O’ZBEKISTON RESPUBLIKASI ALOҚA, AXBOROTLAShTIRISh VA TELEKOMMUNIKATSIYA TEXNOLOGIYALARI DAVLAT QO’MITASI
TOShKENT AXBOROT TEXNOLOGIYALARI UNIVERSITETI

KOMPYUTER INJINIRING FAKULTETI
«Informatika asoslari» kafedrasi

« C++ da Dasturlash » kursi laboratoriya ishlari uchun C++Builderda grafika kompleks masalalarni bajarish uchun va laboratoriya ishlariga uslubiy ko’rsatmalar.

Toshkent-2016

TATU “Informatika asoslari” kafedrasi mudiri prof. A.X.Nishanovning rahbarligi ostida.

Mualliflar: dots.Xaydarova M.Yu., Nazirova M.X.

“ C++ da Dasturlash ” kursi laboratoriya ishlari uchun C++Builderda grafika kompleks masalalarni bajarish uchun va laboratoriya ishlariga metodik ko’rsatmalar “Kompyuter injiniring” fakultetining ilmiy-metodik kengashida ko’rib chiqildi va nashr qilinishiga ruxsat berildi.

“C++ da Dasturlash” kursi laboratoriya ishlari uchun C++Builderda grafika kompleks masalalarni bajarish uchun va laboratoriya ishlariga metodik ko’rsatmalar 1 kurs talabalariga 1 semestr uchun.

Metodik ko’rsatmalar C++ Builder grafikasini ishlatishda muhim bo’lgan tushunchalar va printsiplar haqida qisqa ma’lumotlarni o’z ichiga olgan. “Dasturlash asoslari” qonunlari o’quv rejasiga asoslangan holda, kompleks masalalar va laboratoriya ishlarini bajarish uchun, metodik ko’rsatmada aloxida topshiriqlar xam keltirilgan. Amaliy bilimlarga ega bo’lish uchun qo’llanmada grafik tasvirlar qurishdagi eng sodda grafikalar namunalari xam keltirilgan. Metodik ko’rsatma ma’ruza darslarini o’zlashtirishda, laboratotiya mashg’ulotlarga tayyorgarlik ko’rishda va bajarishda talabalarga qo’l keladi

Toshkent axborot texnologiyalari universiteti 2016 y.

KIRISh

Borland C++ Builder, nafaqat oddiy grafikalar chizadigan, balki tasvirlar, yozuvlar va grafik fayllar bilan ishlaydigan, o'ziga xos komponent va sinflarga ega. Borland C++ Builder da grafik tasvirlar bilan ishlashning printsiplari, Windows operatsion tizimida grafik ob'ektlar yaratishning umumiy printsiplariga asoslangan. Aynan Windows operatsion tizimi dasturchiga grafik qurilmalar interfeysining vositalarini taqdim etadi. Borland C++ Builder komponentlari va sinflari bilan birgalikda grafik funktsiyalardan foydalanish, dasturda grafika yaratish metodikasining yanada samaradorliroqligini o'zida aks ettiradi.

C++ da grafikalardan foydalanishni o'rganish, laboratoriya ishlari va kompleks masalalarni bajarish, quyidagi intizomlarga asoslangan: informatika, dasturlash, oliy matematika,injenerlik grafikasi o'zaklari. Ammo berilgan kursda, ushbu intizom borasidagi bilimlaridan komplesks tarzda foydalanishlari talab etiladi, bu esa o'z navbatida ma'lum bir qiyinchiliklar tug'diradi.

Shu qiyinchiliklarni yengish uchun, C++ dasturlash tilida mustaqil ishlarni bajarish misollari bilan metodik ko'rsatmalar ma'lumotlari yozilgan.

Berilgan metodik ko'rsatmalar, grafik funktsiyalardan foydalanilgan dasturlarining minimal misollar yig'indisini o'z ichiga olgan.Ular, shubxasiz, grafik tasvirlarni dasturlashda va elektrotexnik sxemalar tuzishda talabalarga qo'l keladi.

C++ Builder ning afzalliklari shundaki, u (metod) funktsiyasini ko'rsatkich bo'yicha chaqiradi va yaratilgan sinfning tayyor xususiyatlarini, katta tanlash imkoni bilan, boshqa sinflar yaratadi.

Asosiy nazariy ma’lumotlar
C++grafikasi

C++Builder muhitida, grafikaga aloqasi bor ob’ektlarning 3 xil turi mavjud:

1) Kanva–grafik chiqishi uchun ishlatilishi mumkin bo’lgan, dastur oynasi ustining bitli kartasini, komponentlar, printerlar va h.k larni taqdim etadi.Kanva mustaqil ob’ekt emas, u doim boshqa bir grafik ob’ektning xossa bo’ladi.

2) Grafika–biror bir fayl yoki resursning (bitli obrazni, piktogrammani yoki metafaylni)rastrli tasvirini tashkil qiladi.

3) C++Builder boshlang’ich TGraphic sinfidan quyidagi ob’ektli sinflarning o’zgaruvchilarini aniqlaydi.:

- TBitmap,
- Ticon,
- TMetafile.

4) Tasvir (TPicture) grafik ob’ektlarning istalgan sinfini o’z ichiga olish imkoniga ega, grafiklar uchun konteynerni tashkil qiladi. Natijada TPicture, foydalanuvchi tomonidan belgilangan, bitli obraz, piktogramma, metafayl yoki boshqa bir grafik tipni o’z ichiga olishi mumkin, dastur esa TPicture ob’ekti orqali konteynerning barcha ob’ektlariga murojat qilishi mumkin.

Tbitmap sinfi, Borland C++ Builder da Graphic::TBitmap deb belgilangan, u grafik tasvirlarni yaratish va ularning xususiyatlarini boshqarish, xotirani o’qish va diskda saqlash xususiyat va metodlarga ega.Graphic::Tbitmap piksellar massivi ko’rinishidagi rastrli grafik tasvirlarni qo’llagani kabi, bmp. fortmatidagi tasvirlarni xam qo’llaydi. Sinfning asosiy xususiyati − Canvas. TGraphic sinfi TBitmap grafika bilan ishlash uchun minimal standart interfeysni tashkil etadi.

TIcon va TMetafile sinflari ham Tgraphic sinfining erkin tashkil etuvchilaridir, ular ham grafika bilan ishlash uchun xossa va metodlariga ega, lekin TBitmap dan farqli tomoni shundaki, *Canvas* xossasiga ega emas. Buni o’z navbatida, *.ico *. va wmf (*.emf) formatidagi tasvirlardan foydalanish va o’zgacha qurilishi ko’rsatib beradi.

Tasvir grafika uchun konteynerdir, ya’ni u grafik ob’ektlarning barcha sinflarni o’z ichiga olishi mumkin. Konteynerli TPicture sinf bitli obrazga, piktogrammaga, metafayl va boshqa, foydalanuvchi tomonidan belgilangan grafik tipga ega bo’lishi mumkin, dastur esa “tasvir” ob’ekti orqali, konteynerdagi barcha ob’ektlarga standartlashgan holatda murojat qilishi mumkin.

Xozirgi vaqtda, zamonaviy kompyuterlarda ma’lumot chiqarishning grafik rejimi qo’llaniladi, Windows operatsion tizimi esa simvolli rejimni tanimaydi. DOS operatsion tizimdan reallashtirish bilan, C++ dasturlash tilida, grafik funtsiyalar graphics.lib kutubxonasida saqlanadi, bu funktsiyalarning protiplari (xabarlar) esa graphics.h. faylida joylashgan bo’ladi.

Windows da bir necha o’nlab grafik funktsiyalar bor. Ularning nomlari DOSdagi xos vositalar nomi bilan qisman mos keladi. Bu funktsiyalarning nomidagi bosh xarflarga etibor berish lozim, bu C++ da dastur yig’ishda kerak bo’ladi. DOS muhitida , grafik funktsiyalarning nomlari, yozma lotin xarflari bilan yoziladi, bu C++tiliga hosdir. 1997 yilda C++tili uchun dasturlash muhitining ko’rgazmali namunasi paydo bo’ldi, u xam Borland kompaniyasi tomonidan yaratildi va C++ Builder nomiga ega bo’ldi. C++ Builder afzalliklari, yaratilgan sinfning ko’pkina tayyor vositalari bilan, avtomatik tarzda boshqa bir sinf yaratishi va ko’rsatkich bo’yicha (metod) funktsiyasini chaqirishidir.

Grafika bilan ishlan uchun funktsiyalar

Eng oddiy funktsiyalar grafik primitivlar deb ataladi. Primitivlarni, shartli ravishda quyidagi guruhlarga bo’lish mumkin:konturlar chizishning primitivlari va yuzali figuralar. Grafik primitizlarni chiqarish, shakl yuziga to’g’ri keladi. Bu yuza *Canvas* deb ataladi.

Kontur primitivlari: chiziqlar (line), to’g’riburchaklar (rectangle), yoy (arc), aylanalar (circle), ellipslar (ellipse), ko’pburchaklar (drawpoly) va boshqa ichi bo’yalmaydigan shakllar.

Yuzali ichi to’ldiriluvchan shakllarga, bo’yaladigan to’g’riburchaklar (FillRect), dumaloq va ellipssimon sektorlar (Pie) kiradi. Agarda siz chizadigan shakl yopiq, ammo bo’yalmaydigan bo’lsa, masalan, (Poligon) ko’pburchagi, u holda (FloodFill, FillStyle) bo’yash funktsiyasi yordamida uni shtrixlash mumkin.Chizish rangini tanlash uchun, grafik primitivlarning rang berish funktsiyasidan foydalaniladi (Pen− >Color).

Jadval **1** − kanvaning metod va xossalari

Metod **(**Funktsiya**)**	xossa	harakat
MoveTo	PenPos	Qalamning dastlabki holatini aniqlaydi
LineTo	PenPos	Berilgan nuqtagacha to’g’ri chiziq chizadi

Rectangle		To’g’riburchak chizadi
Ellipse		Ellips chizadi
Arc		Yoy chizadi
Polyline		Siniq chiziq chizadi
PolyBezier		Qiyshiq Beze chizadi
Chord		Sektor chizadi

DrawFocusRect		To'g'ri burchakli to'rtburchak chizadi
FrameRect		To’g’riburchak atrofidan ramka chiqaradi
Pie		Aylana sektorini chiqaradi
TextOut		Tekst chizig’ini chiqaradi
TextHeight		Tekst chizig’i balandligini belgilaydi
TextWidth		Tekst chizig’ini chiqarish uchun kenglikni belgilaydi
TextRect		To’g’ri burchak ichida matn yozish
FillRect		Belgilangan to’g’riburchakni rang bilan va dastlabki cho’tka shakli bilan to’ldirish.
FloodFill		Kanva maydonini (istalgan shaklda) berilgan rang bilan to’ldirish
	Pen	Qalamning rangi, uslubi, kengligi va rejimini o’rnatish uchun ishlatiladi
	Brush	Kanvaning fon va grafik shaklida rang va tuzilishini o’rnatish uchun ishlatiladi
	Font	Berilgan rang, o’lcham, va usuldagi shriftni o’rnatish uchun ishlatiladi
	Pixels	Berilgan kanva pikseli rangini o’qish va yozish uchun ishlatiladi

CopyRect	CopyMode	CopyMode rejimida kanvaning to'g'riburchak maydonini ko'chiradi
BrushCopy		Rang almashtirish bilan kanvaning to'g'riburchak maydonini ko'chiradi
Draw		Kanvaning berilgan joyida, bitli obraz, piktogramma, metafaylni chizadi
StretchDraw		Berilgan to'g'riburchakni to'liq to'ldirish uchun bitli obraz, piktogramma, metafaylni chizadi

C++ Builder muhitida grafika va *Canvas*-grafikasining funktsiyalari

Polotno (*Canvas*) C++ Builder polotno grafikasi, shakl "polotnosi" ustida grafik primitivlarni chizish uchun mo'ljallangan. Dastur grafikani *Canvas*(*Canvas* –chizish uchun polotno) xossasi to'g'ri keladigan shakl (yoki *Image* komponenti) yuziga chiqarishi mumkin. *Image* komponenti yoki shakl yuzida chiziq, aylana, to'g'riburchak, yoki boshqa bir grafik element (primitiv) paydo bo'lishi uchun, *Canvas* xossasiga mos keluvchi metod qo'llash (jad. 2) mumkin.

Masalan,

Forml->Canvas->Rectangle(10,10,50,50);

Shakl ustida to'g'riburchak chizadi.

Jadval 2–grafik primitivlarini o'chirish usullari

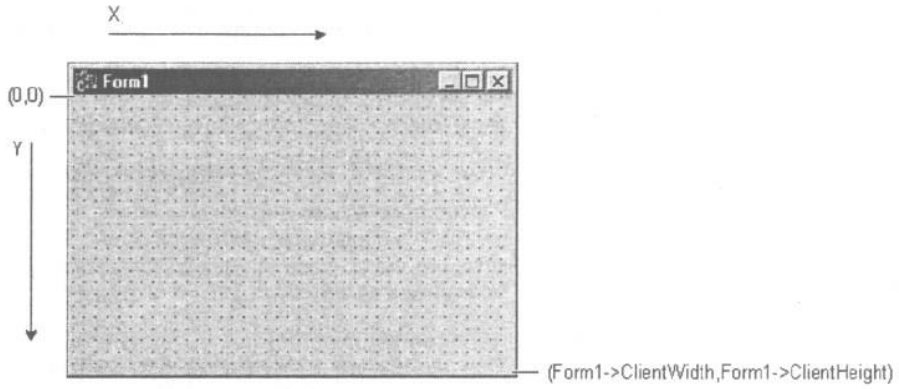
Usul	Xarakat
LineTo(*x*,*y*)	Belgilangan nuqtadan, koordinatalari berilgan nuqtagacha chiziq chizadi
Rectangle (*x*l,*y*l,*x*2,*y*2)	Chap tepa va o'ng past burchaklari *x*l,*y*l,*x*2,*y*2 – koordinatali bo'lgan to'g'ri to'rtburchak chizadi
FillRect (*x*l,*y*l,*x*2,*y*2)	Diagonal burchaklari *x*l,*y*l,*x*2,*y*2 – koordinatali bo'lgan bo'yalgan to'g'ri to'rtburchak chizadi

FrameRect (*x*l,*y*l,*x*2,*y*2)	Diagonal burchaklari *x*l,*y*l,*x*2,*y*2 – koordinatali bo'lgan to'g'ri to'rtburchak konturini chizadi
RoundRect (*x*l, *y*l,*x*2,*y*2,*x*3,*y*3)	Burchaklari dumaloqlashgan to'g'ri to'rtburchak chizadi
Ellipse (*x*l,*y*l,*x*2,*y*2)	Ellips yoki aylana chizadi, *x*l,*y*l,*x*2,*y*2 – ichiga ellips chiziladigan to'g'ri to'rtburchakning, yoki ichiga aylana chiziladigan kvadratning koordinatalari
Polyline (points,*n*)	Siniq chiziq chizadi, *points* – TPoint tipining massivi.

Polyline metodi, ketma ketlikda, koordinatalari massivda joylashgan nuqtalarni tekis kesmalar bilan bog'lab siniq chiziq chizib boradi: birinchisini ikkinchisi bilan, ikkinchisini uchinchisi bilan va h.k.

Grafik primitivlarni chiqarish metodlari *Canvas* xossasiga, ustida chizish mumkin bo'lgan abstrakt polotno sifatida qaraydi. Polotno, alohida nuqtalar-piksellardan tashkil topgan. Polotno yuzasida pikselning o'rni gorizontal (*X*) va vertikal (*Y*) koordinatalar bilan ifodalanadi. Koordinatalar pastdan tepaga va chapdan o'ngga tomon o'sib boradi. (rasm. 1). Shakl yuzasida chapdan tepadagi piksel koordinatalari (0,0), o'ngdan pastdagisi – (*ClientWidth*, *ClientHeight*).

Elementlari polotno nuqtalarining rangi haqida ma'lumot saqlaydigan massivni tashkil etuvchi *Pixels* xossalari orqali alohida piksellarga yo'l ochish mumkin.



Rasm 1 –shakl (polotno) yuzasidagi nuqtalar koordinatasi

Qalam va mo'yqalam

Grafik primitivlarni chizish metodlari faqat chizishnigina amalga oshirib beradi. Grafik element ko'rinishini, metod chizilayotgan (*Canvas*) yuzasining ***Pen***(qalam) va ***Brush*** (mo'yqalam) xossalari, aniqlab beradi.

Pen ob'ektining xossalari (3 jadval) geometrik shaklning chegaralarini yoki chiziq kengligi va tipini, rangini belgilab beradi. ***Brush*** ob'ekti xossalari (jadval 4) to'g'rito'rtburchak, sektor, aylana yoki yopiq konturlar ichlarini bo'yash usuli va rangini belgilab beradi.

Jadval 3 –*Pen* ob'ektining xossalari

Xossasi	Aniqlaydi
Color	Chiziq rangini
Width	Chiziq kengligini (pikselda beriladi)
Style	Chiziq ko'rinishini (*psSolid* – to'liq; *psDash* – punktirli, uzun shtrixlar; *psDot* – punktirli, qisqa shtrixlar; *psDashDot* – punktirli, uzun va qisqa shtrixlar ketmaketligi; *psDashDotDot* – punktirli, bitta uzun va ikkita qisqa shtrixlar ketmaketligi; *psClear* – chiziq ko'rinmaydi (maydon chegaralarini ko'rsatish lozim bo'lmagan xolatlarda ishlatiladi –masalan, to'g'rito'rtburchakda)

Jadval 4 –*Brush* ob'ektining xossalari

Xossasi	Aniqlaydi
Color	Yopiq maydonning bo'yalish rangi
Style	Maydoni to'ldirish uslubi (*bsSolid* – to'liq bo'yash. Shtrixlash: *bsHorizontal* – gorizontal; *bsVertical* – vertikal; *bsFDiagonal* – oldinga egik chiziqlar bilan diagonal; *bsBDiagonal* – orqaga egik chiziqlar bilan diagonal; bsCross – katak; *bsDiagCross* – diagonal katak

Quyidagi buyruqlar olimpiada xalqalari ifoda etuvchi bayroq chizadi – (rasm. 9):

```
bayroq polotnosi
Image1->Canvas->Pen->Width = 1;
Image1->Canvas->Pen->Color q = clBlack;
Image1->Canvas->Brush->Color = clCream;
Image1->Canvas->Rectangle(30,30,150,150);
Image1->Canvas->Pen->Width =2; // xalqalar kengligi
```

```
// Ellipse metodi bilan chizilgan aylana bo'yalmagan bo'lishi uchun
Image1->Canvas->Brush->Style = bsClear;
// xalqalar chizamiz
Image1->Canvas->Pen->Color = clBlue;
Image1->Canvas->Ellipse(40,40,80,80);
Image1->Canvas->Pen->Color = clBlack;
Image1->Canvas->Ellipse(70,40,110, 80);
Image1->Canvas->Pen->Color = clRed;
Image1->Canvas->Ellipse(100,40,140, 80);
Image1->Canvas->Pen->Color = clYellow;
Image1->Canvas->Ellipse(55,65,95,105);
Image1->Canvas->Pen->Color = clGreen;
Image1->Canvas->Ellipse(85,65,125,105);
}
```

Grafik primitivlar

Istalgan tasvir, chizma yoki sxemaga grafik primitivlar (nuqtalar, chiziqlar, aylanalar, yoylar va boshqalar) yig'indisi sifatida qarash mumkin. Demak ekranda kerakli tasvir paydo bo'lishi uchun, dastur, grafik elementlarni (shu tasvirni tashkil etuvchi primitivlarni) chizishni (chiqarishni) ta'minlashi lozim.

TImage sinfi va tasvirlar bilan ishlash

TImage ko'rgazmali komponenti grafik tasvirning (bitli obrazning, piktogrammaning, yoki metafaylning) konteyner formasi ustida yaratadi.

TImage komponentining aksariyat xossalari, ko'pkina boshqa komponentlarnikidek (shaklda tasvirning joylashishi, uning o'lchamlari, yuzasi va h.k.). "Nostandart" xossalar deb, quyidagilarni belgilashimiz mumkin:

Picture. Grafika konteyneri. Tasvirlar fayllarining yuklanish oynasi, *TImage* komponentining Picture xossalari ustunida tugma yordamida ochiladi. Undan tashqari bu xossa, LoadFromFile(), SaveToFile() metodlarga ega.

AutoSize. Tasvirni to'liq holda sig'dirish yo'lida, konteyner o'z o'lchamlarini o'zgartirishi uchun AutoSize xossasining true belgisi o'rnatiladi.

Stretch. Dastlabki tasvir butun konteynerga cho'zilishi uchun Stretch xossasining true belgisi o'rnatiladi.

Center − agar AutoSize false bo'lsa, u holda tasvir konteyner o'rtasiga sig'adi.

IncrementalDisplay − agar true bo’lsa, u holda katta fayllarni yuklashda, yuklanishiga qarab ular qismlarga bo’linib namoyon bo’ladi.

Transparent – pikselning bir burchak rangini ko’rinmas rang deb hisoblaydi.(chapdan pastki).

Image komponenti yuzasida grafik primitivlarni chizish, shu yuzaning *Canvas* xossasiga kerakli metodlarni ishlatish yo’li bilan amalga oshiriladi.

Chiziq

To’g’ri chiziq chizishni **LineTo** metodi amalga oshiradi. Bu metod, ayni damda qalam joylashgan nuqtadan boshlab (bu nuqta qalamning dastlabki xolati deyiladi), koordinatalari, metodni chaqirish ko’rsatmasida berilgan nuqtagacha chiziq chizadi.

Masalan,

```
Image1->Canvas->LineTo(100,200);
```

(100, 200) koordinatali nuqtaga chiziq chizadi, shundan so’ng (100, 200) koordinatali nuqta dastlabki bo’ladi.

Qalamni grafik yuzaning istalgan nuqtasiga keltirib, boshlang’ich (dastlabki) nuqtani belgilash mumkin.Buni **MoveTo** metodi yordamida, parametrlar sifatida, chiziq boshlanishi nuqtasi koordinatalarini qo’rsatib, bajarish mumkin.

Masalan, operatorlar:

```
Image1->Canvas->MoveTo(10,10); // nuqtagaqalamnio’rnatish (10,10)
Image1->Canvas->LineTo(50,10); // (10,10) nuqtadan (50,10)nuqtagachiziqlar
```

(10, 10) nuqtadan (50, 10) nuqtagacha gorizontal chiziq chizadi.

Dastlabki nuqtaning xossalaridan foydalaniladi, siniq chiziq chizish mumkin. Masalan, operatorlar:

```
//Z xarfiga o’xshash chiziq chizishadi.

Image1->Canvas->MoveTo(10,10);
Image1->Canvas->LineTo(50,10);
Image1->Canvas->LineTo(10,20);
Image1->Canvas->LineTo(50,20);
```

Siniq chiziq

Siniq chiziqni **Polyline** metodi chizadi. Chiziqning bog'lamali nuqtalar koordinatalarini o'z ichiga olgan *Tpoint* tipidagi massiv metodiga sozlama sifatida uzatiladi. Koordinatalari massivda joylashgan nuqtalarni ketm aketlikda bog'lab, ***Polyline*** metodi siniq chiziq chizadi. Masalan, pastda keltirilgan kod qismi, uchta uchdan iborat siniq chiziqni chizadi.

```
TPoint p[4]; // boshi, oxiri va buklanish nuqtalar koordinatalari
// siniq chiziq koordinatalarini belgilab berish
r[0].x =100; r[0].y = 100; // boshi
r[1].x = 100; r[1].y = 150; // buklanish nuqtasi
r[2].x = 150; r[2].y = 150; // buklanish nuqtasi
r[3].x = 150; r[3].y = 100; // oxiri
Image1->Canvas->Polyline(r,3); //  uchta uchdan iborat siniq chiziqni chiqaradi
```

Polyline metodini yopiq konturlarni chizish uchun ishlatish mumkin. Buning uchun massivning birinchi va oxirgi elementlari, bitta nuqta koordinatalarini o'z ichiga olgan bo'lishi kerak.

To'g'rito'rtburchak

Rectangle metodi to'g'rito'rtburchak chizadi. Metod chaqiruvi ko'rsatmasida to'g'rito'rtburchak burchaklarining ikkita nuqta koordinatalarini belgilash lozim.

Masalan, buyruq

```
Image1->Canvas->Rectangle(10,10, 50, 50);
```

Chapdan teppa burchagi (10, 10) nuqtada, o'ng tomondan pastdagi burchagi esa (50, 50) nuqtada joylashgan kvadratni chizadi.

To'g'rito'rtburchak konturining rangi, ko'rinishi va kengligini **Pen** xossasi, to'g'rito'rtburchak ichini bo'yash uslubi va rangini esa, metod to'g'rito'rtburchakni chizayotgan yuzaning **Brush** xossasi aniqlaydi. Masalan, quyidagi bayro chizadi

```
Image1->Canvas->Brush->Color = clWhite; // mo'yqalam rangi − oq
Image1->Canvas->Rectangle(10,10,90,30);
Image1->Canvas->Brush->Color = clBlue; // mo'yqalam rangi – ko'k
Image1->Canvas->Rectangle(10,30,90,50);
Image1->Canvas->Brush->Color = clRed; //mo'yqalam rangi – qizil
 Image1->Canvas->Rectangle(10,50,90,70);
```

Kodning quyidagi qismi *TRect* strukturastdan *Rectangle* metodi o'lchamlari sifatida foydalanish keltirilgan.

```
Rect rct; // to'g'rito'rtburchak qism
rct.Top = 10;
rct.Left =10;
rct.Bottom=50;
rct.Right = 50;
Image1->Canvas->Rectangle(rct); //to'g'rito'rtburchakni ko'rsatadi
```

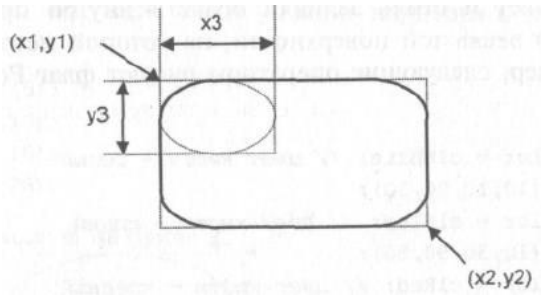
To'g'rito'rtburchak chizishni yana ikki metodi bor. ***FillRect*** metodi, asbob sifatida (*Brush*) mo'yqalamni ishlatib, ichi bo'yalgan to'g'rito'rtburchak, ***FrameRect*** metodi esa, qalam yordamida faqat kontur chizadi. Bu metodlarda faqat bitta, *Trect* tipidagi struktura, o'lchami bor. *Trect* strukturasi chegaralari o'lchamlarini *Rect* funktsiyasi yordamida buyurish mumkin. Masalan:

```
TRect rct; // bo'yash kerak bo'lgan maydon
rct =Rect(10,10,30,50); // maydon koordinatalari
Image1->Canvas->Brush->Color = clRed; // bo'yash rangi
Image1->Canvas->FillRect(rct); // bo'yalgan to'g'rito'rtburchakni yoritib berish
```

RoundRect metodi dumaloqlashkan burchakli to'g'rito'rtburchak chizadi. ***RoundRect*** metodini chaqirish ko'rsatmasi umumiy holda quyidagicha ko'rinishda bo'ladi:

```
Image1->Canvas->RoundRect(xl, yl, x2, y2, x3, y3);
```

*x*l, *y*1, *x*2, *y*2 o'lchamlari o'rnini aniqlab beradi, x3 va y3 –esa ,to'rtdan bir qismi aylanalashgan burchak chizishda ishtirok etayotgan, ellips kattaligini. (rasm. 2).



Rasm 2 − ***RoundRect*** metodi aylanalashgan uchli to'g'rito'rtburchakni chizadi.

Ko’pburchak

Polygon metodi ko’pburchak chizadi. Ushbu metodni chaqirish ko’rsatmasi:

Image1->Canvas->Polygon(p,n); , r – ko’pburchak uchlari koordinatalarini o’z ichiga olgan, *TPoint* tipidagi ro’yxat massivi; n – uchlar soni.

Polygon metodi, koordinatalari massivda joylashgan nuqtalarni tekis chiziqlar yordamida chizish orqali, ko’pburchak chizadi: birinchi nuqtani ikkinchisi bilan, ikkinchisini uchinchisi bilan va h.k.

Polygon metodidan foydalangan xolda romb chizish kodining bir qismi:

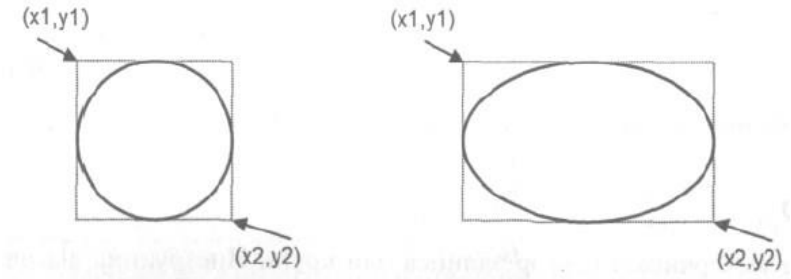
```
TPoint p[4]; // to’rtta uch uchlar koordinatalari
r[0].x = 50;  r[0].y = 100;
 r[1].x = 150; r[1].y = 75;
 r[2].x = 250; r[2].y = 100;
 r[3].x = 150;  r[3].y = 125;
Image1->Canvas->Brush->Color = clRed;
Image1->Canvas->Polygon(p,3); //romb chizildi
```

Ellips va aylana

Aylana yoki ellipsni (ellipsning xususiy holati) ***Ellipse*** metodi yordamida chizish mumkin. Metodni chaqirish ko’rsatmasi umumiy holda quyidagicha bo’ladi:

Image1->Canvas->Ellipse (*x*l,*y*l,*x*2,*y*2);

*x*l, *y*1, *x*2, *y*2 o’lchamlari, ichiga ellips chiziladigan to’g’rito’rtburchakni, yoki ichiga aylana chiziladiga kvadratni koordinatalarini belgilaydi. (rasm. 3).



Rasm 3 –***Ellipse*** metodi o'lchamlari geometrik shaklning ko'rinishini belgilaydi

To'g'rito'rtburchakning diagonal burchaklarining to'rtta o'lchami o'rniga, bitta *TRect* tipidagi ob'ektni ***Ellipse*** metodiga uzatish mumkin. Quyidagi kod qismi *TRect* ob'ektining ***Ellipse*** metodi o'lchamlari sifatida ishlatilishini ko'rsatadi:

```
TRect rec = Rect(10,10,50,50);
Image1->Canvas->Ellipse(rec); //aylanani chizadi
```

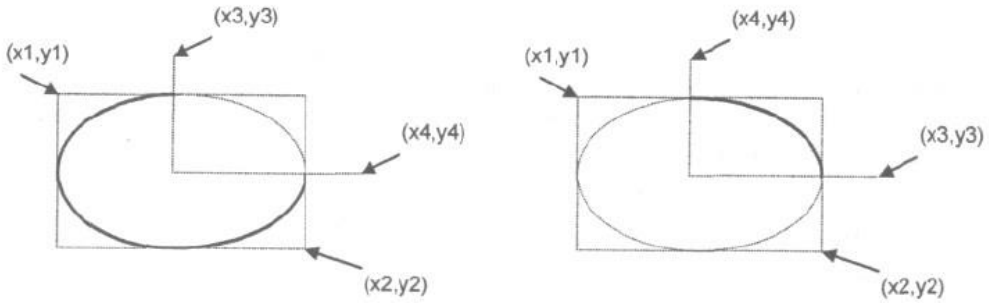
Yoy

Arc metodi yoyni chizadi – ellipsning (aylananing) qismi.

Metodni chaqirish:

Image1->Canvas->Arc(*x*l, *y*l, x2, y2, x3, y3, x4, y4);

*x*1, *y*l, x2, *y*2 o'lchamlari, bir qismi yoy bo'lgan ellipsni (aylanani) belgilaydi.*x3* va *y3* o'lchamlari, yoyning boshlang'ich nuqtalarini, *x4* va *y4* esa yakuniy nuqtalarini belgilaydi.Yoyning boshlang'ich (yakuniy) nuqtasi- bu ellips chegaralari bilan, ellips o'rtasidan, *x3* va *y*3 (*x4*, *y4*) o'lchamli nuqtagacha, to'g'richiziqning kesishgan nuqtasi. ***Arc*** metodi, boshlang'ich nuqtadan yakuniysigacha, soat ko'rsatkichlariga qarshi tomon aylangan holda yoy chizadi (rasm4). Chizildigan yoyning rangi, chiziqlar usuli va qalinligi, amal bajarilayotgan (*Canvas*) maydonining *pen* xossasi yordamida belgilanadi.



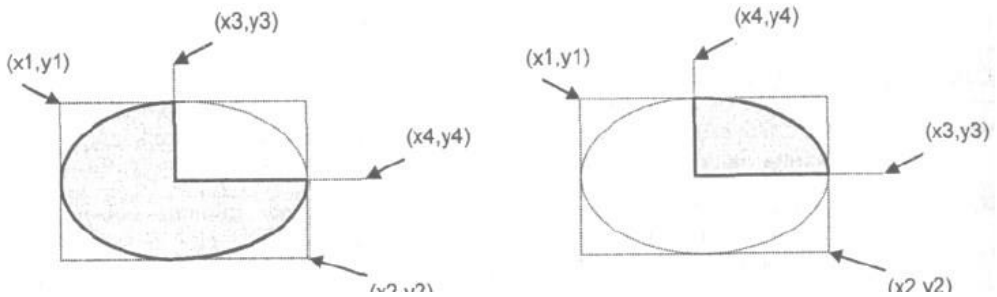
Rasm 4 – ***Arc*** metodining o'lchamlari**,** yoyni , ellipsning (aylananing) bir qismi sifatida aniqlab beradi.

Sektor

Pie metodi ellips yoki aylananing sektorini chizadi. Metodni chaqirish ko'rsatmasi:

Image1->Canvas->Pie(*x*l,*y*l,x2,y2,x3,y3,x4, y4);

*x*l, *y*l, *x*2, y2 bir qismi sektor bo'lgan,ellipsning (aylananing) o'lchamlari. x3, y3, x4 va y4 to'g'richiziqlar – sektorning chegaralari. Chegaralarning boshlang'ich nuqtasi ellipsning o'rtasiga to'g'ri keladi. Sektor, nuqtasi (x3, y3) o'lchamda berilgan to'g'richiziqdan, nuqtasi (x4, y4) o'lchamda berilgan to'g'richiziqgacha, soat aylanishiga qarshi yo'nalishida kesilib boriladi. (rasm. 5).



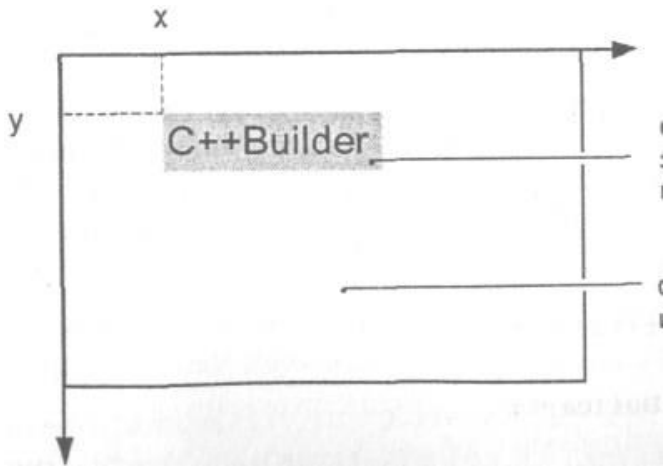
Risunok 5 − ***Pie*** metodi o'lchamlari, sektorni, ellipsning (aylananing) bir qismi sifatida belgilaydi.

Tekst

Polotnoga tekstni chiqarish juda oson. Faqat shriftning xususiyatlarini (polotnoning *Font* xossasi) va AnsiString formatidagi tekstni kiritish kerak xolos. ***TextOut()*** metodi, tekstni grafik ob'ektning yuzasiga chiqarishni amalga oshirib beradi. ***TextOut*** metodini chaqirish, umumiy holda, quyidagi ko'rinishda bo'ladi:

Image1->Canvas->TextOut(x, y, Tekst)

Tekst o'lchami, chaqiriladigan tekstni buyuradi. x va y o'lchamlari, grafik yuzada tekstning chiqish nuqtasining o'lchamlarini belgilaydi. (rasm. 6).



Rasm 6 −tekst chiqish yuzasining o’lchamlari

Matnni chiqarish uchun ishlatiladigan shrift, Canvas ob’ektining *Font* xossasi bilan belgilanadi. *Font* xossasi, *TFont* tipidagi ob’ektni tashkil etadi. 4-jadvalda, *TextOut* metodi orqali matinni chiqarish uchun ishlatiladigan shriftning xususiyatlarini belgilab beradigan *Tfont* ob’ektining xossalari sanab o’tilgan.

4 –jadval TFont ob’ektining xossalari

Xossalar	Aniqlaydi
Name	Foydalanilayotgan shrift. Ma’no sifatida shriftning nomini ishlatish lozim. (masalan, Arial)
Size	(points) punktlarida shriftning kattaligi. Punkt – bu poligrafiyada ishlatiladigan shriftning o’lchov birligi.Bir punkt 1/72 dyuymga teng.
Style	Belgilarning chizilish uslubi.Bo’lishi mumkin: normal, yarimqalin, kursiv, tagi chizilgan, usti chizilgan. Uslub, quyidagi konstantalar yordamida buyuriladi: *fsBold* (yarimqlin), *fsItalic* (kursiv), *fsUnderline* (tagi chizilgan), *fsStrikeOut* (usti chizilgan)
Color	Belgilar rangi. Ma’nosi sifatida *TColor* tipidagi konstantani ishlatish mumkin.

Style xossasi kerakli uslublarni qo’shib foydalanishga yordam beradi. Masalan, "yarimqalin tagi chizilgan" uslubni o’rnatish ko’rsatmasi:

```
Image1->Canvas->Font->Style = TFontStyles () <<fsBold<<fsUnderline;
```

Matnni chiqarishda *TextWidth* va *TextHeight* metodlari yordam qiladi.

1)TextWidth (AnsiString S); − polotno shrifti tomonidan buyurilgan S qatorini ko’rsatish uchun kerak bo’ladigan matn kengligini piksellarda qaytaradi;

2) TextHeight(AnsiString S); − polotno shrifti tomonidan buyurilgan S qatorini ko’rsatish uchun kerak bo’ladigan matn kengligini piksellarda qaytaradi;

Ikkala metodga ham o’lchamlar sifatida, *TextOut* metodi orqali yuzaga chiqarilishi kerak bo’lgan qator yuboriladi.

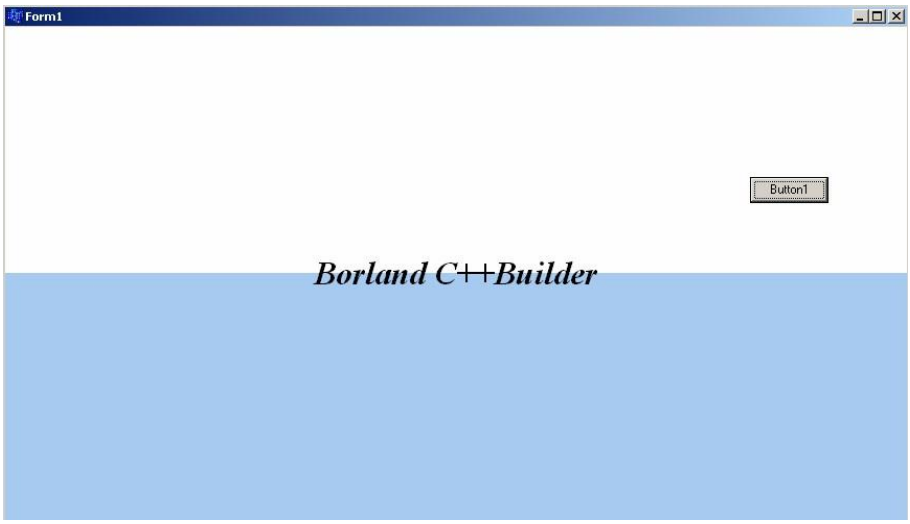
Shriftning kattaligini o’zgartirgan xolda, matn komponent chegaralaridan chiqmasligi uchun, bu amallarning foydasi tegadi.

Pastda keltirilgan dastur misolida ListBox komponentining kengligi o’zgarishi xolati ko’rsatilgan.

```
void __fastcall TForm1::Button1Click(TObject *Sender)
{
//Matnrangi
 ListBox1->Canvas->Font->Color=(TColor)0x00FF7D7D;
//Matn balandligi piksellarda berilgan
```

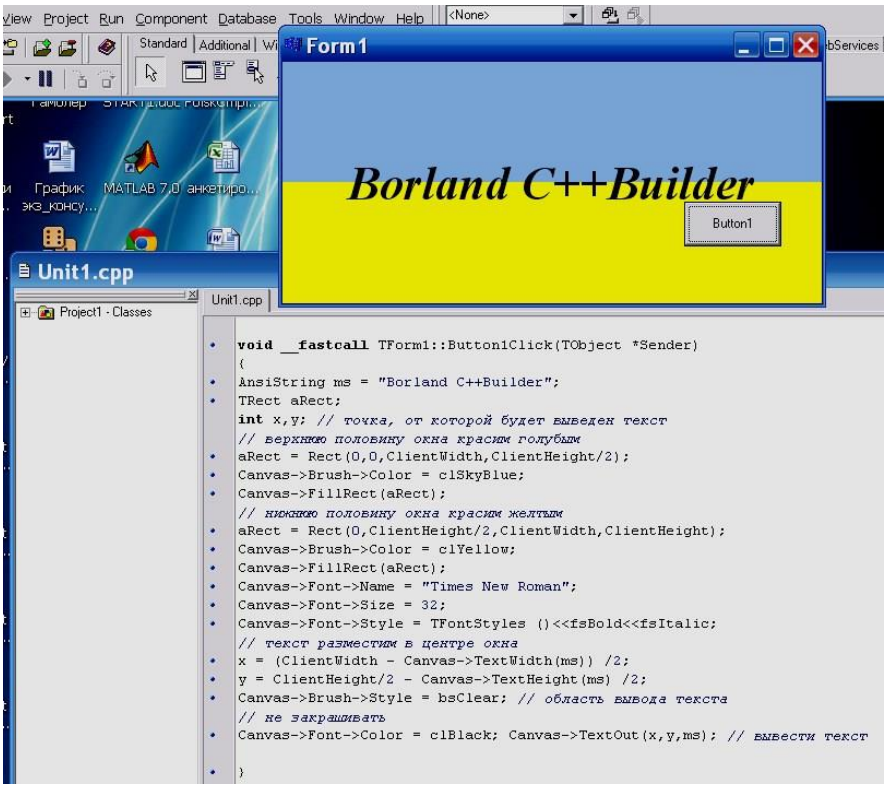
```
 ListBox1->Canvas->Font->Height=25;
 AnsiString S="Matn misoli";
 ListBox1->Width=ListBox1->Canvas->TextWidth(S)=20;
}

void __fastcall TForm1::Button2Click(TObject *Sender) {
 AnsiString S="Matn misoli";
 Canvas->Font->Color=clBlue;
 Canvas->TextOutA(50,50,S);
}

void __fastcall  TForm1::Button3Click(TObject *Sender)
{
 AnsiString S="Matn misoli";
 ListBox1->Canvas->TextOutA(2,10,S); }


void __fastcall TForm1::Button1Click(TObject *Sender)
{
 AnsiString ms = "Borland C++Builder";
 TRect aRect;  int x,y; // Matnning chiqish nuqtasi
 // oynaning tepa qismini  oq rangda bo'yaymiz
 Rect = Rect(0,0,ClientWidth,ClientHeight/2);
 Canvas->Brush->Color = clWhite;
 Canvas->FillRect(aRect);
 // oynaning past qismini ko'k rangda bo'yaymiz
 aRect = Rect(0,ClientHeight/2,ClientWidth,ClientHeight);
 Canvas->Brush->Color = clSkyBlue;
 Canvas->FillRect(aRect);
 Canvas->Font->Name = "Times New Roman";
 Canvas->Font->Size = 24;
 Canvas->Font->Style = TFontStyles ()<<fsBold<<fsltalic;
 // matnnioynaningo'rtasigajoylaymiz
```

```
x = (ClientWidth - Canvas->TextWidth(ms))/2;
y = ClientHeight/2 - Canvas->TextHeight(ms) /2;
Canvas->Brush->Style = bsClear; // matnni chiqarish yuzasi
// bo'yamaslik
Canvas->Font->Color = clBlack;
Canvas->TextOut(x,y,ms); // matnni chiqarish
}
```



Rasm 7 –matnni oynani o'rtasida chiqarish.

Keyigi rasm formani ranga bo'yab chizib, matn chiqarish dasturidan parcha.(rasm.8).



Rasm 8–Dastur proektini bajarish natijasi

Ba’zida xabardan so’ng, dasturni yaratish davomida uzunligi ma’lum bo’lmagan, biror bir matnni chiqarish kerak bo’ladi.Bu xolatda, matn chiqariladigan yuzaning o’ng chegaralarining o’lchamlarini bilish lozim. *TextOut* metodi orqali chiqarilgan matnning o’ng chegaralarining o’lchamlarini, *PenPos*xossasiga murojat qilgan holda aniqlash mumkin.

Kodning quyidagi parchasi, *TextOut* ning ikkita ko’rsatmasi yordamida matn qatorini chiqarish imkonini yoritib bergan:

```
Canvas->TextOut(10,10,"Borland ");
Canvas->TextOut(Canvas->PenPos.x, Canvas->PenPos.y, "C++Builder");
```

Nuqta

Dastur, grafika chiqishini amalga oshirishi mumkin bo’lgan yuzalarga, *Canvas* ob’ekti mos keladi. *Tcolor* turdagi ikki o’lchovli massivni tashkil etuvchi *Pixels* xossasi, grafik yuzaning xar bir nuqtaning rangi haqida ma’lumotlarni o’z ichiga oladi. *Pixels* xossasidan foydalangan xolda, grafik yuzaning istalgan nuqtasiga rang buyursa bo’ladi,ya’ni nuqtani “chizish”.Masalan,

```
Canvas->Pixels[10][10] = clRed;
```

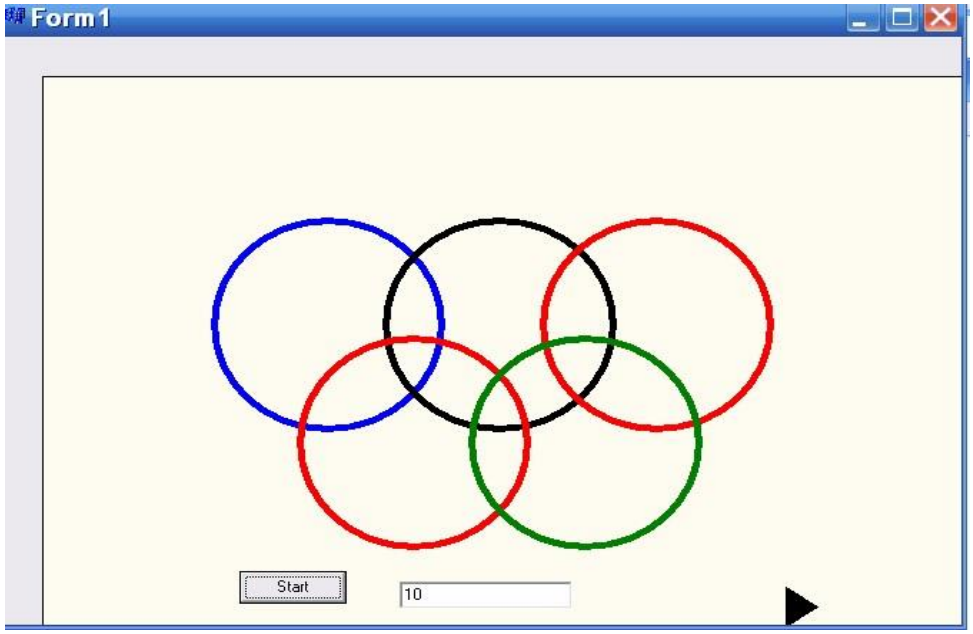
Ko’rastma, shakl yuzasining nuqtasini qizil rangga bo’yaydi.

Pixels massivning o’lchamlari grafik yuzaning o’lchamlari orqali belgilanadi. Shaklning grafik yuzasining (ishchi yuza) o’lchamlarini, *ClientWidth* va *ClientHeight* xossalari belgilaydi, *image* komponentining o’lchamini esa– *Width* va *Height* xossalari.Ishchi yuzaning chapdan teppadagi nuqtasiga *Pixels[0][0]*elementi mos keladi, pastdang o’ng tomondagisiga esa – *Pixels[ClientWidth - 1][ClientHeight-1]*.

«Olimpiya uzuklari»dasturi misoli

Dastur matni:

```
void __fastcall TForm1::Button1Click(TObject *Sender)
{
  Canvas->Pen->Width=1;
  Canvas->Pen->Color=clBlack;
  Canvas->Brush->Color =clCream;
  Canvas->Rectangle(30,30,700,500);
  Canvas->Pen->Width=5;
   Canvas->Brush-> Style=bsClear;
   Canvas->Pen->Color=clBlue;
   Canvas->Ellipse(150,140,310,300);
   Canvas->Pen->Color=clBlack;
   Canvas->Ellipse(270,140,430,300);
   Canvas->Pen->Color=clRed;
   Canvas->Ellipse(380,140,540,300);
   Canvas->Pen->Color=clYellow;
   Canvas->Ellipse(210,230,370,390);
   Canvas->Pen->Color=clGreen;
   Canvas->Ellipse(330,230,490,390);

}
```



Rasm 9 − «Olimpiya uzuklari» dasturini bajarish natijasi

Nazorat savollar

1. Grafik funktsiyalar va ularning prototiplari (xabarlari) kutubxonaning qaysi faylida saqlanadi?
2. Qaysi metod yordamida burchaklari aylanalashgan to’g’rito’rtburchak chiziladi?
3. ***Polyline*** funktsiyasi nima uchun ishlatiladi va qanaqa o’lchamlarga ega?
4. ***Canvas***funktsiyasi nima uchun ishlatiladi va qanaqa grafik funktsiyalarga ega?
5. ***FloodFill***funktsiyasi nima uchun ishlatiladi va qanaqa o’lchamlarga ega?
6. ***Polygon***metodi qaysi shaklni chizadi**?** Bu metodnichaqirish ko’rsatmasini yozing.
7. ***TextWidth***va***TextHeight***metodlari qaysi funktsiyani bajaradi?
8. ***FillRect***va***FrameRect***metodlari qaysi funktsiyani bajaradi?

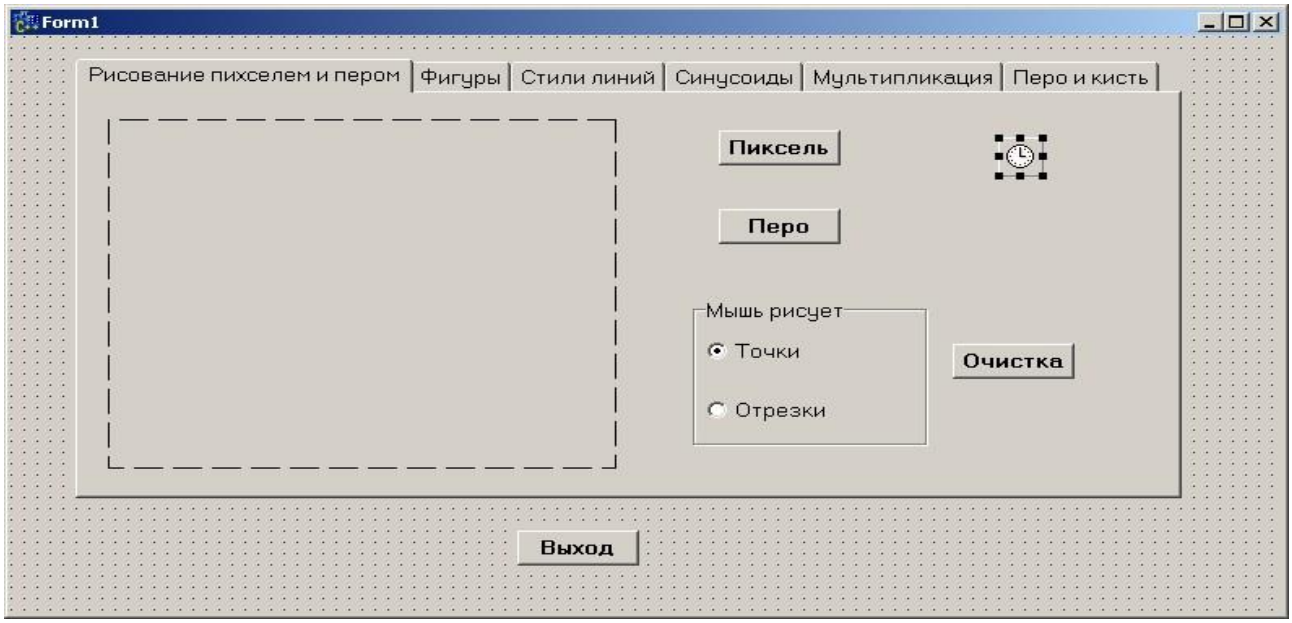
Laboratoriya ishi № 1

C++ Builder da bazali grafikaning asosiy funktsiyalari

Ishdan maqsad**:** C++ da, bo'yoqning rangi va namunalarini boshqaradigan, matnli ma'lumotni ko'rsatadigan, sxemalar elementlarini grafik primitivlar yordamida chizadigan, berilgan dastur misolida grafik funksiyalarning imkoniyatlari bilan aniq tanishuv: ***C*** kandensator, ***R*** rezistor va C++ Builder sohasida induktivlik.

Laboratoriya topshirig'i

1.1. rasmda ko'rsatilgan proekt shaklini yarating. Shaklda PageControl komponentini, uning ***allClient*** ga teng, ***Align*** xossasini o'rnating va oltita bet qo'shing. Bu betlarning xar biri, proektning alohida pog'onasiga mos keladi. Har bir betda Image (punktirlangan to'g'ri to'rtburchak 1.1 rams.)komponentini va ko'rinmaydigan Timer komponentini o'rnating.



Rasm 1.1− Proyektning shakli. Bet «piksel va qalam bilan»

1 «Piksel va qalam bilan chizich»

1.1 Topshiriqni bajarish

“piksel” tugmasi bosilganda grafikaning tashqi qismi “pero” tugmasi bosilganda esa ichki qismi bo’yaladi. Bunda bazaviy nuqtalar metodidan foydalanib Pixels(piksellar massivi) ***LineTo metodi qo’llaniladi.*** RadioGroup1 da ko`rsatilgan punktlardan biri tanlanganda quyida berilgan tenglama asosida figura yoki nuqtalar chiziladi:

$$x(t) = a\cos^3(t)$$
$$y(t) = a\sin^3(t)$$

t [0,2π] oraliqa tegishli , a – musbat son.

*1.2 «*Piksel*» tugmasi*

cx va *cy* – rasim markazi koordinatalari ; *px*, *py* – joriy nuqta koordinatalari .

Dastur matni*:*

```
void __fastcall TForm1::Button1Click(TObject *Sender)
{ int a,i,px,py,cx,cy;
const float pi=3.141563;
a=100;
cx=Image1>Width/2;
cy=Image1->Height/2;
for (int i=0;i<=180;i++)
{px=(int)(a*pow(cos(2*pi*i/180),3)); py=(int)(a*pow(sin(2*pi*i/180),3));
Image1->Canvas->Pixels[cx+px][cy+py]=clBlue; }
}
```

*1.3 «*Pero*» tugmasi*

cx va *cy* – rasim markazi koordinatasi; *px*, *py* – joriy nuqta koordinatasi.

Programma matni*:*

```
void __fastcall TForm1::Button2Click(TObject *Sender)
{ int a,i,px,py,cx,cy;
const float pi=3.141563;
a=100;
cx=Image1->Width/2;
```

```
cy=Image1->Height/2;
Image1->Canvas->MoveTo(cx,cy);
Image1->Canvas->Pen->Color=clRed;
for (int i=0;i<=180;i++)
{px=(int)(a*pow(cos(2*pi*i/180),3));
py=(int)(a*pow(sin(2*pi*i/180),3));
Image1->Canvas->LineTo(cx=px,cy=py); }
}
```

1.4 sichqoncha tugmasini bosilganda bajariladigan programma

```
void __fastcall TForm1::Image1MouseDown(TObject *Sender, TMouseButton
Button, TShiftState Shift, int X, int Y)
{
if (RadioGroup1->ItemIndex==0)
for     (X=0,Y=0;X+Y<300;X+=3,Y+=3)
Image1->Canvas->Pixels[X][Y]=clBlue;
else
{Image1->Canvas->Pen->Color=clRed;
Image1->Canvas->LineTo(X,Y);}
}
```

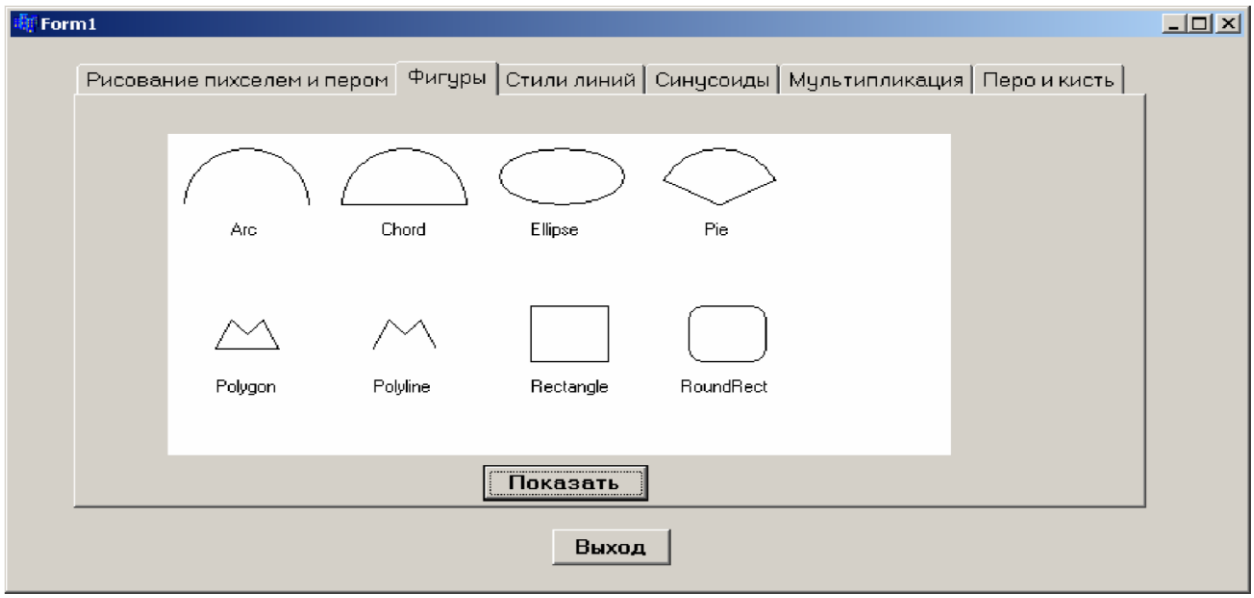
1.5 «Tozalash***» tugmasi***

Programma matni*:*

```
void __fastcall TForm1::Button3Click(TObject *Sender)
{
Image1->Canvas->FillRect(Rect(0,0,Image1->Width,Image1->Height)); }
```

2 «Figura **» bolimi**

2.1 Bo`limning ishlash tartibi



1.2 rasm – «Figuralar» bo'limi

*2.2 «*Pokazat*» tugmasi*

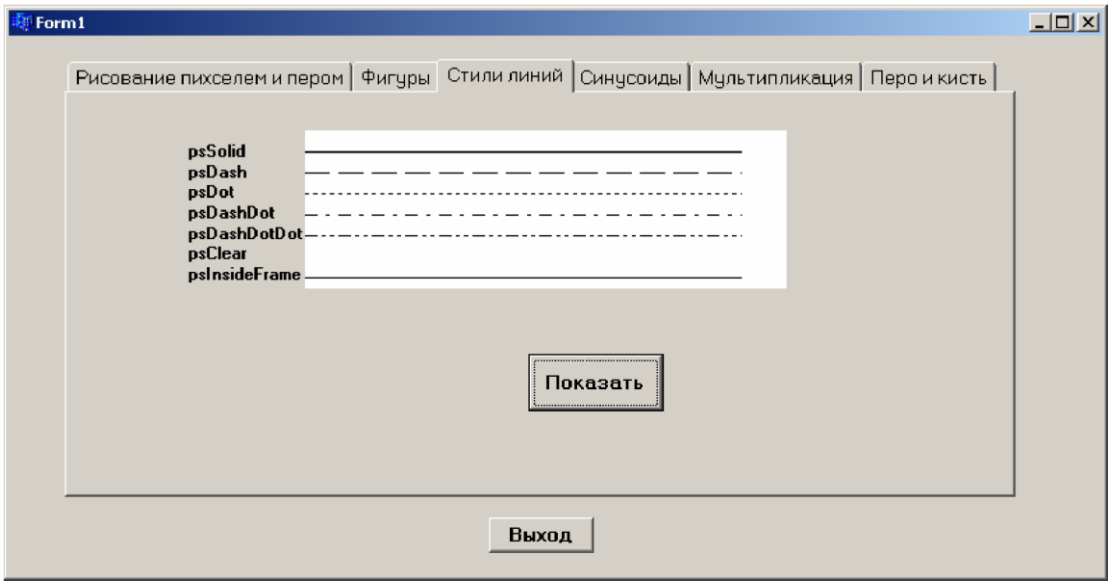
Programma matni*:*

```
void __fastcall TForm1::Button4Click(TObject *Sender)
{ Image2->Canvas->Font->Style << fsBold;
  Image2->Canvas->Arc(10,10,90,90,90,50,10,50);
  Image2->Canvas->TextOut(40,60,"Arc");
  Image2->Canvas->Chord(110,10,190,90,190,50,110,50);
  Image2->Canvas->TextOut(135,60,"Chord");
  Image2->Canvas->Ellipse(210,10,290,50);
  Image2->Canvas->TextOut(230,60,"Ellipse");
  Image2->Canvas->Pie(310,10,390,90,390,30,310,30);
  Image2->Canvas->TextOut(340,60,"Pie");
TPoint points[5];
points[0] = Point(30,150);
points[1] = Point(40,130);
points[2] = Point(50,140);
points[3] = Point(60,130);
```

```
points[4] = Point(70,150);
  Image2->Canvas->Polygon(points,4);
  Image2->Canvas->TextOut(30,170,"Polygon");
points[0].x += 100;
points[1].x += 100;
points[2].x += 100;
points[3].x += 100;
points[4].x += 100;
  Image2->Canvas->Polyline(points,4);
  Image2->Canvas->TextOut(130,170,"Polyline");
  Image2->Canvas->Rectangle(230,120,280,160);
  Image2->Canvas->TextOut(230,170,"Rectangle");
  Image2->Canvas->RoundRect(330,120,380,160,20,20);
  Image2->Canvas->TextOut(325,170,"RoundRect"); }
```

3 «chiziq stili» **bo'limi**

3.1 Programma natni



Risunok 1.3 − Stranitsa «Stililiniy»

3.2 «Pokazat» ***tugmasi***

Programma matni*:*

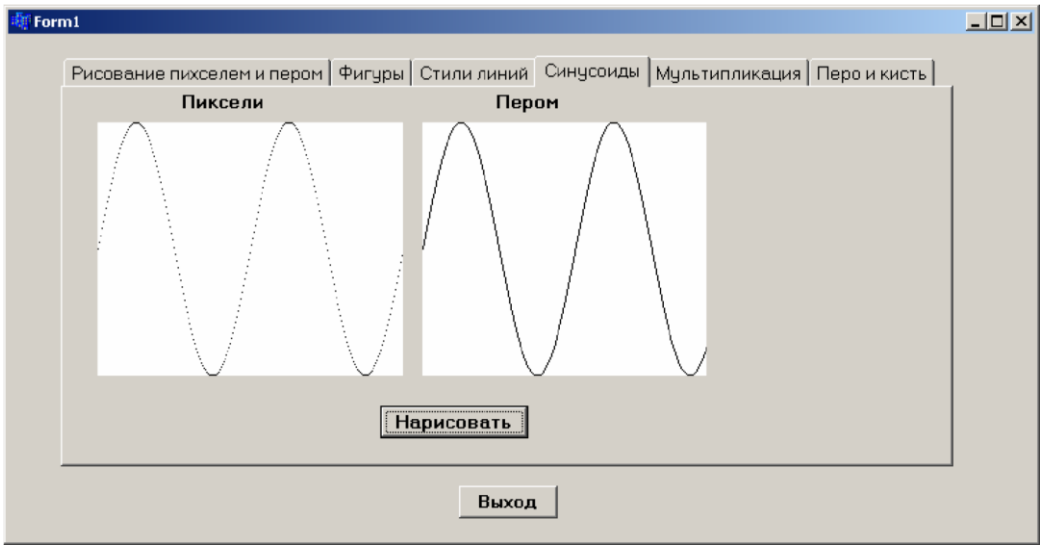
```
void __fastcall TForm1::Button5Click(TObject *Sender)
{
  for (int i = 1; i < 8; i++)
 {  Image3->Canvas->Pen->Style = TPenStyle(i-1);
    Image3->Canvas->MoveTo(0,i*15);
    Image3->Canvas->LineTo(Image1->Width,i*15); }
}
```

4 «Sinusqoidalar» **bo'limi**

*4.1 «*Chizish*» tugmasi*

Pragramma matni:

```
 void __fastcall TForm1::Button6Click(TObject *Sender)
 {
 #define Pi 3.14159 ;
 float X,Y;         // funksiya koordinatalari
 int PX,PY;          // piksel koorsinatalari
Image5->Canvas->MoveTo(0,Image5->Height  /  2);
for (PX = 0; PX <= Image4->Width; PX++)
{
//X – koordinata, PX mos qo'yilgan
X = PX * 4 * Pi / Image4->Width;
Y = /*2 **/ sin(X);
//PY −  Y mos qo'yilgan koordinata
PY = Image4->Height - (Y+1) * Image4->Height/2;

   Image4->Canvas->Pixels[PX][PY] = clBlack;

   Image5->Canvas->LineTo(PX,PY);
  }
}
```



1.4 rasim – “ sinusoida ” saxifasi

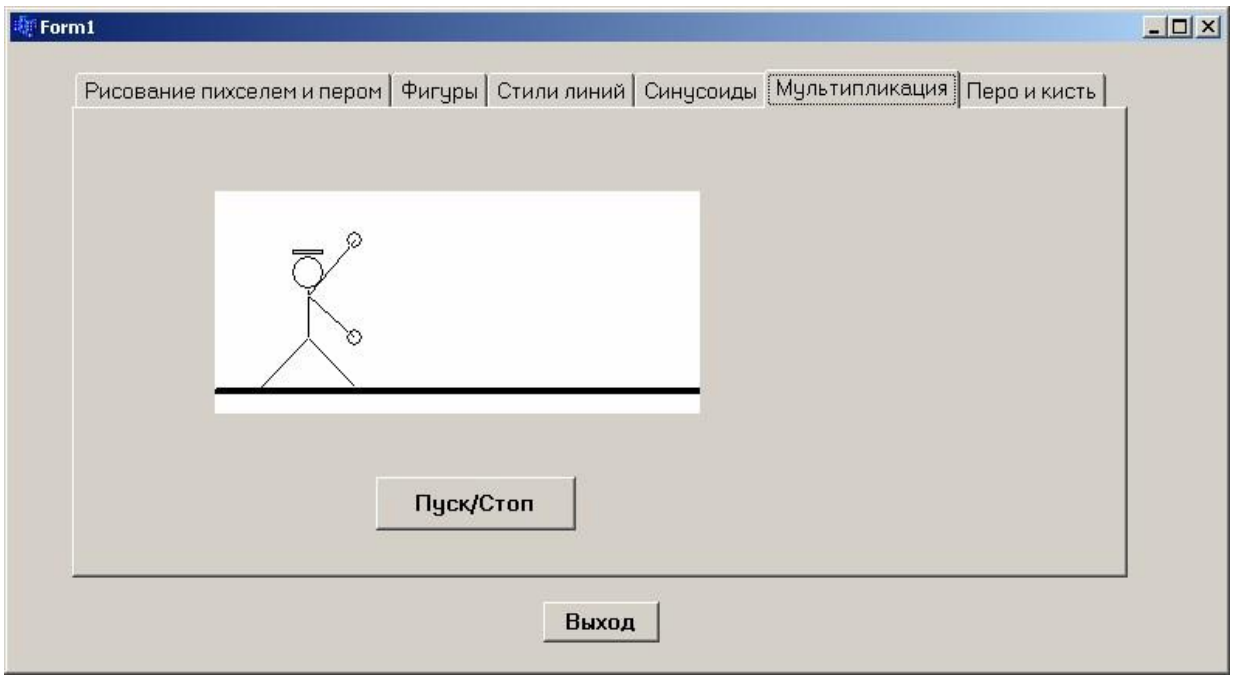
5 «Multiplikatsiya» **bo’limi**

5.1 Bo’limning umumiy tavsifi

Bolimda harakatlanuvchi odamcha tasvirlanadi

Unit.h faylida **Draw() metodinn public** bolimiga qo’shib qo’yish kerak

```
public:    // User declarations
  void __fastcall Draw(); //qo’shimcha metod
```



1.5rasim − «Multiplikatsiya» saxifasi

5.2 Draw() metodi

«Pusk/Stop» tugmasidan foydalanishdan oldin **Draw()** yozib olnishi shart**.**

```
// o'zgaruvchilar:
short int num = 0;
short int H=30;                 // qadam
short int Xpos = 2 * H;       //rasimdagi odamcha qadami
short int Ypos = 120;          // "yer"
short int Hmen = 30;           // odamchaning balandligi
short int Rhead = 10;          // odamcha boshi radusi
short int Rhead2 = Rhead / 2; // radius
 short int revers = 1;              // harakat yo’nalishi
short int L = H * 1.41;          // oyoq uzunligi
```

Pragramma matni*:*

```
void __fastcall TForm1::Draw()
{ short int Yhead;                // koordinata
```

```
 switch (num)
{   case
0:
Yhead = Ypos-H-Hmen;
Image6->Canvas->MoveTo(Xpos-H,Ypos);
Image6->Canvas->LineTo(Xpos,Ypos-H); // oyoq
Image6->Canvas->LineTo(Xpos+H,Ypos);// boshqa oyoq
Image6->Canvas->MoveTo(Xpos,Ypos-H);
Image6->Canvas->LineTo(Xpos,Yhead); // asosiy yo'l
Image6->Canvas->MoveTo(Xpos+revers*H,Yhead-H);
Image6->Canvas->LineTo(Xpos,Yhead+4);  //qo'l
Image6->Canvas->Ellipse(Xpos+revers*H-Rhead2,Yhead-
HRhead2,Xpos+revers*H+Rhead2,Yhead-H+Rhead2);
// boshqa qo'l
Image6->Canvas->LineTo(Xpos+revers*H,Yhead+H);
Image6->Canvas->Ellipse(Xpos+revers*H-Rhead2,Yhead+H-Rhead2,
Xpos+revers*H+Rhead2, