Паньгина Нина Николаевна

ЗАНЯТИЕ 4. РАЗРАБОТКА ИГР

Вы хорошо потрудились и хотите отдохнуть? А отдохнуть для Вас – это поиграть в какую-нибудь интересную игру? Можно, но только создайте эту игру сами!

ПРОЕКТИРОВАНИЕ МЕНЮ

Создание меню в среде Visual Basic является очень легкой задачей. Но именно меню делает собственные программные разработки наиболее похожими на профессиональные Windows-приложения. Создается меню с помощью Проектировщика меню ***Menu Design*** в более ранних версиях VB или Редактора меню ***Menu Editor*** (рисунок 1) в современных версиях VB. Для этого необходимо войти в пункт горизонтального меню *Tools* и активизировать нужное окно.

Caption. Содержит заголовок пункта меню, определяющий или название команды, или название выпадающего меню. Все, что вводится в этом окне, автоматически появляется в нижней части окна.

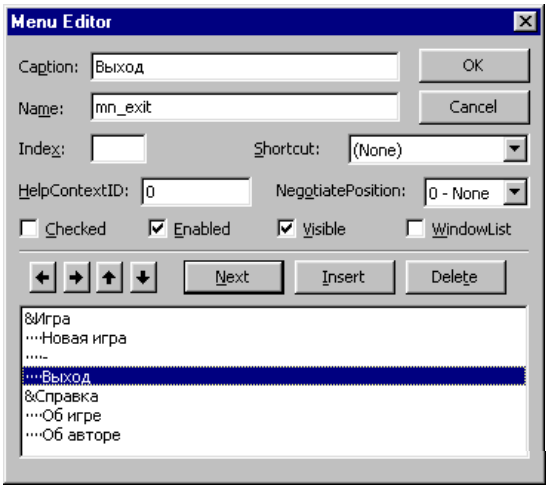


Рисунок 1

Знак «&» около буквы дает подчеркивание этой буквы в пункте меню. Символ «–» создает разделительную черту в меню, например, перед пунктом Выход (рисунок 2).

Name. В текстовом окне для каждого из элементов, входящих в меню, вводится название или специальный идентификатор, который впоследствии используется при написании текста событийной процедуры, вызываемой при выборе этого пункта или команды.

Index. Текстовое окно *Index* позволяет ввести значение этого параметра для некоторого элемента, входящего в состав меню. После того, как введено значение параметра, управляющая структура автоматически становится частью массива подобных объектов.

Shortcut. В этом окне высвечивается полный список всех клавиш и комбинаций клавиш сокращенного доступа (горячие клавиши), которые можно назначать командам меню.

Windowlist. По умолчанию, присвоено значение *false*. Активизируется, если пункт меню содержит некоторые подпункты, образуя ниспадающее меню.

Checked. Рядом с названием будет поставлена галочка (при активизации). При выборе этого режима создается двоичный переключатель для обозначения того, выбрана ли данная команда в меню.

Enabled. Проверочное окно отмечено галочкой (по умолчанию), значение па-

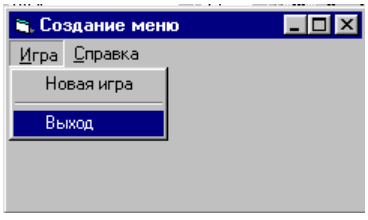


Рисунок 2

раметра *true*. В случае, когда команда в меню доступна, ее выбор влечет за собой выполнение назначенной ей событийной процедуры; если команда недоступна, то ее параметр устанавливается в *false* (выводится с пониженной яркостью и не реагирует на выбор).

Visible. Проверочное окно отмечено галочкой (по умолчанию). Для того чтобы сделать команду невидимой в меню, уберите отметку, после чего она станет недоступной и уберется из меню.

Кнопки для создания пунктов меню.

- ***Next*** **–** кнопка используется при составлении меню, означает переход на следующую строку в созданном меню.
- ***Insert*** **–** с помощью данной кнопки можно вставить в меню новую строку.
- ***Delete*** **–** с помощью этой кнопки можно удалить строку в готовом меню.
- ***Кнопки с изображениями стрелок*** используются при создании подпунктов основного или ниспадающего меню, либо для изменения их порядка.

Существует ***всплывающее меню*** **(popup menu)**, создаваемое с помощью метода PopupMenu. Статус всплывающего может быть присвоен любому пункту меню.

Задание 1. Игра «БЛОКИ».

Создать приложение, представляющее собой игру с оформлением, как на рисунке 3, и со следующими правилами:

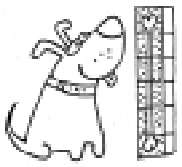
1. Игра начинается при запуске программы или при выборе пунктов меню «Игра», «Новая».

