

Тема: Умовні оператори.

Мета: Познайомитися із правилами алгебри логіки та логічними операторами, які побудовані на її основі.

## 1. Поняття математичної логіки.

Роботи Джорджа Буля 1847 та 1854 років поклали початок алгебрі логіки – булевій алгебрі. Вчений створив систему позначень та правил, користуючись якими, можна закодувати довільні висловлення, а потім маніпулювати ними як звичайними числами, передбачаючи лише два варіанти відповіді – істинно (true, 1) чи хибно (false, 0).

Булева алгебра користується трьома основними операторами – І (and), АБО (or), НІ (not), які дозволяють виконувати множення, додавання та заперечення логічних умов. У 1854 році у своїй роботі «Закони мислення» Буль повністю сформулював закони математичної логіки, скориставшись для її формалізації двійковою системою.

Створюючи мову Pascal, Ніклаус Вірт ввів для позначення логічного типу даних зарезервоване слово Boolean на знак поваги до автора математичної логіки.

Таким чином, математична логіка (булева алгебра) оперує логічними змінними, які можуть набувати значення двох видів – true або false. Значення логічним змінним задаються шляхом присвоєння констант або результатів порівняння.

Розглянемо приклад:

```
Var  
a: integer;  
x, y: Boolean;  
begin  
x:=true;  
a:=7;  
y:=(a>2);  
end;
```

Обидві логічні змінні (x та y) після виконання цього фрагменту програми набудуть значення true.

В операціях порівняння можна застосовувати звичні в математиці знаки порівняння: > (більше), < (менше), = (рівне), >= (більше або рівне), <= (менше або рівне), <> (не рівне). Як правило, умови порівняння в задачах набагато складніші, ніж проста перевірка на рівність чи нерівність. Навіть прості за умовою задачі вимагають одночасного виконання декількох умов.

Наприклад, для визначення того, чи є рік високосним, використовується наступна вербальна умова: «якщо номер року ділиться на 4 без залишку, і при цьому, якщо він ділиться без залишку на 100 (тобто, закінчується на «00»), а число його сотень не ділиться на 4» Як бачимо, ця умова дуже об'ємна; дещо простіше вона буде виглядати в оберненому варіанті – умова «невисокосності» року:

```
year:=2010;  
if (year mod 4 <>0) or (year mod 100 = 0) and ((year div 100) mod 4 <>0)  
then  
label1.caption:='невисокосний рік'  
else
```

```
label1.caption:='високосний рік';
```

У всіх мовах програмування реалізовані як мінімум чотири логічні операції:

- Логічне заперечення – not, заміна істинного виразу на хибний та навпаки.
- Логічне множення – and, потребує одночасного виконання обох умов.
- Логічне додавання – or, потребує виконання хоча б однієї із двох частин.
- «Виняткове АБО» - xor, яка дає істину лише тоді, якщо значення операндів співпадають (істина та істина, хибне та хибне).


Таблиця істинності для логічних операцій у Delphi при усіх можливих значеннях аргументів:

Змінна x	False	False	True	True
Змінна y	False	True	False	True
<b>Not x</b>	True	True	False	False
<b>Not y</b>	True	False	True	False
<b>x or y</b>	False	True	True	True
<b>x and y</b>	False	False	False	True
<b>x xor y</b>	True	False	False	True

Завдання. Внести до таблиці значення true або false відповідно до зразка (перший рядок).

Значення змінної a	1	3	5	7	9
Значення змінної b	10	-1	2	20	7
$(a \bmod 2 = 0) \text{ or } (b < 10)$	False	True	True	False	True
$((a \bmod 2) = (a \text{ div } 2)) \text{ or } (b > 0)$					
$(\text{not } (a > 2)) \text{ and } (b < 11)$					
$(a \leq 2) \text{ or } (a > 7) \text{ or } (b > a)$					
$(a \geq 5) \text{ and } (b \bmod 6 > 2)$					
$(a * b > 10) \text{ and } (a > b)$					
$(a > 2 * b) \text{ or } (b > a * 2)$					

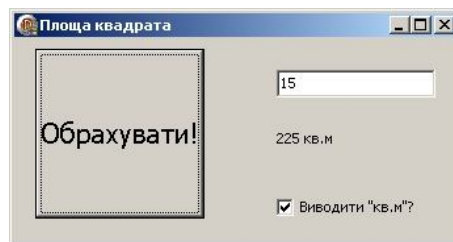
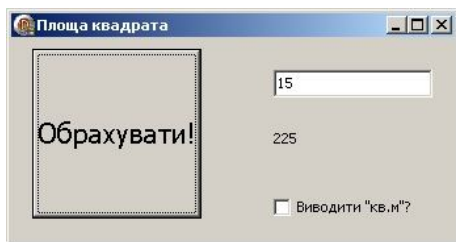
## 2. Формування умов.

