O’ZBEKISTON RESPUBLIKASI OLIY VA O’RTA MAXSUS TA’LIM VAZIRLIGI

FARG’ONA POLITEXNIKA INSTITUTI

“ENERGETIKA” FAKULTETI

“INFORMATIKA VA AXBOROT TEXNOLOGIYALARI” KAFEDRASI

“Informatika va axborot texnologiyalari” fanidan

Kiritilgan sonlarning EKUB va EKUKini hisoblash dasturini tuzish mavzusida yozilgan

REFERAT

REFERAT

Topshirdi: 18-13 gurux tolibi
M.Xoshimov

Qabul qildi: S.Zokirov

REJA:

I. **Nazariy qism**
   1. **Ma’lumotning asosiy turlari, massiv va yozuvlar;**
   2. **Shart operatorlari;**
   3. **Sikl operatorlari;**
   4. **Standart funksiya va proseduralar**

II. **Amaliy qism**
   1. **Masala yechimining Delphi 7 dagi talqini**
   2. **Masala yechimining Turbo Pascaldagi talqini**

III. **Foydalanilgan adabiyotlar**

I. NAZARIY QISM.

MA’LUMOTLARNING ASOSIY TURLARI

Pascal tilining asosiy turlariga quyidagilar kiradi:

- butun sonlar (INTEGER va boshkalar);
- haqiqiy sonlar (REAL va boshqalar);
- belgilar (CHAR);
- satrlar (STRING);
- mantiqiy (BOOLEAN).

Butun va suzuvchi nuqtali sonlar har xil formatlarda berilishi mumkin.

Butun sonlar

Format	Qiymatlar sohasi
SHORTING	- 128 – 127
INTEGER	- 32768 – 32767
LONGINT	- 2147483648 - 2147483647
BYTE	0 – 255
WORD	0 – 65535

Haqiqiy sonlar

Format	Qiymatlar sohasi	Raqamlar qiymati miqdori
REAL	2.9E – 39 – 1.7e 38	11 – 12
SINGLE	1.5E – 45 – 3.4e 38	7 – 8
DOUBLE	5.0E – 324 – 1.7e 108	15 – 16
EXTENDED	3.4E – 4932 – 1.1e 4932	–19 – 20

Satrlar

Satrlar ikki xil ko’rinishda e’lon qilinishi mumkin:

1-ko’rinish:

Ism : String;

255 belgidan iborat uzunlikka ega o’zgaruvchi e’lon qilinadi.

2-ko’rinish:

Ism: String [satr uzunligi];

Ko’rsatilgan uzunlikdagi o’zgaruvchi e’lon qilinadi.

MASSIVLAR

Bir o’lchovli massivni e’lon qilish:

Massiv ismi: array [quyi indeks .. yuqori indeks] of <elementlar turi>;

Ikki o’lchovli massivni e’lon qilish:

Massiv ismi: [quyi indeks 1 .. yuqori indeks 1, quyi indeks 2 .. yuqori indeks 2] of <elementlar turi>;

YOZUVLAR

Yozuvlarni ikki xil ko’rinishda e’lon qilish mumkin:

1-ko’rinish:

O’zgaruvchi Yozuv: **record**

Maydon 1 : 1 tur;

Maydon 2 : 2 tur;

. . . .

Maydon 3 : 3 tur;

end.

2-ko’rinish:

Avval yozuv turi, keyin o’zgaruvchi tur e’lon qilinadi:

type

Tur Yozuvi Ismi: **record**

Maydon 1 : tur 1;

Maydon 2 : tur 2;

. . . .

Maydon J : tur J;

end;

var

O’zgaruvchi Yozuv : Tur Yozuvi Ismi;

SHART OPERATORLARI

1-ko’rinish:

if shart

then

begin

{bu ko’rsatmalar, agar shart to’g’ri bo’lsa, bajariladi}

end

else

begin

{bu ko’rsatmalar shart xato bo’lsa, bajariladi}

end;

2-ko’rinish:

IF shart

then

begin

{bu ko’rsatmalar shart to’g’ri bo’lganda bajariladi}

end

Izoh: Agar **begin** va **end** o’rtasida faqat bitta ko’rsatma (operator) bo’lsa, **begin** va **end** ni yozmaslik mumkin.

CASE ko’rsatmasi

1-ko’rinish:

Case ifoda **of**

begin

1 – o’zgarmaslar ro’yxati: **begin**
{1-ko’rsatmalar}
end;

2 – o’zgarmaslar ro’yxati: **begin**
{2-ko’rsatmalar}
end;

. . .

J – o’zgarmaslar ro’yxati: **begin**
{J - ko’rsatmalar}
end;

else

begin
{ko’rsatmalar}
end;

end

Case so’zidan keyin keluvchi ifoda qiymati, tegishli ro’yxat o’zgarmasi bilan mos kelgandagina **begin** va **end** o’rtasidagi ko’rsatmalar bajariladi, aks holda **else** so’zidan keyin keluvchi **begin** va **end** so’zlari o’rtasidagi ko’rsatmalar bajariladi.

SIKLLAR

3) FOR ko’rsatmasi

1-ko’rinish: (hisobchi qiymati o’sadi);

for Hisobchi: = Boshlang’ich Qiymat **to** So’nggi Qiymat **do**

begin

{ko’rsatmalar}

end;

begin va **end** o’rtasidagi ko’rsatmalar [(So’nggi Qiymat – Boshlang’ich Qiymat) + 1] marta bajariladi.

Agar boshlang’ich qiymat so’nggi qiymatdan katta bo’lsa, **begin** va **end** o’rtasidagi ko’rsatmalar bajarilmaydi.

Izoh: Agar **begin** va **end** o’rtasida faqat bitta ko’rsatma bo’lsa, **begin** va **end** so’zlarini yozmaslik mumkin.

