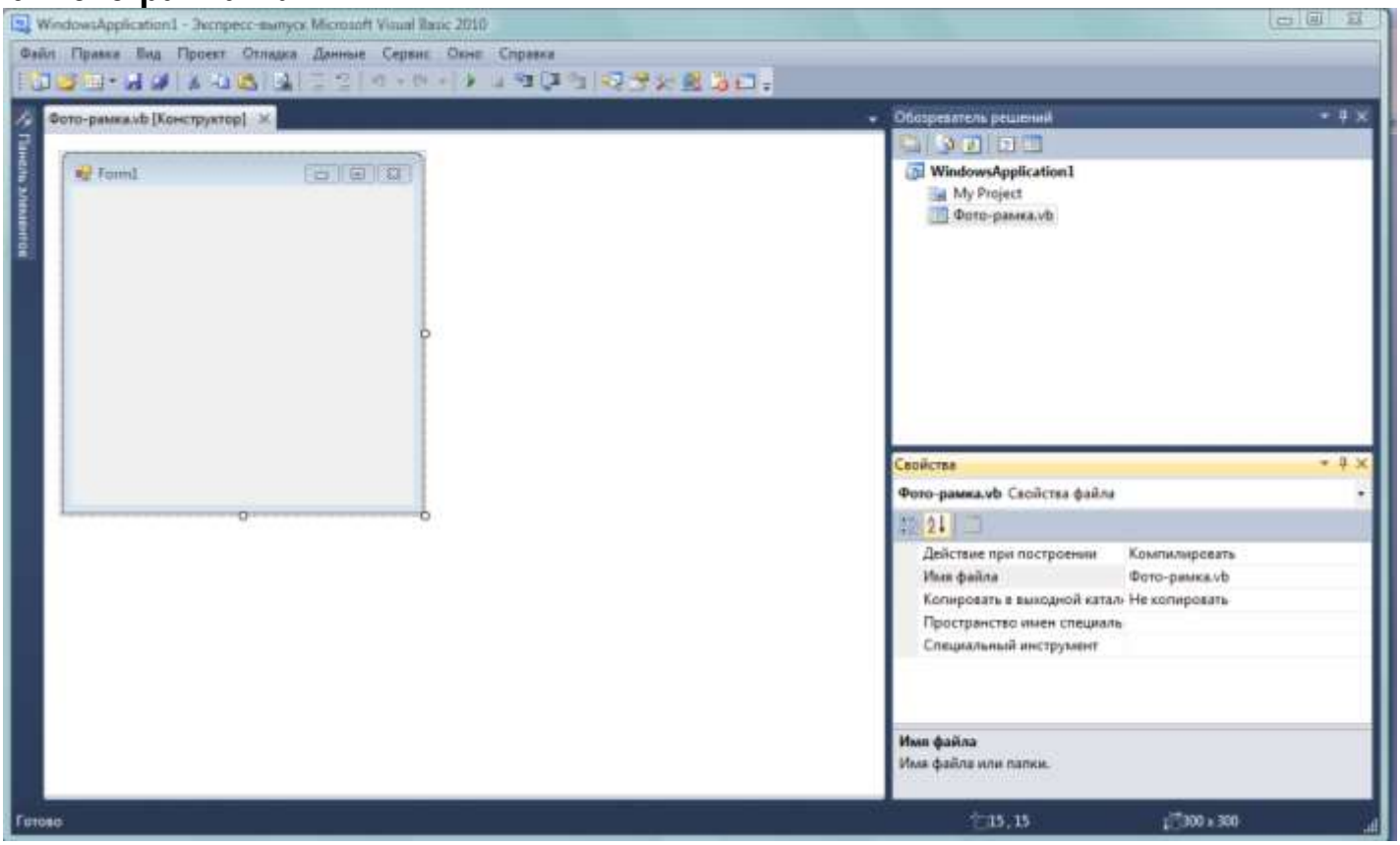
- Форма располагается в центре экрана;
- Большую часть формы занимает графическое окно;
- Под окном располагаются пять элементов управления (флажок и кнопки);
- Для каждой кнопки и флажка прописывается определённый код;
- Проект сохраняется и публикуется как автономное приложение.

Выполнение проекта:

1 этап. Создание проекта. Задание свойств формы в окне Свойства.

1. Открыть VB 2010, Создать проект..., в открывшемся окне Создать проект выберите Приложение Windows Forms.

2. По умолчанию открывается форма **Form1**. В окне **Свойства** измените **Имя файла** на **Фото-рамка.vb**:

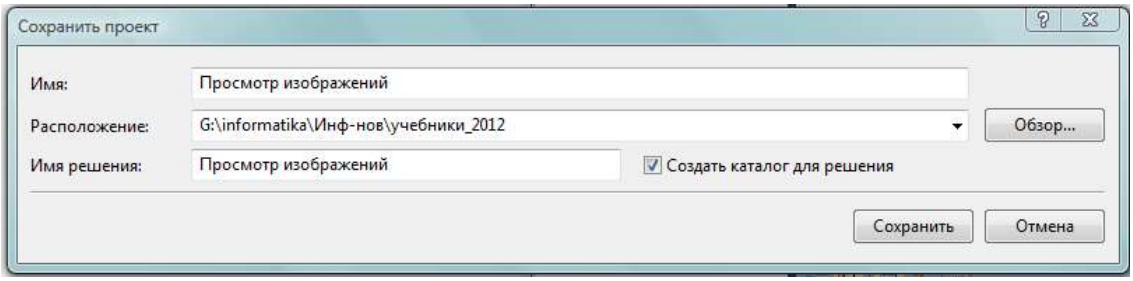


3. Выделите форму. В окне Свойства появятся все свойства формы по умолчанию. Измените свойства **Text** с **Form1** на **Фото-рамка** и **Size** с 300; 300 на, например, 500; 400.

4. Измените значение свойства **StartPosition** с **WindowsDefaultLocation** на **CenterScreen**.

5. Сохраните проект в свою папку: **Файл** – **Сохранить всё** (или на панели инструментов нажмите кнопку ). Измените **Имя** на **Просмотр изображений** и

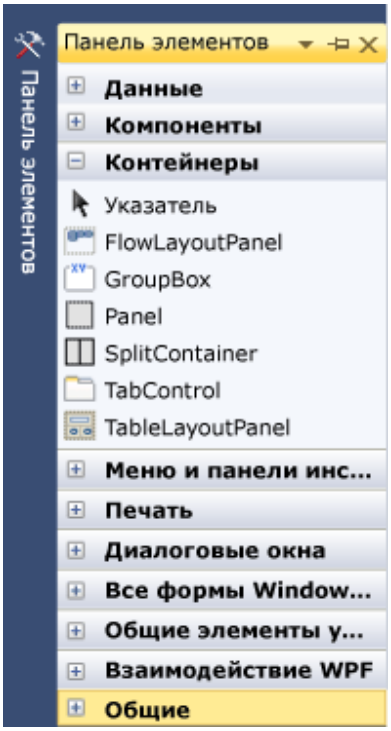
укажите расположение. Обязательно поставьте **флажок** на **Создать каталог для решения**:

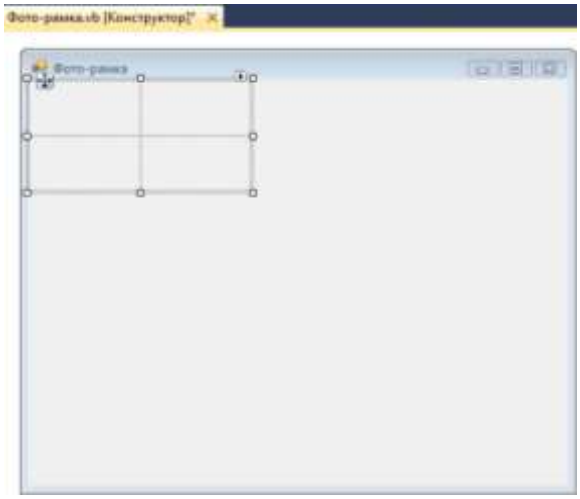


6. Запустите **Отладку** (кнопка на панели инструментов). Обратите внимание: окно приложения появляется в центре экрана и его можно увеличить.

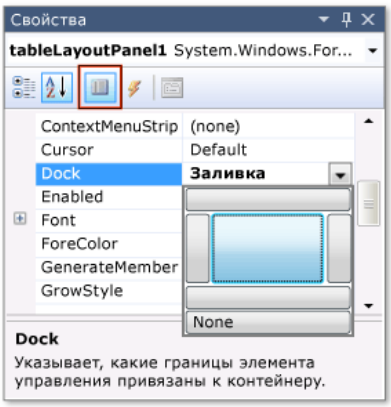
2 этап. Создание макета формы с помощью элемента управления TableLayoutPanel.

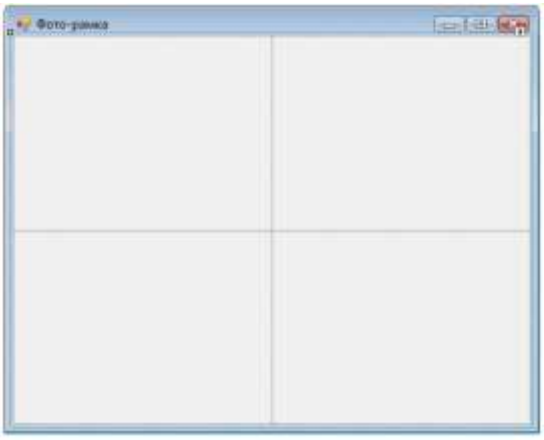
1. Слева от формы находится **Панель элементов**. Нажмите на неё и в выдвинувшейся панели выберите группу Контейнеры, найдите элемент **TableLayoutPanel** и дважды быстро щёлкните по нему. В форму добавится элемент управления:



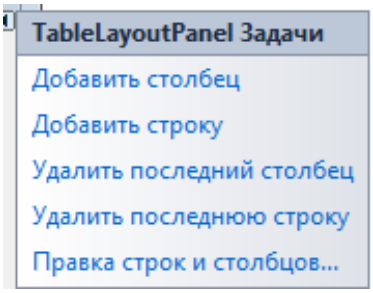


2. Щёлкните по элементу и в окне Свойства установите свойство **Dock – Fill** (Заливка) (**TableLayoutPanel** заполнит всю форму):

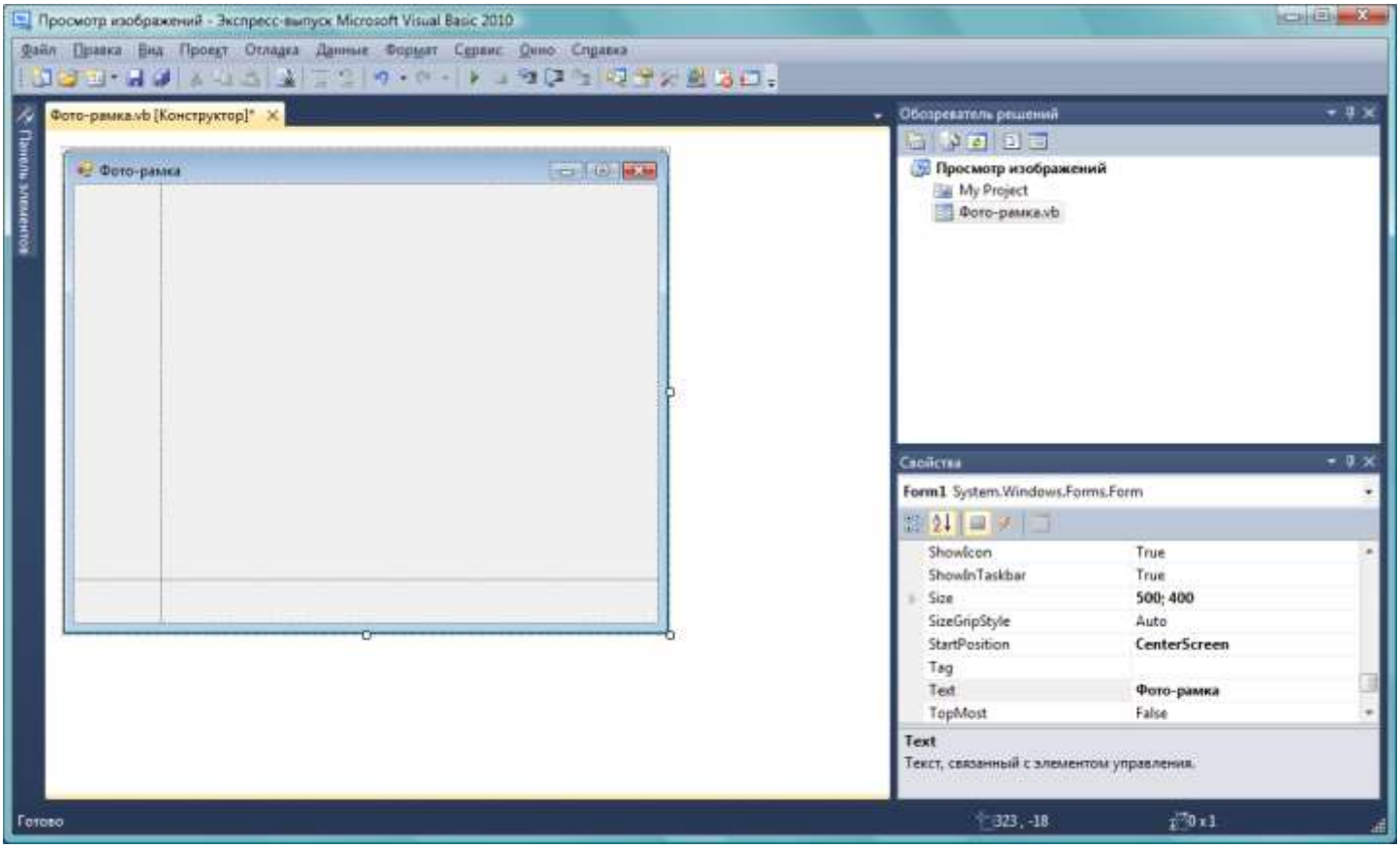




3. В верхнем правом углу выделенного элемента находится ▶. Нажмите на него, откроются **TableLayoutPanel Задачи**. Выберите Правка строк и столбцов. Для левого столбца установите размер 15%, для левого – 85%, для верхней строки – 90%, для нижней – 10%

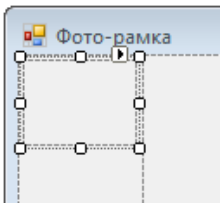


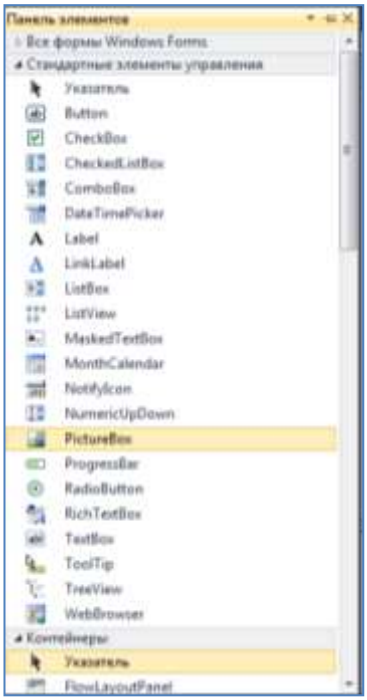
4. Получится:



3 этап. Добавление элементов управления PictureBox и CheckBox в форму.

1. На Панели элементов и разверните **Стандартные элементы управления**. Дважды щелкните элемент управления **PictureBox**. В левой верхней ячейке TableLayoutPanel появится элемент:



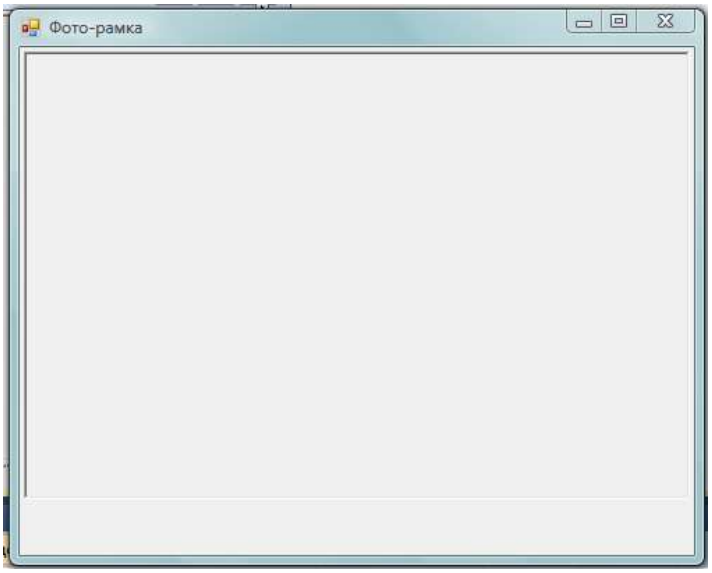


Откройте **PictureBox Задачи** и щёлкните элемент **Закрепить в родительском контейнере**. В результате этого действия у элемента управления PictureBox свойство **Dock** принимает значение **Fill**.

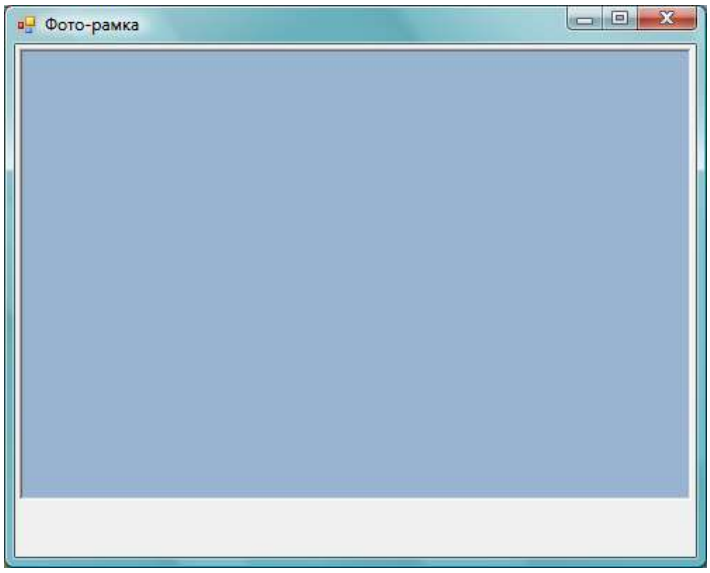
2. Чтобы элемент управления PictureBox занимал два столбца установите значение свойства **ColumnSpan равным 2**.

3. Чтобы по умолчанию элемент управления PictureBox был пустым установите для свойства **BorderStyle** значение **Fixed3D**.

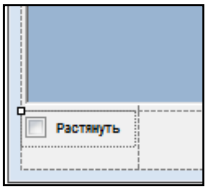
4. Запустите Отладку приложения. Должно получиться:



5. Можно установить для **PictureBox** значение свойства **BackColor**.

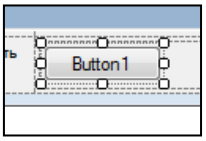


6. На Панели элементов дважды щелкните элемент **CheckBox** на панели элементов (элемент управления добвится в нижнюю левую ячейку). Выделите элемент управления **CheckBox** и установите для свойства **Text** значение **Растянуть** (можно поменять шрифт и размер шрифта).

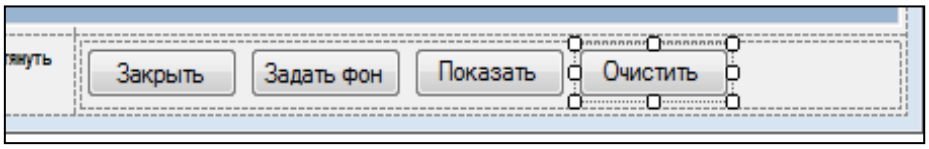


7. Для добавления кнопок на панели элементов перейдите к группе **Контейнеры** дважды щёлкните элемент управления **FlowLayoutPanel**. Затем выберите в списке задач **Закрепить в родительском контейнере**.

8. Выделите элемент **FlowLayoutPanel** и на Панели элементов в группе Стандартные элементы управления дважды щёлкните на элемент **Button**. Скопируйте его (Ctrl+C) и вставьте ещё три кнопки (Ctrl+V).

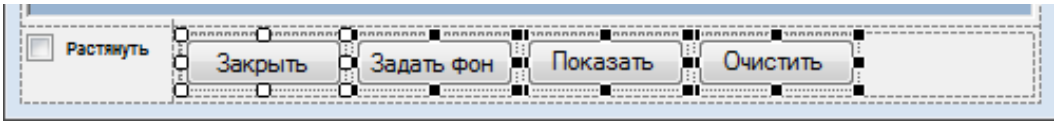


9. Последовательно выделяя каждую кнопку меняйте значение свойства **Text** для **Button 1** – на **Закрыть**, далее – **Задать фон**, **Показать рисунок**, **Очистить рисунок**. Получится:

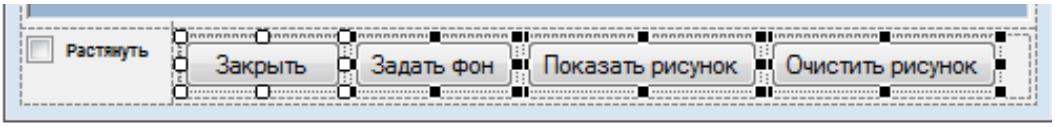


Проверьте, чтобы для элемента управления **FlowLayoutPanel** свойство **FlowDirection** имело значение **RightToLeft**.

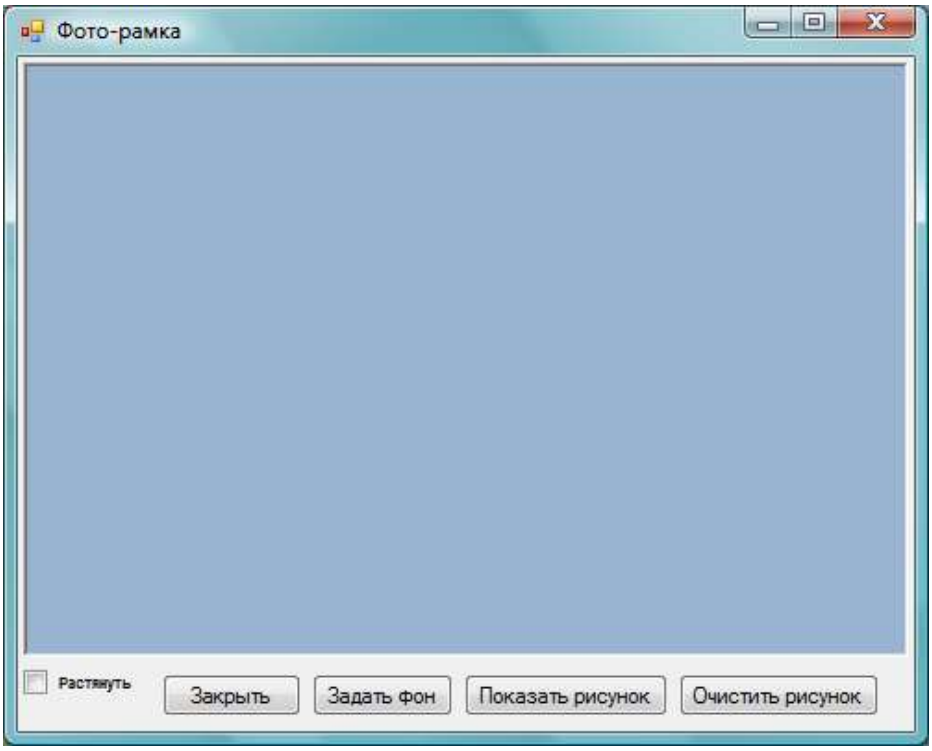
10. Чтобы надписи на кнопках были видны полностью, выделите кнопку **Закрыть** и, не отпуская клавишу **Ctrl,** выделите все остальные кнопки:



В окне **Свойства** установите значение свойства **AutoSize** – **True**:



11. Сохраните проект (Сохранить всё). Запустите отладку. Должно получиться:



4 этап. Присвоение имен элементам управления "Кнопка". Создание событийных процедур (Private Sub).

