Тип низ

Код рада са већом количином података врло често се јавља потреба за великим бројем променљивих које би чувале потребне вредности. У већини таквих случајева број неопходних променљивих је или јако велики или се и не зна колико променљивих ће бити употребљено у програму. Да би се олакшало решавање оваквог типа проблема уводи се појам низа. Низ је уређени скуп података једног типа са заједничким именом које се назива именом низа. Тип низ има следеће особине:

- *Садржи коначан, унапред одређен број чланова (низ може имати произвољно много чланова, што засвиси од потреба проблема који се решава и од расположиве меморије)*
- *Сви чланови једног низа морају бити истог типа*
- *Сваки члан се означава именом низа и својим индексом (редним бројем)*
- *Индекси чланова једног низа морају бити истог типа.*

Да би се низ могао користити у програму мора бити претходно дефинисан (имплицитно или експлицитно). Експлицитна дефиниција подразумева да се дефинише нови тип података који има своје име. Имплицитна дефиниција се користи када не желимо да *измишљамо* име новом типу података, него га само описујемо кроз декларацију променљивих које му припадају. Декларација променљиве типа низ се састоји у давању имена низа и одређивању типа индекса и типа чланова низа. Тип индекса мора бити интервални подтип уређених простих типова (***boolean, char, integer***, набројани). Тип чланова низа може бити било који дефинисани тип. Декларацијом низа резервише се количина меморије потребна за чување свих његових елемената без обзира да ли ће се сви они користити у програму или не.

```
ARRAY[1..100] of REAL
```

низ може имати највише сто чланова који су реални бројеви, резервише се меморија за чување сто реалних бројева, а у програму можемо користити, рецимо, само првих 15 елемената низа

```
ARRAY['a'..'z'] of INTEGER
```

низ може имати највише 26 чланова који су целобројних и чији индекси су мала слова енглеске абецеде, резервише се меморија за чување 26 целих бројева

Службена реч ***array*** је обавезна код декларације низовног типа.

Ако се некој речи придружи декларација низовног типа онда она постаје нови, дефинисани (изведени) тип и може се употребљавати као да је стандардни тип (експлицитна дефиниција иза).

```
TYPE NIZ=ARRAY[1..50] of INTEGER;
{ дефинишемо нови тип података који се зове NIZ и представља низ који може имати
  највише педесет целобројних елемената }
     ZNAK=(pik,karo,herc,tref);
{ дефинишемо набројани тип са именом ZNAK са четири елемента који су наведени у
  загради }
     BROJ=1..14;
{ дефинишемо интервални тип са именом BROJ са четрнаест елемента који су дати у
  облику интервала }
     SPIL=ARRAY[znak] of broj;
{ дефинишемо нови тип података низ  SPIL и који може имати највише четири елемента
  који могу бити цели бројеви од 1 до 14 }
```

Када смо дефинисали нове типове података, помоћу њих можемо да дефинишемо променљиве које ћемо користити у програму на уобичајен начин:

```
VAR a:INTEGER;
    b:NIZ;
    karta:SPIL;
```

Све операције које су дозвољене у раду са промењивама типа којег су чланови низа дозвољене су и у раду са члановима тог низа.

Два низа се могу упоређивати ако су им једнаки типови. У том случају кажемо да су два низа једнака ако имају једнак број елемената и одговарајући чланови су им једнаки (први са првим, други са другим, итд).

Једнодимензионални низови.

Линеарно уређени низ једноиндексних променљивих чини једнодимензионални низ.

При декларацији сваког низа преводилац резервише потребну количина меморије за чување вредности свих елемената низа, без обзира да ли се они користе или не у програму, зато максималан број чланова једнодимензионалног низа (исто тако и сваког другог низа) зависи од расположиве количине радне меморије рачунара. У случају да нема довољно меморије за чување декларисаног низа добићемо једну од следећих порука:

```
"Error 22: Structure too large"  или
"Error 96: Too many variables"
```

Грешке се отклањају смањењем дужине низова или променом типа елемената низова или се другачијим алгоритмом број низова смањи и ослободи довољна количина меморије.

У делфију програм може користити целокупну слободну ***RAM*** меморију, па се овакве грешке неће јављати, али и овде важе нека ограничења, међутим у задацима са којима ћемо се сретати у *животу обичних смртника* нећемо имати проблема са недостатком меморије за чување вредности чланова низа.

Примери декларација једнодимензионалних низова

Декларација промењиве типа низ (тип низ је имплицитно дефинисан, нема своје име и не може се користити као тип података улазних и излазних параметара функција и процедура):

```
VAR x:ARRAY[1..20] of REAL;
```

једнодимензионални низ од највише двадесет реалних бројева.

```
VAR slovo:ARRAY[1..55] of CHAR;
```

једнодимензионални низ типа карактер са највише 55 елемената.

Експлицитно дефинисани тип низ и декларација промењивих типа низ помоћу њега (овако дефинисан тип низ може се користити за декларацију улазних и излазних параметара функција и процедура):

```
TYPE   rniz=ARRAY[1..20] of REAL;
                  cniz=ARRAY[1..55] of CHAR
VAR y:rniz;
```

једнодимензионални низ од највише двадесет реалних бројева.

```
VAR slovo:cniz;
```

једнодимензионални низ типа карактер са највише 55 елемената.

Промењиве ***x*** и ***y*** из претходних примера формално нису истог типа и не могу се упоређивати или замењивати у позивима функција и процедура.

Основне операције са низовима

- ***израчунавање просечне вредности***

Ако треба одредити просечну вредност елемената низа то значи да треба сабрати све елементе низа и поделити тај збир са бројем елемената низа. Као и увек, код рачунања збирова у циклусу, морамо за почетну вредност узети неки број који неће утицати на решење проблема, у овом случају то ће бити 0. Затим ћемо у циклусу ***for*** сабирати све елементе низа. На крају ћемо исписати тражени резултат.

```
zbir:=0;                // почетна вредност за збир елемената низа
For i:=1 to n do        // n је број елемената низа
begin // овде треба учитати или генерисати нови елемент низа ако низ није претходно формиран
      zbir:=zbir+a[i];  // нови елемент низа се сабира са свим претходним
end;
arsred:=zbir/n          // одређује се просечна вредност елемената низа, наравно,
                        // треба водити рачуна о декларисаним типовима променљивих
```

Нешто је већи проблем ако се низ уноси све до испуњења неког услова, односно, низ има највише 100 елемената (на пример, не мора бити 100, може бити било који унапред задати број), а елементи се уносе све док се не унесе 0 (када се унесе 0 то није елемент низа, већ то значи да су унети сви елементи низа). Тада би овај задатак морао да се решава помоћу циклуса са условом, најбоље са *while*.

```
zbir:=0;                        // почетна вредност за збир елемената низа
n:=0;                           // број елемената низа је 0 јер још није унет ни један
broj:=...                       // овде треба унети вредност помоћне променљиве broj
While (n<=100)and(broj<>0) do   // ако није унета 0 формира се неки елемент низа
begin n:=n+1;                   // редни број елемента низа увећава се за један
      a[n]:=broj;               // додаје се нови елемент низа
      zbir:=zbir+a[n];          // нови елемент низа се сабира са свим претходним
      broj:=...                 // овде треба унети нову вредност помоћне променљиве broj
end;
arsred:=zbir/n                  // одређује се просечна вредност елемената низа, наравно,
                                // треба водити рачуна о декларисаним типовима променљивих
```

- ***израчунавање минималне / максималне вредности низа***

Код одређивања максималне и минималне вредности низа користићемо две помоћне променљиве које ће на почетку имати вредност првог елемента низа, а затим ћемо их упоређивати са осталим елементима низа и кад год пронађемо елемент који је већи од прве променљиве доделићемо јој ту вредност, односно, када нађемо елемент низа који је мањи од друге променљиве доделићемо јој ту вредност.

```
min:=a[1];
max:=a[1];
For i:=2 to n do        // циклус креће од 2 јер нема потребе тестирати први елемент
      If a[i]>max
       then max:=a[i]   // ако је елемент низа већи од највећег, онда је он највећи
       else If a[i]<min
       then min:=a[i];  // ако је елемент низа мањи од најмањег, онда је он најмањи
```

За почетну вредност променљивих ***min*** и ***max*** не мора се узети први елемент низа, сасвим би исправно радио програм и ако бисмо узели било који елемент низа, али је овако једноставније и јасније. Оваква метода налажења екстремних вредности позната је као метода лажне претпоставке.

- ***претраживање у низу***

Претраживање у низу који није уређен своди се на претходни пример, разлика је само што не тражимо највећи или најмањи елемент низа већ елемент који задовољава неки услов, већи је или је мањи од неког задатог броја или је једнак неком броју. Програм би нам могао дати одговор да ли такав елемент постоји и колико је таквих елемената у низу.

Ако треба одредити редни број елемента у уређеном низу који испуњава неки услов онда можемо применити различите методе претраживања, али је најједноставније користити методу половљења. Проверавамо да ли је елемент на средини низа тражени, па ако није одређујемо у којој половини низа се он налази, делимо низ на пола и узимамо половину низа у којој се налази тражени елемент. Сада ту половину низа посматрамо као неки низ и понављамо претходни поступак. Овако организован програм би могао дати одговор да ли постоји одговарајући елемент и који је његов редни број, али и колико таквих елемената има.

- ***сортирање низа***

Код сортирања низа потребно је упоређивати узастопне елементе и наместити да свака два узастона елемента буду у одговарајућем поретку. Низ може бити уређен у растућем или опадајућем поретку. Но, ова два поретка, строго математички гледано, подразумевају да у низу нема истих елемената. Пошто ми унапред не знамо да ли ће у низу бити истих елемената или не, уместо појма растући користимо појам неопадајући, а уместо појма опадајући користимо појам нерастући.

Размотримо случај неопадајућег поретка (јер читањем неопадајућег низа од краја према почетку добијамо нерастуће поређане елементе низа). Да бисмо били сигурни да је низ уређен у неопадајући поредак није довољно једном проћи кроз низ и наместити да сваки други од два узастопна елемента буде већи или једнак првом јер не знамо какав је однос претходних елемената низа са мањим од ова два. Показаћемо то на простом примеру:

Нека је дат низ 2, 3, 1. Упоређиваћемо први са другим и други са трећим. Пошто је први мањи од другог ништа се не мења, а пошто је други већи од трећег заменићемо им места. Добили смо низ 2, 1, 3 који, очигледно, није неопадајући јер је први елемент низа већи од другог.

Зато, код уређивања низа морамо сваки елемент низа упоређивати са свима иза њега. То ћемо остварити двоструким циклусом. Први циклус ће почети од првог елемента низа и извршаваће се све док не стигнемо до претпоследњег, тј. узимаће први елемент у пару елемената низа. Други циклус почиње са првим следећим елементом и извршаваће се док не стигнемо до последњег, тј. редни број другог елемента у пару биће увек већи од редног броја првог елемента. Затим ћемо упоређивати ова два елемента, једног из првог циклуса и једног из другог, и кад год нам њихов редослед не одговара заменићемо им места. Практично ћемо то записати овако:

```
// нека је n укупан број елемената у низу
For i:=1 to n-1 do      // i је редни број првог у пару елемената низа који се упоређују
 For j:=i+1 to n do     // j је редни број другог у пару елемената низа
  If a[i]>a[j] then     // проверавамо да ли је нарушен поредак елемената низа
  begin                 // ако јесте елементима мењамо места
        p:=a[i];        // вредност првог елемента чувамо у помоћној променљивој p
        a[i]:=a[j];     // први елемент узима вредност другог елемента низа
        a[j]:=p;        // други елеменат узима вредност првог сачувану у помоћној
  end;
```

Уређивање у нерастући поредак остварило би се на исти начин, једино би се уместо релације *веће од* користила релација *мање од*:

```
// нека је n укупан број елемената у низу
For i:=1 to n-1 do      // i је редни број првог у пару елемената низа који се упоређују
 For j:=i+1 to n do     // j је редни број другог у пару елемената низа
  If a[i]<a[j] then     // проверавамо да ли је нарушен поредак елемената низа
  begin                 // ако јесте елементима мењамо места
        p:=a[i];        // вредност првог елемента чувамо у помоћној променљивој p
        a[i]:=a[j];     // први елемент узима вредност другог елемента низа
        a[j]:=p;        // други елеменат узима вредност првог сачувану у помоћној
  end;
```

Постоје и другачији алгоритми за сортирање низа. Овај је најефикаснији када низ има мањи број елемената. У случају сортирања низа са јако великим бројем елемената треба изабрати неки други алгоритам јер је овај прилично спор.

Дводимензионални низови

Дводимензионалним низом зовемо низ двоиндексних променљивих. Ова врста низова обично се назива матрица или таблица, хоризонтални елементи чине редове, а вертикални колоне матрице. Индекси низа имају своје границе, али оне не морају бити једнаке. Ако су границе једнаке матрицу зовемо квадратном. Елементи од горњег левог до доњег десног у квадратној матрици чине главну, а елементи од горњег десног до доњег левог споредну дијагоналу. Типови индекса не морају да буду једнаки, тј. један индекс може бити цео број, а други, на пример, типа карактер. Укупан број декларисаних променљивих, за које се резервише меморија, код дводимензионалног низа добија се када се највећи редни бројеви оба индекса помноже.

```
TYPE MATRICA=ARRAY [1..10,1..20] of INTEGER;
```

дефинише се тип дводимензионални низ са 10 * 20 = 200 целобројних чланова.

```
TYPE FIGURA=(prazno,pesak,top,konj,lovac,kralj,dama);
     SAHTABLA=ARRAY['a'..'h',1..8] of FIGURA;
VAR  mesto:SAHTABLA;
```

дефинишу се типови фигура и шахтабла и декларише се дводимензионални низ са 8 * 8 = 64 елемената типа фигура.

Аналогно би се могао дефинисати низ са више индекса, на пример троиндексни низ је низ троиндексних променљивих, итд. Начин рада са таквим, вишедимензионалним низовима, би био аналоган раду са једнодимензионалним или дводимензионалним низовима. Низ се најпре декларише, члановима се доделе неке вредности и обави се задатак због којег је низ формиран. Но, мора се признати да се врло ретко јавља потреба за низовима са више од три димензије (индекса), али добро је знати да је и то могуће. Наравно са повећањем броја димензија низа повећава се потреба за меморијом, па се смањује максимална, реално могућа, дужина таквих низова.

Алгоритми за израчунавања и трансформације на табели и њеним деловима.