2-ko’rinish (hisobchi qiymati kamayadi)

for Hisobchi: = Boshlang’ich Qiymat **downto** So’nggi Qiymat **do**

begin

{ko’rsatmalar}

end;

begin va **end** o’rtasidagi ko’rsatmalar [(Boshlang’ich Qiymat – So’nggi Qiymat) + 1] marta bajariladi. Agar boshlang’ich qiymat so’nggi qiymatdan kichik bo’lsa, **begin** va **end** o’rtasidagi ko’rsatmalar bajarilmaydi.

4) REPEAT ko’rsatmasi

repeat

begin

{ko’rsatmalar}

end

until SHart;

begin va **end** o’rtasidagi ko’rsatmalar (tsikl ko’rsatmalari) bajariladi, shundan keyin SHart ifodaning qiymati tekshiriladi. Agar u **False** ga teng bo’lsa

(ya’ni shart bajarilmasa), tsikl ko’rsatmasi yana bir marta bajariladi. Xuddi shunday tarzda SHart haqiqat bo’lguncha davom ettiriladi. SHunday qilib, **until** so’zidan keyin tsiklni tugatish sharti yoziladi.

Izoh: Agar **begin** va **end** o’rtasida faqat bitta ko’rsatma yozilgan bo’lsa, **begin** va **end** so’zlarini yozmaslik mumkin.

5) WHILE ko’rsatmasi

while SHart **do**
    begin
        {shartlar}
    end;

SHart ifodasidagi qiymat tekshiriladi, agar u **True**ga teng bo’lsa (shart bajarilsa), **begin** va **end** o’rtasidagi ko’rsatmalar (tsikl ko’rsatmalari) bajariladi. SHundan keyin yana SHart ifodasining qiymati tekshiriladi, shunday tarzda SHart ifodasining qiymati **False** bo’lguncha davom ettiriladi. SHunday qilib, **while** so’zidan keyin tsikl ko’rsatmalarining bajarish sharti yoziladi.

Izoh: Agar **begin** va **end** o’rtasida faqat bitta ko’rsatma yozilgan bo’lsa, **begin** va **end** so’zlarini yozmaslik mumkin.

6) GOTO shartsiz o’tish ko’rsatmasi

GoTo metka (nishona)

Oldida, **label** bo’limida e’lon qilingan, metka yozilgan ko’rsatmaga o’tish amalga oshiriladi.

STANDART funksiya VA PROTSEDURALARI

Matematik	Izoh
ABS (ifoda)	Argumentning absolyut qiymati (butun yoki haqiqiy tur)
SQR (ifoda)	Argument kvadrati (butun yoki haqiqiy tur)
SQRT (ifoda : **real**) : **real**	Argumentdan kvadrat ildiz chiqarish
SIN (ifoda : **real**) : **real**	Radianlarda ifodalangan argumentning sinusi
COS (ifoda : **real**) : **real**	Radianlarda ifodalangan argumentning kosinusi
ARCTAN (ifoda : **real**) : **real**	Radianlarda ifodalangan argumentning arktangensi

EXP (ifoda : **real**) : **real**	Argument eksponentasi
LN (ifoda : **real**) : **real**	Argumentning natural logarifmi
INT (ifoda : **real**) : **real**	Argument butun qismi
TRUNC (ifoda:**real**) : **longint**	Argument butun qismi
ROUND (ifoda:**real**) : **longint**	Argument qiymatini eng yaqin butun songacha yaxlitlash
STR (ifoda: **var** Satr : **string)**	Sonli ifodani satrga aylantirish
Val (Satr:**string**; **var** O'zgaruvchi, **var** Xato kodi : **integer**);	Butun yoki haqiqiy sonni ifodalovchi satrni songa aylantirish

Satr va belgilar bilan ishlash uchun	Izohlar
Concat (Satr 1 : **string**; …; Satr N : **string**): **string**	Bir necha satrni bittaga birlashtirish
Copy (Satr : **string**; Belgi Raqami : **integer**): **string**	Ichki satrni ajratish
Delete (**var** Satr : **string**; Belgi Raqami : **integer**; Qancha : **integer**)	Satr qismini olib tashlash
Length (Satr : **string**) : **integer**	Satr uzunligi
Pos (Satr : **string**; Ichki satr : **string**) : **byte**	Ichki satrning satrdagi o'rni
Chr (Belgi Kodi : **byte**) : **Char**	Ko'rsatilgan koddagi belgi

Funksiya va protseduralar bayonida quyidagi belgilashlar qabul qilingan:

- funksiya va protseduralar ismi yarim yo'g'on shrift bilan ajratilgan;
- parametrlar kursiv bilan ajratilgan; (parametr sifatida o'zgarmaslar, o'zgaruvchilar yoki tegishli turdagi ifodalar ishlatilishi mumkin. Agar parametr sifatida asosiy dasturning o'zgaruvchisi bo'lishi shart bo'lsa, uning oldiga, albatta, **var** so'zi, parametrdan keyin uning turi yozilishi kerak);
- shart bo'lmagan parametrlar kvadrat qavslarda yozilgan;
- funksiya parametrlari ro'yxatidan keyin (ikki nuqta orqali:) funksiya qaytaradigan natija turi ko'rsatiladi.

MATEMATIK FUNKSIYALAR

1. Abs

Sintaksis;

function Abs (x)

Butun yoki haqiqiy turdagi argumentning absolyut qiymatini qaytaradi.

2. Arctan

Sintaksis;

function Arctan (x : real) : real;

Radianlarda ifodalangan argument burchak miqdorining arktangensini hisoblaydi.

3. Cos

Sintaksis:

function Sos (x : real) : real;

Radianlarlarda ifodalangan argument burchak miqdorining kosinusini hisoblaydi.

4. EXP

Sintaksis:

function EXP (x : real) : real;

Argument eksponentasiga teng qiymatni hisoblaydi.