Умови для виконання тієї чи іншої дії можуть формуватись шляхом порівняння вибраних значень, як результат обрахування логічних виразів або шляхом опитування відповідних відеокomпонентів. Найбільш простим та розповсюдженим компонентом для введення умов одно- чи двох-альтернативного розгалуження є компонент CheckBox ( TCheckBox).

На формі цей компонент має вигляд квадратної комірки із написом. Клацанням мишки у цю комірку можна вставити відповідну помітку – «галочку». У процесі виконання програми наявність або відсутність «галочки» в елементі CheckBox перевіряється (а при необхідності – й встановлюється) властивістю Checked.

Розглянемо приклад.

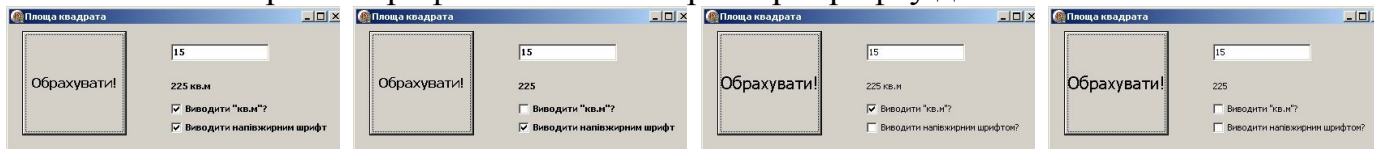
На формі в полі Edit вводиться число – довжина сторони квадрата. При натискуванні кнопки Обрахувати! В полі Label виводиться число - його площа. Якщо в розміщеному на формі компоненті CheckBox стоїть «галочка», то до числового значення площі додається підпис «кв.м».



Фрагмент коду, який буде виконуватись при клацанні мишкою на кнопці **Обрахувати!**:

```
procedure TForm1.Button1Click(Sender: TObject);  
begin  
  label1.Caption:=floattostr (sqr(strtfloat(edit1.Text)));  
  if checkBox1.Checked then  
    label1.Caption:=label1.Caption+' кв.м';  
end;
```

Ускладнимо завдання. Для зручності читання написи у всіх компонентах вивести напівжирним шрифтом (властивість `font.bold` має значення `true`). При цьому напис «Обрахувати!» може не вміститись у розміри кнопки; тому при виведенні написів напівжирним шрифтом зменшити розмір шрифту до 12 пт.



Фрагмент коду, який буде виконуватись при клацанні мишкою на кнопці **Обрахувати!**:

```
procedure TForm1.Button1Click(Sender: TObject);  
begin  
  label1.Caption:=floattostr (sqr(strtfloat(edit1.Text)));  
  if checkBox1.Checked then  
    label1.Caption:=label1.Caption+' кв.м';  
  if checkbox2.Checked then  
    begin  
      button1.Font.Size:=12;  
      button1.Font.Style:=[fsBold];  
      label1.Font.Style:=[fsBold];  
      edit1.Font.Style:=[fsBold];  
      checkbox1.Font.Style:=[fsBold];  
      checkbox2.Font.Style:=[fsBold];  
    end  
  else  
    begin  
      button1.Font.Size:=16;  
      button1.Font.Style:=[];  
      label1.Font.Style:=[];  
      edit1.Font.Style:=[];  
      checkbox1.Font.Style:=[];  
      checkbox2.Font.Style:=[];  
    end  
end;
```

### 3. Алгоритмічні конструкції одно-, двох- та багато-альтернативного розгалужень.

У ряду випадків одно альтернативне розгалуження може виявитись недостатнім: в алгоритмі може ставитись задача вибору однієї із двох альтернатив. Подібні алгоритмічні структури реалізуються двох-альтернативним оператором If, який має наступну структуру:

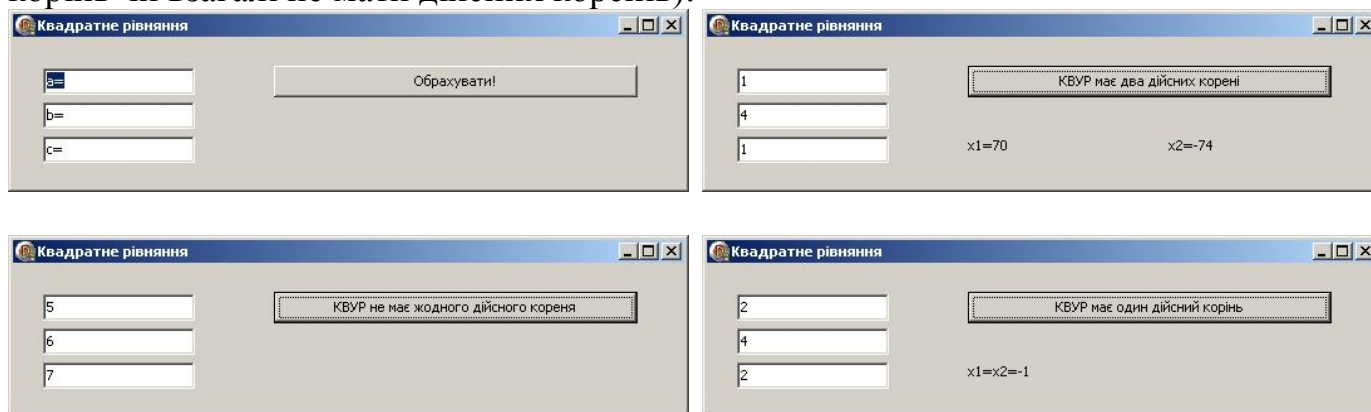
If логічний вираз  
Then оператор  
Else оператор;

Тут логічний вираз – це булева змінна, константа чи логічний вираз, що являє собою операцію (чи декілька операцій) порівняння, об'єднаних операторами булевої алгебри.

Оператори, розміщені після ключових слів then та else, можуть бути простими або складеними (тобто, групою операторів, об'єднаних операторними дужками begin .. end).

Приклад. Скласти програму для визначення коренів квадратного рівняння за заданими значеннями коефіцієнтів a, b, c.

Форма програми має наступний вигляд (у залежності від значень коефіцієнтів a, b та c квадратне рівняння може мати два різних дійсних корені, один дійсний корінь чи взагалі не мати дійсних коренів):



Фрагмент програмного коду:

```
var a,b,c,d: real;
procedure TForm1.Button1Click(Sender: TObject);
begin
  a:=strtofloat(edit1.Text);
  b:=strtofloat(edit2.Text);
  c:=strtofloat(edit3.Text);
  d:=sqr(b)-4*a*c;
  if d>0 then
  begin
    label2.Caption:='x1='+floattostr((-b+sqr(d))/(2*a));
    label3.Caption:='x2='+floattostr((-b-sqr(d))/(2*a));
    button1.Caption:='КВУР має два дійсних корені';
  end
  else
  if d<0 then
  begin
```

```
    button1.Caption:='КВУР не має жодного дійсного кореня';  
    label2.Caption:='';  
    label3.Caption:=''  
end  
else  
begin  
    button1.Caption:='КВУР має один дійсний корінь';  
    label2.Caption:='x1=x2='+floattostr(-b/(2*a));;  
    label3.Caption:=''  
end;  
end;
```

Інколи в алгоритмах недостатньо й двох альтернатив, тоді доцільно використовувати багато-альтернативне розгалуження Case, яке має наступну структуру:

```
Case вираз-перемикач of  
    Значення перемикача 1: оператор;  
    Значення перемикача 2: оператор;  
    ...  
End;  
або  
Case вираз-перемикач of  
    Значення перемикача 1: оператор;  
    Значення перемикача 2: оператор;  
    ...  
Else  
    Оператор;  
End;
```