- ***Збир два дводимензионална низа***

Најједноставнија математичка операција са два дводимензионална низа је одређивање њиховог збира. Збир је, такође, дводимнезионални низ чији се елементи добијају сабирањем одговарајућих елемената оба низа. Да би се два низа могла сабирати морају имати исте димензије, односно, број редова и колона једног мора бити једнак броју редова и колона другог низа (не морају бити квадратни као на слици).

$$\begin{bmatrix}1 & 3 & 2\\ 1 & 0 & 0\\ 1 & 2 & 2\end{bmatrix}+\begin{bmatrix}0 & 0 & 5\\ 7 & 5 & 0\\ 2 & 1 & 1\end{bmatrix}=\begin{bmatrix}1+0 & 3+0 & 2+5\\ 1+7 & 0+5 & 0+0\\ 1+2 & 2+1 & 2+1\end{bmatrix}=\begin{bmatrix}1 & 3 & 7\\ 8 & 5 & 0\\ 3 & 3 & 3\end{bmatrix}$$

Алгоритам за сабирање низова ћемо записати на следећи начин:

```
For i:=1 to n do
 For j:=1 to m do
  c[i,j]:=a[i,j]+b[i,j];
```

- ***Производ два дводимензионална низа***

Да би се два дводимензионална низа могла помножити обавезно мора број колона првог бити једнак броју редова другог низа (број редова првог и број колона другог дводимензионалног низа не морају бити једнаки). Производ два дводимензионална низа димензија *NxK* и *KxM* је дводимензионални низ димензије *NxM* чији се елементи добијају као збир производа свих елемената одговарајуће колоне првог низа и елемената одговарајућег реда другог низа.

$$\begin{bmatrix}1 & 0 & 2\\ -1 & 3 & 1\end{bmatrix}\cdot\begin{bmatrix}3 & 1\\ 2 & 1\\ 1 & 0\end{bmatrix}=\begin{bmatrix}(1\cdot 3+0\cdot 2+2\cdot 1) & (1\cdot 1+0\cdot 1+2\cdot 0)\\ ((-1)\cdot 3+3\cdot 2+1\cdot 1) & ((-1)\cdot 1+3\cdot 1+1\cdot 0)\end{bmatrix}$$

Алгоритам за множење низова ћемо записати на следећи начин (***n*** - број редова првог, ***m*** - број колона другог низа):

```
For i:=1 to n do
 For j:=1 to m do
 begin c[i,j]:=0;
       For k:=1 to p do
        c[i,j]:=c[i,j]+a[i,k]*b[k,j];
 end;
```

- ***Додавање колоне или реда***

Нека треба додати један ред дводимензионалном низу димензија N x M, после његовог к-тог реда. То значи да ћемо имати нови низ са димензијама (N+1) x M. Додавање реда ћемо остварити тако што ћемо редове од *k+1* до *n* преписати у редове *k+2* do *n+1*, а на место *k+1* ћемо уметнути нови ред низа. Ради једноставнијег записа алгоритма, посматраћемо ред који треба уметнути као једнодимензионални низ.

Алгоритам за додавање реда ћемо записати на следећи начин:

```
For j:=1 to m do
begin For i:=n+1 downto k+2 do
       a[i,j]:=a[i-1,j];
      a[k+1,j]:=b[j];
end;
n:=n+1;
```

Алгоритам за додавање колоне ћемо записати на следећи начин:

```
For i:=1 to n do
begin For j:=m+1 downto k+2 do
       a[i,j]:=a[i,j-1];
      a[i,k+1]:=b[i];
end;
n:=n+1;
```

- ***Брисање колоне или реда***

Нека треба обрисати један ред дводимензионалном низу димензија N x M, на пример к-ти ред. То значи да ћемо имати нови низ са димензијама (N-1) x M. Брисање реда ћемо остварити тако што ћемо редове од *k+1* до *n* преписати у редове *k* do *n-1*, а затим прву променљиву (N) смањити за један. Брисање колоне ћемо остварити тако што ћемо колоне од *k+1* до *m* преписати у колоне *k* do *m-1*, а затим другу променљиву (M) смањити за један.

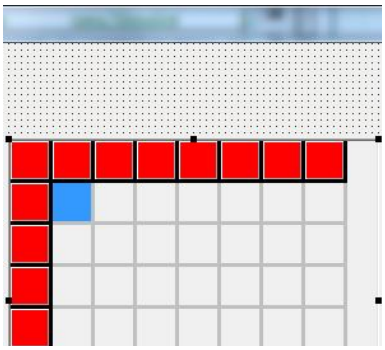
Алгоритам за брисање реда ћемо записати на следећи начин:

```
For j:=1 to m do
 For i:=k to n-1 do
  a[i,j]:=a[i+1,j];
n:=n-1;
```

Алгоритам за брисање колоне ћемо записати на следећи начин:

```
For i:=1 to n do
 For j:=k to m-1 do
  a[i,j]:=a[i,j+1];
m:=m-1;
```

Визуелна компонента за табеларни приказ текста.



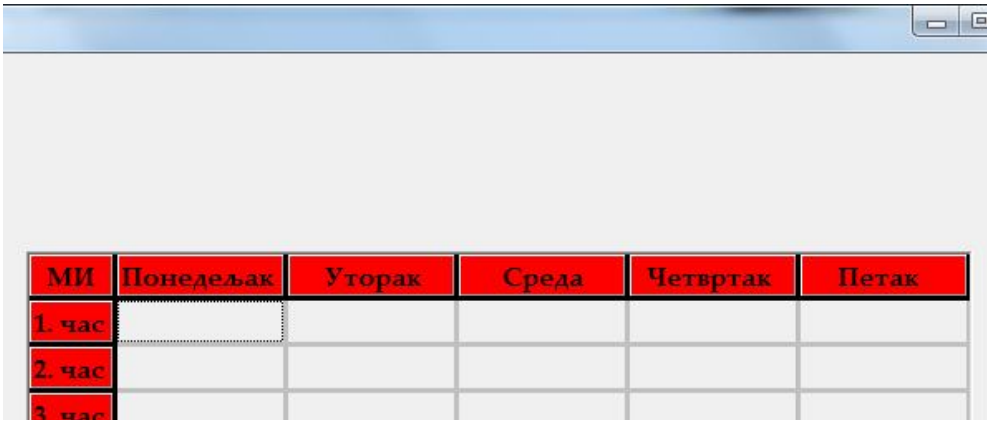
За приказивање елемената дводимензионалног низа може се користити мемо поље, при чему морамо форматизовати сваки елемент и сваки ред низа како бисмо добили табелу или можемо користити компоненту ***StringGrid***. Црвено обојена поља на слици су фиксна, тј. у фази извршавања програма кориснику није дозвољен приступ овим пољима. Карактеристике које би требало подесити су:

Align - положај грида у односу на рубове форме
ColCount - број колона
Color - боја позадине
DefaultColWidth - предефинисана ширина колона
DefaultRowHeight - предефинисана висина редова
Enabled - доступност активностима корисника
FixedColor - боја фиксних поља
FixedCols - број фиксних колона
FixedRows - број фиксних редова
Font - врста фонта којим се попуњава грид
GridLineWidth - дебљина линија које ограничавају поља грида
Height - висина грида
Left - растојање од левог руба форме
Name - име грида
RowCount - број редова грида
ScrollBars - да ли се приказују хоризонтални и вертикални клизач ако сва поља грида нису видљива
Top - растојање од врха форме
Visible - да ли је видљив грид
Width - ширина грида

Ако желимо да дводимензионални низ буде исписан у гриду, онда морамо знати да је први аргумент грида редни број колоне, а други реда (супротно од дводимензионалног низа). Попуњавање остварујемо са:

```
For i:=1 to 6 do
      For j:=1 to 7 do
            StringGrid1.Cells[j,i]:=IntToStr(a[i,j]);
```

Компонента ***StringGrid*** се може користити за табеларно приказивање текста. У том случају могу се користити и фиксни ред и фиксна колона да би се дефинисало заглавље табеле. Редни бројеви и редова и колона почињу од нуле. Подаци у пољима грида се исписују лево оријентисано, па, ако нам то не одговара, морамо форматизовати исписивање садржаја. На слици је приказан образац распореда часова.

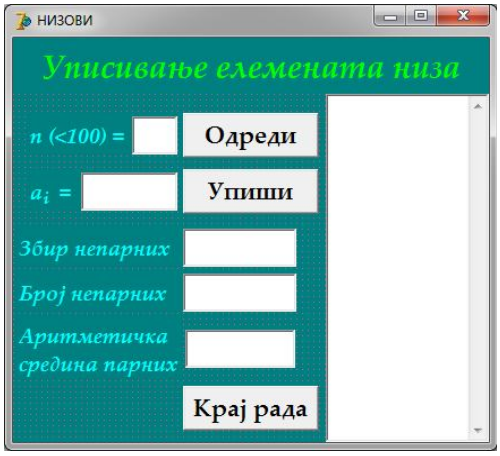


Задаци са низовима

Претходни задаци решени су без коришћења типа низ и предности које он доноси. Следе задаци који ће бити решавани са променљивама типа низ. Формирање низа може се остварити на три начина: директним уписом преко тастатуре, генерисањем бројева помоћу функције ***Random*** или преузимањем података из датотеке (мада се и та датотека мора некако попунити). У зависности од изабраног начина формирања низа форма може бити дизајнирана на два начина: са делом за унос елемената низа или без тог дела. Урадићемо први задатак са уносом елемената помоћу тастатуре, а све остале ћемо решавати генерисањем елемената низа.

- ***Саставити програм који формира низ са највише 100 троцифрених бројева, а затим одређује број и збир непарних и аритметичку средину парних елемената низа***

За унос елемената помоћу тастатуре морамо предвидети: едит за унос дужине низа и едит за унос елемената низа и тастере за потврду оба уноса. Задатак ће бити решен онако како би кориснику апликације било најједноставније. Креираћемо форму као на слици, а затим написати комплетан програм. Најпре ћемо дефинисати нови тип података - ***низ*** у делу за дефиниције типова (на самом почетку јунита - глобални тип):




```
type
  niz=array[1..100] of integer;
  TForm1 = class(TForm)
       ...
```

Затим ћемо декларисати глобалне променљиве ***n*** - број елемената низа и ***k*** - бројач елемената низа, које су цели бројеви и ***a*** - променљива типа низ у делу за декларације:

```
var
  Form1: TForm1;
  n,k:integer;
  a:niz;
```

Да бисмо имали потпуну контролу над програмом, поставићемо следеће карактеристике објеката на форми: *Edit2.**Enabled** - false, Button2.**Enabled** - false, Edit3.**ReadOnly** - true, Edit4.**ReadOnly** - true, Edit5.**ReadOnly** - true, Memo1.**ReadOnly** - true*. Да не бисмо имали четири тастера на форми, уместо тастера за брисање написаћемо процедуру која ће брисати претходни унос низа када се курсор постави у први едит.

```
procedure TForm1.Edit1Enter(Sender: TObject);
begin Memo1.Clear;Edit1.Clear;
      Edit2.Clear;Edit3.Clear;
      Edit4.Clear;Edit5.Clear;
end;
```

Забранићемо кориснику да уноси слова и знакове у прва два едита.

```
procedure TForm1.Edit1KeyPress(Sender: TObject; var Key: Char);
begin If key=#13
       then If sender=edit1
            then Button1.Click
            else Button2.Click
       else If not (key in ['0'..'9',#8,#128])
            then key:=#27;
end;
```

Када се унесе дужина низа глобалне променљиве ***n*** и ***k*** добиће одговарајуће вредности.

```
procedure TForm1.Button1Click(Sender: TObject);
begin n:=StrToIntDef(Edit1.Text,10);
      If n>100                 // забрањујемо унос више од 100 елемената низа због дефинције низа
       then n:=100;
      Edit1.Text:=IntToStr(n);
      Edit1.Enabled:=false;    // забрањује се приступ едиту 1
      Edit2.Enabled:=true;     // дозвољава се приступ едиту 2
      Button1.Enabled:=false; // забрањује се употреба тастера 1
      Button2.Enabled:=true;  // дозвољава се употреба тастера 2
      k:=0;                    // бројач елемената низа постављамо на 0
      Edit2.SetFocus;          // курсор постављамо у други едит
end;
```

Сада се уносе елеменати низа помоћу тастатуре, уносе се у мемо поље, одређују се тражени резултати и исписују у предвиђене едите.

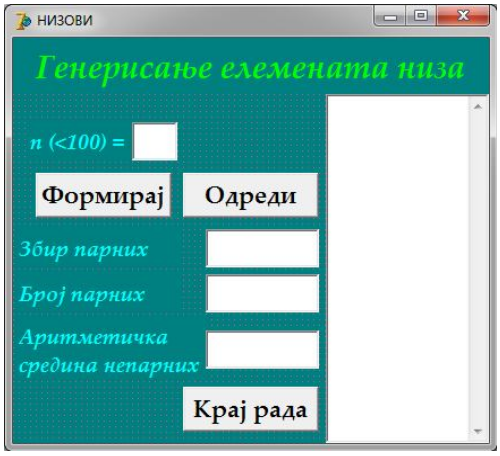
```
procedure TForm1.Button2Click(Sender: TObject);
var par,zp,zn:integer;
begin k:=k+1;
      a[k]:=StrToIntDef(Edit2.Text,100);
      If a[k]<100              // забрањујемо унос једноцифрених и двоцифрених бројева
       then a[k]:=100;
```

```
      Edit2.Text:=IntToStr(a[k]);
      Memo1.Lines.Add(IntToStr(k)+'. '+Edit2.Text);     // исписујемо елемент низа у мемо поље
      If k<n                                            // припремамо унос следећег елемента
       then begin Edit2.Clear;Edit2.SetFocus;
            end
       else begin Edit2.Enabled:=false;                 // забрањује се приступ едиту 2
                  Button2.Enabled:=false;               // забрањује се употреба тастера 2
                  par:=0;zp:=0;zn:=0;                   // почетне вредности бројача и збирова
                  For k:=1 to n do
                   If Odd(a[k])
                    then zn:=zn+a[k]                    // сабирамо непарне елементе низа
                   else begin par:=par+1;               // бројимо парне елементе
                              zp:=zp+a[k];              // сабирамо парне елементе
                        end;
                  Edit3.Text:=IntToStr(zn);
                  Edit4.Text:=IntToStr(n-par);          // број непарних елемената
                  If par>0                              // аритметичку средину тражимо ако има парних
                   then Edit4.Text:=Format('%1.5f',[zp/par])
                   else Edit4.Text:='0';
                  Button1.Enabled:=true;                // дозвољава се употреба тастера 1
                  Edit1.Enabled:=true;                  // дозвољава се приступ едиту 1
                  Button3.SetFocus;                     // жижу постављамо на тастер крај рада
            end;
end;
```

Још само недостаје процедура за крај рада.

```
procedure TForm1.Button3Click(Sender: TObject);
begin Application.Terminate;
end;
```

- ***Саставити програм који генерише низ са највише 100 троцифрених бројева, а затим одређује збир и број парних и аритметичку средину непарних елемената низа***



Задатак је готово идентичан претходном, само што ћемо овде генерисати троцифрене бројеве помоћу *Random* функције. Креираћемо форму као на слици, а затим написати комплетан програм. Најпре ћемо дефинисати нови тип података - ***низ*** у делу за дефиниције типова (на самом почетку јунита - глобални тип):

```
type
  niz=array[1..100] of integer;
  TForm1 = class(TForm)
       ...
```

Затим ћемо декларисати глобалне променљиве ***n*** - број елемената низа и ***a*** - променљива типа низ у делу за декларације:

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

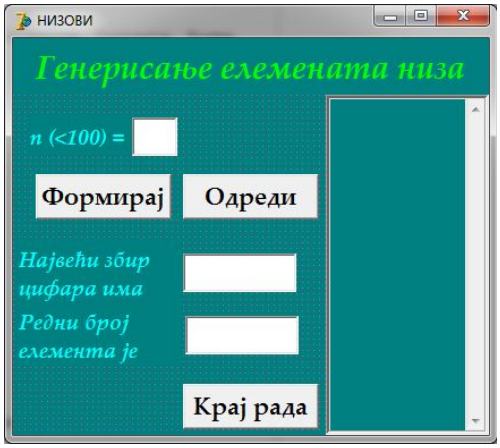
Уносимо само број елемената низа, па су сви, осим тог едита заштићени од уноса. Можемо додати тастер за брисање или задржати процедуру за брисање из претходног задатка.

```
procedure TForm1.Edit1Enter(Sender: TObject);
begin Memo1.Clear;Edit1.Clear;
      Edit2.Clear;Edit3.Clear;
end;
procedure TForm1.Edit1KeyPress(Sender: TObject; var Key: Char);
begin If key=#13
       then Button1.Click
       else If not (key in ['0'..'9',#8,#128])
             then key:=#27;
end;
procedure TForm1.Button1Click(Sender: TObject);
var k:integer;
begin n:=StrToIntDef(Edit1.Text,10);
      If n>100
       then n:=100;
      Edit1.Text:=IntToStr(n);
      Edit1.Enabled:=false;
      Button1.Enabled:=false;
      Randomize;
      For k:=1 to n do
      begin a[k]:=Random(900)+100;
            Memo1.Lines.Add(IntToStr(k)+'. '+IntToStr(a[k]));
      end;
      Button2.SetFocus;
end;
procedure TForm1.Button2.Click(Sender: TObject);
```

```
var nep,zp,zn,k:integer;
begin par:=0;
      zp:=0;zn:=0;
      For k:=1 to n do
        If Odd(a[k])
         then zp:=zp+a[k]
         else begin nep:=nep+1;
                    zn:=zn+a[i]
              end;
      Edit2.Text:=IntToStr(zp);
      Edit3.Text:=IntToStr(n-nep);
      If par>0
       then Edit4.Text:=Format('%1.5f',[zn/nep])
       else Edit4.Text:='0';
      Button1.Enabled:=true;
      Edit1.Enabled:=true;
      Button3.SetFocus;
end;
procedure TForm1.Button3Click(Sender: TObject);
begin Application.Terminate;
end;
```

- ***Саставити програм који формира низ од 100 природних бројева, а затим одређује редни број елемента који има највећи (најмањи) збир цифара, у случају да више елемената има исти збир цифара исписује се најмањи (највећи).***

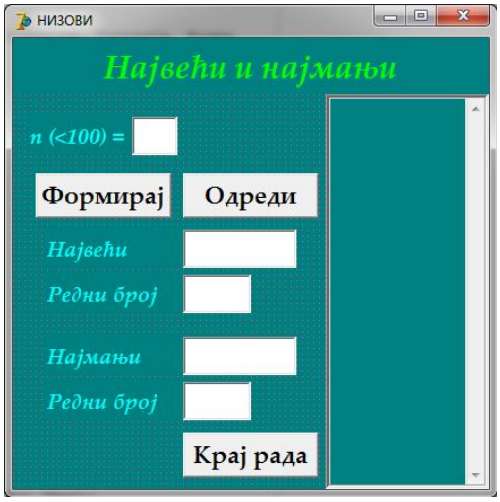
Креираћемо форму као на слици, а затим написати комплетан програм. Најпре ћемо дефинисати нови тип података - ***низ*** у делу за дефиниције типова, затим ћемо декларисати глобалне промењиве ***n*** - број елемената низа и ***a*** - промењива типа низ у делу за декларације.




```
type
  niz=array[1..100] of integer;
  TForm1 = class(TForm)
       ...
var
  Form1: TForm1;
  n:integer = 0;
  a:niz;
procedure TForm1.Button3Click(Sender: TObject);
begin Application.Terminate;
end;
procedure TForm1.Edit1Change(Sender: TObject);
begin Memo1.Clear;
      Edit2.Clear;Edit3.Clear;
end;
procedure TForm1.Edit1KeyPress(Sender: TObject; var Key: Char);
begin If key=#13
       then Button1.Click
       else If not (key in ['0'..'9',#8,#128])
        then key:=#27;
end;
procedure TForm1.Button1Click(Sender: TObject);
begin n:=StrToIntDef(Edit1.Text,10);
      If n>100 then n:=100;
      Edit1.Enabled:=false;
      Edit1.Text:=IntToStr(n);
      FormirajNiz;
      Button1.Enabled:=false;
      Button2.Enabled:=true;
      Button2.SetFocus;
end;
procedure TForm1.FormirajNiz;
begin k:=0;Randomize;
      While k<n do
      begin k:=k+1;a[k]:=Random(900)+100;
            Memo1.Lines.Add(Format('%2d.%4d',[k,a[k]]));
      end;
end;
procedure TForm1.Button2Click(Sender: TObject);
begin Button2.Enabled:=false;
      Odredi(k);
      Edit2.Text:=IntToStr(a[k]);
      Edit3.Text:=IntToStr(k);
      Button1.Enabled:=true;
      Edit1.Enabled:=true;
      Edit1.SetFocus;
end;
```

```
procedure TForm1.Odredi(VAR j:integer);
var i,nz,zc:integer;
begin j:=1;
      nz:=ZbirCifara(a[j]);
      For i:=2 to n do
       begin zc:=ZbirCifara(a[i]);
             If (nz<zc)or(nz=zc)and(a[j]>a[i]) // највећи збир цифара
                // (nz>zc)or(nz=zc)and(a[j]<a[i]) ако желимо да одредимо најмањи збир цифара
              then begin nz:=zc;
                         j:=i;
                   end;
      end;
end;
function TForm1.ZbirCifara(a:integer):integer; // рекурзивна процедура за збир цифара
begin If a=0
       then zbircifara:=0
       else zbircifara:=ZbirCifara(a div 10)+a mod 10;
end;
function TForm1.ZbirCifara(a:integer):integer; // процедура за збир цифара која није рекурзивна
var z:integer;
begin z:=0;
      Repeat z:=z+a mod 10;
             a:=a div 10;
      until a=0;
      zbircifara:=z;
end;
```

- ***Саставити програм који формира низ од 100 природних бројева, а затим одређује редни број највећег и најмањег елемента, у случају да има више једнаких елемената исписује се елемент са најмањим редним бројем.***



Тражење екстремних вредности је једна од важнијих процедура са низовима. Креираћемо форму као на слици, а затим написати комплетан програм.

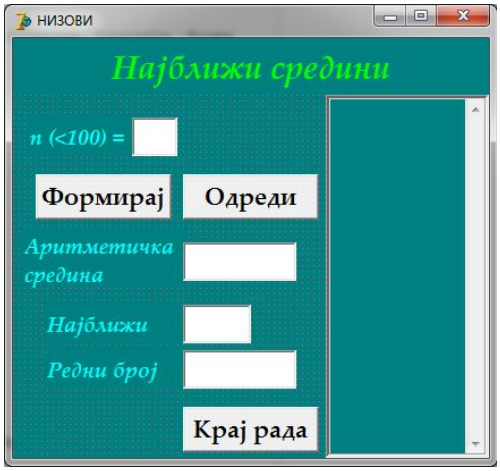
```
type
  niz=array[1..100] of integer;
  TForm1 = class(TForm)
       ...
var
  Form1: TForm1;
  n:integer = 0;
  a:niz;
procedure TForm1.Edit1Change(Sender: TObject);
begin Memo1.Clear;
      Edit2.Clear;Edit3.Clear;
      Edit4.Clear;Edit5.Clear;
end;
procedure TForm1.Button3Click(Sender: TObject);
begin Application.Terminate;
end;
procedure TForm1.Edit1KeyPress(Sender: TObject; var Key: Char);
begin If key=#13
       then Button1.Click
       else If not (key in ['0'..'9',#8,#128])
        then key:=#27;
end;
procedure TForm1.FormirajNiz;
var k:integer;
begin Randomize;
      k:=0;
      While k<n do
      begin k:=k+1;
            a[k]:=Random(1000);     // уместо 1000 могао је бити било који други број
            Memo1.Lines.Add(Format('%2d.%4d',[k,a[k]]));
      end;
end;
procedure TForm1.Button1Click(Sender: TObject);
begin n:=StrToIntDef(Edit1.Text,10);
      If n>100
       then n:=100;
      Edit1.Text:=IntToStr(n);
      FormirajNiz;
      Edit1.Enabled:=false;
      Button1.Enabled:=false;
      Button2.Enabled:=true;
      Button2.SetFocus;
end;
```

```
procedure TForm1.Button2Click(Sender: TObject);
var k:integer;
begin Button2.Enabled:=false;
      Odredi(k,true);
      Edit2.Text:=IntToStr(a[k]);
      Edit3.Text:=IntToStr(k);
      Odredi(k,false);
      Edit4.Text:=IntToStr(a[k]);
      Edit5.Text:=IntToStr(k);
      Button1.Enabled:=true;
      Edit1.Enabled:=true;
      Edit1.SetFocus;
end;
procedure TForm1.Odredi(VAR j:integer; m:boolean);
var i:integer;
begin j:=1
      For i:=2 to n do
       If m and (a[i]>a[j]) or
          not m and (a[i]<a[j])
        then j:=i;
end;
```

Параметар м, логичка променљива, одређује да ли процедура *Одреди* тражи највећи (*m:=true*) или најмањи (*m:=false*) елемент низа. Променљива не мора бити логичка, могла је бити и целобројна, где би 1 било тражење највећег, а било шта друго најмањег, а могла је бити и било ког другог типа. Овако троши најмање меморије и то је био једини разлог да буде логичког типа.

- ***Саставити програм који формира низ од највише 100 троцифрених бројева, а затим одређује аритметичку средину низа и редни број елемента низа који јој је најближи, у случају да има више елемената исписује се већи елемент са највећим редним бројем.***

Креираћемо форму као на слици, а затим написати комплетан програм.




```
type
  niz=array[1..100] of integer;
  TForm1 = class(TForm)
       ...

var
  Form1: TForm1;
  n:integer = 0;
  a:niz;

procedure TForm1.Edit1Change(Sender: TObject);
begin Memo1.Clear;
      Edit2.Clear;Edit3.Clear;
      Edit4.Clear;Edit5.Clear;
end;
procedure TForm1.Button3Click(Sender: TObject);
begin Application.Terminate;
end;
procedure TForm1.Edit1KeyPress(Sender: TObject; var Key: Char);
begin If key=#13
       then Button1.Click
       else If not (key in ['0'..'9',#8,#128])
        then key:=#27;
end;
procedure TForm1.FormirajNiz;
var k:integer;
begin Randomize;
      k:=0;
      While k<n do
      begin k:=k+1;
            a[k]:=Random(900)+100;
            Memo1.Lines.Add(Format('%2d.%4d',[k,a[k]]));
      end;
end;
procedure TForm1.Button1Click(Sender: TObject);
begin n:=StrToIntDef(Edit1.Text,10);
      If n>100
       then n:=100;
      Edit1.Text:=IntToStr(n);
      FormirajNiz;
      Edit1.Enabled:=false;
      Button1.Enabled:=false;
      Button2.Enabled:=true;
      Button2.SetFocus;
end;
```

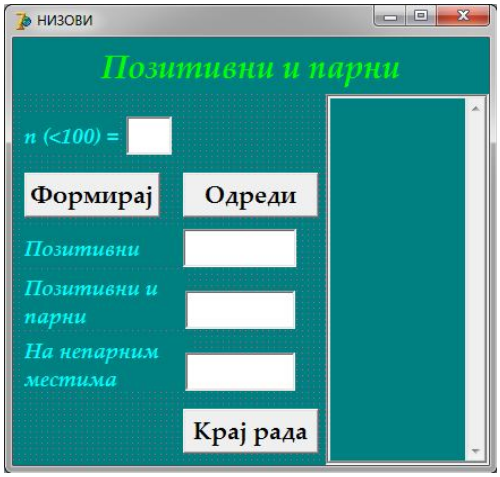
```
procedure TForm1.Button2Click(Sender: TObject);
var ars:real;
begin Button2.Enabled:=false;
      Odredi(ars,k);
      Edit2.Text:=Format('%1.3f',[ars]);
      Edit3.Text:=IntToStr(a[k]);
      Edit4.Text:=IntToStr(k);
      Button1.Enabled:=true;
      Edit1.Enabled:=true;
      Edit1.SetFocus;
end;
procedure TForm1.Odredi(VAR ars:real; VAR k:integer);
var i,k:integer;
begin ars:=0;
      For i:=1 to n do
       ars:=ars+a[i]/n;
      k:=1;
      For i:=2 to n do
       If (Abs(a[i]-ars)<Abs(a[k]-ars)) or
          (Abs(a[i]-ars)=Abs(a[k]-ars)) and (a[i]>a[k])
        then k:=i;
end;
```

Израчунавање аритметичке средине низа могла је бити и посебна функција;

```
function TForm1.ASred:real;              // нема аргумената јер су n и a глобалне променљиве
var i:integer;
    s:integer;
begin s:=0;
      For i:=1 to n do
       s:=s+a[i];
      asred:=s/n;
end;
```

- ***Саставити програм који формира низ од највише 100 целих бројева, а затим одређује број парних и позитивних који имају непаран редни број.***

Креираћемо форму као на слици, а затим написати комплетан програм.