5. Ln

Sintaksis:

function Ln (x : real) : real;

Argumentning natural logarifmiga teng bo’lgan qiymatni hisoblaydi.

6. Sin

Sintaksis:

function Sin (x : real) : real;

Radianlarda ifodalangan argument burchak miqdorining sinusini hisoblaydi.

7. SQR

Sintaksis:

function SQR (x);

Butun yoki haqiqiy turdagi ifoda argumentining kvadratini hisoblaydi.

8. SQRT

Sintaksis:

function SQRT (x : real) : real;

Argumentdan kvadrat ildiz chiqarilgan qiymatni aniqlaydi.

9. Random

Sintaksis:

function Random [(coha : word)]

Agar Soha parametri ko’rsatilmagan bo’lsa, $0 \leq x \leq 1$ shartni qanoatlantiruvchi X tasodifiy sonni beradi. Agar Soha parametri ko’rsatilgan bo’lsa, funksiya $0 \leq x \leq$ Soha shartni qanoatlantiruvchi **Word** turidagi tasodifiy sonni qabul qiladi.

Izoh: **Random** funksiyasiga birinchi marta murojaat qilishda, **Randomize** protsedurasini chaqirish yordamida tasodifiy sonlar dastur generatorini initsializatsiya qilish zarur.

10. Randomize

Sintaksis:

prosedure Randomize;

Tasodifiy sonlar dastur generatorini initsializatsiyalaydi.

O’ZGARTIRISHNING FUNKSIYA VA PROTSEDURALARI

1. INT

Sintaksis:

function Int (x : real) : real;

Haqiqiy turdagi qiymat sifatida argumentga butun qismni beradi. SHakl almashtirishda argumentning kasr qismi hisobga olinmaydi, ya’ni funksiya yaxlitlamaydi.

2. Round

Sintaksis:

function Round (x : real) : real;

Argumentni eng yaqin butun songa yaxlitlaydi.

3. Str

Sintaksis:

procedure Str (x [:2 Belgilar Jami [:Kasr Qism]]; **var** Satr : **string**);

Sonli ifodani uning satrli bayoniga aylantiradi. Belgilar Jami va Kasr Qism – butun turdagi belgilarning umumiy sonini va son tasviridagi kasr qism belgilari miqdorini beruvchi, shart bo’lmagan, ifodadir.

4. Trunc

Sintaksis:

function Trunc (x : real) : longint;

Argumentning butun qismini butun turdagi qiymat sifatida beradi. SHakl almashtirishda argumentning kasr qismi hisobga olinmaydi, ya’ni funksiya yaxlitlanmaydi.

5. Val

Sintaksis:

Procedure Val (Satr : **string**; **var** O’zgaruvchi; **var** Xato : integer);

Butun yoki haqiqiy sonni ifodalovchi satrni songa aylantiradi. Hosil bo’lgan qiymat protsedurani chaqirishda ko’rsatilgan o’zgaruvchi tomonidan o’zlashtiriladi.

«Xato» o’zgaruvchi, agar shakl almashtirish bajarilmasa, uning amalga oshmasligiga sabab bo’lgan satr belgisi raqamini o’zlashtiradi. Agar shakl almashtirish muvaffaqiyat bilan bajarilsa, «Xato» ning qiymati nolga teng.

SATR VA BELGILAR BILAN ISHLASH UCHUN FUNKSIYALAR VA PROTSEDURALAR

1. Chr

Sintaksis:

function Chr (Belgi Kodi : byte) : char;

Ko’rsatilgan kod bilan belgini qabul qiladi.

2. Concat

Sintaksis:

function Concat (S1 [, S2, …, SN] : **string**) : **string**;

Funksiyani chaqirishda ko’rsatilgan qatorlar birlashmasi bo’lgan satrni qabul qiladi

3. Copy

Sintaksis:

function Copy (s : **string**; n : integer; l : integer) : **string**;

S satr qismi - ichki satrni qabul qiladi. Ichki satr n – raqamli belgidan boshlanadi va l belgidan iborat bo’ladi.

4. Delete

Sintaksis:

procedure Delete (**var** S: **string**; Belgi Raqami:integer; Qancha:integer);

S satrdan n raqami bilan boshlanuvchi va l belgidan iborat bo’lgan qismni o’chiradi.

5. Length

Sintaksis:

function Length (satr : **string**) : integer;

Argument – satr belgilari miqdoriga teng qiymatni qaytaradi.

6. POS

Sintaksis:

function POS (satr : **string:** Ichki satr : **string**) : byte;

Satrdagi ichki satr o’rnini (belgi raqamini) qaytaradi.

7. Satr va belgilar ishlatilgan dasturlarda ko’pincha grafik rejimning DetectGraph protsedurasi ishlatiladi.

DetectGrap

Sintaksis:

DetectGrap (**var** Drayver, Rejim : **integer**);

Grafik adapterni tekshiradi va qaysi grafik drayverb va rejim ishlatilayotganini aniqlaydi.

BOSHQA FUNKSIYALAR

1. ClrEol

Sintaksis:

procedure ClrEol;

Joriy (kursor turgan) satrni, kursor turgan xonadan satr oxirigacha TextBackGrand protsedurasi bergan rang bilan bo’yab, tozalaydi.

2. ClrScr

Sintaksis:

procedure ClrScr;

Ekranning joriy oynasini (Windows protsedurasi o’rnatgan) TextBackGrand protsedurasi bergan rang bilan bo’yab, tozalaydi.

3. Halt

Sintaksis:

procedure Halt [(Tugatish kodi : word)];

Dastur bajarilishini tugatadi va boshqarishni operatsion tizimga uzatadi.

4. ReadKey

Sintaksis:

function ReadKey : char;

Bosilgan klavishga mos belgini beradi. Ord funksiyasidan foydalanib, bosilgan klavish kodini hosil qilish mumkin. Agar xizmat klavishi bosilgan bo'lsa (masalan, kursorni siljitish klavishi), ReadKey funksiya O kodlik belgini beradi. Bu holda xizmat klavishini o'xshatish uchun yana bir marta ReadKey funksiyani chaqirish kerak, bu holda u xizmat kodi klavishini beradi. Quyida tez-tez ishlatiladigan ba'zi bir klavishlarning kodlari keltirilgan.

Klavish	Kod
<**Esc**>	27
<**Backspace**>	8
<**Enter**>	13
<**Space**>	32
<↑>	0; 72
<↓>	0; 80
<←>	0; 75
<→>	0; 77

5. TextBackGround

Sintaksis:

procedure TextBackGround (rang : byte);

Write va writeln ko'rsatmalari bilan chiqariladigan axborotlar tag rangini beradi. Rang parametri sifatida quyida nomlangan o'zgarmaslarning birontasi ishlatilishi mumkin.

O'zgarmas	Rang	Rang raqami
Black	Qora	0
Blue	Ko'k	1
Green	Yashil	2
Cyan	Firuza	3
Red	Qizil	4
Magenta	Binafsha	5
Brown	Jigarrang	6
LightGray	Oq	7

6. TextColor

Sintaksis:

procedure TextColor (rang : byte);

Write va writeln ko’rsatmalari bilan chiqariladigan axborot belgilarining rangini o’rnatadi. Rang parametri sifatida quyida keltirilgan nomlangan o’zgarmaslarning biridan foydalanish mumkin:

O’zgarmas	Rang	Rang raqami
Black	Qora	0
Blue	Ko’k	1
Green	Yashil	2
Cyan	Firuza	3
Red	Qizil	4
Magenta	Binafsha	5
Brown	Jigarrang	6
LightGray	Oq	7
DarkGray	Kulrang	8
LightBlue	Havorang	9
LightGreen	Ochiq yashil	10
LightCyan	Ochiq firuza	11
LightRed	Ochiq qizil (alvon rang)	12
LightMagenta	Ochiq binafsha	13
Yellow	Sariq	14
White	Oq, yarqiroq	15

II. AMALIY QISM.

MASALA YECHIMINING DELPHI 7 DAGI TALQINI.

Yaratmoqchi bo’lgan dasturimizning ko’rinishi ixcham, sodda va qulay bo’lishi kerak.

Dastur interfeysini 6 ta **Label,** 2 ta **Edit** va 1 ta **Button** komponenti yordamida hosil qilamiz. Аввал уанги форма hосил қилингда, унга юқоридаги компонентларни жойлашtиринг ва уларнинг hусусиуатини қуйидаги жадваллар асосида созланг:

Form1

Hususiyat	Qiymati	Izoh
Caption	EKUB va EKUKni hisoblash	*Forma sarlavhasi*
ClientHeight	172	*Forma ishchi maydoni balandligi*
Position	poDesktopCenter	*Dastur ishga tushganda odatdagi pozitsiyasi*
BorderStyle	bsDialog	*Forma stili (dialog oynasi bo’rinishida)*
Name	Form1	*Dastur matnida komponentga murojaat qilish imkonini beruvchi unikal nom*
ClientWidth	350	*Forma ishchi maydoni uzunligi*

Label1

Hususiyat	Qiymati	Izoh
Caption	Birinchi son:	*Izoh*
Left	24	*Chapdan komponentgacha masofa*
Top	16	*Yuqoridan komponentgacha masofa*

Label2

Hususiyat	Qiymati	Izoh
Caption	Ikkinchi son:	*Izoh*
Left	144	*Chapdan komponentgacha masofa*
Top	16	*Yuqoridan komponentgacha masofa*

Label3

Hususiyat	Qiymati	Izoh
Caption	EKUB	*Izoh*
Left	24	*Chapdan komponentgacha masofa*
Top	80	*Yuqoridan komponentgacha masofa*

Label4

Hususiyat	Qiymati	Izoh
Caption		*Izoh*
Left	144	*Chapdan komponentgacha masofa*
Top	80	*Yuqoridan komponentgacha masofa*

Label5

Hususiyat	Qiymati	Izoh
Caption	EKUK:	*Izoh*
Left	24	*Chapdan komponentgacha masofa*
Top	112	*Yuqoridan komponentgacha masofa*

Label6

Hususiyat	Qiymati	Izoh
Caption		*Izoh*
Left	144	*Chapdan komponentgacha masofa*
Top	112	*Yuqoridan komponentgacha masofa*

Edit1

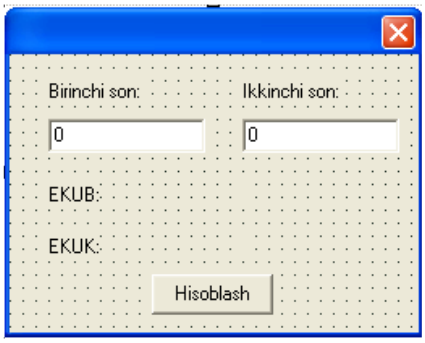
Hususiyat	Qiymati	Izoh
Text	0	*Yozuv maydonidagi odatdagi yozuv*
Cursor	clBeam	*Kursor turini aniqlash (tahrir rejimi)*
TabOrder	0	*Tab tugmasi bosilganda komponentga fokusni o'tkazishdagi tartib raqam*
Left	24	*Chapdan komponentgacha masofa*
Top	40	*Yuqoridan komponentgacha masofa*
Width	97	*Komponent uzunligi*

Edit2

Hususiyat	Qiymati	Izoh
Text	0	*Yozuv maydonidagi odatdagi yozuv*
Cursor	clBeam	*Kursor turini aniqlash (tahrir rejimi)*
TabOrder	1	*Tab tugmasi bosilganda komponentga fokusni o'tkazishdagi tartib raqam*
Left	24	*Chapdan komponentgacha masofa*
Top	40	*Yuqoridan komponentgacha masofa*
Width	97	*Komponent uzunligi*

Button1

Hususiyat	Qiymati	Izoh
Caption	Hisoblash	*Tugma sarlavhasi*
Cursor	crHandPoint	*Kursor turini aniqlash (ko'rsatkich barmoq)*
TabOrder	2	*Tab tugmasi bosilganda komponentga fokusni o'tkazishdagi tartib raqam*
Left	88	*Chapdan komponentgacha masofa*
Top	136	*Yuqoridan komponentgacha masofa*
Width	75	*Komponent uzunligi*



DASTURIY QISM.