1. До сих пор при разработке проекта можно было обходиться теми возможностями, которые предлагает VisualBasic. Но, если мы хотим, чтобы расположенные на форме элементы начали работать, потребуется для каждого элемента прописать определённый код.

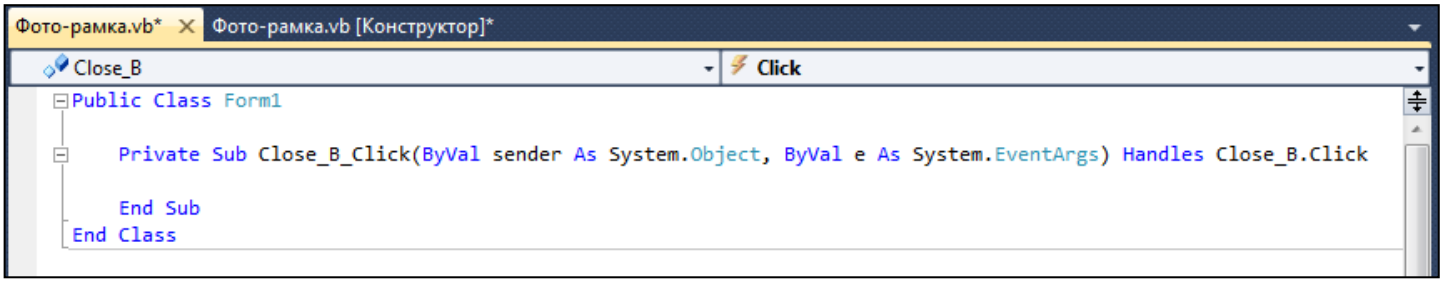
В форме существует только один элемент управления **PictureBox**. Когда он был добавлен, интегрированная среда разработки присвоила ему имя **PictureBox1** (это можно увидеть в окне **Свойства** – свойство **(Name)**). Также имеется только один элемент управления **CheckBox** с именем **CheckBox1**[1]. Так как существует только по

[1] Если бы соответствующих элементов было бы больше, то к имени была бы добавлена цифра 2, 3 и т.д., по порядку.

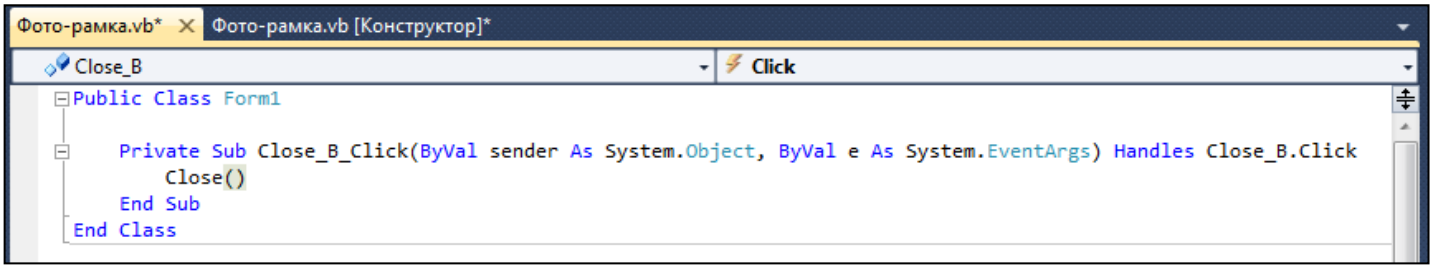
одному экземпляру каждого такого компонента, то при написании кода для **PictureBox1** или **CheckBox1** вполне понятно, к какому элементу формы обращаются.

На нашей форме содержится целых четыре кнопки. И, не смотря на то, что мы меняли для каждой кнопки свойство **Text**, значение свойства **(Name)** для каждой кнопки не изменилось: **Button1**, **Button2**, **Button3** и **Button4**. Поэтому, когда мы перейдём к написанию кода, будет не очень понятно, какой код прописывать для какой кнопки[2]. Удобнее всего для элементов задавать «говорящие» имена: для кнопки **Закрыть** значение свойства **(Name)** можно установить, например, **Close_B** (close – закрыть), для кнопки **Задать фон** – **Backgr_B**, **Показать рисунок** – **Show_B**, **Очистить рисунок** – **Clear_B**[3].

2. Дважды кликните на элемент **Закрыть** на форме. Откроется вкладка:



- **Public Class Form1** – объект является экземпляром **Класса Form**.
- Создана процедура **Private Sub** для элемента **Close_B**, в которой используется метод[4] **Click**: **Close_B_Click()**. В принципе, то, что стоит внутри круглых скобок, можно удалить.
- Там, где мигает курсор, можно уже прописывать код. Так как данный элемент должен закрывать форму, то код выглядит так:



Этот код содержит всего один оператор – **Close()** (закрыть всё).

Обратите внимание: когда вы начинаете набирать первую букву кода, то появляется меню с набором нужных команд, и вам достаточно просто выбрать

---

[2] В нашем случае разных экземпляров одного элемента немного, но если проект станет более сложным, то присвоение говорящих имён разным элементам станет особенно актуальным.

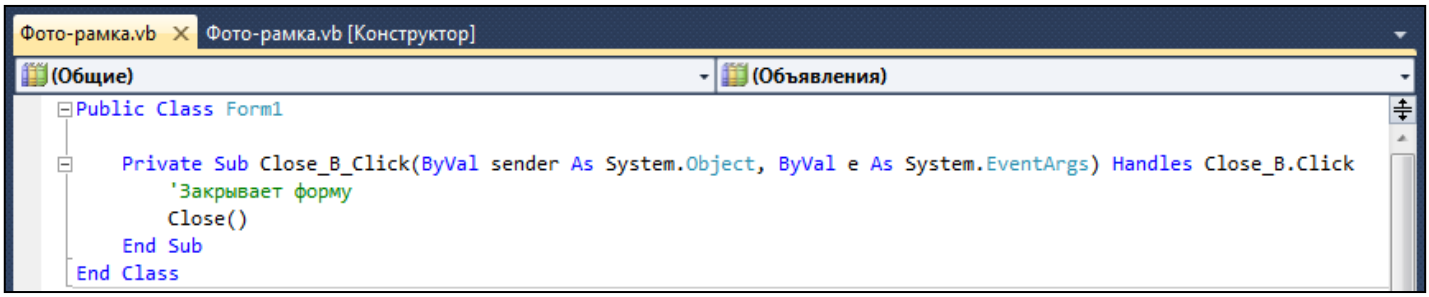
[3] Имя элемента должно состоять только из одного слова. Букву "_B" мы оставим как указание на то, что данный элемент является кнопкой.

[4] Метод – действие, которое выполняет объект.

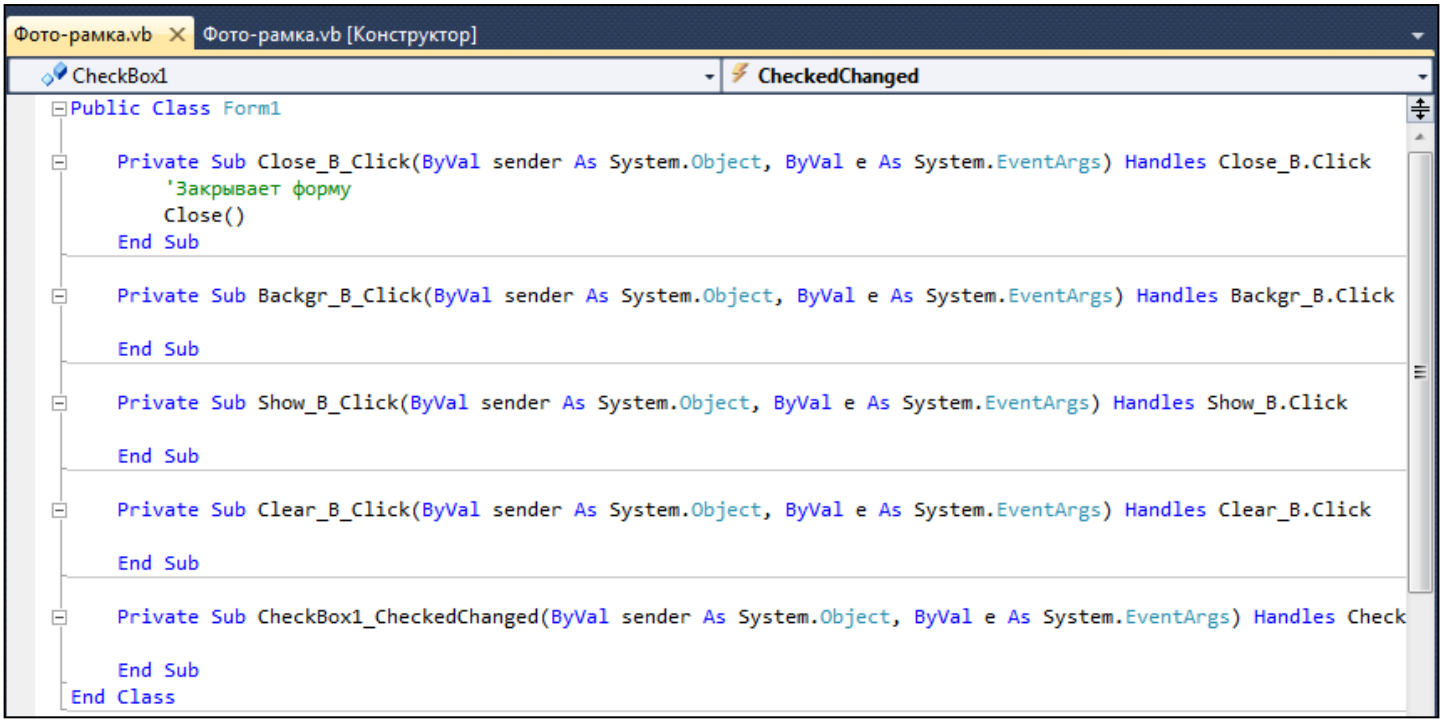
нужное. Это работает система **IntelliSense**, которая даёт возможность вместо ручного набора **выбрать** нужный метод (), свойство () или событие().

Запустите **Отладку** и нажмите на кнопку **Закрыть**. Форма должна закрыться.