2. На форме расположены 16 кнопок-блоков, под которыми скрыты парные картинки. Их необходимо открывать по одной. Если вторая открытая картинка не является парой для первой открытой картинки, то первую картинку нужно снова закрыть. Если же открытые картинки совпадают, то они должны исчезнуть.

3. Во время игры необходимо считать количество сделанных ходов и учитывать время. По окончании игры все картинки можно открыть и вывести Окно сообщения с поздравлением.

4. Игру завершить после выбора пункта меню «Выход».

Выполнение задания 1.



Первая часть – *визуальное программирование*.

1.Необходимо создать форму и переименовать ее в соответствии с рисунком 3, свойству Caption присвоив значение «Игра БЛОКИ».

2. Создать меню из одного главного пункта «Игра» и двух подпунктов: «Новая» и «Выход». В соответствующих пунктах меню свойству Name присвоить такие значения: mnuGame, mnuNew, mnuExit.

3. Нанести на форму массив из 16 Окон рисунка, присвоив свойству Name значение pic, а в свойство picture первых восьми Окон рисунков занести изображения 8 различных картинок (по величине могут удачно подходить файлы рисунков с расширением .ico).

4. Нанести на форму массив из 16 Командных кнопок, закрыв ими Окна рисунков, присвоив свойству Name значение cmd, а в свойство Caption занести пустую строку.

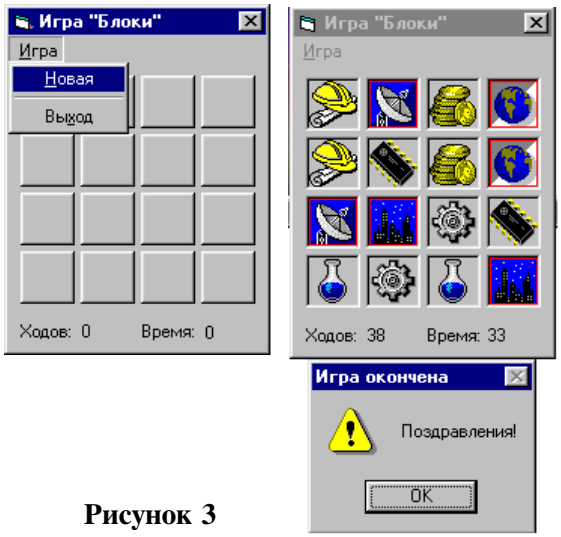


Рисунок 3

5. Нанести на форму Метку 1, изменив ее свойство Caption на «Ходов», а свойство Name – на lbl1.

6. Нанести на форму Метку 2, изменив ее свойство Caption на «0», а свойство Name – на lblMoves.

7. Нанести на форму Метку 3, изменив ее свойство Caption на «Время», а свойство Name – на lbl3.

8. Нанести на форму Метку 4, изменив ее свойство Caption на «0», а свойство Name – на lblTime.

9. Нанести на форму таймер, присвоив свойству Enabled значение false (первоначально отключить), свойству Interval – 1000 миллисекунд.

Вторая часть – *написание кода программы.*

10. В данном приложении будут использоваться две глобальные переменные Opened – номер открытой картинки и Pairs – количество открытых пар картинок. Описываем их в общем разделе описаний General Declarations:

```
Dim Opened
Dim Pairs
```

11. Первое событие – загрузка Формы. В процедуре обработки данного события последние восемь Окон рисунков заполнить картинками из первых восьми Окон рисунков. Начать новую игру обращением к событийной процедуре mnuNew_Click (щелчок на пункт меню «Новая»).

```
Private Sub Form_Load()
  For i = 0 To 7
     Pic(i + 8).Picture = Pic(i).Picture
  Next
  Call mnuNew_Click
End Sub
```

12. Второе событие – Выбор пунктов меню «Игра» «Новая». В процедуре обработки события Щелчок на данный пункт меню необходимо выполнить следующие действия:

- сделать видимыми все командные кнопки,
- двадцать раз поменять местами случайно выбранные пары картинок, используя случайные числа для индексов (номеров) Окон рисунков,
- переменной Opened присвоить значение –1, обозначая тем самым, что все картинки закрыты,
- переменной Pairs присвоить значение 0, обозначая, что открытых пар картинок еще нет,
- присвоить начальное значение «0» для Меток 2 и 4 (количество ходов и время),
- запустить Таймер.

```
Private Sub mnuNew_Click()
  For i = 0 To 15
    cmd(i).Visible = True
  Next
  For i = 1 To 20
    a = Int(Rnd * 16)
    b = Int(Rnd * 16)
    If a <> b Then
      Set Temp = Pic(a).Picture
      Pic(a).Picture = Pic(b).Picture
      Pic(b).Picture = Temp
    End If
  Next
  Opened = -1
  Pairs = 0
  lblTime.Caption = "0"
  lblMoves.Caption = "0"
  Timer.Enabled = True
End Sub
```