Birinchi navbatda sonlarni kiritishdagi xatoliklarni tekshirib ko'rishimiz lozim. Bunda
- kiritilgan son butun son bo'lishi;
- ulardan xech biri nolga teng bo'lmasligi kerak.

Yuqoridagi muammolarni hal qilish uchun dasturlash tilining standart imkoniyatlaridan voz kechgan holda o'zimiz hatolikka tekshirish jarayonini yaratib olamiz. Birinchi navbatda **Edit** ga kiritilgan qiymat butun son ekanligini tekshirishimiz kerak. Buning uchun **Hisoblash** tugmasi bosilgan holda Edit1 va Edit2 lardagi matnning har bir belgisi 0 dan 9 gacha bo'lgan sonlardan biri ekanligini aniqlashimiz kerak (k=1 dan Edit1 hamda Edit2 matni uzunligigacha sikl ochish yordamida). Hato aniqlangan holda quyidagi ishlarni amalga oshiramiz:

1. Xato haqida habar beramiz;
2. Edit1 va Edit2 ning matnini '0' ga tenglaymiz;
3. Boshqaruvni dastur oxiriga uzatamiz.

```
…
procedure TForm1.Button1Click(Sender: TObject);
label t;                          //Boshqaruv uzatilishi uchun metka e'lon qilindi
var     k:integer;                //sikl uchun o'zgaruvchi
```

```
begin
        for k := 1 to Length(Edit1.Text) do   {k=1 dan Edit1 matni uzunligigacha sikl ochildi}
        if not (Edit1.Text[k] in ['0'..'9']) then
        {agar Edit1 matnining k o'rinda turgan belgisi 0 da 9 gacha bo'lgan raqamlardan biri bo'lmasa}
        begin
          ShowMessage('Qiymat no"to"g"ri kiritildi');      //Hato haqida habar berildi
          xato:=true;
         Edit1.Text:='0';
         Edit2.Text:='0';
          goto t;              //Boshqaruv dastur oxiridagi t metkaga uzatildi. Bu holda
        end;                   //oradagi operatorlar bajarilmaydi
t:
end;
…
```

Yuqoridagi tekshirishni Edit2 uchun ham yozamiz:

```
for k := 1 to Length(Edit2.Text) do
if not (Edit2.Text[k] in ['0'..'9']) then
begin
  ShowMessage('Qiymat no"to"g"ri kiritildi');
  Edit1.Text:='0';
  Edit2.Text:='0';
  goto t;
end;
```

Navbatdagi vazifamiz kiritilgan sonlarning noldan farqli ekanligini tekshirishdan iborat. Chunki dastur davomida bo'lish amalini bajaramiz. Qiymatlardan birining 0 ga teng bo'lishi esa bo'lish amalining hato bilan yakunlanishiga olib keladi. Buni amalga oshirish uchun ikki sondan biri 0 ga teng bo'lgan holda ularning ko'paytmasi nolga teng bo'lishini yodga olishimiz kifoya. Shu hususiyatdan foydalangan holda, Edit1 va Edit2 matnidagi yozuvning tipini **Integer** ga aylantirib olib, ularning ko'paytmasini noldan farqli ekanligini tekshiramiz. Agar ko'paytma nolga teng bo'lsa quyidagi ishlarni amalga oshiramiz:

1. Xato haqida habar beramiz;
2. Edit1 va Edit2 ning matnini '0' ga tenglaymiz;
3. Boshqaruvni dastur oxiriga uzatamiz.

```
if (StrToInt(Edit1.Text)*StrToInt(Edit1.Text)=0) then
begin
  ShowMessage('Qiymat no"to"g"ri kiritildi');
  Edit1.Text:='0';
```

```
  Edit2.Text:='0';
  goto t;
end;
```

Shu bilan hatolikka tekshirish jarayoni tugadi va tekshirish jarayonining kodini to’liq keltiraman:

```
…
        procedure TForm1.Button1Click(Sender: TObject);
        label t;
        var i,j,k,katta, kichik:integer;
        begin

                for k := 1 to Length(Edit1.Text) do
                if not (Edit1.Text[k] in ['0'..'9']) then
                begin
                  ShowMessage('Qiymat no"to"g"ri kiritildi');
                  Edit1.Text:='0';
                 Edit2.Text:='0';
                 goto t;
                end;

                for k := 1 to Length(Edit2.Text) do
                if not (Edit1.Text[k] in ['0'..'9']) then
                begin
                  ShowMessage('Qiymat no"to"g"ri kiritildi');
                  Edit1.Text:='0';
                  Edit2.Text:='0';
                 goto t;
                end;

                if (StrToInt(Edit1.Text)*StrToInt(Edit2.Text)=0) then
                begin
                  ShowMessage('Qiymat no"to"g"ri kiritildi');
                  Edit1.Text:='0';
                 Edit2.Text:='0';
                 goto t;
                end;

        t:
        end;
…
```

Ikki sonning EKUB va EKUK ini aniqlashga kirishishdan oldin ulardan qaysi biri katta, qaysi biri kichik ekanini aniqlab olishimiz lozim. Chunki

algoritmimizning keyingi qismlari shuni taqozo etadi (sababini dasturni tuzish davomida tushunib olasiz). Agar ularning qiymati o'zaro teng bo'lsa, bu haqida habar berilib, Edit larning qiymatini '0' tenglangandan keyin boshqaruv dastur oxiriga uzatilishi kerak. Ular teng bo'lmagan holda esa kichigining qiymati **kichik**, kattasining qiymati **katta** o'zgaruvchisiga o'zlashtiriladi:

```
// Agar edit1 ning matni edit2 ning matnida son sifatida katta bo'lsa
if StrToInt(Edit1.Text)>StrToInt(Edit2.Text) then
  begin
    katta:=StrToInt(Edit1.Text);
    kichik:=StrToInt(Edit2.Text);
  end
else
//aks holda
// Agar edit1 ning matni edit2 ning matnida son sifatida kichik bo'lsa
if StrToInt(Edit1.Text)<StrToInt(Edit2.Text) then
  begin
    katta:=StrToInt(Edit2.Text);
    kichik:=StrToInt(Edit1.Text);
  end
// aks holda, ya'ni ikkisi o'zaro teng bo'lgan holda
else
  begin
  ShowMessage('Sonlar teng!');
  Edit1.Text:='0';
  Edit2.Text:='0';
  goto t;
  end;
```

Ikki sonning EKUBini topishning eng oson usuli – bu Evklid usuli bo'lib, u katta sondan kichik sonni ayirib tashlash hisobiga katta son tarkibidagi ortiqcha qiymatdan qutulish natijasida har ikki sonni bosqichma-bosqich kichraytirib borish va nihoyat ikki son o'zaro teng bo'lgan holda jarayonni to'xtatish evaziga EKUBni topishga asoslangan. Bunda sonlarning o'zaro teng bo'lgandagi qiymati EKUB bo'ladi:

```
while katta<>kichik do
  if katta>kichik then
    katta:=katta-kichik
  else
    kichik:=kichik-katta;
```

Yuqoridagi holda **katta** va **kichik** o'zgaruvchilarining qiymati ahamiyatga ega emas. Shuning uchun keltirilgan kodni quyidagicha yozish ham hato hisoblanmaydi:

```
while kichik<>katta do
    if  kichik>katta  then
       kichik:=kichik-katta
   else
      katta:=katta-kichik;
```

Har ikki holda ham EKUB ni **kichik** yoki **katta** o’zgaruvchisining qiymatidan olishimiz mumkin. Faqat natijani **Label4** komponenti sarlavhasida e’lon qilishidan avval, natija 1 ga teng bo’lib qolmaganiga ishonch hosil qilishimiz kerak. Chunki faqat o’zaro tub sonlardagina EKUB 1 ga teng bo’ladi:

```
if kichik<>1 then
    Label4.Caption:=IntToStr(kichik)
  else
    Label4.Caption:='O"zaro tub son';
```

yoki

```
if katta<>1 then
    Label4.Caption:=IntToStr(katta)
  else
    Label4.Caption:='O"zaro tub son';
```

Sonlarning EKUK ini topish ham unchalik murakkab masala emas. Faqat maktab darsliklarida o’rgatilgan an’anaviy usulni “unutib”, biroz mantiqiy fikrlasak, hammasi osongina hal bo’ladi. Tasavvur qilamiz, bizga 14 va 4 soni berilgan bo’lsin. Avvalo ulardan kattasi kichigiga qoldiqsiz bo’linish-bo’linmasligini tekshirib ko’ramiz. Agar qoldiqsiz bo’linganda edi, ulardan kattasi EKUK, kichigi esa EKUB bo’lar edi (masalan 8 va 4, 12 va 6 va h.k.). Natijada EKUB va EKUK ni topishga qaratilgan siklli jarayonlarni amalga oshirishning hojati qolmasdi. Keling shu fikrimiz asosida EKUB ni topishga kirishishdan ham oldin sonlar o’zaro qoldiqsiz bo’lingan hol uchun natijani **Label4** va **Label6** larda e’lon qilib, boshqaruvchi dastur oxiriga uzatish algoritmini kod sifatida kiritib qo’yamiz:

```
  if (katta mod kichik = 0) then   // mod amali qoldiqli bo’lishni ifodalaydi. Masalan 8 ni
    begin                          // 4 ga bo’lishdagi qoldiq 0 ga teng
     Label4.Caption:=IntToStr(kichik);
     Label6.Caption:=IntToStr(katta);
     goto t;
    end;
```

Yuqoridagi hususiy hol yuzaga kelmagan taqdirda sikl yordamida EKUK ni qidirish jarayonini yaratishimizga to’g’ri keladi. Shu o’rinda Hususiy holni kengroq tahlil qilib ko’raylik:

8 va 4 sonining EKUKini topish kerak bo’lsin.

8 ni 4 ga bo’lsak, qoldiq 0 ga teng.

Shu ifodani o’zgartirib yozaman:

8 mod 4 = 0

Yana biroz o’zgartirish kiritaman:

*8*1 mod 4 = 0*

Nima demoqchi ekanligimni tushunmadingiz, to’g’rimi?! Unday bo’lsa yana misol keltiraman: 12 va 8

*12*1 mod 8 = 4* *// 12 ni 1 ga ko’paytirsak, 12 ga teng. Uni 8 ga bo’lsak, qoldiq 4.*
// natija bizni qoniqtirmagani uchun davom etamiz

*12*2 mod 8 = 0* *// 12 ni 2 ga ko’paytirsak, 24 ga teng. Uni 8 ga bo’lsak, qoldiq 0.*
// demak 12 va 8 ga bo’linadigan eng kichik son – 24 ekan

Yuqoridagilardan kelib chiqqan holda hulosa qilamiz:

Sonlarning EKUKini topish uchun ulardan birini 1, 2, 3, … , IkkinchiSon ga ko’paytirib, har safar ko’paytirgandan so’ng **natija**ni ikkinchi songa bo’lib ko’ramiz. Birinchi qoldiqsiz bo’lingan **natija** shu sonlarning EKUKi bo’ladi. Ko’paytirish jarayoni ikkinchi songa ko’paytirguncha davom etishi yoki 1 dan IkkinchiSon gacha bo’lgan oraliqda to’xtalishi mumkin. Ko’paytirish IkkinchiSon gacha davom etgan taqdirda jarayon juda uzoq davom etmasligi uchun IkkinchiSon sifatida sonlardan kichigini tanlash maqsadga muvofiq (avvalgi o’rinlarda **katta** va **kichik** qiymatlarni aniqlab olgandik). Masalan 12 va 123121 sonlari uchun EKUK topishda ko’paytirishni 12 ga nisbatan olinsa, bu jarayon 123121 marta amalga oshiriladi. Chunki ularning EKUKi 12x123121=1477452 ga teng. Aksincha, 123121 ga nisbatan ko’paytirish amalga oshirilsa, bu jarayon atigi 12 marta takrorlanadi:

```
for k := 1 to kichik do
    if (katta*k mod kichik = 0) then
      begin
        Label6.Caption:=IntToStr(katta*k);
        Break
      end;
```

Kod tarkibidagi **Break** birinchi qanoatlantruvchi natija kelib chiqishi bilan siklni to’xtarishga hizmat qiladi.

Yana bir eslatma: garchi EKUKni topish algoritmini dastur so’ngida ko’rib chiqqan bo’lsakda, uning kodini EKUB ni topishdan avval yozilishi kerak. Chunki EKUBni topish jarayonida **katta** va **kichik**larning qiymati o’zgarib, o’zaro teng qiymatgacha kichraytirilgan bo’ladi.

Dasturning to’liq kodi:

```
unit Unit1;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls;

type
  TForm1 = class(TForm)
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    Label5: TLabel;
    Label6: TLabel;
    Label4: TLabel;
    Edit1: TEdit;
    Edit2: TEdit;
    Button1: TButton;
    procedure Button1Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;

implementation

{$R *.dfm}
```

```
procedure TForm1.Button1Click(Sender: TObject);
label t;
var k,katta, kichik:integer;

begin

  for k := 1 to Length(Edit1.Text) do
    if not (Edit1.Text[k] in ['0'..'9']) then
      begin
        ShowMessage('Qiymat no''to''g''ri kiritildi');
        Edit1.Text:='0';
        Edit2.Text:='0';
        goto t;
      end;

  for k := 1 to Length(Edit2.Text) do
    if not (Edit2.Text[k] in ['0'..'9']) then
      begin
        ShowMessage('Qiymat no''to''g''ri kiritildi');
        Edit1.Text:='0';
        Edit2.Text:='0';
        goto t;
      end;

    if (StrToInt(Edit1.Text)*StrToInt(Edit2.Text)=0) then
      begin
        ShowMessage('Qiymat no''to''g''ri kiritildi');
        Edit1.Text:='0';
        Edit2.Text:='0';
        goto t;
      end;

    if StrToInt(Edit1.Text)>StrToInt(Edit2.Text) then
      begin
        katta:=StrToInt(Edit1.Text);
        kichik:=StrToInt(Edit2.Text);
      end
    else
    if StrToInt(Edit1.Text)<StrToInt(Edit2.Text) then
      begin
        katta:=StrToInt(Edit2.Text);
        kichik:=StrToInt(Edit1.Text);
      end
    else
      begin
```

```
        ShowMessage('Sonlar teng!');
        Edit1.Text:='0';
        Edit2.Text:='0';
        goto t;
      end;

    if (katta mod kichik = 0) then
     begin
      Label4.Caption:=IntToStr(kichik);
      Label6.Caption:=IntToStr(katta);
      goto t;
     end;

    for k := 1 to kichik do
      if (katta*k mod kichik = 0) then
        begin
          Label6.Caption:=IntToStr(katta*k);
          Break
        end;

    while katta<>kichik do
      if katta>kichik then
        katta:=katta-kichik
      else
        kichik:=kichik-katta;

    if kichik<>1 then
      Label4.Caption:=IntToStr(kichik)
     else
      Label4.Caption:='O"zaro tub son';
t:
end;
end.
```

MASALA YECHIMINING TURBO PASKAL 7.0 DAGI TALQINI.

Dasturning Delphidagi talqinini yaratib olganimizni hisobga oladigan bo'lsak, bizning oldimizda muammo deyarli qolmadi. Faqat foydalanuvchi tomonidan kiritilgan qiymat butun sondan iborat bo'lmagan holda qiymatni kiritish haqidagi habarni qaytadan chiqarish jarayonini yaratsak bo'lgani. Dastur kodini murakkablashtirmaslik va mustaqil ravishda "kompyuterni aldash" ning yo'llarini ko'rib chiqish maqsadida dastur algoritmini quyidagicha tuzishni ma'qul ko'rdim:

1. EKUB va EKUKlari topilishi kerak bo'lgan sonlar uchun o'zgaruvchilarni **string** sifatida e'lon qilamiz. Shunda kompilyator uchun foydalanuvchi kiritayotgan belgining ahamiyati bo'lmaydi (harf yozib yuborilgan yoki o'nli kasr ko'rinishidagi son yozilgan taqdirda kompilyator hato tufayli dastur ishini to'xtatib qo'ymaydi);
2. O'zgaruvchilarga qabul qilingan qiymatlar tarkibida 0 dan 9 gacha bo'lgan raqamlardan boshqa belgi qatnashmaganligini tekshiramiz. Hato aniqlangan taqdirda boshqaruvni qiymat kiritish kerakligi haqida habar beriladigan qatorga qaytaramiz. Shu usul bilan to'g'ri qiymat kiritilguncha dasturning keyingi qismi bajarilmaydi;
3. Bizni qanoatlantiruvchi qiymat kiritilgan taqdirda **string** tipidagi o'zgaruvchilarning tipini **integer** ga aylantirib qo'yamiz.