Можно добавить комментарий (он начинается знаком «‘» и не исполняется):



3. Перейдите в **Конструктор** и таким же образом добавьте процедуры для оставшихся кнопок и элемента **CheckBox1**:



Обратите внимание: для кнопок также используется метод **Click()**, а для **CheckBox1** (который будет впоследствии «растягивать» рисунок по размерам рамки) используется другой метод – **CheckedChanged()**: **CheckBox1_CheckedChanged().**

4. Пропишем процедуру для элемента **CheckBox1**:

```
Private Sub CheckBox1_CheckedChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Ch
    'Если пользователь поставил флажок, рисунок приобретает
    'свойство "растянут";
    'Если флажок очищен, рисунок имеет нормальные размеры.
    If CheckBox1.Checked Then
        PictureBox1.SizeMode = PictureBoxSizeMode.StretchImage
    Else
        PictureBox1.SizeMode = PictureBoxSizeMode.Normal
    End If
End Sub
```

PictureBox1.SizeMode = PictureBoxSizeMode.* - свойство «изменение размера» элемента PictureBox1 принимает заданное значение (либо «растянутое», либо нормальное).

Обратите внимание: как только вы начинаете писать код, среда **VisualBasic** опять «помогает» вам его прописывать, система **IntelliSense** начинает работать автоматически.

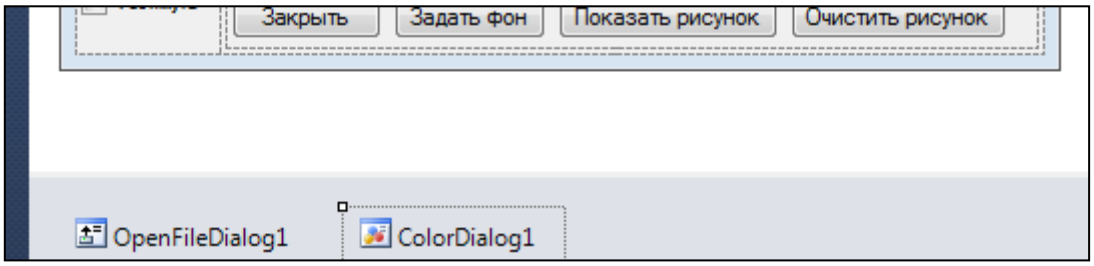
Комментарии к коду можно и не писать.

5. Пропишем процедуру для элемента **Clear_B**:

```
Private Sub Clear_B_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Clear_B.Click
    ' Убирает рисунок из рамки.
    PictureBox1.Image = Nothing
End Sub
```

6. Чтобы прописать два последних метода (**Show_B_Click()** и **Backgr_B_Click()**), требуется добавить в форму два **Диалоговых окна** – для открытия рисунка и для выбора цвета фона.

- Перейдите в **Конструктор** и в **Панели элементов** найдите группу **Диалоговые окна**;
- Дважды щёлкните компонент **OpenFileDialog** (этот компонент запускает стандартное диалоговое окно для выбора нужного файла) и **ColorDialog** (запускается стандартное диалоговое окно выбора цвета);
- Внизу под формой появится серое поле с двумя компонентами:



- Щёлкните по значку и установите значение свойства **Filter**: JPEG Files (*.jpg)|*.jpg|PNG Files (*.png)|*.png|BMP Files (*.bmp)|*.bmp|All files (*.*)|*.* - это даст возможность выбирать файлы только заданных типов, и значение свойства **Title** "**Выберите файл:**".
- Вернитесь в **Конструктор** и дважды кликните на кнопку **Показать рисунок**. Откроется конструктор кода непосредственно для событийной процедуры **Show_B_Click**. Введите следующий код:

```
Private Sub Show_B_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Show_B.Click
    If OpenFileDialog1.ShowDialog() = DialogResult.OK Then
        PictureBox1.Load(OpenFileDialog1.FileName)
    End If
End Sub
```

То же самое, но с комментариями:

```
Private Sub Show_B_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Show_B.Click
    'При нажатии открывается Диалог открытия файла.
    'Пользователь после выбора файла нажимает OK.
    'В окно загружается выбранный файл.
    If OpenFileDialog1.ShowDialog() = DialogResult.OK Then
        PictureBox1.Load(OpenFileDialog1.FileName)
    End If
End Sub
```

Данная процедура работает так: при нажатии на кнопку **Показать рисунок** запускается диалоговое окно **OpenFileDialog1** (вы увидите это окно с возможностью поиска нужного рисунка). После того, как нужный файл будет найден (нажата кнопка **ОК** или **Enter**), выполнится оператор **Load()** для элемента **PictureBox1**: **PictureBox1.Load** (открыть выбранный рисунок в окне **PictureBox1**).

- Запустите **Отладку**, посмотрите, как работает кнопка **Показать рисунок**.
- Установите курсор в оставшуюся процедуру для метода **Backgr_B_Click** (Задать фон) и введите код:

```
Private Sub Backgr_B_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Backgr_B.Click
    If ColorDialog1.ShowDialog() = DialogResult.OK Then
        PictureBox1.BackColor = ColorDialog1.Color
    End If
End Sub
```

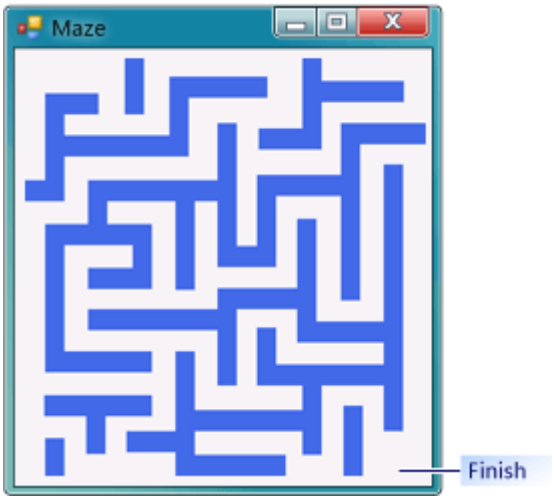
То же самое, с комментариями:

```
Private Sub Backgr_B_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Backgr_B.Click
    'Вызывает диалоговое окно Изменения цвета фона окна.
    'Пользователь выбирает цвет, нажимает OK, цвет фона окна меняется на выбранный.
    If ColorDialog1.ShowDialog() = DialogResult.OK Then
        PictureBox1.BackColor = ColorDialog1.Color
    End If
End Sub
```

- Запустите **Отладку**, посмотрите, как работает кнопка **Задать фон**.
- В режиме **Отладки** откройте какой-нибудь рисунок, увеличьте рамку, попробуйте режим **Растянуть**.
- Не забудьте **Сохранить всё**.
- Далее: **Проект – Опубликовать Просмотр изображений** – (укажите место, в котором предполагаете опубликовать проект или согласитесь с предложенным) **– Готово**. Откроется папка, в которую выполнена публикация вашего проекта. Файл **setup.exe** можно будет потом установить на ваш компьютер и тогда созданное вами Приложение может быть использовано автономно.******(проверить).

Проект 2. Создание игры «Лабиринт»

Содержание проекта. Игра заключается в том, что пользователь должен переместить указатель мыши от старта к финишу и не коснуться при этом стен лабиринта. Указатель мыши появляется в левом верхнем углу лабиринта. Пользователь проходит по лабиринту, аккуратно, стараясь не задеть стены указателем мыши. Если указатель касается стены, он автоматически возвращается в исходную позицию. Если указатель достигает метки **Финиш** в конце лабиринта, отображается окно сообщений с поздравлением и игра заканчивается.



В результате ваша программа будет выглядеть так, как показано на следующем рисунке.

Этапы проекта:

1) Создание макета формы с помощью контейнера **Panel**.

2) Создание лабиринта с помощью элементов управления **Label**.

3) Создание кода для отображения окна сообщений и настройка обработчика событий для поведения мыши.

4) Добавление обработчика события для перезапуска игры.

5*) Воспроизведение в программе звука.

1 этап. Создание макета формы с помощью контейнера Panel.

1. Создайте новый проект **Приложение Windows Forms,** затем установите **Имя файла Лабиринт.vb**.

2. **Сохраните всё** (не забывайте это делать периодически в процессе работы над проектом). Задайте имя приложения, например, **Игра Лабиринт**.

3. Задайте нужные вам размеры формы (значение свойства **Size**, например – 650; 650).

4. Измените значение свойства **Text** (например, Лабиринт) и установите значение свойства **StartPosition** – **SenterScreen**.

5. Установите для свойства **FormBorderStyle** значение **Fixed3D**. Это исключит возможность изменения размеров окна обычным образом с помощью курсора мыши.

6. Отключите кнопку **Развернуть** в заголовке окна. Для этого установите для свойства **MaximizeBox** значение **False**.

7. Запустите **Отладку** (▶) и посмотрите, как работает созданная форма. Убедитесь, что вы не можете ее развернуть или изменить размер. Единственное, что оставлено пользователю – возможность свернуть окно приложения и поместить его на Панель задач Windows. Таким образом, получившееся приложение будет тех размеров, которые предусмотрены автором.

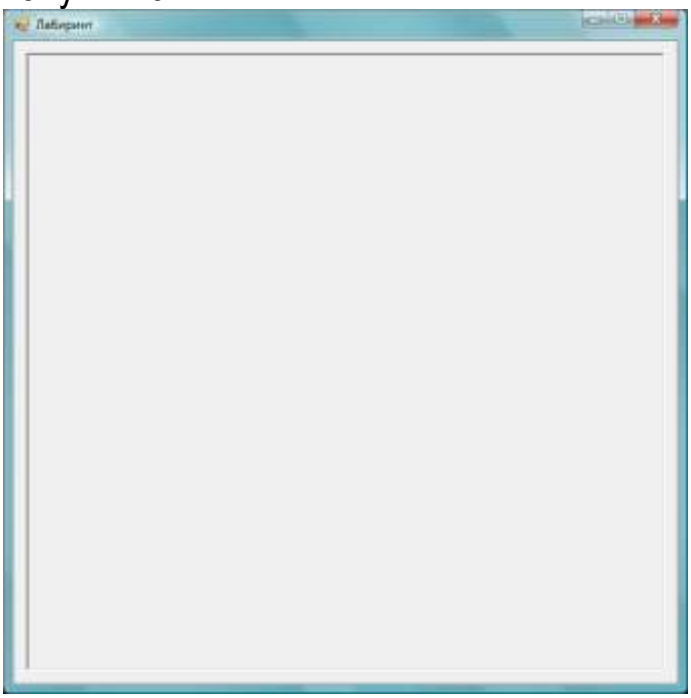
8. На **Панели элементов** откройте группу **Контейнеры** и дважды щёлкните на элемент **Panel** (он также позволяет размещать группы элементов управления. Но в

отличие от известных нам контейнеров (**TableLayoutPanel** и **FlowLayoutPanel**) **Panel** не выполняет переупорядочивание элементов управления, которые содержит. Это дает возможность располагать элементы управления в нужных местах. **Panel** не рекомендуется использовать, если оставлена возможность изменять размеры окна).

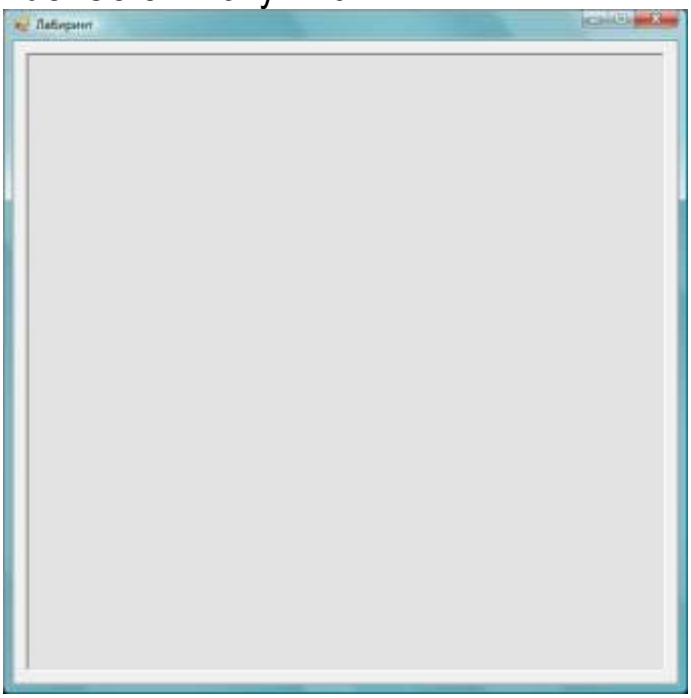
9. При выделенной **Panel** в левом верхнем углу размещается **Маркер перемещения** (). Передвиньте **Panel** за этот маркер немного вниз, а затем перетаскивайте нижний правый маркер переноса, пока панель не разместится в правой и нижней области.

10. Чтобы была видна граница лабиринта, выделите **Panel** и установите для свойства **BorderStyle** значение **Fixed3D**.

11. Запустите отладку. Должно получиться:



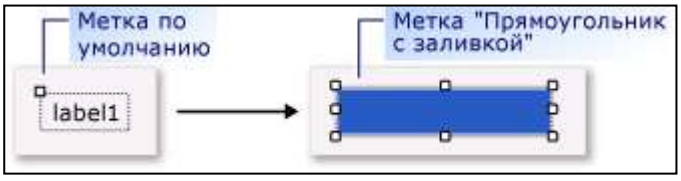
12. Можно изменить значение свойства **BackColor**. Получится:



2 этап. Создание лабиринта с помощью элементов управления Label.

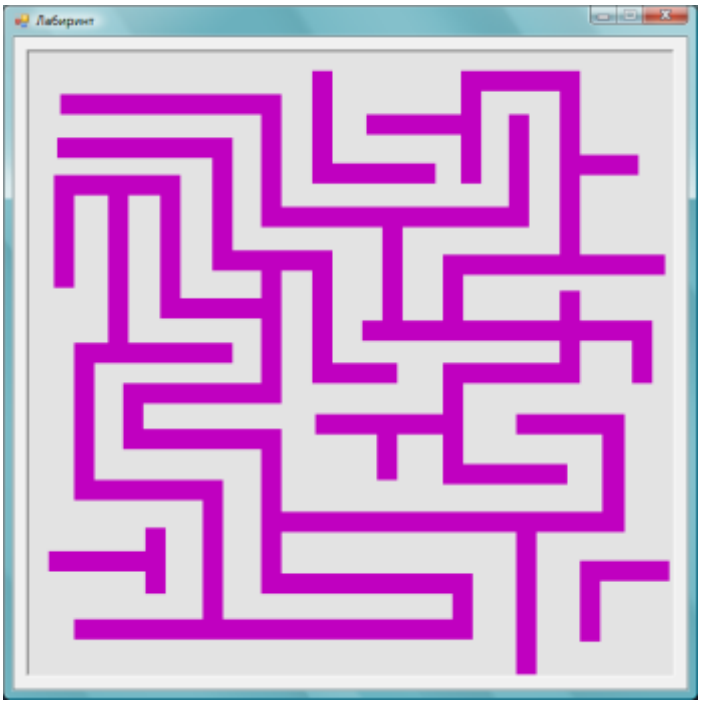
1. Перейдите в **Конструктор**, откройте **Панель элементов** и в группе **Стандартные элементы управления** дважды щёлкните по элементу управления **Label**.

2. Выделите его и задайте свойству **AutoSize** значение **False** (тогда размеры этого элемента можно будет изменить), свойству **BackColor** необходимый вам цвет (контрастный по отношению к цвету формы) и измените значение свойства **Text** на пустое (на элементе не будет надписи).



3. Скопируйте этот элемент (Ctrl+C) и вставьте (Ctrl+V). Измените размеры элемента так, чтобы он стал вертикальным. Далее, копируйте горизонтальные и вертикальные элементы, располагая их в необходимом порядке, постройте лабиринт.

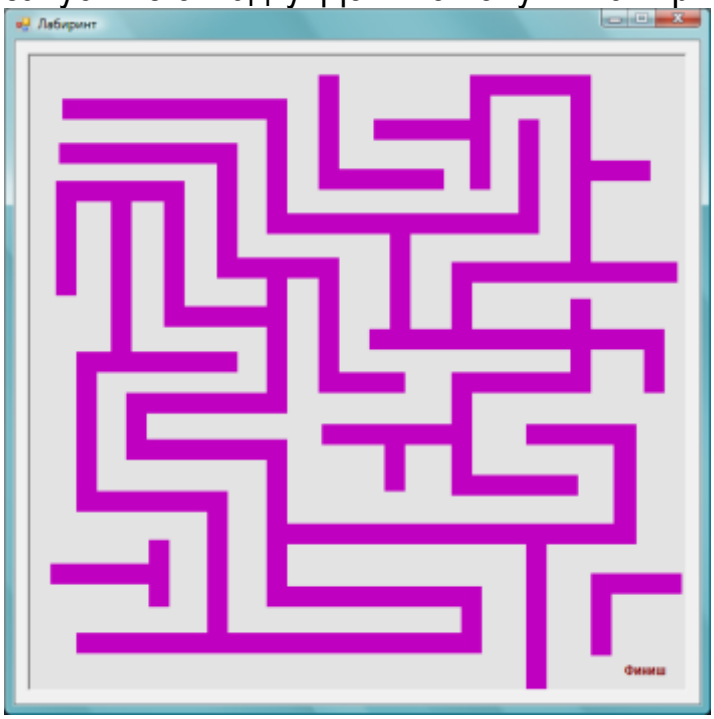
Должно получиться примерно следующее (точное расположение элементов повторять не нужно):



Обратите внимание: в верхнем левом углу оставлено больше места – там будет располагаться курсор в начале игры.

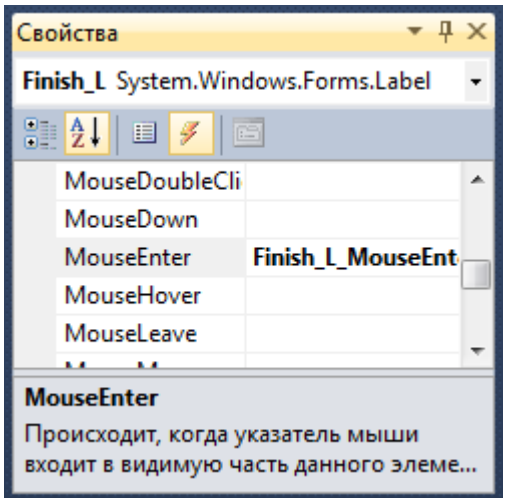
4. Ещё раз вызовите **Панель элементов** и в группе **Стандартные элементы управления** дважды щёлкните элемент управления **Label**. Установите значение свойства **(Name)** – **Finish_L** (эта метка будет обозначать **Финиш**, и для неё нужно будет писать код), а свойству **Text** задайте значение **Финиш**. Расположите этот элемент в конце лабиринта. Можно изменить шрифт и цвет шрифта (свойства **Font** и **FontColor**)

5. Сохраните всё и запустите отладку. Должно получиться примерно так:



3 этап. Создание кода для отображения окна сообщений. Настройка обработчика событий для поведения мыши.

1. Выделите элемент управления **Finish_L** и в верхней части окна Свойства щёлкните на значок **Событие** (⚡). Тогда в окне Свойства будут отображаться события для данного элемента управления (по умолчанию предлагается событие **Click()**, с которым мы уже знакомы).



2. Выберите событие **MouseEnter** и дважды щёлкните не него. Откроется событийная процедура для метода **Finish_L_MouseEnter()**. Этот метод будет работать тогда, когда указатель мыши войдёт в метку:

```
Public Class Лабиринт

    Private Sub Finish_L_MouseEnter(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Finish_L.MouseEr

    End Sub
End Class
```

3. Т.к. при попадании курсора мыши в метку должно появляться сообщение «**Поздравляем! Вы выиграли!**», а после этого окно приложения должно закрыться, добавим в эту процедуру код, который будет содержать оператор **MessageBox.Show()** (Показать сообщение) и оператор **Close()** (закрывает окно приложения).

```
    Private Sub Finish_L_MouseEnter(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Finish_L.Mo
        'Показывается Сообщение (MessageBox), после чего форма закрывается
        MessageBox.Show("Поздравляем! Вы выиграли!")
        Close()
    End Sub
```

4. Сохраните всё и запустите отладку. Установите курсор мыши на метку **Финиш** и посмотрите, как работает программа.

Замечание. В окне сообщения **MessageBox** в зависимости от того, какое приложение вы создаёте, могут содержаться самые различные сообщения, например, инструкции пользователю или какие-либо сведения.

4 этап. Добавление обработчика события для перезапуска игры

1. Откройте код формы, установите курсор после метки End Sub и нажмите Enter.

2. Самостоятельно наберите новую событийную процедуру:

```
    End Sub
    Private Sub MoveToStart()
        Dim startingPoint = Panel1.Location
        startingPoint.Offset(10, 10)
        Cursor.Position = PointToScreen(startingPoint)
    End Sub
```

Этот метод написан специально, чтобы к нему можно было обращаться каждый раз, когда требуется перезапустить программу. Иначе, придётся каждый раз прописывать код для перемещения курсора мыши при перезапуске приложения.

3*(этот шаг можно не выполнять). Создадим специальный XML-комментарий, который будет отображаться в **Подсказке** и в системе **IntelliSense**. Этот комментарий добавляется перед вновь созданным методом следующим образом: перед **Private Sub** вставляется строка и набираются три апострофа («’’’»). Тогда автоматически вставится следующий текст:

```
''' <summary>
'''
''' </summary>
''' <remarks></remarks>
Private Sub MoveToStart()
```

После этого между тегами <summary></summary> можно добавить примерно такой комментарий:

```
''' <summary>
''' Перемещает курсор мыши в левый верхний угол Лабиринта
''' на расстоянии 10 пикселов сверху и слева.
''' </summary>
''' <remarks></remarks>
Private Sub MoveToStart()
    Dim startingPoint = Panel1.Location
```

4. Теперь новый метод необходимо добавить в код **Формы**. Для этого **перед** процедурой для метода **Finish_L_MouseEnter()** наберите такую процедуру:

```
Public Class Лабиринт
    Public Sub New()
        ' Это требуется для проектировщика форм.
        InitializeComponent()
        ' Можете инициализировать любой метод после вызова метода InitializeComponent ().
    End Sub
```

Метод **New()** – это специальный метод, который называется **конструктором**. Этот метод выполняется один раз при создании процедуры. Сейчас он только вызывает метод с именем **InitializeComponent()** (вызов, инициализация методов).

5. Теперь добавьте в эту процедуру вызов метода **MoveToStart():**

```
Public Class Лабиринт
    Public Sub New()
        ' Это требуется для проектировщика форм.
        InitializeComponent()
        ' Можете инициализировать любой метод после вызова метода InitializeComponent ().
        MoveToStart()
    End Sub

    Private Sub Finish_L_MouseEnter(ByVal sender As System.Object, ByVal e As System.EventArgs)
        'Показывается Сообщение (MessageBox), после чего форма закрывается
        MessageBox.Show("Поздравляем! Вы выиграли!")
        Close()
```

Обратите внимание, при наведении курсора на слово **MoveToStart()** появляется подсказка, что этот метод выполняет (если вы выполняли Шаг 3).

6. Сохраните всё и запустите Отладку приложения. Обратите внимание, где появляется курсор, попробуйте «походить» по Лабиринту.

7. Теперь усложним игру, сделаем так, чтобы каждый раз, когда курсор касается стены, происходил перезапуск приложения, и курсор возвращался бы в исходное положение. Для этого перейдите в **Конструктор** и выделите любую из «стен» Лабиринта.

8. В окне **Свойства** щёлкните по значку **Событие** (⚡) и найдите событие **MouseEnter**. Введите значение события **wall_MouseEnter** и нажмите **Enter** (если вы выбрали «стену» с именем Label1, то процедура будет выглядеть так, хотя это может быть и другой элемент, например Label12 или Label61):

```
    End Sub

    Private Sub wall_MouseEnter() Handles Label1.MouseEnter

    End Sub
```

9. Теперь добавим в этот метод метод **MoveToStart()**:

```
    End Sub

    Private Sub wall_MouseEnter() Handles Label1.MouseEnter
        'Если курсор касается "стены", то программа вызывает метод MoveToStart().
        MoveToStart()
    End Sub
```

10. **Сохраните всё** и запустите **Отладку**. Попробуйте коснуться курсором той «стены» для которой прописан метод **wall_MouseEnter**.

11. Перейдите в **Конструктор** и удерживая клавишу **Ctrl** щёлкните на те элементы, которые должны быть выделены (стены и **Panel1**).

12. Не снимая выделение в окне **Свойства** нажмите значок **События** (⚡). Найдите событие **MouseEnter** и выберите для него значение **wall_MouseEnter** (только что написанный обработчик события).

13. Сохраните всё и запустите Отладку приложения. Попробуйте провести курсором между стенами, посмотрите, что будет происходить, если курсор случайно коснётся «стены» Лабиринта.

5 этап*(этот этап можно не выполнять). Воспроизведение звука в программе Добавление объекта SoundPlayer.

1. Добавьте следующий код перед методом **New()**:

```
Public Class Лабиринт
    ' Этот звук запускается компонентом SoundPlaer, если курсор коснётся стены.
    Dim startSoundPlayer = New System.Media.SoundPlayer("C:\Windows\Media\Аккорд.wav")
    ' Этот звук запускается компонентом SoundPlaer, если игра завершена.
    Dim finishSoundPlayer = New System.Media.SoundPlayer("C:\Windows\Media\tada.wav")

    Public Sub New()
```

2. Теперь добавим оператор **Play()**, в метод **MoveToStart()** и в метод **Finish_L_MouseEnter**:

```
Private Sub MoveToStart()
    startSoundPlayer.Play()
    Dim startingPoint = Panel1.Location
    startingPoint.Offset(10, 10)
    Cursor.Position = PointToScreen(startingPoint)
End Sub
```

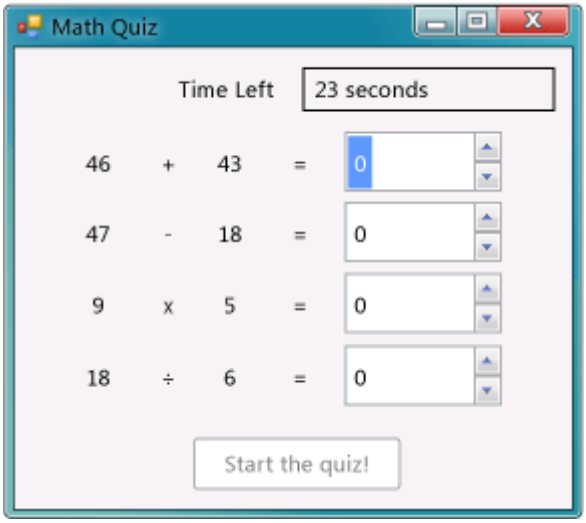
```
Private Sub Finish_L_MouseEnter(ByVal sender As System
    'Показывается Сообщение (MessageBox), после чего ф
    finishSoundPlayer.Play()
    MessageBox.Show("Поздравляем! Вы выиграли!")
    Close()
End Sub
```

3. Запустите Отладку Приложения и посмотрите, как это работает.

Выполните публикацию приложения (см. предыдущий проект).

Проект 3. Математическая викторина*

Содержание проекта. Игра заключается в том, что пользователь должен за ограниченное время должен выполнить четыре арифметические действия со случайными числами.



В процессе выполнения проекта мы познакомимся с:

- Работой генератора случайных чисел **Random**.
- Вызовом события с помощью элемента управления **Timer**.
- Управлением работой программы с помощью операторов **if else**.
- Настройкой основных арифметических операций.

1 этап. Создание проекта и добавление в форму элементов управления Label.

1. Файл – Создать проект – **Приложение Windows Forms.**
2. Введите имя файла – Математическая головоломка.vb.
3. Выделите форму. Задайте для свойства **Text** значение **Математическая головоломка**, значение свойства **Size** на 550; 550 (можно этот размер установить просто перетаскивая границы формы), значение свойства **StartPosition** – **SenterScreen**.
4. Запретите пользователю изменять размеры формы: измените значение свойства **FormBorderStyle** на **Fixed3D**, а значение свойства **MaximizeBox** на **False**.
5. **Сохраните всё (** **)** в свою папку, задайте **Имя** – **Математическая головоломка**, установите флажок на **Создать каталог для решения.**
6. На **Панели элементов** из группы **Стандартные элементы управления** установите элемент управления **Label** (Метка). Установите значение свойств:
   - **(Name) – Time_L**(это будет окно обратного отсчёта секунд);
   - **AutoSize – False** (можно будет установить размеры этого поля);
   - **BorderStyle** – **FixedSingle** (видна линия вокруг поля);
   - **Size** – 200; 30;
   - **Text** – установите пустое значение;
   - **Font** – размер 16 (необходимо кликнуть на значение, откроется окно выбора шрифта).
   - Перетащите **Label** в правый верхний угол формы (пока не появятся синие линии-разделители).
7. Введите ещё один элемент управления **Label** и установите значение свойств:
   - Размер шрифта – 16 (можно поменять цвет текста – **ForeColor**);
   - **Text – Оставшееся время:**;

- Перетащите и выровняйте метку рядом с меткой **Time_L**.

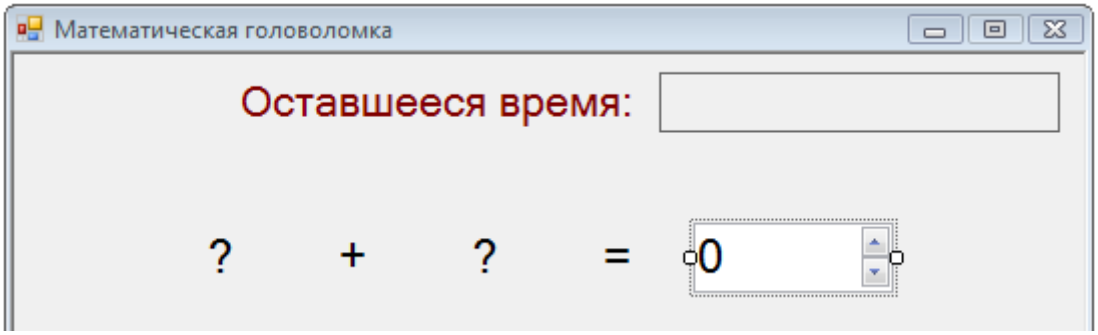
8. Добавим элементы управления для задачи сложения. Добавьте новый элемент управления **Label** и установите значение свойств:

- **Text** - **?** (вопросительный знак);
- **AutoSize** – **False**;
- **Size** – **60, 50**;
- Размер шрифта – **18**;
- **TextAlign** – **MiddleCenter**;
- **Location** – **75; 75** (определено место элемент на форме);
- **(Name)** – **PlusL_L** («метка плюс левая»);
- Скопируйте элемент и вставьте его три раза, затем выровняйте их правее первой метки;
- Для второй метки измените значение свойства **Text** на значение **+** (знак плюс);
- Для третьей метки измените значение свойства (**Name)** на значение **PlusR_L** («метка плюс правая»);
- Для четвёртой метки установите значение свойства **Text** на значение **=** (знак равенства).

9. На Панели элементов найдите элемент управления **NumericUpDown** из группы Стандартные элементы управления и измените значения свойств:

- Размер шрифта – **18**;
- Установите ширину – **100**;
- Выровняйте его по элементами управления **Label** для задачи сложения;
- **(Name)** – **sum**.

10. Должно получиться следующее:



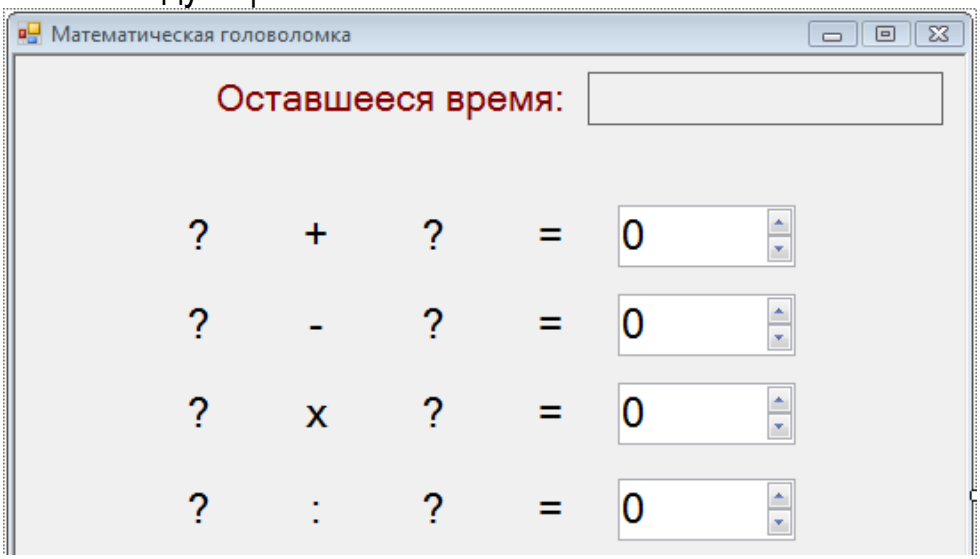
11. Выделите все пять элементов управления задачи на сложение, скопируйте их, затем вставьте и выровняйте по элементам управления задачи на сложение. Измените значения следующих свойств новых элементов:

- У первой метки: **(Name) – MinusL_L**.
- У второй метки: **Text** на значение **–** (знак минус).
- У третьей метки: **(Name) – MinusR_L**.
- У элемента **NumericUpDown**: **(Name) – Dif**.

12. Вставьте пять элементов управления еще два раза и выполните следующие действия:

- В третьей строке назовите первую метку как **UmnL_L**, измените у второй метки значение свойства **Text** на **×** (знак умножения), назовите третью метку как **UmnR_L**, назовите элемент управления **NumericUpDown** как **Prod**;
- В четвертой строке назовите первую метку как **DivL_L**, измените у второй метки значение свойства **Text** на **÷** (знак деления), назовите третью метку как **DivR_L**, назовите элемент управления **NumericUpDown** как **Quot**.

13. Получится следующее:



14. Чтобы закончить создание исходной формы **Приложения**, добавим из **Стандартных элементов управления** элемент **Button.** Эта кнопка будет запускать головоломку. Измените значение следующих свойств:

- **(Name) – Start_B;**
- **Text – Начать опрос**;
- Размер шрифта – **14**;
- **AutoSize – True** (размер кнопки автоматически изменится при изменении размера текста на кнопке);
- **TabIndex**[5] **– 1**;
- Разместите кнопку посередине и внизу формы**.**

15. Выделите элемент управления **суммы NumericUpDown** и установите для свойства **TabIndex** значение **2**.