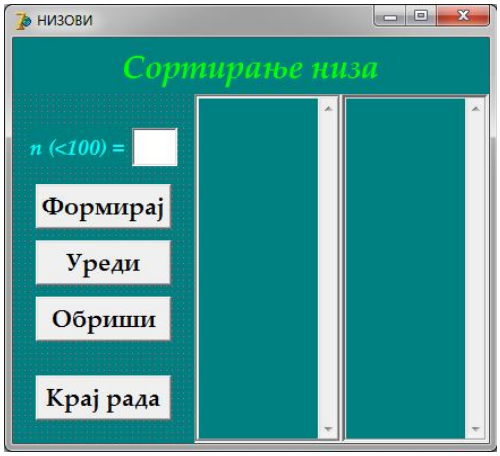



```
type
  niz=array[1..100] of integer;
  TForm1 = class(TForm)
      ...
var
  Form1: TForm1;
  n:integer = 0;
  a:niz;
procedure TForm1.Edit1Change(Sender: TObject);
begin Memo1.Clear;
      Edit2.Clear;Edit3.Clear;
      Edit4.Clear;Edit5.Clear;
end;
procedure TForm1.Button3Click(Sender: TObject);
begin Application.Terminate;
end;
procedure TForm1.Edit1KeyPress(Sender: TObject; var Key: Char);
begin If key=#13
       then Button1.Click
       else If not (key in ['0'..'9',#8,#128])
        then key:=#27;
end;
procedure TForm1.FormirajNiz;
var k:integer;
begin k:=0;Randomize;
      While k<n do
      begin k:=k+1;
            a[k]:=Random(2001)-1000;            // формира бројеве од -1000 до 1000
            Memo1.Lines.Add(Format('%2d.%4d',[k,a[k]]));
      end;
end;
procedure TForm1.Button1Click(Sender: TObject);
begin n:=StrToIntDef(Edit1.Text,10);
      If n>100
       then n:=100;
      Edit1.Text:=IntToStr(n);
      FormirajNiz;
      Edit1.Enabled:=false;
      Button1.Enabled:=false;
      Button2.Enabled:=true;
      Button2.SetFocus;
end;
```

```
procedure TForm1.Button2Click(Sender: TObject);
var ars:real;
    k:integer;
begin Button2.Enabled:=false;
      Odredi(ars,k);
      Edit2.Text:=Format('%1.3f',[ars]);
      Edit3.Text:=IntToStr(a[k]);
      Edit4.Text:=IntToStr(k);
      Button1.Enabled:=true;
      Edit1.Enabled:=true;
      Edit1.SetFocus;
end;
procedure TForm1.Odredi(VAR p,pp,ppn:integer);
var i:integer;
begin p:=0;       // број позитивних
      pp:=0;      // број парних позитивних
      ppn:=0;     // број парних позитивних на непарним местима
      For i:=1 to n do
       If a[i]>0
        then begin p:=p+1;
                   If not Odd(a[i])
                    then begin pp:=pp+1;
                               If Odd(i)
                                then ppn:=ppn+1;
                         end;
             end;
end;
```

- ***Саставити програм који формира низ од највише 100 троцифрених бројева, а затим га уређује у неопадајући или нерастући поредак.***

Креираћемо форму као на слици, а затим написати комплетан програм.




```
type
  niz=array[1..100] of integer;
  TForm1 = class(TForm)
      ...
var
  Form1: TForm1;
  n:integer = 10;
  k:integer = 0;
  a:niz;
procedure TForm1.Button4Click(Sender: TObject);
begin Application.Terminate;
end;
procedure TForm1.Button3Click(Sender: TObject);
begin Memo1.Clear;Memo2.Clear;
      Edit1.Clear;Edit1.Enabled:=true;
      Button1.Enabled:=true;
      Button2.Enabled:=false;
      Edit1.SetFocus;
end;
procedure TForm1.Edit1KeyPress(Sender: TObject; var Key: Char);
begin If key=#13
       then Button1.Click
       else If not (key in ['0'..'9',#8,#128])
             then key:=#27;
end;
procedure TForm1.Button1Click(Sender: TObject);
begin n:=StrToIntDef(Edit1.Text,10);
      If n>100
       then n:=100;
      Edit1.Text:=IntToStr(n);
      FormirajNiz;
      k:=0;
      Edit1.Enabled:=false;
      Button1.Enabled:=false;
      Button2.Enabled:=true;
      Button2.SetFocus;
end;
procedure TForm1.FormirajNiz;
begin Randomize;
      k:=0;
      While k<n do
      begin k:=k+1;
            a[k]:=Random(900)+100;
            Memo1.Lines.Add(Format('%2d.%4d',[k,a[k]]));
      end;
end;
```

```
procedure TForm1.Button2Click(Sender: TObject);
begin k:=1-k;
      UrediNiz(k);
      IspisiNiz;
end;
procedure TForm1.UrediNiz(k:integer);
var i,j,p:integer;
begin For i:=1 to n-1 do
       For j:=i+1 to n do
        If (a[i]>a[j])and(k=1) or              // за уређење у неопадајући поредак
           (a[i]<a[j])and(k=0)                 // за уређење у нерастући поредак
         then begin p:=a[i];a[i]:=a[j];a[j]:=p;
              end;
end;
procedure TForm1.IspisiNiz;
var i:integer;
begin Memo2.Clear;
      For i:=1 to n do
       Memo2.Lines.Add(Format('%2d.%4d',[i,a[i]]));
end;
```

Уређење се мења сваки пут када се кликне тастер *Уреди*. Можда је једноставније решење са два тастера за уређење, један за растући, један за опадајући поредак, а клик на било који од њих би позивао исту процедуру *Button2Click*. У том случају требало би ред:

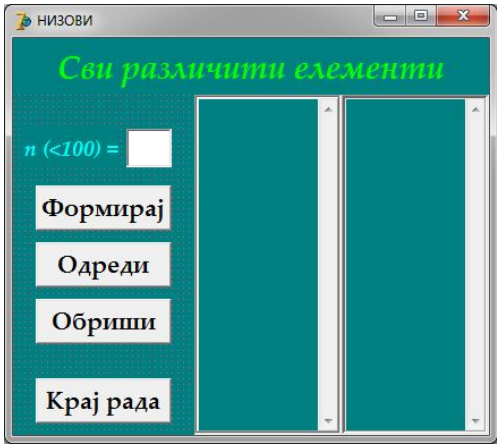
```
      k:=1-k;UrediNiz(k);
```

заменити једном оваквом *IF* наредбом:

```
      If sender=Button2
       then UrediNiz(1)
       else UrediNiz(0)
```

- ***Саставити програм који формира низ од највише 100 двоцифрених бројева, а затим од тог низа формира нови низ чији су сви елементи различити.***

Креираћемо форму као на слици, а затим написати комплетан програм.

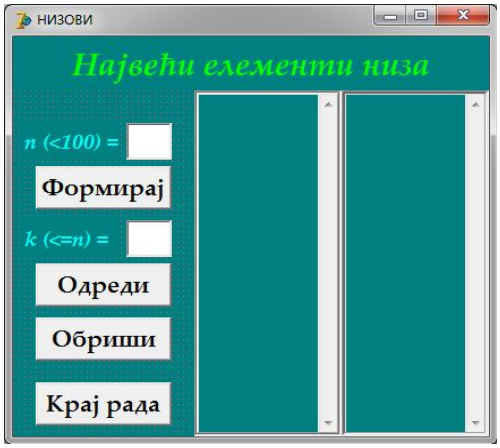



```
type
  niz=array[1..100] of integer;
  TForm1 = class(TForm)
       ...
var
  Form1: TForm1;
  n,k:integer;
  a,b:niz;
procedure TForm1.Button4Click(Sender: TObject);
begin Application.Terminate;
end;
procedure TForm1.Button3Click(Sender: TObject);
begin Memo1.Clear;Memo2.Clear;
      Button1.Enabled:=true;
      Edit1.Enabled:=true;
      Edit1.Clear;
      Edit1.SetFocus;
end;
procedure TForm1.Edit1KeyPress(Sender: TObject; var Key: Char);
begin If key=#13
       then Button1.Click
       else If not (key in ['0'..'9',#8,#128])
         then key:=#27;
end;
procedure TForm1.Button1Click(Sender: TObject);
begin n:=StrToIntDef(Edit1.Text,10);
      If n>100
       then n:=100;
      Edit1.Enabled:=false;
      Edit1.Text:=IntToStr(n);
      FormirajNiz;
      Button1.Enabled:=false;
      Button2.Enabled:=true;
      Button2.SetFocus;
end;
procedure TForm1.FormirajNiz;
begin Randomize;
      k:=0;
      While k<n do
      begin k:=k+1;a[k]:=Random(90)+10;
            Memo1.Lines.Add(Format('%2d.%4d',[k,a[k]]));
      end;
end;
```

```
procedure TForm1.Button2Click(Sender: TObject);
begin Razliciti;
      IspisiNiz;
      Button2.Enabled:=false;
      Button3.SetFocus;
end;
procedure TForm1.Razliciti;
var i,j:integer;
    dodaj:boolean;
begin k:=1;b[k]:=a[k];
      i:=2;
      While i<=n do
      begin j:=0;
            Repeat j:=j+1;dodaj:=a[i]=a[j];
            until (dodaj)or(j=i-1);
            If not dodaj
             then begin k:=k+1;b[k]:=a[i];
                        end;
            i:=i+1;
          end;
end;
procedure TForm1.IspisiNiz;
var i:integer;
begin Memo2.Clear;
      For i:=1 to k do
           Memo2.Lines.Add(Format('%2d.%4d',[i,b[i]]));
end;
```

- ***Саставити програм који формира низ од највише 100 природних бројева, а затим неуређујући низ исписује к највећих.***

Овде ћемо приказати једну другачију процедуру за испис елемената низа која као параметре има низ који се исписује, број елемената који се исписују и редни број мемо поља у које ће се исписати низ. Креираћемо форму као на слици, а затим написати комплетан програм.



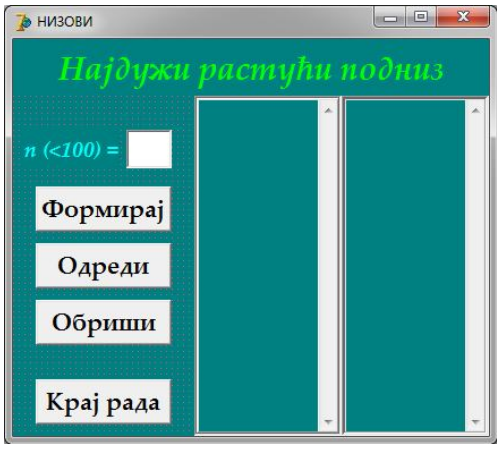

```
type
  niz=array[1..100] of integer;
  TForm1 = class(TForm)
       ...
var
  Form1: TForm1;
  n,k:integer;
  a,b:niz;
procedure TForm1.Button4Click(Sender: TObject);
begin Application.Terminate;
end;
procedure TForm1.Button3Click(Sender: TObject);
begin Memo1.Clear;Memo2.Clear;
      Button1.Enabled:=true;
      Edit1.Enabled:=true;
      Edit1.Clear;
      Edit1.SetFocus;
end;
procedure TForm1.Edit1KeyPress(Sender: TObject; var Key: Char);
begin If key=#13
       then If sender=edit1
             then Button1.Click
             else Button2.Click
       else If not (key in ['0'..'9',#8,#128])
         then key:=#27;
end;
procedure TForm1.Button1Click(Sender: TObject);
begin n:=StrToIntDef(Edit1.Text,10);
      If n>100
       then n:=100;
      Edit1.Text:=IntToStr(n);
      FormirajNiz;
      Edit1.Enabled:=false;
      Button1.Enabled:=false;
      Button2.Enabled:=true;
      Edit2.Enabled:=true;
      Edit2.SetFocus;
end;
procedure TForm1.FormirajNiz;
begin Randomize;
      k:=0;
      While k<n do
```

```
      begin k:=k+1;
            a[k]:=Random(900)+100;
      end;
      IspisiNiz(a,n,1);
end;
procedure TForm1.IspisiNiz(a:niz;n,k:integer);
var i:integer;
begin TMemo(FindComponent('Memo'+IntToStr(k))).Clear;   // празни мемо поље број k
      For i:=1 to n do
       TMemo(FindComponent('Memo'+IntToStr(k))).Lines.Add(Format('%2d.%4d',[i,a[i]]));
end;
procedure TForm1.Button2Click(Sender: TObject);
begin k:=StrToIntDef(Edit2.Text,10);
      If k>n
       then k:=n;
      Edit2.Text:=IntToStr(k);
      Najveci(k);
      IspisiNiz(b,k,2);
      Edit2.Enabled:=false;
      Button2.Enabled:=false;
      Button3.SetFocus;
end;
procedure TForm1.Najveci(k:integer);
var i,j,p:integer;
    dodaj:boolean;
begin For i:=1 to k do
       b[i]:=a[i];                          // првих k елемената низа a формира низ b
      For i:=k+1 to n do           // проверавамо да ли има већих у остатку низа a
      begin dodaj:=false;
            j:=1;p:=1;
            While j<=k do
            begin If a[i]>b[j]              // ако је већи елемент у остатку низа a
                   then begin dodaj:=true;      // памтимо да треба додати елемент
                              If b[j]<b[p]
                               then p:=j;                 // тражи се најмањи у b
                        end;
                  j:=j+1;
            end;
            If dodaj
             then begin For j:=p to k-1 do
                         b[j]:=b[j+1];                          // брише најмањи у b
                        b[k]:=a[i];       // додаје се нови елемент на крај низа b
                  end;
      end;
end;
```

Додавањем елемената на крај низа ***b*** учинили смо да се редослед ***k*** највећих елемената низа ***a*** не мења, тј. поредак елемената у низу ***b*** је исти као у низу ***a***.

- ***Саставити програм који формира низ од највише 100 троцифрених бројева, а затим из низа издваја најдужи растући подниз.***

Креираћемо форму као на слици, а затим написати комплетан програм.



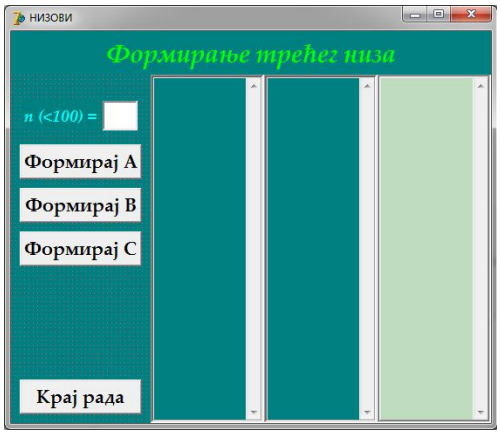

```
type
  niz=array[1..100] of integer;
  TForm1 = class(TForm)
      ...

var
  Form1: TForm1;
  n,k:integer;
  a,b:niz;

procedure TForm1.Button4Click(Sender: TObject);
begin Application.Terminate;
end;
procedure TForm1.Button3Click(Sender: TObject);
begin Memo1.Clear;Memo2.Clear;
      Button1.Enabled:=true;
      Edit1.Enabled:=true;
      Edit1.Clear;
      Edit1.SetFocus;
end;
procedure TForm1.Edit1KeyPress(Sender: TObject; var Key: Char);
begin If key=#13
       then Button1.Click
       else If not (key in ['0'..'9',#8,#128])
        then key:=#27;
end;
```

```
procedure TForm1.Button1Click(Sender: TObject);
begin n:=StrToIntDef(Edit1.Text,10);
      If n>100
       then n:=100;
      Edit1.Text:=IntToStr(n);
      FormirajNiz;
      Edit1.Enabled:=false;
      Button1.Enabled:=false;
      Button2.Enabled:=true;
      Button2.SetFocus;
end;
procedure TForm1.FormirajNiz;
begin Randomize;
      k:=0;
      While k<n do
      begin k:=k+1;a[k]:=Random(900)+100;
      end;
      IspisiNiz(a,n,1);
end;
procedure TForm1.IspisiNiz(a:niz;n,k:integer);
var i:integer;
begin TMemo(FindComponent('Memo'+IntToStr(k))).Clear;   // празни мемо поље број k
      For i:=1 to n do
       TMemo(FindComponent('Memo'+IntToStr(k))).Lines.Add(Format('%2d.%4d',[i,a[i]]));
end;
procedure TForm1.Button2Click(Sender: TObject);
begin Button2.Enabled:=false;
      OdrediPodniz(b,k);
      IspisiNiz(b,k,2);
      Button1.Enabled:=true;
      Edit1.Enabled:=true;Edit1.SetFocus;
end;
procedure TForm1.OdrediPodniz(VAR b:niz;VAR k:integer);
var i,p,q,pp:integer;
begin k:=0; pp:=1;                // број елемената и редни број првог у најдужем поднизу
      p:=1; q:=1;                 // број елемената и редни број првог у помоћном поднизу
      For i:=2 to n do
        If a[i-1]<a[i]            // проверава да ли следећи елемент припада растућем поднизу
         then p:=p+1              // повећава се дужина растућег подниза
         else begin If np<p       // ако следећи елемент није у текућем поднизу
                    then begin k:=p;pp:=q;    // памти се тренутно најдужи подниз
                         end;
                   p:=1;q:=i;    // тражи се следећи растући подниз
              end;
      For i:=1 to k do            // формира се низ елемената из најдужег подниза
       b[i]:=a[pp+i-1];
end;
```

Није обавезно формирати нови низ, могли смо га исписати директно из низа a, али би у том случају процедура за исписивање морала да се промени да би се додао редни број почетног елемента за испис.

- ***Саставити програм који формира два низа са по n (n<=100) троцифрених бројева, а затим од њих формира трећи низ узимајући парне елементе из првог и непарне елементе из другог низа.***

Дефинисаћемо тип низ са највише 200 елемената јер се може десити да су у првом низу свих 100 елемената парни, а у другом сви непарни, па трећи низ може да има 200 елемената. Креираћемо форму као на слици, а затим написати комплетан програм.

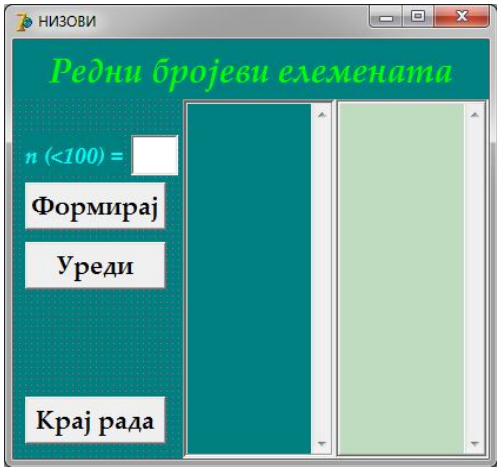



```
type
  niz=array[1..200] of integer;
  TForm1 = class(TForm)
       ...
var
  Form1: TForm1;
  n,k:integer;
  a,b,c:niz;
procedure TForm1.Button4Click(Sender: TObject);
begin Application.Terminate;
end;
procedure TForm1.Edit1Change(Sender: TObject);
begin Memo1.Clear;
      Memo2.Clear;
      Memo3.Clear;
end;
procedure TForm1.Edit1KeyPress(Sender: TObject; var Key: Char);
begin If key=#13
       then Button1.Click
```

```
       else If not (key in ['0'..'9',#8,#128])
        then key:=#27;
end;
procedure TForm1.FormirajNiz(VAR a:niz); // параметар је обавезан јер процедура формира оба низа
begin Randomize;
      k:=0;
      While k<n do
      begin k:=k+1;a[k]:=Random(900)+100;
      end;
end;
procedure TForm1.IspisiNiz(a:niz;n,k:integer);
var i:integer;
begin TMemo(FindComponent('Memo'+IntToStr(k))).Clear;   // празни мемо поље број k
      For i:=1 to n do
       TMemo(FindComponent('Memo'+IntToStr(k))).Lines.Add(Format('%2d.%4d',[i,a[i]]));
end;
procedure TForm1.Button1Click(Sender: TObject);       // формира први низ
begin n:=StrToIntDef(Edit1.Text,10);
      If n>100
       then n:=100;
      Edit1.Text:=IntToStr(n);
      FormirajNiz(a);IspisiNiz(a,n,1);
      Edit1.Enabled:=false;
      Button1.Enabled:=false;
      Button2.Enabled:=true;
      Button2.SetFocus;
end;
procedure TForm1.Button2Click(Sender: TObject);       // формира други низ
begin Button2.Enabled:=false;
      FormirajNiz(b);IspisiNiz(b,n,2);
      Button2.Enabled:=false;
      Button3.Enabled:=true;
      Button3.SetFocus;
end;
procedure TForm1.Button3Click(Sender: TObject);       // формира трећи низ
var i:integer;
begin k:=0;
      For i:=1 to n do
      begin If not Odd(a[i])
             then begin k:=k+1;c[k]:=a[i];
                  end;
            If Odd(b[i])
             then begin k:=k+1;c[k]:=b[i];
                  end;
      end;
      IspisiNiz(c,k,3);
      Button3.Enabled:=false;
      Button1.Enabled:=true;
      Edit1.Enabled:=true;
      Edit1.SetFocus;
end;
```

У задатку није конкретно дефинисан начин додавања елемената у низ ***c***, па је овде праћен паралелни редослед елемената у оба низа при чему се прво тестирају елементи низа ***a***. Могао се применити и неки други принцип, на пример, прво се додају сви из првог, па онда сви из другог низа или другачије.

- ***Саставити програм који формира низ од највише 100 троцифрених бројева, а затим неуређујући га исписује редне бројеве елемента од најмањег до највећег .***



Један од начина да се реши овај задатак је тражити најмањи елемент низа и његов редни број сместити у нови низ, а на његово место ставити ***maxint***, поступак понављамо све до сви елементи првог низа не постану исти. Ово ћемо урадити у процедури, при чему ће почетни низ бити улазни параметар процедуре да се не би изгубиле вредности елемената низа, па касније не бисмо могли да испишемо низ у новоодређеном поретку. Креираћемо форму као на слици, а затим написати комплетан програм.

```
type
  niz=array[1..100] of integer;
  TForm1 = class(TForm)
      ...

var
  Form1: TForm1;
  n,k:integer;
  a,b:niz;
```

```
procedure TForm1.Button3Click(Sender: TObject);
begin Application.Terminate;
end;
procedure TForm1.Edit1Change(Sender: TObject);
begin Memo1.Clear;
      Memo2.Clear;
end;
procedure TForm1.Edit1KeyPress(Sender: TObject; var Key: Char);
begin If key=#13
       then Button1.Click
       else If not (key in ['0'..'9',#8,#128])
        then key:=#27;
end;
procedure TForm1.FormirajNiz;
begin Randomize;
      k:=0;
      While k<n do
      begin k:=k+1;a[k]:=Random(900)+100;
            Memo1.Lines.Add(Format('%3d.%4d',[k,a[k]]);
      end;
end;
procedure TForm1.Button1Click(Sender: TObject);
begin n:=StrToIntDef(Edit1.Text,10);
      If n>100
       then n:=100;
      Edit1.Enabled:=false;
      Edit1.Text:=IntToStr(n);
      FormirajNiz;
      Button1.Enabled:=false;
      Button2.Enabled:=true;
      Button2.SetFocus;
end;
procedure TForm1.Button2Click(Sender: TObject);
begin Button2.Enabled:=false;
      Odredi(a);
      Button1.Enabled:=true;
      Edit1.Enabled:=true;
      Edit1.SetFocus;
end;
procedure TForm1.Odredi(a:niz);
var i,p:integer;
begin k:=0;                                          // број елемената низа b је на почетку 0
      While k<n do                      // циклус се завршава када се формира n елемената низа b
      begin p:=1;
            For i:=2 to n do
             If a[i]<a[p]                                      // тражи се најмањи елемент низа a
              then p:=i;
            k:=k+1;b[k]:=p;                      // памти се редни број најмањег елемента низа a
            Memo2.Lines.Add(Format('%3d.4d',[b[k],a[b[k]]]));                // исписује низ b
            a[p]:=maxint;                        // најмањем елементу низа a се увећава вредност
      end;
end;
```

Задатак се врло лепо може решити и уз помоћ скупова. Тражи се најмањи елемент чији редни број се не налази у скупу индекса и када се нађе његов редни број се смести у нови низ и дода у скуп индекса. У овом случају почетни низ не мења вредности, па не мора бити улазни параметар процедуре (наравно, није забрањено ако нам се то свиђа). Ево и овако организоване процедуре:

```
procedure TForm1.Odredi;
var i,p:integer;
    si:set of 1..100;
begin k:=0;si:=[];               // број елемената низа b је 0 и скуп редних бројева је празан
      While k<n do                    // циклус се завршава када се формира n елемената низа b
      begin p:=0;
            Repeat p:=p+1;
            until not(p in si);               // тражи се први редни број који недостаје скупу
            For i:=1 to n do                                  // тражи се најмањи елемент низа a
              If not(i in si)and(a[i]<a[p])                   // чији редни број није у скупу
               then p:=i;
            k:=k+1;b[k]:=p;                      // памти се редни број најмањег елемента низа a
            Memo2.Lines.Add(Format('%3d.4d',[b[k],a[b[k]]]));                // исписује низ b
            si:=si+[p];                          // редни број најмањег елемента се смешта у скуп
      end;
end;
```

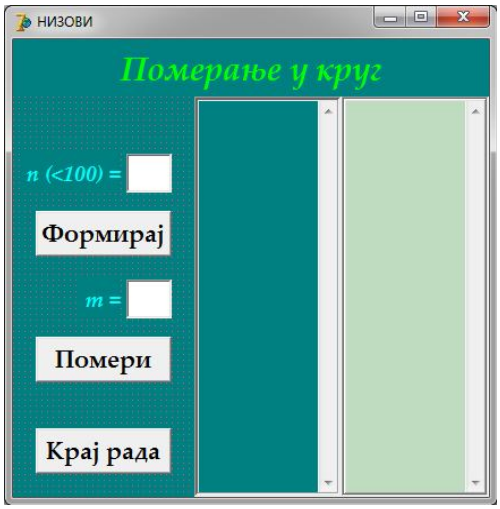
Ево још једног решење за које се може рећи да задовољава слободније тумачење услова задатака. Формира се низ чије су вредности индекси првог низа, а затим новоформирани низ уредити у неопадајући помоћу одговарајућих вредности првог низа. И овде се не мењају вредности почетног низа, па га не морамо уводити као улазни параметар (наравно, није забрањено ако нам се то свиђа).

```
procedure TForm1.Odredi;
var i,j,p:integer;
begin For i:=1 to n do b[i]:=i;
      For i:=1 to n-1 do
       For j:=i+1 to n do
        If a[b[i]]>a[b[j]]
         then begin p:=b[i];b[i]:=b[j];b[j]:=p;
              end;
      For i:=1 to n do
       Memo2.Lines.Add(Format('%3d.4d',[b[k],a[b[k]]]));
end;
```

Има још различитих и лепих решења овог (*као и сваког другог*) задатка. Читаоцима се оставља на вољу да истражују и пронађу бар још једно решење које им се чини лепим.

- ***Саставити програм који формира низ са највише 100 троцифрених бројева, а затим елементе циклично помера за m места (m∈N).***

Креираћемо форму као на слици, а затим написати комплетан програм.




```
type
  niz=array[1..100] of integer;
  TForm1 = class(TForm)
      ...
var
  Form1: TForm1;
  n,k:integer;
  a,b:niz;
procedure TForm1.Button3Click(Sender: TObject);
begin Application.Terminate;
end;
procedure TForm1.Edit1Change(Sender: TObject);
begin Memo1.Clear;
      Memo2.Clear;
end;
procedure TForm1.Edit2Change(Sender: TObject);
begin Memo2.Clear;
end;
procedure TForm1.Edit1KeyPress(Sender: TObject; var Key: Char);
begin If key=#13
       then Button1.Click
       else If not (key in ['0'..'9',#8,#128])
              then key:=#27;
end;
procedure TForm1.Edit2KeyPress(Sender: TObject; var Key: Char);
begin If key=#13
       then Button2.Click
       else If not (key in ['-','0'..'9',#8,#128])
              then key:=#27;
end;
procedure TForm1.IspisiNiz(a:niz;n,k:integer);
var i:integer;
begin TMemo(FindComponent('Memo'+IntToStr(k))).Clear;   // празни мемо поље број k
      For i:=1 to n do
       TMemo(FindComponent('Memo'+IntToStr(k))).Lines.Add(Format('%2d.%4d',[i,a[i]]));
end;
procedure TForm1.FormirajNiz;
begin Randomize;
      k:=0;
      While k<n do
      begin k:=k+1;a[k]:=Random(900)+100;
      end;
end;
procedure TForm1.Button1Click(Sender: TObject);
begin n:=StrToIntDef(Edit1.Text,10);
      If n>100
       then n:=100;
      Edit1.Text:=IntToStr(n);
      FormirajNiz;IspisiNiz(a,n,1);
      Button1.Enabled:=false;
      Button2.Enabled:=true;
      Edit1.Enabled:=false;
      Edit2.Enabled:=true;
      Edit2.SetFocus;
end;
procedure TForm1.Button2Click(Sender: TObject);
begin k:=StrToIntDef(Edit2.Text,10);Edit2.Text:=IntToStr(k);
      If k<0
       then PomeriLevo(-k)
```

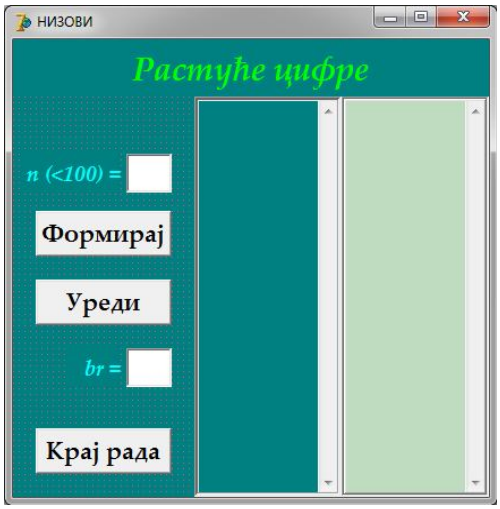
```
       else PomeriDesno(k);
      IspisiNiz(a,n,2);
      Button1.Enabled:=true;
      Button2.Enabled:=false;
      Edit1.Enabled:=true;
      Edit2.Enabled:=false;
      Edit1.SetFocus;
end;
procedure TForm1.PomeriLevo(k:integer);
var i,j,p:integer;
begin For i:=1 to k do
      begin p:=a[1];
            For j:=2 to n do
             a[j-1]:=a[j];
            a[n]:=p;
      end;
end;
procedure TForm1.PomeriDesno(k:integer);
var i,j,p:integer;
begin For i:=1 to k do
      begin p:=a[n];
            For j:=n downto 2 do
             a[j]:=a[j-1];
            a[1]:=p;
      end;
end;
```

Процедуре за померање лево и десно могу се написати као једна ако се уведе помоћна променљива s са вредностима s=0 за померање у лево, s=1 за померање у десно. Процедура је мало компликованија, али много ефикаснија:

```
procedure TForm1.PomeriLevo(s,k:integer);
var i,j,p:integer;
begin For i:=1 to k do
      begin p:=a[1+(n-1)*s];
            For j:=1 to n-1 do
             a[Abs((n+1)*s-j)]:=a[Abs((n+1)*s-j-1)];
            a[n-(n-1)*s]:=p;
      end;
end;
```

- ***Саставити програм који на случајан начин формира низ са највише 100 троцифрених бројева, а затим од њега формира нови низ тако што цифре сваког елемента уређује у растући поредак (на пример: 453 постаје 345), а затим исписује број елемената другог низа који су једнаки неком елементу почетног низа.***

Креираћемо форму као на слици, а затим написати комплетан програм.

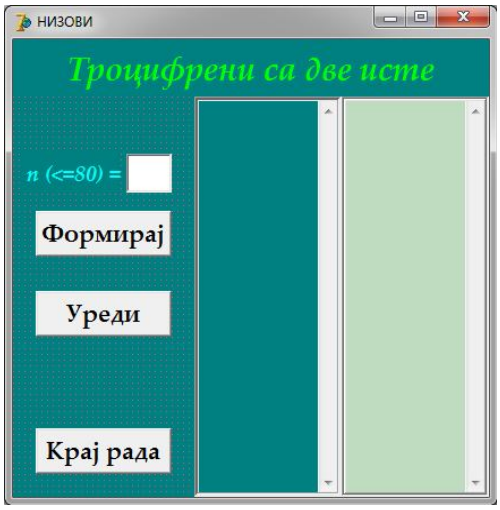



```
type
  niz=array[1..100] of integer;
  TForm1 = class(TForm)
      ...
var
  Form1: TForm1;
  n,k:integer;
  a,b:niz;
procedure TForm1.Button3Click(Sender: TObject);
begin Application.Terminate;
end;
procedure TForm1.Edit1Change(Sender: TObject);
begin Memo1.Clear;
      Memo2.Clear;
      Edit2.Clear;
end;
procedure TForm1.Edit1KeyPress(Sender: TObject; var Key: Char);
begin If key=#13
       then Button1.Click
       else If not (key in ['0'..'9',#8,#128])
        then key:=#27;
end;
procedure TForm1.IspisiNiz(a:niz;n,k:integer);
var i:integer;
begin TMemo(FindComponent('Memo'+IntToStr(k))).Clear;   // празни мемо поље број k
      For i:=1 to n do
       TMemo(FindComponent('Memo'+IntToStr(k))).Lines.Add(Format('%2d.%4d',[i,a[i]]));
end;
procedure TForm1.FormirajNiz;
begin Randomize;
      k:=0;
```

```
      While k<n do
      begin k:=k+1;a[k]:=Random(900)+100;
      end;
end;
procedure TForm1.Button1Click(Sender: TObject);
begin n:=StrToIntDef(Edit1.Text,10);
      If n>100
       then n:=100;
      Edit1.Text:=IntToStr(n);
      Edit1Change(sender);
      FormirajNiz;IspisiNiz(a,n,1);
      Edit1.Enabled:=false;
      Button1.Enabled:=false;
      Button2.Enabled:=true;
      Button2.SetFocus;
end;
procedure TForm1.Button2Click(Sender: TObject);
var i,j,br:integer;
begin For i:=1 to n do
        b[i]:=Rastuce(a[i]);
      IspisiNiz(b,n,2);
      br:=0;
      For i:=1 to n do
        For j:=1 to n do
          If a[i]=b[j]
           then br:=br+1;
      Edit2.Text:=IntToStr(br);
      Button1.Enabled:=true;
      Button2.Enabled:=false;
      Edit1.Enabled:=true;
      Edit1.SetFocus;
end;
function TForm1.Rastuce(p:integer):integer;
var c1,c2,c3:integer;
begin c1:=p mod 10;
      c2:=p div 10 mod 10;
      c3:=p div 100;
      If c1<c2
       then begin p:=c1;c1:=c2;c2:=p;
            end;
      If c1<c3
       then begin p:=c1;c1:=c3;c3:=p;
            end;
      If c2<c3
       then begin p:=c3;c3:=c2;c2:=p;
            end;
      rastuce:=c1+c2*10+c3*100;
end;
```

- ***Саставити програм који формира низ од 80 троцифрених бројева који имају бар две исте цифре, а затим га уређује у опадајући поредак.***

Овде, наравно, треба променити до сада стандардну процедуру за генерисање елемената низа. Други део задатка је већ решаван. Креираћемо форму као на слици, а затим написати комплетан програм.



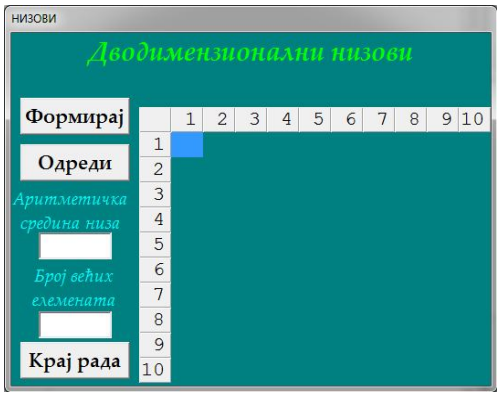

```
type
  niz=array[1..100] of integer;
  TForm1 = class(TForm)
      ...
var
  Form1: TForm1;
  n,k:integer;
  a:niz;
procedure TForm1.Button3Click(Sender: TObject);
begin Application.Terminate;
end;
procedure TForm1.Edit1Change(Sender: TObject);
begin Memo1.Clear;
      Memo2.Clear;
end;
procedure TForm1.Edit1KeyPress(Sender: TObject; var Key: Char);
begin If key=#13
       then Button1.Click
       else If not (key in ['0'..'9',#8,#128])
              then key:=#27;
end;
procedure TForm1.FormirajNiz;
begin Randomize;
```

```
      k:=1;
      While k<=n do
      begin a[k]:=Random(900)+100;
            If Iste(a[k]) then k:=k+1;;
      end;
end;
function TForm1.Iste(p:integer):boolean;
var c1,c2,c3:integer;
begin c1:=p mod 10;
      c2:=p div 10 mod 10;
      c3:=p div 100;
      iste:=(c1=c2)or(c2=c3)or(c3=c1);
end;
procedure TForm1.IspisiNiz(a:niz;n,k:integer);
var i:integer;
begin TMemo(FindComponent('Memo'+IntToStr(k))).Clear;   // празни мемо поље број k
      For i:=1 to n do
       TMemo(FindComponent('Memo'+IntToStr(k))).Lines.Add(Format('%2d.%4d',[i,a[i]]));
end;
procedure TForm1.Button1Click(Sender: TObject);
begin n:=StrToIntDef(Edit1.Text,10);
      If n>100
       then n:=100;
      Edit1.Text:=IntToStr(n);
      Edit1Change(sender);
      FormirajNiz;IspisiNiz(a,n,1);
      Edit1.Enabled:=false;
      Button1.Enabled:=false;
      Button2.Enabled:=true;
      Button2.SetFocus;
end;
procedure TForm1.Button2Click(Sender: TObject);
var i,j,p:integer;
begin For i:=1 to n-1 do
        For j:=i+1 to n do
          If a[i]<a[j]
           then begin p:=a[i];a[i]:=a[j];a[j]:=p
                end;
      IspisiNiz(a,n,2);
      Button1.Enabled:=true;
      Button2.Enabled:=false;
      Edit1.Enabled:=true;
      Edit1.SetFocus;
end;
```

Има још пуно лепих и карактеристичних задатака са једнодимензионалним низовима, али не могу се баш сви урадити и сервирати читаоцима, зато сада прелазимо на нешто сасвим ново, на групу од четири задатка са дводимензионалним низовима да се упознамо са начином рада са њима и неким лепим начинима размишљања у две димензије. Користићемо низове двоцифрених бројева и димензије 10x10 због лакшег исписа, а решења су универзална и могу се применити на било који низ елемената.

- ***Саставити програм који формира дводимензионални низ 10 x 10 двоцифрених бројева, а затим одређује аритметичку средину и број елемената већих од ње.***

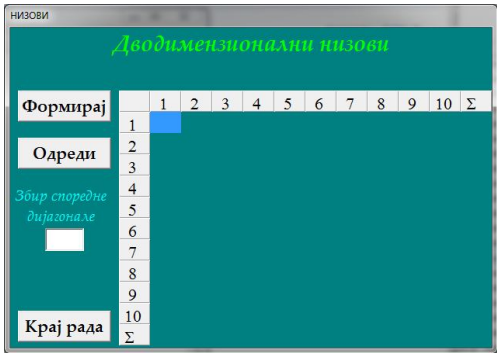
За исписивање елемената дводимензионалног низа могу се користити компоненте ***Memo*** или ***ListBox***. У том случају редови дводимензионалног низа формирају стринг који се затим исписује у одговарајући ред изабраног објекта. Ми ћемо за испис користити објекат ***StringGrid***. Овај објекат подсећа на ***excel*** табелу у којој колоне и редови нису означени (водимо рачуна код исписа низа да је у гриду прва координата колона, а друга ред, супротно од уобичајеног). Најпре ћемо поставити неколико стандардних карактеристика како би приказивање елемената низа било једноставније: ***Align*** = *alRight (подаци ће бити десно поравнати);* ***BorderStyle*** = *bsNone (нема оквира);* ***ColCount*** = *11 (десет колона за елементе низа и једна за редне бројеве);* ***Color*** = *clTeal (боја позадине);* ***DefaultColWidth*** = *35 (предефинисана ширина колона);* ***DefaultRowHeight*** = *27 (предефинисана висина редова);* ***Enabled*** = *False (корисник нема права да мења садржај било којег поља);* ***Font*** = *Courier New, Regular, 16;* ***GridLineWidth*** = *0 (нема линија између редова и колона);* ***Height*** = *300;* ***RowCount*** = *11 (десет редова за елементе низа и један за редне бројеве);* ***ScrollBars*** = *ssNone (нема клизача јер се сви елементи виде);* ***Width*** = *385*. У процедури ***FormCreate*** ћемо попунити нулти ред и колону редним бројевима (додајемо ознаке редова и колона да бисмо се лакше сналазили у низу, није обавезно и овај ред и колона се могу искључити, ако тако желимо). Креираћемо форму као на слици, а затим написати комплетан програм.



```
type
  niz=array[1..10,1..10] of integer;
  TForm1 = class(TForm)
      ...
var
  Form1: TForm1;
  a:niz;


procedure TForm1.FormCreate(Sender: TObject);
var i:integer;
begin For i:=1 to 10 do
      begin StringGrid1.Cells[0,i]:=Format('%2d',[i]);
            StringGrid1.Cells[i,0]:=Format('%2d',[i]);
      end;
end;
procedure TForm1.Button3Click(Sender: TObject);
begin Application.Terminate;
end;
procedure TForm1.Button1Click(Sender: TObject);
var i,j:integer;
begin Randomize;
      For i:=1 to 10 do
       For j:=1 to 10 do
       begin a[i,j]:=Random(90)+10;
             StringGrid1.Cells[j,i]:=Format('%2d',[a[i,j]]);
       end;
      Button1.Enabled:=false;
      Button2.Enabled:=true;
      Button2.SetFocus;
end;
procedure TForm1.Button2Click(Sender: TObject);
var asr:real;
    i,j,n:integer;
begin asr:=0;n:=0;
      For i:=1 to 10 do
       For j:=1 to 10 do
        asr:=asr+a[i,j]/100;
      For i:=1 to 10 do
       For j:=1 to 10 do
        If asr<a[i,j]
         then n:=n+1;
      Edit1.Text:=FloatToStr(asr);
      Edit2.Text:=IntToStr(n);
      Button2.Enabled:=false;
      Button1.Enabled:=true;
      Button1.SetFocus;
end;
```

- ***Саставити програм који формира дводимензионални низ 10 x 10 двоцифрених бројева, а затим одређује збирове елемената по колонама, по редовима и по дијагоналама.***

Основни низ ћемо допунити још једним редом и колоном у којима ће се исписати одговарајући збирови елемената. Збир споредне дијагонале ћемо исписати у посебном едиту. Грид ће имати дванаест редова и колона да бисмо све суме уписали у одговарајућа поља. Последње поље у првом реду и првој колони ћемо попунити симболом Σ (велико слово S). Да би се симбол видео у делфи програму за грид ћемо подесити фонт *Symbol*. Креираћемо форму као на слици, а затим написати комплетан програм.

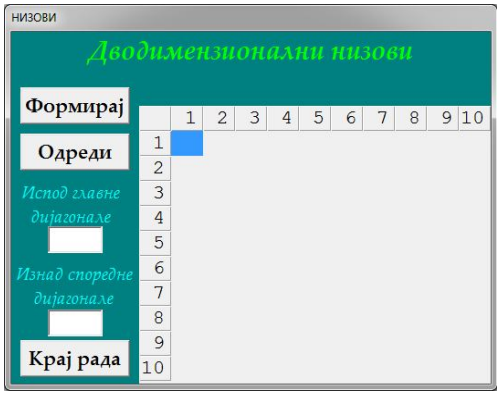



```
type
  niz=array[1..11,1..11] of integer;
  TForm1 = class(TForm)
      ...
var
  Form1: TForm1;
  a:niz;
procedure TForm1.FormCreate(Sender: TObject);
var i:integer;
begin For i:=1 to 10 do
      begin StringGrid1.Cells[0,i]:=Format('%2d',[i]);
            StringGrid1.Cells[i,0]:=Format('%2d',[i]);
      end;
      StringGrid1.Cells[0,11]:='Σ';
      StringGrid1.Cells[11,0]:='Σ';
end;
```

```
procedure TForm1.Button3Click(Sender: TObject);
begin Application.Terminate;
end;
procedure TForm1.Button1Click(Sender: TObject);
var i,j:integer;
begin Randomize;
      For i:=1 to 10 do
       For j:=1 to 10 do
       begin a[i,j]:=Random(90)+10;
             StringGrid1.Cells[j,i]:=Format('%2d',[a[i,j]]);
       end;
      Edit1.Clear;
      For i:=1 to 11 do
      begin StringGrid1.Cells[11,i]:='';
            StringGrid1.Cells[i,11]:='';
      end;
      Button1.Enabled:=false;
      Button2.Enabled:=true;
      Button2.SetFocus;
end;
procedure TForm1.Button2Click(Sender: TObject);
var i,j,sd:integer;
begin a[11,11]:=0;sd:=0;
      For i:=1 to 10 do
      begin a[11,11]:=a[11,11]+a[i,i];
            sd:=sd+a[i,11-i];
            a[11,i]:=0;
            a[i,11]:=0;
      end;
      For i:=1 to 10 do
       For j:=1 to 10 do
       begin a[11,j]:=a[11,j]+a[j,i];
             a[j,11]:=a[j,11]+a[i,j];
       end;
      For i:=1 to 11 do
      begin StringGrid1.Cells[11,i]:=Format('%3d',[a[11,i]]);
            StringGrid1.Cells[i,11]:=Format('%3d',[a[i,11]]);
      end;
      Edit1.Text:=Format('%3d',[sd]);
      Button2.Enabled:=false;
      Button1.Enabled:=true;
      Button1.SetFocus;
end;
```

- ***Саставити програм који формира дводимензионални низ 10 x 10 двоцифрених бројева, а затим одређује збир елемената испод главне и збир елемената изнад споредне дијагонале.***

Креираћемо форму као на слици а затим написати комплетан програм.

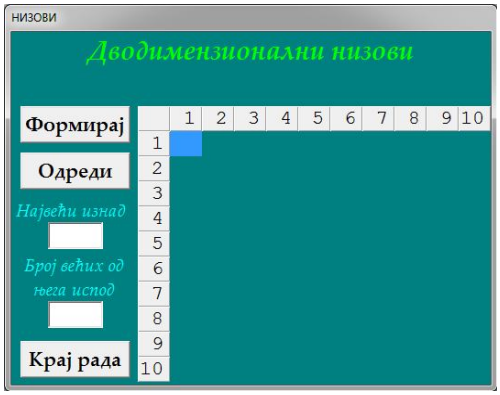



```
type
  niz=array[1..10,1..10] of integer;
  TForm1 = class(TForm)
      ...
var
  Form1: TForm1;
  a:niz;
procedure TForm1.FormCreate(Sender: TObject);
var i:integer;
begin For i:=1 to 10 do
      begin StringGrid1.Cells[0,i]:=Format('%2d',[i]);
            StringGrid1.Cells[i,0]:=Format('%2d',[i]);
      end;
end;
procedure TForm1.Button3Click(Sender: TObject);
begin Application.Terminate;
end;
procedure TForm1.Button1Click(Sender: TObject);
var i,j:integer;
begin Randomize;
      For i:=1 to 10 do
       For j:=1 to 10 do
       begin a[i,j]:=Random(90)+10;
             StringGrid1.Cells[j,i]:=Format('%2d',[a[i,j]]);
       end;
      Edit1.Clear;Edit2.Clear;
      Button1.Enabled:=false;
      Button2.Enabled:=true;
      Button2.SetFocus;
end;
```

```
procedure TForm1.Button2Click(Sender: TObject);
var i,j,g,s:integer;
begin g:=0;s:=0;
      For i:=2 to 10 do                          // елементи испод главне дијагонале
        For j:=1 to i-1 do
         g:=g+a[i,j];
      For i:=1 to 9 do                          // елементи изнад споредне дијагонале
        For j:=1 to 10-i do
         s:=s+a[i,j];
      Edit1.Text:=IntToStr(g);
      Edit2.Text:=IntToStr(s);
      Button2.Enabled:=false;
      Button1.Enabled:=true;
      Button1.SetFocus;
end;
```

- ***Саставити програм који формира дводимензионални низ 10 x 10 двоцифрених бројева, а затим одређује број елемената испод главне и споредне дијагонале који су већи од највећег елемента изнад главне и споредне дијагонале.***

Креираћемо форму као на слици, а затим написати комплетан програм.

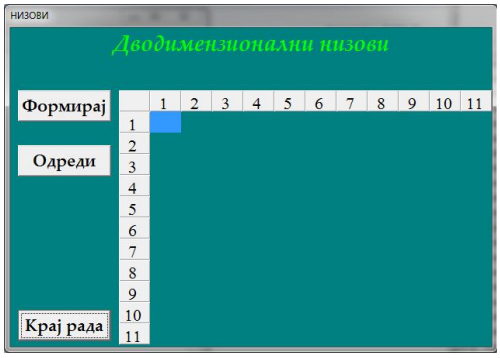



```
type
  niz=array[1..10,1..10] of integer;
  TForm1 = class(TForm)
       ...
var
  Form1: TForm1;
  a:niz;
procedure TForm1.FormCreate(Sender: TObject);
var i:integer;
begin For i:=1 to 10 do
      begin StringGrid1.Cells[0,i]:=Format('%2d',[i]);
            StringGrid1.Cells[i,0]:=Format('%2d',[i]);
      end;
end;
procedure TForm1.Button3Click(Sender: TObject);
begin Application.Terminate;
end;
procedure TForm1.Button1Click(Sender: TObject);
var i,j:integer;
begin Randomize;
      For i:=1 to 10 do
       For j:=1 to 10 do
       begin a[i,j]:=Random(90)+10;
             StringGrid1.Cells[j,i]:=Format('%2d',[a[i,j]]);
       end;
      Edit1.Clear;Edit2.Clear;
      Button1.Enabled:=false;
      Button2.Enabled:=true;
      Button2.SetFocus;
end;
procedure TForm1.Button2Click(Sender: TObject);
var i,j,g,s:integer;
begin g:=0;s:=0;
      For i:=1 to 4 do              // елементи изнад главне и споредне дијагонале
        For j:=i+1 to 10-i do
         If a[i,j]>g
          then g:=a[i,j];
      For i:=7 to 10 do             // елементи испод главне и споредне дијагонале
        For j:=12-i to i-1 do
         If a[i,j]>g
          then s:=s+1;
      Edit1.Text:=IntToStr(g);
      Edit2.Text:=IntToStr(s);
      Button2.Enabled:=false;
      Button1.Enabled:=true;
      Button1.SetFocus;
end;
```

- ***Саставити програм који формира дводимензионални низ 10 x 10 двоцифрених бројева, а затим га проширује за један ред и колону највећим елементима у одговарајућем реду, колони или главној дијагонали.***

Креираћемо форму као на слици на следећој страни, а затим написати комплетан програм.

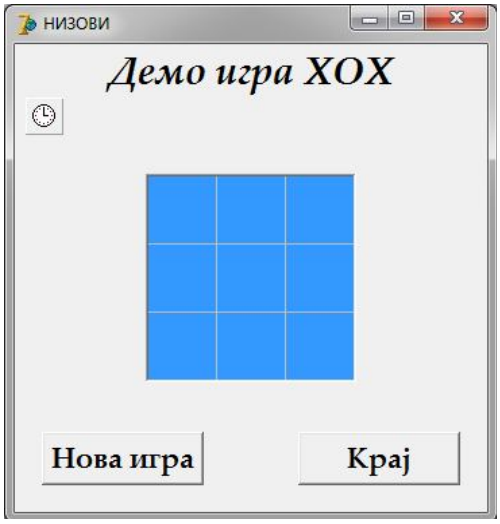
```
type
  niz=array[1..11,1..11] of integer;
  TForm1 = class(TForm)
       ...
```




```
var
  Form1: TForm1;
  a:niz;
procedure TForm1.FormCreate(Sender: TObject);
var i:integer;
begin For i:=1 to 11 do
      begin StringGrid1.Cells[0,i]:=Format('%2d',[i]);
            StringGrid1.Cells[i,0]:=Format('%2d',[i]);
      end;
end;
procedure TForm1.Button3Click(Sender: TObject);
begin Application.Terminate;
end;
procedure TForm1.Button1Click(Sender: TObject);
var i,j:integer;
begin Randomize;
      For i:=1 to 10 do
       For j:=1 to 10 do
       begin a[i,j]:=Random(90)+10;
             StringGrid1.Cells[j,i]:=Format('%2d',[a[i,j]]);
       end;
      Button1.Enabled:=false;
      Button2.Enabled:=true;
      Button2.SetFocus;
end;
procedure TForm1.Button2Click(Sender: TObject);
var i:integer;
begin For i:=1 to 10 do
      begin a[i,11]:=Najveci(a,i,1);
            StringGrid1.Cells[11,i]:=Format('%2d',[a[i,11]]);
            a[11,i]:=Najveci(a,i,2);
            StringGrid1.Cells[i,11]:=Format('%2d',[a[11,i]]);
      end;
      a[11,11]:=Najveci(a,1,3);
      StringGrid1.Cells[11,11]:=Format('%2d',[a[11,11]]);
      Button2.Enabled:=false;
      Button1.Enabled:=true;
      Button1.SetFocus;
end;
function TForm1.Najveci(a:niz; i,j:integer):integer;
var k,n:integer;
begin Case j of
        1 : n:=a[i,1];
        2 : n:=a[1,i];
       else n:=a[1,1];
      end;
      For k:=2 to 10 do
      begin If (j=1)and(a[i,k]>n)   // тражимо највећи у реду i
             then n:=a[i,k];
            If (j=2)and(a[k,i]>n)   // тражимо највећи у колони i
             then n:=a[k,i];
            If (j=3)and(a[k,k]>n)   // тражимо највећи на главној дијагонали
             then n:=a[k,k];
      end;
      najveci:=n;
end;
```

- ***Саставити програм који симулира игру ИксОкс.***

Следе два примера коришћења низова у једноставнијим логичким играма. У овим примерима нећемо направити нешто *револуционарно*, јер у овом тренутку нити можемо нити хоћемо да тако нешто постигнемо. Циљ је да се само демонстрира моћ коју у рукама програмера могу имати низови. Први пример је игра *ИксОкс*. Ово је једна од *савршених* игара. Термин *савршена* игра се користи за игре у којима, ако играчи играју увек најбољи потез онда на крају игре нема победника. Постоје три могућа нивоа игре у зависности од функције коју има рачунар: рачунар игра и као први и као други играч, рачунар игра као један од играча док је други играч човек, корисник програма и игру играју два играча, а рачунар има улогу посматрача и судије. Према сложености поља за игру, такође, можемо да направимо нивое, постоје варијанте у две и три димензије са различитим бројем позиција на којима може да се игра. Овде ће бити приказана најједноставнија варијанта 3x3 игре, демо ниво, тј. рачунар игра сам против себе и без икаквог коефицијента интелигенције, тј. сви потези се бирају на случајан начин. Да би се добила права игра мора се додати и мало *мозгања* (постоје потези које никако не смемо одиграти ако не желимо да изгубимо). Као симболи играча коришћени су карактери из фонта *Webdings* ● и ✕. У коду програма користи се само основни фонт, па ће се уместо симбола из фонта *Webdings* приказивати карактери = и r. Креираћемо форму као на слици на следећој страни, а затим написати комплетан програм без неког претераног објашњавања.

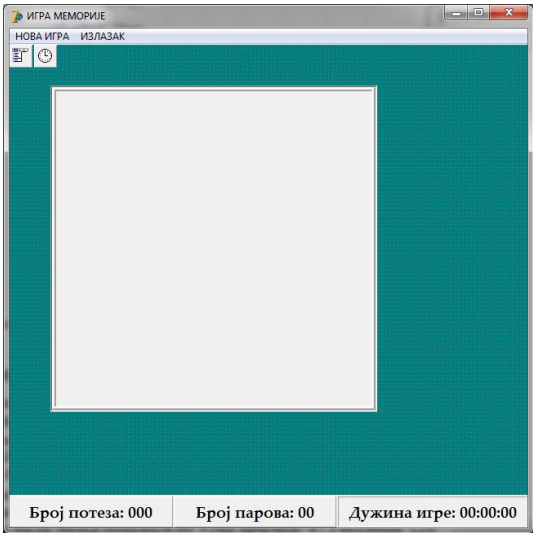



```
type
  niz=array[0..3,0..3] of integer;
  TForm1 = class(TForm)
      ...
var
  Form1: TForm1;
  a:niz;
  slob:integer;
  ch:char;
procedure TForm1.Button2Click(Sender: TObject);
begin Application.Terminate;
end;
procedure TForm1.Button1Click(Sender: TObject);
var i,j:integer;
begin For i:=0 to 3 do
       For j:=0 to 3 do
       begin a[i,j]:=0;
// почетно стање низа, сви елементи су на нули
             If (i<3)and(j<3)
              then StringGrid1.Cells[i,j]:='';
// празне се сва поља стринг грида
       end;
      slob:=9;                 // свих девет поља је слободно
      ch:='r';                 // на потезу је први играч
      Timer1.Enabled:=true;    // укључује се тајмер који контролише одигравање потеза
      Button1.Enabled:=false;
      Button2.Enabled:=true;
      Button2.SetFocus;
end;
function TForm1.Kraj:char;     // функција која одређује завршетак игре
var i,j,p:integer;
begin kraj:=' ';               // нема победника
      p:=0;                    // контрола споредне дијагонале
      a[3,3]:=0;               // контрола главне дијагонале
      For i:=0 to 2 do
      begin a[3,i]:=0;         // контрола колона
            a[i,3]:=0;         // контрола редова
            p:=p+a[i,3-i];     // збир елемената на споредној дијагонали
            a[3,3]:=a[3,3]+a[i,i];  // збир елемената на главној дијагонали
            For j:=0 to 2 do
            begin a[3,i]:=a[3,i]+a[j,i];  // збирови елемената по колонама
                  a[i,3]:=a[i,3]+a[i,j];  // збирови елемената по редовима
            end;
      end;
      For i:=0 to 3 do         // проверава да ли је неко од играча победио
       If (Abs(a[3,i])=3)      // први играч је победио ако је збир неке колоне, реда или
        or(Abs(a[i,3])=3)      // дијагонале 3, а други ако је -3, зато користимо функцију Abs
        or(Abs(p)=3)
        then kraj:=ch;         // ако јесте, проглашава победника
end;
procedure TForm1.Timer1Timer(Sender: TObject);
var i,j:integer;
    pl:string;
begin Randomize;               // потези се бирају на случајан начин
      slob:=slob-1;            // број слободних поља се смањује након сваког потеза
      Repeat i:=Random(3);
             j:=Random(3);
      Until a[i,j]=0;          // одређује се потез
      StringGrid1.Cells[j,i]:=ch;   // исписује се потез играча
      If ch='='                // одређује ко је на потезу и вредност потеза 1 или -1
       then begin a[i,j]:=-1;ch:='r';
                  // вредност потеза једног играча је -1
            end
       else begin a[i,j]:=1;ch:='=';
                  // вредност потеза другог играча је 1
            end;
      If Kraj=ch               // тестира да ли је игра завршена
       then begin If ch='='    // ако јесте проглашава победника
                   then pl:='први'
                   else pl:='други';
                  Timer1.Enabled:=false;
                  Button1.Enabled:=true;
                  ShowMessage('Победио је '+pl+' играч');
            end
       else If slob=0          // ако нико није победио тестира да ли је завршено нерешено
             then begin Timer1.Enabled:=false;
                        Button1.Enabled:=true;
                        ShowMessage('Нерешено!');
                  end;
      // ако игра није завршена одиграва се следећи потез
end;
```

- ***Саставити програм који симулира игру Меморије.***

Игра меморије састоји се из различитог броја парова сличица (број парова зависи од нивоа). Задатак играча је да открије све парове у што краћем времену и са што је могуће мање отварања парова. У игри разликујемо два потеза:

- прво отварање - отвара се изабрано поље, ако су претходно отворена два различита поља затварају се оба поља и
- друго отварање - отвара се изабрано поље и упоређује се са претходно отвореним, ако су иста обрачунавају се поени и забрањује се касније употреба ових поља.



Да бисмо ово постигли дефинисаћемо дводимензионални низ и доделити му вредности редних бројева парова и неколико помоћних променљивих које ће водити рачуна о редном броју отварања, претходно отвореном пољу и броју поља која још нису отворена. У примеру који се овде приказује, да бисмо олакшали примену програма на рачунару читаоца, уместо сличица користићемо цифре декадног бројног система (значи играће се на пољу димензија 4 x 5). Логика програма је иста као и да се ради са сличицама само код писања програма не морамо проналазити и погодне сличице. Наравно, са мало више знања, игра би се могла допунити листом најбољих резултата, музиком у позадини, звуцима када се погоди или промаши пар и слично. Границе не постоје. Креираћемо форму као на слици, а затим написати комплетан програм без неког претераног објашњавања.

```
type
  niz=array[1..4,1..5] of integer;
  TForm1 = class(TForm)
      ...
var
  Form1: TForm1;
  h,m,s,ss,po,pa,b1,b2,aI,aP,aD,mk,pp:integer;
  polje:niz;
procedure TForm1.FormCreate(Sender: TObject);
var i:integer;
begin aD:=Min(Screen.Width*2 div 3,Screen.Height*2 div 3);
      Form1.Left:=aD div 4;Form1.Width:=aD;
      Form1.Top :=aD div 4;Form1.Height:=aD;
      aP:=aD*3 div 4;aI:=aD*3 div 20;
      Panel1.Left:=aD div 9;Panel1.Width:=aP+15;
      Panel1.Top :=aD div 9;Panel1.Height:=aP*4 div 5+15;
      For i:=1 to 20 do
       begin TButton.Create(Self).Name := 'BTN' + IntToStr(i);
             with TButton(FindComponent('BTN' + IntToStr(i))) do
             begin Left:=aI*((i-1) div 4)+5;Width :=aI;
                   Top :=aI*((i-1) mod 4)+5;Height:=aI;
                   Parent := Panel1;
                   Caption:=IntToStr(i mod 10);
                   Font.Name:='Comic Sans MS';
                   Font.Size:=aI div 2;
                   OnClick:=MouseClick;
                   Enabled:=false;
             end;
       end;
end;
procedure TForm1.Timer1Timer(Sender: TObject);
begin ss:=ss+1;
      If ss=100
       then begin s:=s+1;ss:=0;
           end;
      If s=60
       then begin s:=0;m:=m+1;
            end;
       If m=60
        then begin m:=0;h:=h+1;
             end;
      StatusBar1.Panels[2].Text:='Дужина игре: '+RightStr('00'+IntToStr(h),2)+':'
                                                +RightStr('00'+IntToStr(m),2)+':'
                                                +RightStr('00'+IntToStr(s),2);
      If pa=10
       then begin Timer1.Enabled:=false;
                  ShowMessage('Игра је завршена!');
            end;
end;
```

```
procedure TForm1.Exit1Click(Sender: TObject);
begin Application.Terminate;
end;
procedure TForm1.New1Click(Sender: TObject);
var i:integer;
begin h:=0;m:=0;s:=0;ss:=0;
      po:=0;pa:=0;
      pp:=0;mk:=1;
      StatusBar1.Panels[1].Text:='Број парова: '+RightStr('00'+IntToStr(pa),2);
      StatusBar1.Panels[0].Text:='Број потеза: '+RightStr('00'+IntToStr(po),2);
      Rasporedi;
      For i:=1 to 20 do
        TButton(FindComponent('BTN'+IntToStr(i))).Enabled:=true;
      Timer1.Enabled:=true;
end;
procedure TForm1.Rasporedi;
var x,y,z:integer;
    i:char;
    ch:array[0..9] of char;
begin Randomize;
      For x:=1 to 4 do
        For y:=1 to 5 do polje[x,y]:=-1;
      For z:=0 to 9 do ch[z]:='.';
      For z:=1 to 20 do TButton(FindComponent('BTN'+IntToStr(z))).Caption:='';
      For i:='0' to '9' do
      begin Repeat z:=Random(10);
            until ch[z]='.';
            ch[z]:=i;
            Repeat x:=Random(4);
                   y:=Random(5);
            until polje[x,y]=-1;
            polje[x,y]:=StrToInt(i);
      end;
      z:=0;
      For x:=1 to 4 do
        For y:=1 to 5 do
          If (polje[x,y]=-1)and(z<10)
            then begin polje[x,y]:=StrToInt(ch[z]);z:=z+1;
                 end;
end;
procedure TForm1.MouseClick(Sender: TObject);
var i,x,y,b:integer;
begin x:=0; y:=0; b:=0;
      If sender.ClassName='TButton' then
       begin For i:=1 to 20 do
              If sender=TButton(FindComponent('BTN'+IntToStr(i))) then
               begin x:=(i-1) mod 4+1;
                     y:=(i-1) div 4+1;
                     TButton(FindComponent('BTN'+IntToStr(i))).Caption:=IntToStr(polje[x,y]);
                     b:=i;
               end;
              If mk=1 then
               begin If b1>0 then
                      begin TButton(FindComponent('BTN'+IntToStr(b1))).Caption:=' ';
                            TButton(FindComponent('BTN'+IntToStr(b2))).Caption:=' ';
                            b1:=0;b2:=0;
                      end;
                     pp:=polje[x,y];
                     mk:=2;b1:=b;
               end
               else If b1<>b then
                     begin mk:=1;b2:=b;
                           If pp=polje[x,y] then
                            begin TButton(FindComponent('BTN'+IntToStr(b1))).Enabled:=false;
                                  TButton(FindComponent('BTN'+IntToStr(b2))).Enabled:=false;
                                  b1:=0;b2:=0;
                                  pa:=pa+1;
                                  StatusBar1.Panels[1].Text:='Број парова: '+
                                                             RightStr('00'+IntToStr(pa),2);
                            end;
                            po:=po+1;
                            StatusBar1.Panels[0].Text:='Број потеза: '+
                                                       RightStr('00'+IntToStr(po),2);
                     end;
       end;
end;
```

У програму је коришћен панел на коме се у фази извршења креирају тастери, мени компонента, тајмер и статусна линија. Форма је направљена да се у програму препозна резолуција монитора и подесе димензије тако да се форма види комплетна. Панел и тастери, такође, мењају димензије у зависности од резолуције монитора. Могла су се додати и још нека подешавања, пре свега панели у статусној

линији који су овде фиксни, па се на нижим резолуцијама може десити да се не види број потеза или парова. Али, ово је само демонстрација дела могућности програмског окружења.

Задаци за самосталан рад

- ***Саставити програм који уписани петоцифрени број раздваја на цифре, а затим од тих цифара формира све различите бројеве и одређује који од њих су прости.***
- ***Саставити програм који на случајан начин формира низ од највише 100 бројева од -500 до 500, а затим одређује најмањи и највећи по апсолутној вредности двоцифрени број, број позитивних и број негативних троцифрених бројева.***
- ***Саставити програм који на случајан начин формира низ са највише 100 бројева мањих од 1000, а затим од њега формира нови низ тако што сваку цифру елемената првог низа мења допуном до 9 (на пример: 473 постаје 526), а затим исписује највећи и најмањи елемент који се налази у оба низа, ако такви постоје.***
- ***Саставити програм који на случајан начин формира низ са највише 100 троцифрених бројева, а затим од њега формира нови низ чији су елементи збирови цифара одговарајућих елемената првог низа (на пример: 453 постаје 12), а затим одређује елемент другог низа који се појављује највише пута.***
- ***Саставити програм који на случајан начин формира низ од највише 100 троцифрених бројева, а затим формира низ чији и-ти елемент представља број елемената првог низа који су већи од и-тог елемента првог низа и низ чији и-ти елемент представља број елемената првог низа који су мањи од и-тог елемента првог низа.***
- ***Саставити програм који формира низ од 80 природних бројева (мањих од 1000), а затим одређује низ чији први и последњи елемент су једнаки одговарајућим елементима првог низа, а остали се добијају по формули: $b_i = (a_{i-1} + a_{i+1}) / 2$.***
- ***Саставити програм који на случајан начин формира низ од највише 100 четвороцифрених бројева, а затим одређује број елемената који почињу и завршавају истом цифром и њихову аритметичку средину.***
- ***Саставити програм који на случајан начин формира два низа са по н<=100 чланова, а затим одређује укупан број парова елемената са истим индексом у оба низа који су једнаки ($a_i = b_i$).***
- ***Саставити програм који на случајан начин формира низ са највише 100 четвороцифрених бројева, а затим од њега формира нови низ чији се елементи добијају од елемената првог низа заменом прве две са последњим двема цифрама (на пример: 1453 постаје 5314), а затим исписује елементе који се појављују у оба низа.***
- ***Саставити програм који формира низ од највише 100 природних бројева (мањих од 1000), а затим одређује низ чији је први елемент једнак првом елементу првог низа, а остали се добијају по формули: $b_i = (a_1 + a_2 + a_3 + \ldots + a_{i-1}) / i$.***
- ***Саставити програм који формира низ од највише 100 природних бројева (мањих од 1000), а затим одређује низ чији елементи се добијају по формули:***
$$b_i = (a_1 + a_2 + a_3 + \ldots + a_{i-1} + a_{i+1} + a_{i+2} + a_{i+3} + \ldots + a_n) / (n-1).$$
- ***Саставити програм који са тастатуре прихвата три природна броја n, p, q, а затим формира два низа по формулама: $a_1 = p$, $a_i = (a_{i-1} + b_{i-1}) / 2$, $1 < i <= n$; $b_1 = q$, $a_i = Sqrt(a_{i-1} + b_{i-1})$, $1 < i <= n$.***
- ***Саставити програм који формира низ од највише 100 двоцифрених бројева, а затим одређује низ чији елементи се добијају по формули: $b_{2i} = (a_2 + a_4 + a_6 + \ldots + a_{2i}) / (2i)$***
$$b_{2i-1} = (a_1 + a_3 + a_5 + \ldots + a_{2i-1}) / (2i-1).$$
- ***Саставити програм који формира низ од највише 100 природних бројева (мањих од 1000), а затим одређује низ чији елемент на месту і представља број елемената првог низа који су мањи од елемента a_i.***
- ***Саставити програм који формира два низа са по од највише 100 природних бројева (мањих од 1000), а затим одређује низ чији елементи се добијају по формули:*** $c_i = \begin{cases} -1, & a_i < b_i \\ 0, & a_i = b_i \\ 1, & a_i > b_i \end{cases}$.
- ***Саставити програм који на случајан начин формира низ целих двоцифрених бројева, формира низ бинарних записа чланова тог низа и исписује оба низа.***

- ***Саставити програм који на случајан начин формира низа са највише 100 целих бројева од -1000 до 1000, а затим формирати нови низ тако сто се члановима низа са парним индексима промени знак , а чланови са непарним индексима се умање за свој индекс.***
- ***Саставити програм који у несортираном низу одређује највећи и најмањи члан и затим исписује низ без та два елемента.***
- ***Саставити програм који формира низ од највише 100 троцифрених бројева, а затим одређује колико пута у низу следећи елемент почиње цифром којом претходни завршава.***
- ***Саставити програм којим се уносе елементи низа од н елемената, а затим се од тога низа формира матрица n x n која има елементе првог низа на главној дијагонали, а сви остали су јединице.***
- ***Саставити програм који у матрица са n врста и m колона налази и исписује највећи и најмањи елемент матрице, као и врсту и колону где се они налазе.***
- ***Саставити програм којим се уносе елементи низа са n^2 елемената, затим се од тога низа формира матрица која има n врста и n колона.***
- ***Саставити програм који формира матрицу n x n, а затим израчунава која колона има највећи збир и исписује колону и збир.***
- ***Саставити програм којим се проверава да ли је збир две матрице n x n јединична матрица.***
- ***Саставити програм којим се проверава да ли је производ две матрице n x n јединична матрица.***
- ***Саставити програм који симулира игру Ханојске куле (три штапа и дискови који се премештају).***
- ***Саставити програм који симулира игру Otello (познатија као реверси).***
- ***Саставити програм који симулира игру GoMoKo (као хох на табли 40x30 или више поља).***
- ***Саставити програм који симулира игру Потапање бродова.***
- ***Саставити програм који симулира игру MasterMind ("Скочко", са бојама или цифрама).***
- ***Саставити програм који симулира игру Дама (на шаховској табли по 8 белих и црних жетона).***
- ***Саставити програм који симулира игру Мица (традиционална игра са зрнима пасуља и кукуруза).***
- ***Саставити програм који симулира игру Не љути се човече.***
- ***Саставити програм који симулира игру Монопол.***
- ***Саставити програм који симулира игру Јамб.***
- ***Саставити програм који симулира игру Connect 4.***
- ***Саставити програм који симулира игру Шах.***