13. Следующее событие – срабатывающий Таймер. В процедуре обработки данного события на 1 секунду увеличивается время, отображаемое в Метке 4.

```
Private Sub Timer_Timer()
   lblTime.Caption = lblTime.Caption + 1
End Sub
```

14. Следующее событие – щелчок на одну из массива кнопок-блоков. В процедуре обработки данного события необходимо выполнить следующее:

- сделать невидимой кнопку с текущим индексом, то есть показать картинку;
- в Метке 2 увеличить на единицу количество ходов;
- если открытая картинка не первая, то проверить, совпадает ли текущая картинка с уже открытой;
- если совпадает, то сделать обе картинки невидимыми, увеличить количество открытых пар (значение переменной Pairs)

на 1; проверить, не равно ли количество открытых пар 8, и если равно, то остановить Таймер, показать все картинки, вывести в Окне сообщений поздравления с окончанием игры;

- присвоить переменной Opened значение –1 и выйти из процедуры;
- если же текущая картинка не совпала с предыдущей, то предыдущую картинку необходимо закрыть (сделать видимой соответствующую кнопку);
- переменной Opened присвоить значение индекса текущей картинки.

```
Private Sub cmd_Click(Index As Integer)
cmd(Index).Visible = False
lblMoves.Caption = lblMoves.Caption + 1
If Opened <> -1 Then
   If Pic(Opened).Picture = _
   Pic(Index).Picture Then
      Pic(Opened).Visible = False
      Pic(Index).Visible = False
      Pairs = Pairs + 1
      If Pairs = 8 Then
         Timer.Enabled = False
         For i = 0 To 15
            Pic(i).Visible = True
         Next
         MsgBox "Поздравления!", _
         vbExclamation, "Игра окончена"
      End If
      Opened = -1
      Exit Sub
   Else
      cmd(Opened).Visible = True
   End If
End If
Opened = Index
End Sub
```

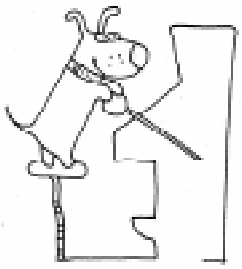
Примечание. Поскольку ширина страницы книги не бесконечна, некоторые операторы программы разбиты на несколько строк. В конце таких строк используется символ подчеркивания (_) с предваряющим пробелом, что является стандартным символом продолжения языка Visual Basic. При вводе текста в окно редактора вы вполне можете опустить символы продолжения и записать оператор в одной строке. Обе записи правомерны, и компилятор правильно их воспримет.

15. Последнее событие – щелчок на пункт меню «Выход». В процедуре обработки этого события необходимо завершить работу программы.

```
Private Sub mnuExit_Click()
   End
End Sub
```

16. Запустить программу на выполнение, протестировать ее и завершить.

Задание 2. Игра «Jack Pot» (для самостоятельного выполнения).



Создать приложение, представляющее собой игру с оформлением, как на рисунке 4, и со следующими правилами:

«Вы указываете начальный капитал, который имеется на Вашем счету, и затем можете делать ставки, не превышающие сумму Вашего счета. После нажатия на кнопку ПУСК в окнах меняются картинки. По нажатию на кнопку или по истечении 25 секунд картинки останавливаются. При выпадении трех одинаковых картинок Ваша ставка утраивается, а двух одинаковых – удваивается. Если все картинки разные, то Вы теряете свои деньги. Игра прекращается либо по Вашему желанию, либо при обнулении Вашего счета».

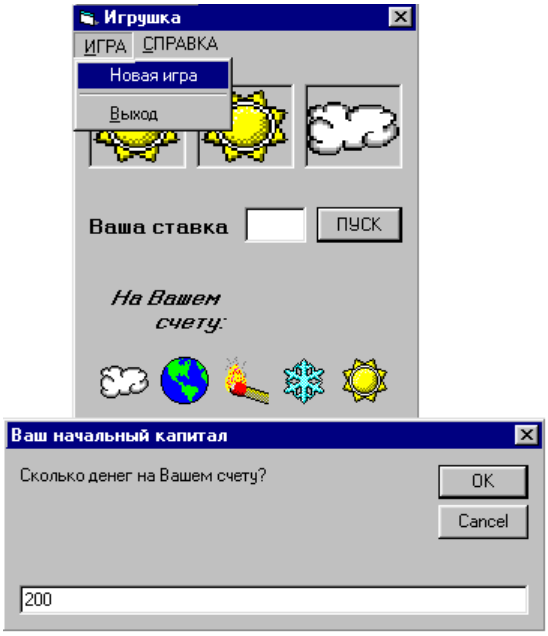
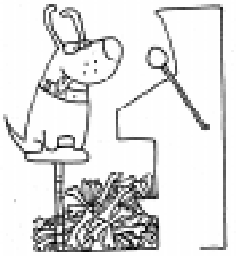


Рисунок 4

Указания к выполнению задания 2.



1. Создайте форму и переименуйте ее в соответствии с рисунком 4.

2. Создайте меню из двух главных пунктов «ИГРА» и «СПРАВКА» и следующих подпунктов:

- «Новая игра» и «Выход» для пункта «ИГРА»;
- «Правила игры» и «Об авторе» для пункта «СПРАВКА».

3. Для пунктов «Правила игры» и «Об авторе» предусмотрите вывод Окна сообщения с соответствующей информацией.

4. Нанесите на форму массив из трех Окон изображений и массив из пяти Окон рисунков, в свойство picture Окон рисунков занесите изображения пяти различных картинок.

5. Нанесите на форму три Метки, поместив в две из них названия «Ваша ставка» и «На Вашем счету», а в третью произвольное число для обозначения начального капитала.

6. Нанесите на форму Текстовое окно для дальнейшего занесения в него информации о ставке.

7. Нанесите на форму Таймер, чтобы использовать его в дальнейшем для смены картинок, и Командную кнопку для запуска или остановки Таймера.

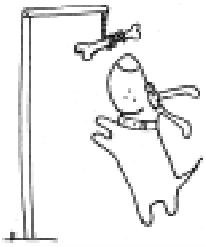
8. При выборе пункта меню «Новая игра» предусмотрите очистку Текстового окна для новой ставки и задание Окна ввода для начального капитала.

9. При обработке события Щелчок на кнопку предусмотрите запуск Таймера, проверку на ввод недопустимой ставки (меньшей либо равной нулю или большей количества денег на счету), учет денег на счету игрока, учет времени.

10. При обработке события Таймер реализуйте случайный выбор картинок из пяти Окон рисунков для трех Окон изображений, не забудьте увеличить время на величину свойства Interval объекта Таймер, а по истечении заданного времени остановить работу Таймера. Осуществите проверку на совпадение картинок в Окнах изображений, при необходимости подсчитайте выигрыш. Проверьте сумму денег на счету и при их отсутствии завершите игру.

11. Предусмотрите выход из программы с подтверждением при выборе пункта меню «Выход».

Задание 3. Игра «Отгадай слово» (для самостоятельного выполнения).



Создать приложение, представляющее собой игру «Отгадай слово», известную среди школьников под названием «Виселица» с оформлением, как на рисунке 5, и со следующими правилами:

1. Компьютер загадывает слово на определенную тему, например, «Страны и города», и показываются начальные буквы данного слова.

2. Отгадывать необходимо по буквам, щелкая на них курсором мыши. Вер-

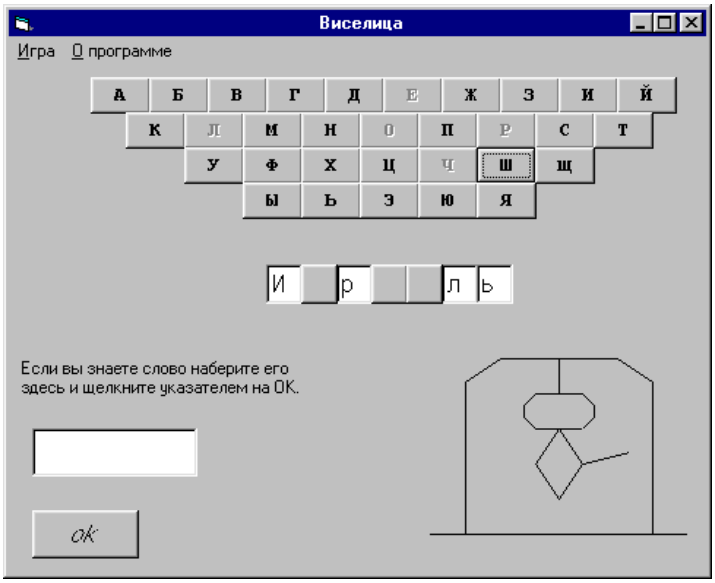


Рисунок 5

но угаданная буква вставляется в слово. Если же буква выбрана неправильно, то рисуется одна деталь виселицы.