16. Установите для элемента управления **разностью NumericUpDown** значение свойства **TabIndex – 3**.

17. Установите для элемента управления **произведением NumericUpDown** значение свойства **TabIndex – 4**.

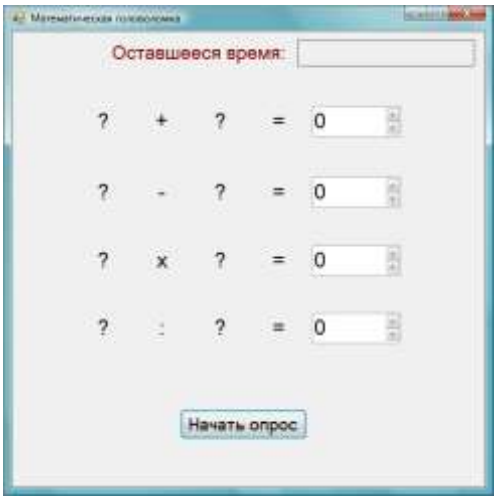
---

[5] Назначением свойства **TabIndex** является установка порядка перемещения курсора между элементами при нажатии пользователем клавиши TAB.

18. Установите для элемента управления **частным NumericUpDown** установите значение свойства **TabIndex – 5**.

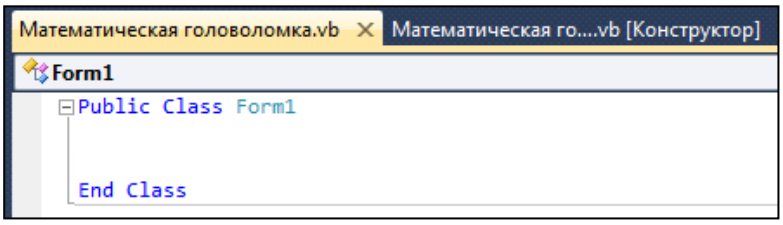
19. В итоге должно получиться примерно такая исходная форма Приложения.

20. Сохраните всё и запустите Отладку приложения. Нажмите несколько раз клавишу Tab и посмотрите, как передвигается курсор.



2 этап. Создание задачи на сложение случайных чисел.

1. Щёлкнем два раза на форме. Откроется вкладка Конструктора кода для экземпляра класса Form1:



2. Создадим **объект Random** с помощью оператора **New**. Имя этого объекта задано как **randomizer** (хотя вы можете задать любое другое имя).

```
Public Class Form1
    ' Создание объека Random для генерации случайных чисел.
    Dim randomizer As New Random

End Class
```

3. Программе нужно запомнить числа, которые она выбрала для задачи. Добавьте в форму целые числа (Integer[6] в Visual Basic) с именем **add1** и целое число с именем **add2** (первое и второе слагаемые):

```
' Создание объекта Random для генерации случайных чисел.
Dim randomizer As New Random
' Эти целые числа будут использованы в задаче на сложение.
Dim add1 As Integer
Dim add2 As Integer
```

4. Добавим метод с именем **StartTheQuiz()**, который использует у объекта **Random** метод **Next()** для выбора двух чисел и размещения их в метках. Он заполнит все задачи и затем запустит таймер. Обратите внимание, если вы зададите **Next(50)**, тогда целое число будет выбрано из промежутка от 0 до 49 (всего 50 чисел). Поэтому код должен выглядеть следующим образом:

```
Dim add2 As Integer
''' <summary>
''' Запомнить числа для задачи.
''' Запуск таймера.
''' </summary>
''' <remarks></remarks>
Public Sub StartTheQuiz()
    ' Выбирается произвольное целое число от 0 до 50
    add1 = randomizer.Next(51)
    add2 = randomizer.Next(51)

    PlusL_L.Text = add1.ToString
    PlusR_L.Text = add2.ToString

    sum.Value = 0
End Sub
```

[6] Целое число Integer используется для хранения целого положительного или отрицательного значения числа. Оно может содержать любое целое число из диапазона от -2 147 483 648 до 2 147 483 647.

Обратите внимание на операторы **PlusL_L.Text = add1.ToString** и **PlusR_L.Text = add2.ToString**. Они устанавливают значение свойств **Текст** у двух меток сложения — **PlusL_L** и **PlusR_L**, поэтому эти метки отображают два случайных числа. Метод **ToString()** используется для преобразования целых чисел в текст (в программировании ***string*** (строка) означает **текст**), так как элементы управления **Label** могут отображать только текст, а не числа.

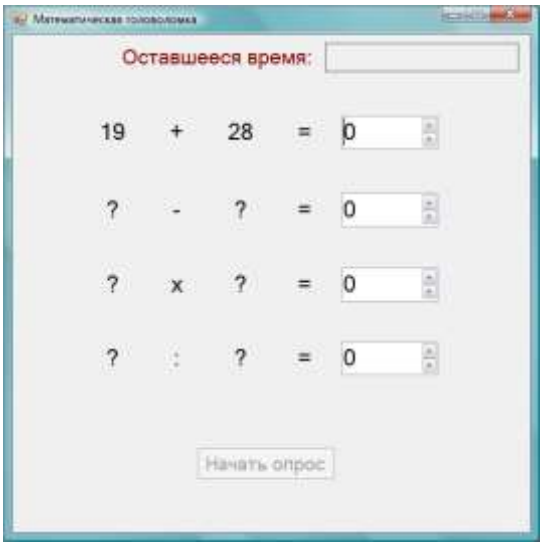
5. Теперь напишем обработчик события для кнопки **Начать опрос** (**Start_B**). Перейдём в **Конструктор** и дважды кликнем на кнопку **Начать опрос**, а затем введём следующий код:

```
        sum.Value = 0
    End Sub

    Private Sub Start_B_Click() Handles Start_B.Click
        Start_B.Enabled = False
        StartTheQuiz()
    End Sub
```

Первый оператор устанавливает у элемента управления **Start_B** значение свойства **Enabled** равное **False** (кнопка отключается, пользователь может нажать Начать опрос только один раз, после этого кнопка отображается затемнённой и недоступной до завершения головоломки или закрытия программы). Второй оператор вызывает новый метод **StartTheQuiz()**.

6. **Сохраните всё** и запустите программу. Нажмите на кнопку **Начать опрос** (должна появиться задача сложения случайных чисел).

Закройте программу и запустите её заново. Опять нажмите кнопку **Начать опрос**. Должна появиться другая пара чисел.



7. Теперь создадим метод для проверки правильности введённого значения суммы чисел. Назовём его, например, **CheckTheAnswer()** (проверка ответа). Он выполнит сложение переменной **add1** и переменной **add2** и выполнит проверку, равна ли сумма значению **sum** элемента управления **NumericUpDown**. Если значение суммы равно значению **sum** этот метод возвратит значение **true** иначе — **false**.

```
    End Sub
    ''' <summary>
    ''' Проверяет правильность введённого ответа
    ''' </summary>
    ''' <returns>Если ответ верен - True, иначе - false.</returns>
    ''' <remarks></remarks>
    Public Function CheckTheAnswer() As Boolean

        If add1 + add2 = Sum.Value Then
            Return True
        Else
            Return False
        End If

    End Function
```

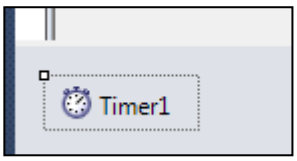
Обратите внимание: вместо ключевого слова **Sub** использовано ключевое слово **Function**, т.к. в данном случае происходит «возвращение значения» (складываются числа и получается некоторый результат). А процедуры (**Sub**) значение не возвращают, в них происходят некоторые события.

3 этап. Добавление таймера с обратным отсчетом.

1. Добавьте целое число **Integer** с именем **timeLeft**:

```
Dim add2 As Integer
' Это число Integer послужит для проверки обратного отсчёта таймера.
Dim timeLeft As Integer
```

2. Перейдите в **Конструктор** и на **Панели элементов** в группе **Компоненты** найдите элемент **Timer**. Кликните на него два раза. Внизу конструктора появится серое поле с добавленным компонентом **Timer1**.



3. Щёлкните по только что добавленному значку **Timer1** и установите для свойства **Interval** значение равное **1000**. Это вызывает событие **Tick** каждую секунду.

4. Добавим обработчик событий **Tick**. Для этого дважды щёлкните по значку. В редакторе кода добавьте следующие операторы:

```
    End Function

    Private Sub Timer1_Tick(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Timer1.Tick
        If (timeLeft > 0) Then
            ' Начнётся новый отсчёт
            ' Обновится надпись в окне таймера.
            timeLeft = timeLeft - 1
            Time_L.Text = timeLeft & " сек"
        Else
            ' Если пользователь превысит лимит, остановить таймер и показать
            ' сообщение о превышении времени.
            Timer1.Stop()
            Time_L.Text = "Время вышло!"
            MessageBox.Show("Вы не уложились в допустимое время", "Очень жаль!")
            Sum.Value = add1 + add2
            Start_B.Enabled = True
        End If

    End Sub
End Class
```

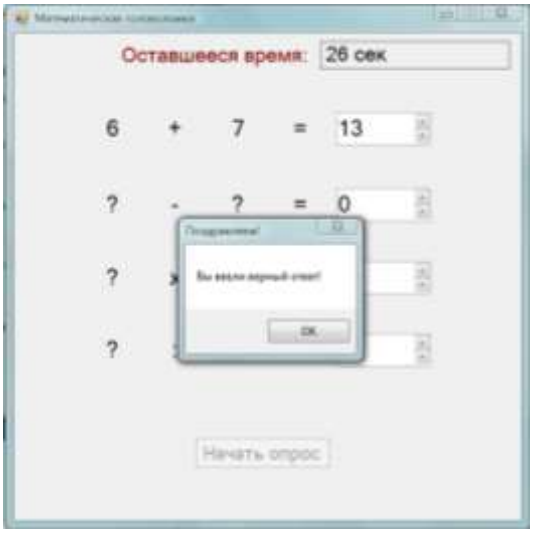
5. Добавим ещё три строки в конец метода **StartTheQuiz()**. Теперь при запуске головоломки он устанавливает целому числу Integer с именем **timeLeft** значение 30, и изменяет у элемента управления **Time_L** свойство **Text** на значение равное 30 сек. Затем, чтобы начать обратный отсчет, он вызывает у элемента управления **Timer** метод **Start()**.

```
    PlusL_L.Text = add1.ToString
    PlusR_L.Text = add2.ToString

    sum.Value = 0

    ' Запуск таймера.
    timeLeft = 30
    Time_L.Text = "30 сек"
    Timer1.Start()

End Sub
```

6. Изменим обработчик событий **Tick** для проверки ответа. Новый обработчик с проверкой ответа должен содержать следующее:

```
    End Function

    Private Sub Timer1_Tick(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Timer1.Tick
        If CheckTheAnswer() Then
            ' If the user got the answer right, stop the timer
            ' and show a MessageBox.
            Timer1.Stop()
            MessageBox.Show("Вы ввели верный ответ!", "Поздравляем!")
            Start_B.Enabled = True
        ElseIf (timeLeft > 0) Then
            ' Начнётся новый отсчёт
            ' Обновится надпись в окне таймера.
            timeLeft = timeLeft - 1
            Time_L.Text = timeLeft & " сек"
        Else
            ' Если пользователь превысит лимит, остановить таймер и показать
            ' сообщение о превышении времени.
            Timer1.Stop()
            Time_L.Text = "Время вышло!"
            MessageBox.Show("Вы не уложились в допустимое время", "Очень жаль!")
            Sum.Value = add1 + add2
            Start_B.Enabled = True
        End If
```

7. **Сохраните всё** и запустите **Отладку**. Нажмите кнопку **Начать опрос** и введите верный ответ. Должно получиться примерно так:



Обратите внимание: если вы вводите верный ответ, таймер останавливается и выводится сообщение с поздравлением. После нажатия на **ОК** кнопка **Начать опрос** опять становится активной (за это отвечает оператор **Start_B.Enabled = True**) Если же вы вводите неверный ответ, программа пока никак не реагирует на это.