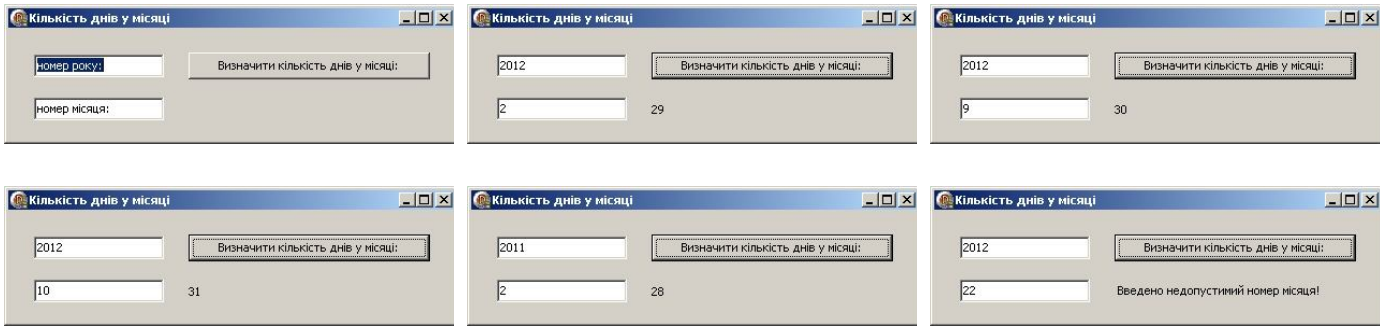
Вираз-перемикач у цьому операторі може бути змінною або виразом порядкового типу, наприклад, цілого, літерального тощо. Тип перемикача повинен співпадати із типом значень перемикача, які стоять перед двокрапкою в альтернативах оператора Case. Частина Else – необов'язкова, вона додається у випадку, коли значення виразу-перемикача не співпадає ні з одним із зарезервованих значень. Таким чином, якщо вираз-перемикач не рівний жодному із цих значень і присутня частина Else, то виконується оператор, який слідує за Else. Якщо ж частина Else в оператор Case не включена, то Case не виконує ніяких дій.

Оператори, розміщені після двокрапок в альтернативах Case, можуть бути простими або складеними (тобто, групою операторів, об'єднаних операторними дужками begin .. end).

Приклад. За введеними двома цілими числами - значеннями року та місяцю, вивести кількість днів у цьому місяці.

Пояснення. Перший, третій, п'ятий, сьомий, восьмий, десятий та дванадцятий місяці кожного року мають по 31 дню; четвертий, шостий, дев'ятий та одинадцятий – по 30 днів. Другий місяць (лютий) у високосному році має 29 днів, а в не високосному – 28 днів.

Форма програми має наступний вигляд (в залежності від введених значень року та місяця):