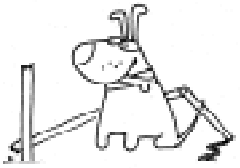
3. Если играющий не успевает отгадать слово до того, как рисунок будет закончен, то он проиграл.

4. Имеется возможность отгадать слово целиком, набрав его в специальном текстовом окне.

5. Игру можно повторить, выбрав в пункте меню «Игра» подпункт «Новая».

6. Выход из программы осуществляется через пункт меню «Выход».

Указания к выполнению задания 3.



1. Вы можете использовать в качестве букв массив Командных кнопок и их свойство Caption. Массив кнопок можно нанести на форму на стадии визуального программирования, но можно создать и программным путем следующим образом:

- Создать первый элемент массива еще на стадии разработки программы, то есть поместить одну Командную кнопку на форму, изменить размеры на желаемые, в свойство Caption занести русскую букву «А», в свойство Name занести cmd и в свойстве Index указать значение 0.
- В процедуре загрузки формы использовать для создания и загрузки в память новых элементов массива кнопок оператор Load. Новый элемент будет располагаться в той же позиции, что и первый, поэтому необходимо изменить значение свойства Left. Для изменения свойства Caption удобно использовать функции Chr и Asc, так как коды ASCII букв последовательно возрастают.

```
Private Sub Form_Load()
  For i = 1 To 9
    Load Cmd(i)
    Cmd(i).Caption = Chr(Asc(Cmd _
    (i - 1).Caption) + 1)
    Cmd(i).Left = Cmd(i - 1).Left _
    + Cmd(i - 1).Width
  Next i
End Sub
```

2. В качестве деталей рисунка виселицы Вы можете использовать объекты управления и контроля Линия, сделав их на стадии разработки невидимыми, а во время выполнения программы при необходимости показывать.

3. Слова для игры храните либо в массиве, либо в файле по Вашему усмотрению.

ТЕХНОЛОГИЯ DRAG-AND-DROP

Visual Basic поддерживает способ взаимодействия с формой, называемый *«перетащить и отпустить»* (***drag and drop***). Этот термин относится к перемещению мышью какого-либо объекта (чаще всего Окна рисунка или изображения). Перемещение объекта разрешено, если его свойству DragMode присвоено значение 1 (Automatic). Перемещаемый объект управления называется *объектом-источником* (***Source***), а объект, воспринимающий событие DragDrop (на который опускается источник), называется *целью* (***Target***). Когда перемещаемый объект проходит над любым другим объектом, для того объекта генерируется событие DragOver.

Задание 4. Игра «Поместите в нужный раздел».

Разработайте игру с пользовательским интерфейсом, приведенном на рисунке 6, и следующими правилами:

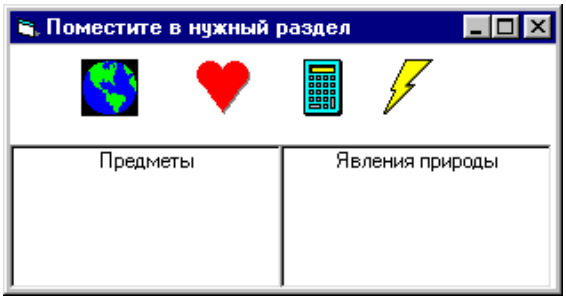


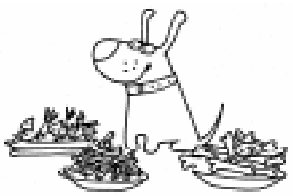
Рисунок 6

· Из предложенного набора рисунков выбрать те, на которых изображены предметы, и переместить их в левый раздел с соответствующим названием.

· Из предложенного набора рисунков выбрать те, на которых изображены явления природы, и переместить их в правый раздел с соответствующим названием.

· Если изображение на рисунке не подходит ни под какой из разделов, сделать перемещение недопустимым.

Выполнение задания 4.



Первая часть – *визуальное программирование*.

1. Необходимо создать форму и переименовать ее в соответствии с рисунком 6.

2. Нанести на форму массив из 4 Окон рисунка, задав соответствующие свойства BackColor и BorderStyle, присвоив свойству Name значение pic, а в свойство picture занести различные картинки (смотрите на рисунке 6: земля, сердце, калькулятор, молния). Свойству DragMode для Окон рисунков придать значение 1.

3. Нанести на форму Метку 1, изменив ее свойство Caption на «Предметы».

Вторая часть – *написание кода программы.*

4. В процедуре обработки события DragDrop для Метки 1, являющейся целью, параметр Source является объектом-источником, а параметры X иY – координатами курсора мыши относительно Метки 1. Для того чтобы разрешить «приземление» источнику, нужно проверить, является ли он предметом (предмет же в данном случае – только калькулятор, изображенный в Picture Box с индексом 2). Почти к любому объекту управления и контроля можно применять метод ***Move***.