4 этап. Добавление обработчиков событий входа для элементов управления NumericUpDown.

Обратите внимание: когда вы начинаете вводить ответ, 0 остаётся, а если вы ввели число больше 100, то введённое число тут же меняется на 100 (это максимальная возможная сумма). Это не очень удобно, поэтому требуется добавит обработчик события (назовём его, например, **Ans_Enter**), который можно будет потом использовать для **всех** элементов **NumericUpDown**.

1. В Конструкторе выделите элемент **NumericUpDown** для задачи нахождения суммы. В окне **Свойства** нажмите на значок ⚡ (**События**).

2. Выберите событие **Enter** и наберите имя **Ans_Enter,** затем нажмите ввод и откроется вкладка редактирования кода. Наберите его:

```
Private Sub Ans_Enter(ByVal sender As System.Object, ByVal e As System.EventArgs)
    ' Выбирается для контроля ввода в элемент NumericUpDown.
    Dim answerBox As NumericUpDown = TryCast(sender, NumericUpDown)

    If answerBox IsNot Nothing Then
        Dim lengthOfAnswer As Integer = answerBox.Value.ToString().Length
        answerBox.Select(0, lengthOfAnswer)
    End If

End Sub
```

Обратите внимание на верхнюю часть метода — sender As System.Object (внутри этого метода, когда используется **sender**, он будет указывать на элемент **NumericUpDown**). Поэтому в первой строке метода указывается, что это не просто объект, а именно элемент управления **NumericUpDown**. В следующей строке кода выполняется проверка, что **answerBox** был преобразован из объекта в элемент управления **NumericUpDown**. Если операция не выполнена, то он будет иметь значение **Nothing**. В третьей строке кода выполняется определение длины ответа (**Length**), который в настоящий момент отображается в элементе управления **NumericUpDown**. В четвертой строке указывается элементу управления **NumericUpDown** выбрать ответ. Теперь, когда пользователь переходит к элементу управления, он вызывает это событие, которое приводит к выбору ответа. Как только пользователь начинает ввод данных, предыдущий ответ стирается и заменяется на новый ответ.

3. **Сохраните всё** и запустите **Отладку**. Обратите внимание, что исходное значение 0 теперь выделено, а когда вы начинаете вводить ответ, 0 заменяется на вводимое значение.

4. Перейдите в **Конструктор** и выделите другой элемент управления **NumericUpDown**. Перейдите на страницу **События** в диалоговом окне **Свойства**, прокрутите содержимое окна до события **Enter** и выберите обработчик событий, который был только что добавлен (**Ans_Enter**).

5. Проделайте такое же действие для оставшихся элементов управления **NumericUpDown**, в которых отображаются значения переменных **Prod** (произведения) и **Quot** (частного).

6. **Сохраните всё** и запустите **Отладку**. Убедитесь, что теперь все окна ввода работают одинаково правильно.

5 этап. Добавление задачи на вычитание.

1. Добавим два целых числа (**Integer**) с именами **vychit** и **umensh** для задачи на вычитание между целыми числами для сложения и целым числом для таймера:

```
    Dim add1 As Integer
    Dim add2 As Integer
    ' Эти целые числа будут использованы
    ' в задаче на вычитание.
    Dim vychit As Integer
    Dim umensh As Integer
    ' Это число Integer послужит для проверки обратного отсчёта таймера.
```

2. Изменим метод **StartTheQuiz()**, добавив в него код для задачи на вычитание случайных чисел. Новый код расположим между заполнением задачи на сложение и запуском таймера. В данном случае обратите внимание, что метод Next() задан не так, как для задачи на сложение – для **umensh** он выберет числа от 1 до 100 (без 0), а **vychit** обязательно задаст меньше или равным значению **umensh**.

```
        PlusL_L.Text = add1.ToString
        PlusR_L.Text = add2.ToString
        Sum.Value = 0

        ' Заполнение задачи на вычитание.
        umensh = randomizer.Next(1, 101)
        vychit = randomizer.Next(1, umensh)
        MinusL_L.Text = umensh.ToString
        MinusR_L.Text = vychit.ToString
        Dif.Value = 0

        ' Запуск таймера.
        timeLeft = 30
```

3. Теперь изменим метод **CheckTheAnswer()** для проверки правильности ответа задачи на вычитание. Код должен выглядеть следующим образом:

```
    Public Function CheckTheAnswer() As Boolean

        If ((add1 + add2 = Sum.Value) AndAlso (umensh - vychit = Dif.Value)) Then
            Return True
        Else
            Return False
```

Обратите внимание: оператор **AndAlso** в языке Visual Basic — это **логическое И**; это можно перевести примерно так: «если к значению переменной add1 прибавить значение переменной add2, то получим значение переменной Sum, **И** если из значения переменной umensh вычесть значение переменной vychit, то получим значение переменной Dif, тогда…»

4. Осталось изменить последнюю часть обработчика событий таймера **Tick**, чтобы он заполнял верный ответ, если пользователь не уложится в отведённое время. Код должен выглядеть следующим образом:

```
        Timer1.Stop()
        Time_L.Text = "Время вышло!"
        MessageBox.Show("Вы не уложились в допустимо
        Sum.Value = add1 + add2
        Dif.Value = umensh - vychit
        Start_B.Enabled = True
    End If
```

5. **Сохраните всё** и запустите **Отладку**. Проверьте, что будет в случае введения правильных ответов и что будет, если не уложиться в заданное время.

6 этап. Добавление задач на умножение и деление.

1. Добавьте в форму ещё четыре целых (**Integer**) числа – два для умножения, два для деления:

```
    Dim umensh As Integer
    ' Эти целые числа будут использованы в задаче на умножение.
    Dim mult1 As Integer
    Dim mult2 As Integer
    ' Эти целые числа будут использованы в задаче на деление.
    Dim delim As Integer
    Dim delit As Integer
```

2. Теперь изменим метод **StartTheQuiz()**. Добавим код для задач на умножение и деление.

```
Dif.Value = 0
' Заполнене задачи на умножение.
mult1 = randomizer.Next(2, 11)
mult2 = randomizer.Next(2, 11)
UmnL_L.Text = mult1.ToString
UmnR_L.Text = mult2.ToString
Prod.Value = 0
' Заполнение задачи на деление.
delit = randomizer.Next(2, 11)
Dim Koeff As Integer = randomizer.Next(2, 11)
delim = delit * Koeff
DivL_L.Text = delim.ToString
DivR_L.Text = delit.ToString
Quot.Value = 0
' Запуск таймера.
```

Если в задаче на умножение всё просто, то для того, чтобы результатом деления было целое число, надо, чтобы делимое было кратно делителю. Поэтому внутри кода для задачи на деление объявлен целый (**Integer**) **Koeff** и делимое находится как результат умножения целого **delit** (делителя) и целого **Koeff** (коэффициента).

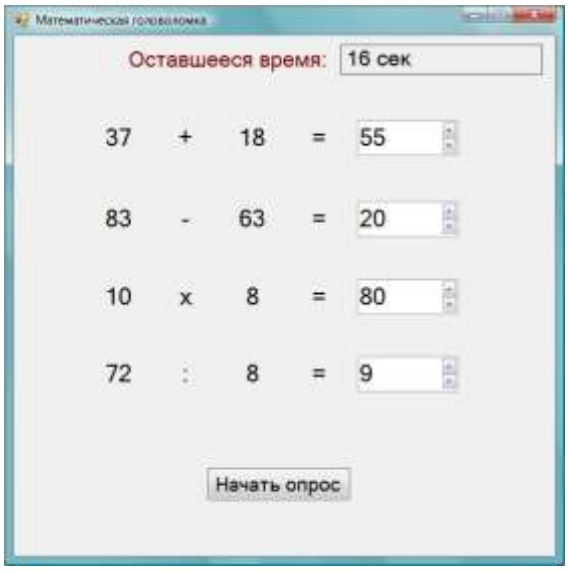
3. Теперь добавим в метод **CheckTheAnswer()** код для проверки результата умножения и деления:

```
If ((add1 + add2 = Sum.Value) AndAlso (umensh - vychit = Dif.Value) AndAlso (mult1 * mult2 = Prod.Value) AndAlso (delim / delit = Quot.Value)) Then
    Return True
Else
```

4. Теперь изменим последнюю часть обработчика событий таймера **Tick**, чтобы он заполнял верный ответ, если пользователь не уложится в отведённое время:

```
        ' Если пользователь превысит лимит, остановить таймер и показать
        ' сообщение о превышении времени.
        Timer1.Stop()
        Time_L.Text = "Время вышло!"
        MessageBox.Show("Вы не уложились в допустимое время", "Очень жаль!")
        Sum.Value = add1 + add2
        Dif.Value = umensh - vychit
        Prod.Value = mult1 * mult2
        Quot.Value = delim / delit
        Start_B.Enabled = True
    End If
```

5. **Сохраните всё** и запустите **Отладку**. Теперь должны заработать все четыре задачи.



*Измените цвет формы, измените цвет шрифта и цвет элемента **Начать опрос**.

*Попробуйте написать код для вывода сообщения-подсказки о том, что после ввода ответа необходимо нажимать клавишу **Tab.**

6. Выполните публикацию проекта.

Проект выполняется в форме многостраничного HTML-документа (сайта) в среде MSFrontPage.

Указания к проекту содержатся в презентации на сайте http://matem-109.ru – Информатика – Материалы – Проект выпускника.

Основы сайтостроения изложены в учебнике для 10 класса (можно скачать на сайте http://matem-109.ru – Информатика – Материалы).

Удачной работы!

Литература:

1. Информатика и ИКТ. 10 класс. Базовый уровень. Угринович Н.Д.
2. Информатика и информационные технологии. Учебник для 10-11 классов. Угринович Н.Д.
3. http://msdn.microsoft.com/ru-ru/library/dd460654.
4. http://msdn.microsoft.com/ru-ru/library/y4wf33f0.

Приложение 1

Таблица единиц измерения информации

Бит	**наименьшая единица измерения количества информации**. Принимает значение 0 или 1 (bit – binary digit — двоичный знак)
1 байт	= 2^3 бит = 8 бит
1 кбайт	= 2^{10} байт = 1024 байт
1 Мбайт	= 2^{10} Кбайт = 1024 Кбайт = 2^{20} байт
1 Гбайт	= 2^{10} Мбайт = 1024 Мбайт = 2^{30} байт

Приложение 2

Таблица соответствия десятичной, шестнадцатеричной, восьмеричной и двоичной систем счисления

Десятичная	0	1	2	3	4	5	6	7
Восьмеричная	0	1	2	3	4	5	6	7
Шестнадцатеричная	0	1	2	3	4	5	6	7
Двоичная	0000	0001	0010	0011	0100	0101	0110	0111

Десятичная	8	9	10	11	12	13	14	15
Восьмеричная	10	11	12	13	14	15	16	17
Шестнадцатеричная	8	9	A	B	C	D	E	F
Двоичная	1000	1001	1010	1011	1100	1101	1110	1111

Приложение 3

Таблица степеней числа 2

2^{10}	2^9	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
1024	512	256	128	64	32	16	8	4	2	1