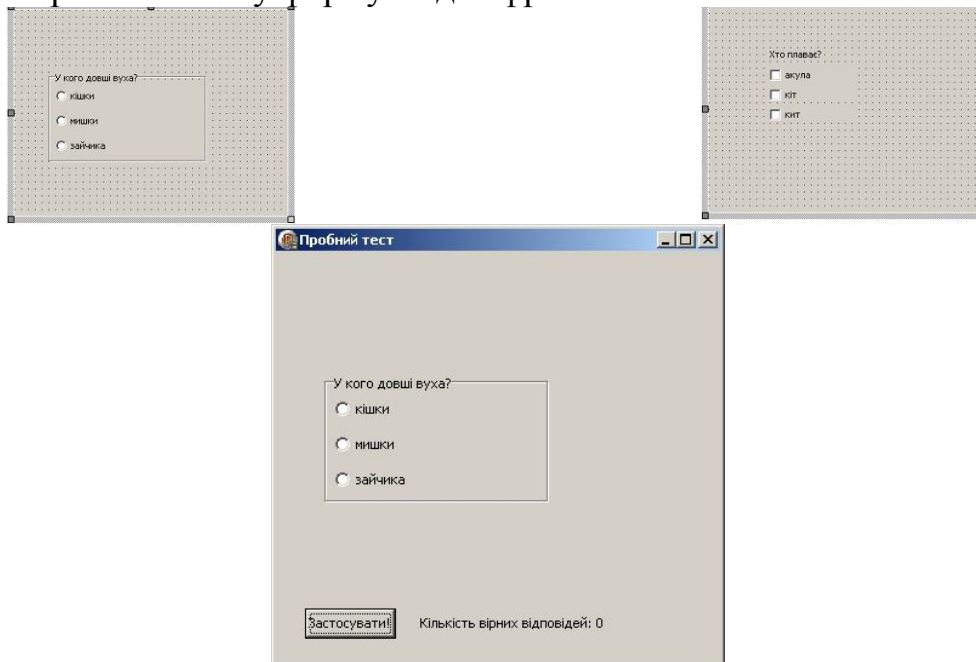


### Фрагмент програмного коду:

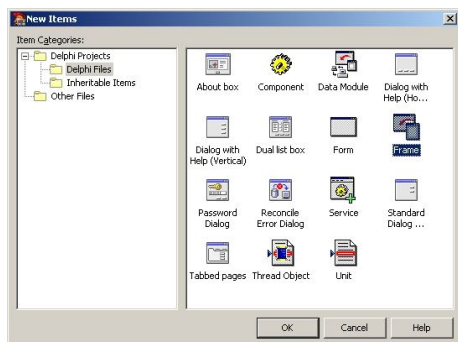
```
procedure TForm1.Button1Click(Sender: TObject);
var mesyac, god: integer;
begin
  god:=strtoint (edit1.Text);
  mesyac:=strtoint (edit2.Text);
  case mesyac of
    1,3,5,7,8,10,12: label1.Caption:='31';
    4,6,9,11: label1.Caption:='30';
    2: if (god mod 4 <>0) or (god mod 100 = 0) then
        label1.Caption:='28'
      else
        label1.Caption :='29';
    else
      label1.Caption:='Введено недопустимий номер місяця!';
  end;
end;
```

### 4. Виконання програм із розгалуженням у покроковому режимі. Вкладені оператори розгалуження.

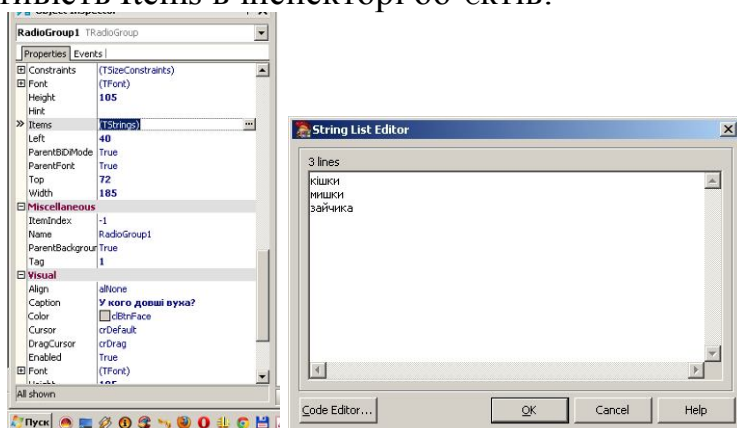
Розглянемо виконання умовних операторів при програмуванні тесту на фреймах. Створимо основну форму та два фрейми:



Для створення нового фрейму потрібно виконати File – New – Other – Frame:



На першому фреймі розмістимо компонент RadioGroup. Назву компоненту змінимо на «У кого довші вуха?», а для опису варіантів відповіді змінимо властивість Items в інспекторі об'єктів:



На другому фреймі розмістимо один об'єкт Label, внісши до нього текст «Хто плаває», а також три об'єкти CheckBox, внісши до них відповідно «акула», «кіт», «кіт».

Виносимо на форму два компоненти Frames, в діалоговому вікні вибираючи Frame2 та Frame3 (у них наскрізна нумерація з формами, тому якби у нас було дві форми, то фрейми отримали б нумерацію 3 та 4). Потрапивши на форму, фрейми змінюють ім'я: вони стають Frame21 та Frame31 (тобто, 2-й фрейм на 1-й формі та 3-й фрейм на 1-й формі).

Також розміщуємо на формі компонент Label, надавши йому початкового значення «Кількість вірних відповідей: 0», та компонент Button із написом «Застосувати!».