Метод ***Move*** служит для перемещения объектов управления и контроля или формы. Синтаксис метода ***Move***:

```
Объект.Move left, top[, width, height]
```

где left – x-координата объекта, top – y-координата объекта, width – новая ширина объекта, height – новая высота объекта.

В рассматриваемой процедуре необходимо left и top задавать относительно формы.

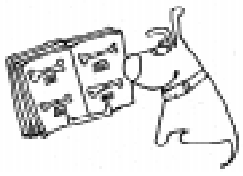
```
Private Sub Label1_DragDrop(Source _
As Control, X As Single, Y As Single)
   If Source.Index = 2 Then _
   Source.Move X + Label1.Left, Y + _
   Label1.Top
End Sub
```

5. В процедуре обработки события DragDrop для Метки 2 необходимо проверить, относится ли перемещаемое изображение к явлениям природы (таковыми являются изображения земли – индекс 0 и изображение молнии – индекс 3).

```
Private Sub Label2_DragDrop(Source _
As Control, X As Single, Y As Single)
   If Source.Index = 0 _
    Or Source.Index = 3 Then
      Source.Move X + Label2.Left, _
       Y + Label2.Top
   End If
End Sub
```

6. Изображение сердца, имеющее индекс 1, не будет помещаться ни в один из разделов, так как к нему не применяется метод Move.

ОБЪЕКТ COLLECTION



Collection (Коллекция) – упорядоченный набор элементов, к которому можно обращаться как к единому целому.

Создать объект Collection, так же как и любой другой объект VB, можно двумя способами:

1. Использовать оператор Dim с ключевым словом New, например,

```
Dim C As New Collection.
```

2. Использовать оператор Dim без ключевого слова New, но с последующим оператором ***Set*** (данный оператор служит для назначения переменной или свойству ссылки на объект), например,

```
Dim C As Collection
…
Set C = New Collection.
```

С коллекциями можно совершать следующие операции:

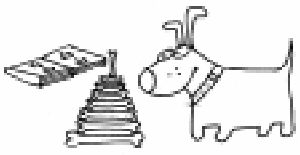
- Добавлять новый элемент с помощью метода **Add** (например, чтобы добавить в коллекцию C объект форму с именем Form1, можно написать C.Add Form1).
- Получать элемент, используя метод **Item** (для коллекции данный метод установлен по умолчанию, поэтому равнозначны следующие записи: Print C.Item(1).Caption и Print C(1).Caption).
- Удалять существующий элемент с помощью метода **Remove** (например, чтобы удалить второй элемент коллекции, можно записать C.Remove (2)).
- Определять количество элементов с помощью ключевого слова **Count** (чтобы вывести на форму количество элементов в коллекции, нужно записать Print C.Count).
- Пробегать по всем элементам коллекции с помощью оператора цикла **For Each ... Next** (например, для вывода на форму названий всех объектов коллекции можно записать:

```
Dim a As Object
For Each a In C
Print a.Caption
Next a)
```

Задание 5. Игра «Ханойские башни».

Разработайте игру с пользовательским интерфейсом, приведенным на рисунке 7, и следующими правилами:

- Из предложенного в меню списка, показывающего количество дисков, выбирается желаемое, после чего на левом стержне появляются диски в выбранном количестве.
- Требуется перенести все диски на правый стержень, используя средний стержень в роли вспомогательного.
- Диски можно перекладывать только по одному, причем каждый диск можно класть лишь на диск большего размера.
- Разработанная программа должна реализовать выше указанные правила, вести счет ходов в игре, а при ее завершении вывести поздравление с указанием минимально возможного числа ходов для данного количества дисков.
- Предусмотреть возможность многократной игры.



Выполнение задания 5.

Первая часть – *визуальное программирование*.

1. Необходимо создать форму и переименовать ее в соответствии с рисунком 7.

2. Создать меню с главным пунктом «Игра» (свойство Name – mnuGame), подпунктами «Количество дисков» (свойство Name – mnuDisks) и «Выход» (свойство Name – mnuExit), а также массивом подпунктов 2, 3, …, 9 (свойство Name – mnuCount с индексами от 2 до 9).

3. Нанести на форму массив из 3 Окон изображений, присвоив свойству Name значение ImgRod, а в свойство picture занести картинку с изображением стержня.

4. Нанести на форму Метку 1, изменив ее свойство Caption на «Ходов:».

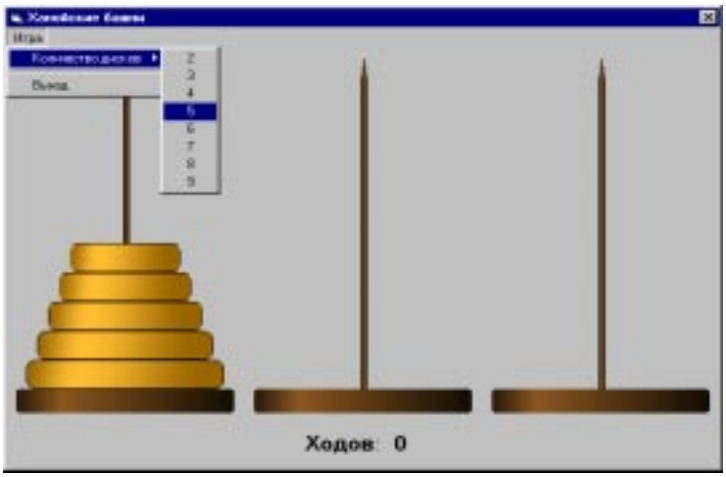


Рисунок 7

5. Нанести на форму Метку 2, изменив ее свойство Caption на «0», а свойство Name на lblMoves.

6. Нанести на форму Окно изображения, присвоив свойству Name значение ImgDisk, в свойство picture занести картинку с изображением диска, свойству Index присвоить значение 0 (таким образом, Окно изображения становится массивом объектов с пока лишь одним нулевым элементом), свойству Visible присвоить значение False (сделать диск невидимым).

Вторая часть – *написание кода программы.*

7. В общем разделе описаний объявить две переменные целого типа N – количество дисков, DragFrom – текущий номер стержня, с которого взят диск. Объявить массив из трех коллекций для запоминания текущего количества дисков на всех трех стержнях (0 – левый стержень, 1 – средний, 2 – правый)

```
Dim DragFrom, N As Integer
Dim Stick(0 To 2) As Collection
```

8. В процедуре обработки события Щелчок на пункт меню с числом дисков необходимо выполнить следующие действия:

- В Метке 2 обнулить свойство Caption (необходимо, если начата новая игра).
- Выгрузить массив дисков, оставшихся от предыдущей игры (если игра первая, то N равно на данный момент 0).
- В N занести количество выбранных дисков для данной игры (совпадает с индексом элемента из массива подпунктов меню).
- Инициализировать коллекции (дисков) для всех трех стержней.
- Создать программно в цикле массив из N Окон изображений (массив дисков), используя нулевой элемент, созданный на стадии разработки, присвоить для всех элементов массива различные свойства Left, Top, Width (диски уменьшаются по ширине и накладываются друг на друга).
- Применить ко всем дискам метод Zorder, помещающий Окна рисунков на передний план (перед стержнем), сделать все диски видимыми.
- Добавить все диски по очереди в коллекцию для первого стержня.

```
Private Sub mnuCount_Click _
(Index As Integer) 'New game
  lblMoves.Caption = "0"
  For i = 1 To N
    Unload imgDisk(i)
  Next
  N = Index
  For i = 0 To 2
    Set Stick(i) = New Collection
  Next
  For i = 1 To N
    Load imgDisk(i)
    imgDisk(i).Left = imgDisk _
    (i - 1).Left + 10
    imgDisk(i).Top = imgDisk(i - _
    1).Top - imgDisk(i - 1).Height
    imgDisk(i).Width = _
    imgDisk(i - 1).Width - 20
    imgDisk(i).ZOrder
    imgDisk(i).Visible = True
    Stick(0).Add i
  Next
End Sub
```

9. Для удобства целесообразно описать функцию, возвращающую номер верхнего диска для текущего стержня. Сначала переменной k присваивается количество элементов в коллекции для стержня с номером Index. Если количество равно 0, то функция принимает значение 0, иначе функции присваивается значение последнего элемента коллекции для данного стержня.

```
Function TopDisk(ByVal Index As _
Integer) As Integer
  k = Stick(Index).Count
  If k = 0 Then TopDisk = 0 Else _
  TopDisk = Stick(Index).Item(k)
End Function
```

10. В процедуре обработки события Нажатие кнопкой мыши на диск (элемент с номером Index из массива Окон изображений imgDisk) необходимо проверить, является ли этот диск верхним. В цикле по всем трем стержням проверить,

если текущий диск верхний, то есть TopDisk(i) = Index, то применить к нему метод Drag, разрешающий перетаскивание (изначально свойство DragMode было равно 0, то есть запрещено автоматическое перетаскивание мышью), запомнить в переменной DragFrom номер стержня, на котором находится этот диск и выйти из процедуры.

```
Private Sub imgDisk_MouseDown(Index _
As Integer, Button As Integer, Shift _
As Integer, X As Single, Y As Single)
For i = 0 To 2
   If TopDisk(i) = Index Then
      DragFrom = i
      imgDisk(Index).Drag
      Exit Sub
   End If
Next
End Sub
```

11. В процедуре обработки события «прохождения» диска (параметр Source – источник) над стержнем (элемент массива Окон изображений ImgRod с номером Index) необходимо проверить, снимается со стержня или одевается на этот стержень диск. Эта информация содержится в параметре State (State=0 при «вхождении» и State=1 при «покидании»). Если диск вносится на стержень, а его номер меньше верхнего диска на этом стержне (меньшему номеру соответствует больший диск), то курсору мыши над диском присвоить другой вид (запрещающий). Если диск снимается со стержня, то курсор приобретает первоначальный вид.

```
Private Sub ImgRod_DragOver(Index As _
Integer, Source As Control, X As _
Single, Y As Single, State As Integer)
   Select Case State
   Case 0       'Drag Enter
      If Source.Index < TopDisk(Index) _
      Then Source.MousePointer = 12
   Case 1       'Drag Leave
      Source.MousePointer = 0
   End Select
End Sub
```

12. В процедуре обработки события «отпустить» диск (параметр Source – источник) на стержень (элемент массива Окон изображений ImgRod с номером Index) необходимо выполнить следующее:

- Если указатель мыши на диске запрещающий (.MousePointer = 12), то ничего не делать, только заменить вид курсора на обычный.
- Если указатель мыши на диске обычный, то
  a. удалить из коллекции стержня с номером DragFrom (с которого снят диск) элемент с наибольшим номером,
  b. в коллекцию текущего стержня добавить этот диск (Source.Index),
  c. переместить диск на новое место, изменив его свойства Left и Top,
  d. в Метке 2 увеличить количество ходов на 1,
  e. проверить, если количество дисков на текущем стержне стало равно N, то закончить игру выводом сообщения с поздравлением и числом минимально возможных ходов в данной игре (вычисляется по формуле 2^N–1, где N – число дисков).

```
Private Sub ImgRod_DragDrop(Index As _
Integer, Source As Control, X As _
Single, Y As Single)
If Source.MousePointer = 12 Then_
'Cannot drop
   Source.MousePointer = 0
Else
   Stick(DragFrom).Remove _
   Stick(DragFrom).Count
   Stick(Index).Add Source.Index
   Source.Left = ImgRod(Index).Left _
   + Source.Index * 10
   Source.Top = imgDisk(0).Top - _
   Stick(Index).Count * Source.Height
   lblMoves.Caption = _
   lblMoves.Caption + 1
   If (Stick(Index).Count = N) And _
   (Index <> 0) Then MsgBox _
   "Поздравляю!" + vbCr + _
   "Минимальное количество ходов: " _
   + Str(2 ^ N - 1), vbInformation, _
   "Игра окончена"
End If
End Sub
```

13. В процедуре обработки события Нажатие на пункт меню «Выход» осуществить завершение работы программы.

```
Private Sub mnuExit_Click()
  End
End Sub
```

Задание 6. Обучающая игра «Переливайка» (для самостоятельного выполнения).

Создать обучающую игру, предназначенную для решения популярных математических задач на переливание, с оформлением, как на рисунке 8, и со следующими правилами:

1. В пункте меню «Задачи» играющий может выбрать одну из задач, которые представляются в виде текста и сопроводительного рисунка на открывающейся второй форме (смотрите рисунок 9).

2. После выбора задачи на первой форме сосуды приобретают соответствующие объемы и уровни наполнения.

3. Играющий выполняет переливание из сосуда в сосуд, решая задачу. Все ходы отображаются в списке. Возможна отмена одного или нескольких последних шагов.

4. В пункте меню «Данные» можно поставить свою задачу – задать объемы сосудов и начальные уровни жидкости в них.

Указания к выполнению задания 6.

В задании могут использоваться дополнительные элементы управления ListView и StatusBar.

Задание 7. Любимая игра (для самостоятельного выполнения).

Создать приложение, представляющее собой Вашу любимую игру, будь то игра «Угадай задуманное число», либо карточная игра «21» («Очко»), либо телевизионная версия игры «Спортлото» и т.п. Для того чтобы осуществить задуманное, у Вас на данный момент имеется уже достаточно большой багаж знаний. Удачи Вам!

Рисунок 8

Рисунок 9

Паньгина Нина Николаевна,
преподаватель ОИиВТ
школы-лицея № 8, г. Сосновый Бор.