Agar tushunarli bo'lgan bo'lsa, boshlaymiz.
Demak hozircha kerakli o'zgaruvchilar quyidagilar:

```
uses crt;
label 1,2;
var
  code : integer;
  I : byte;
  a, b : string;          {qiymatlarni string sifatida qabul qilish uchun}
  _a, _b : integer;  {string tipli o'zgaruvchilarni integer tipli o'zgaruvchiga}
                     {o'zlashtirish uchun}
…
```

Foydalanuvchiga birinchi qiymatni kiritish haqidagi habarni berib, u kiritgan qiymatni **a** o'zgaruvchilarga o'zlashtiramiz:

```
Writeln('Birinchi butun sonni kiriting:');
Readln(a);
```

Kiritilgan qiymat butun son ekanligini tekshiramiz. Buning uchun qiymatning har bir belgisi 0 dan 9 gacha bo'lgan raqamlardan biri ekanligi aniqlaymiz. Aks holda boshqaruvni qiymat kiritish kerakligi haqida habar beriladigan qatorga uzatamiz:

```
…
```

```
1:
Writeln('Birinchi butun sonni kiriting:');
Readln(a);
for i:=1 to length(a) do
if not (a[i] in ['0'..'9']) then goto 1;
…
```

Qiymat to’g’ri kiritilgan taqdirda **string** tipli **a** o’zgaruvchining qiymatini **integer** sifatida **_a** o’zgaruvchiga o’zlashtiramiz:

```
val(a,_a,_code);
```

Yuqoridagi kodni ikkinchi son uchun ham qayta yozamiz:

```
…
2:
Writeln('Ikkinchi butun sonni kiriting:');
Readln(b);
for i:=1 to length(b) do
if not (b[i] in ['0'..'9']) then goto 2;
val(b,_b,code);
…
```

Endi algoritmning asosiy qismiga o’tishimiz mumkin. Avval ta’kidlaganimdek, dasturning Delphidagi talqinini yaratib bo’lganimiz uchun, uni biroz o’zgartirgan holda ko’chirib qo’ysak bo’lgani:

```
uses crt;
label 1, 2, 3;
var
  …
  _a, _b : integer;

begin

 …
   if _a>_b then
     begin
       katta:=_a;
       kichik:=_b;
     end
   else
   if _a<_b then
     begin
       katta:=_b;
       kichik:=_a;
```

```
    end
  else
    begin
      Writeln('Sonlar teng!');
      goto 3;
    end;

  if (katta mod kichik = 0) then
   begin
    writeln('EKUB ',kichik);
    writeln('EKUK ',katta);
    goto 3;
   end;

  for i := 1 to kichik do
    if (katta*i mod kichik = 0) then
      begin
        writeln('EKUK ',katta*i);
        Break
      end;

  while katta<>kichik do
    if katta>kichik then
      katta:=katta-kichik
    else
      kichik:=kichik-katta;

  if kichik<>1 then
    writeln('EKUB ',kichik)
   else
    writeln('O''zaro tub son');

3:

readln;
end.
```

Yuqoridagi kodning tuzilishida xech qanday mantiqiy o’zgarish yo’q. Faqat **Edit1.Text** va **Edit2.Text** o’rnida mos ravishda **_a** va **_b** o’zgaruvchilardan foydalanilgan. Natijani e’lo qilish uchun esa Lebel komponenti o’rnida Writeln protsedurasidan foydalandim.

Natijani olishdan avval ekranni tozalab olish uchun dastur boshida **ClrScr** protsedurasini yozib, undan keyin dastur nomini ekranga chiqarib qo’yamiz:

```
…
```

```
begin
        ClrScr;
        Writeln('EKUB va EKUK ni hisoblash:');
        Writeln('-------------------------');
…
```

Dasturning to’liq kodi:

```
uses crt;
label 1,2,3;
var
  i:byte;
  a, b:string;
  code, _a, _b, katta, kichik:integer;
begin
ClrScr;
Writeln('EKUB va EKUK ni hisoblash:');
Writeln('-------------------------');

1:
Writeln('Birinchi butun sonni kiriting:');
Readln(a);
for i:=1 to length(a) do
if not (a[i] in ['0'..'9']) then goto 1;
val(a,_a,code);

2:
Writeln('Ikkinchi butun sonni kiriting:');
Readln(b);
for i:=1 to length(b) do
if not (b[i] in ['0'..'9']) then goto 2;
val(b,_b,code);

   if _a>_b then
     begin
       katta:=_a;
       kichik:=_b;
     end
   else
   if _a<_b then
     begin
       katta:=_b;
       kichik:=_a;
```

```
    end
  else
    begin
      Writeln('Sonlar teng!');
      goto 3;
    end;

  if (katta mod kichik = 0) then
   begin
    writeln('EKUB ',kichik);
    writeln('EKUK ',katta);
    goto 3;
   end;

  for i := 1 to kichik do
    if (katta*i mod kichik = 0) then
      begin
        writeln('EKUK ',katta*i);
        Break
      end;

  while katta<>kichik do
    if katta>kichik then
      katta:=katta-kichik
    else
      kichik:=kichik-katta;


  if kichik<>1 then
    writeln('EKUB ',kichik)
   else
    writeln('O''zaro tub son');

3:

readln;
end.
```

III. FOYDALANILGAN ADABIYOTLAR

1. Д.Прайс. Программирование на языке Паскаль. Мир, 1987 г.

2. Окулов С. М. Программирование в алгоритмах. БИНОМ. Лаборатория знаний. 2002 г.

3. Фленов М. Е. Библия Delphi. — 2-е изд. БХВ-Петербург. 2008 г