Програмуємо правила проходження та накопичення балів за тест у методі натискування кнопки «Застосувати»:

```
var
  Form1: TForm1;
  n: integer=0;

implementation

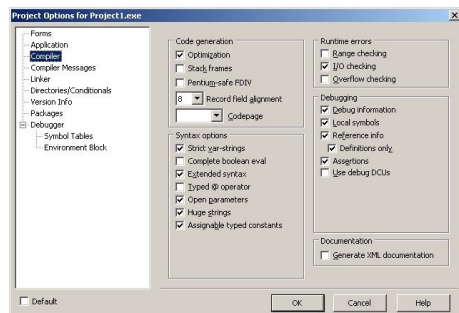
{$R *.dfm}

procedure TForm1.Button1Click(Sender: TObject);
begin
  if frame21.Visible then
  begin
    if frame21.radiogroup1.itemindex=2 then
```

```
n:=n+1;  
frame21.visible:=false;  
frame31.Visible:=true;  
end  
else  
begin  
if frame31.checkbox1.checked and frame31.checkbox3.checked then  
n:=n+1;  
frame31.Visible:=false;  
end;  
label1.Caption := 'Кількість вірних відповідей: '+inttostr(n);  
end;  
  
end.
```

Деякі зауваження щодо роботи даної програми.

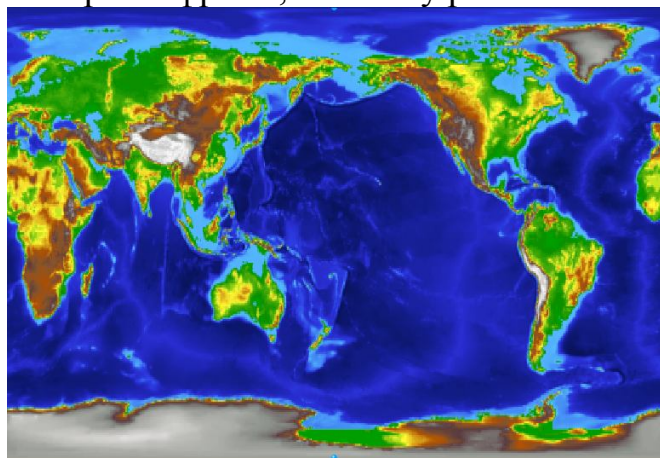
У розділі var ми описали типізовану константу n:integer=0; в якій буде накопичуватись кількість вірних відповідей. Для того, щоб це було дозволено, потрібно поставити галочку в налаштуванні Project – Options – Assignable typed constant:



Ще одна особливість програми – компоненти фреймів із форми «невидимі», тому їх потрібно іменувати повним іменем, наприклад, frame31.checkbox3.checked чи frame21.radiogroup1.itemindex

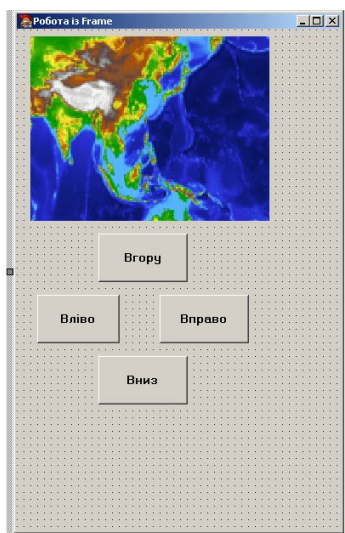
Приклад. Розглянемо використання фреймів та умовних операторів на прикладі виведення великого малюнку на невеликій формі.

Створимо фрейм, на якому розмістимо малюнок – карту світу:



Створимо форму, на якій розмістимо фрейм, а також чотири кнопки Button для переміщення по малюнку вгору, вниз, вправо та вліво:





Запрограмуємо події OnClick для кожної кнопки – переміщення у відповідному напрямку на 5 пікселів. Наприклад, кнопка вгору:

```
procedure TForm1.Button1Click(Sender: TObject);  
begin  
    frame21.image1.Top:=frame21.image1.Top-5;  
end;
```

кнопка вліво:

```
procedure TForm1.Button2Click(Sender: TObject);  
begin  
    frame21.image1.Left:=frame21.image1.Left-5;  
end;
```

кнопка вправо:

```
procedure TForm1.Button3Click(Sender: TObject);  
begin  
    frame21.image1.Left:=frame21.image1.Left+5;  
end;
```

кнопка вниз:

```
procedure TForm1.Button4Click(Sender: TObject);  
begin  
    frame21.image1.top:=frame21.image1.top+5;  
end;
```

Запустивши програму на виконання, виявимо певні недоліки: натискуючи на кнопки, можна «загнати» малюнок за край вікна. Їх можна виправити умовними операторами, встановивши «доступність» кнопок. Також не забуваємо ввімкнути потрібну кнопку, коли малюнок відсувається від краю – ліва кнопка вмикає праву, права – ліву; кнопка вгору вмикає кнопку вниз і навпаки:

Кнопка вгору:

```
procedure TForm1.Button1Click(Sender: TObject);  
begin  
    frame21.image1.Top:=frame21.image1.Top-5;  
    if abs(frame21.image1.top+10)>frame21.Image1.Height-frame21.height  
        then Button1.Enabled:=false;  
    Button4.Enabled:=true;  
end;
```

Кнопка вліво:

```
procedure TForm1.Button2Click(Sender: TObject);  
begin  
  frame21.image1.Left:=frame21.image1.Left-5;  
  if abs(frame21.image1.Left+10)>frame21.Image1.Width-frame21.Width  
  then Button2.Enabled:=false;  
  button3.Enabled := true;  
end;
```

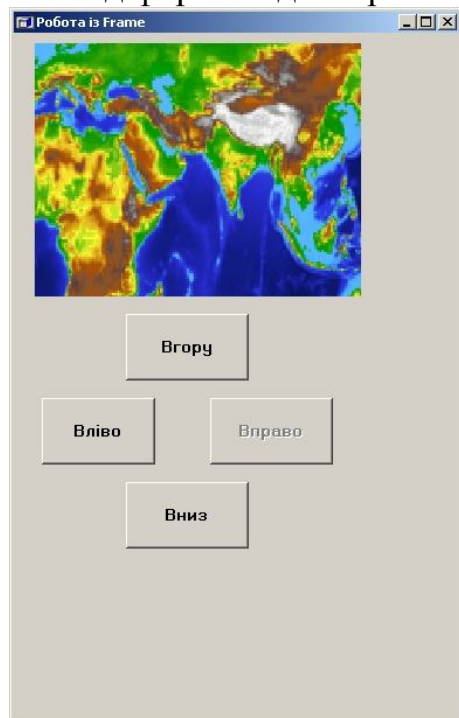
Кнопка вправо:

```
procedure TForm1.Button3Click(Sender: TObject);  
begin  
  frame21.image1.Left:=frame21.image1.Left+5;  
  if frame21.image1.Left>0  
  then Button3.Enabled:=false;  
  Button2.Enabled:=true;  
end;
```

Кнопка вниз:

```
procedure TForm1.Button4Click(Sender: TObject);  
begin  
  frame21.image1.top:=frame21.image1.top+5;  
  if frame21.image1.top>0  
  then Button4.Enabled:=false;  
  Button1.Enabled:=true;  
end;
```

Вигляд форми під час роботи програми:



## 5. Теоретичні питання для самоконтролю.

1. Якими змінними оперує математична логіка?
2. Перерахуйте можливі операції порівняння.
3. Назвіть чотири основні логічні операції.
4. Що більше - true чи false?

5. Як можуть формуватись умови для виконання деякої дії?
6. Для чого призначений компонент CheckBox?
7. Назвіть основну властивість компоненту CheckBox. Як вона використовується?
8. Назвіть властивість компоненту CheckBox, яка змінюється при помітці «галочкою». Яке значення (true чи false) буде записане в цю властивість?
9. Яка властивість компоненту CheckBox відповідає за виведення текстового питання поруч із діалоговим елементом?
10. Опишіть, які властивості потрібно налаштувати, щоб написи в Label та CheckBox виконувались більш крупним шрифтом.
11. Які особливості використання оператора If?
12. Як вставити в оператор If більше одного оператора?
13. Яке призначення оператора багато-альтернативного розгалуження Case? У яких випадках він використовується?
14. Наведіть приклади використання чотирьох альтернатив за допомогою операторів If та Case.
15. В чому полягає особливість використання frame?
16. Які дії потрібно виконати для дозволу присвоєння значень типізованим константі?
17. Як перемістити малюнок вправо на 5 пікселів?
18. Як перемістити малюнок вгору на 7 пікселів?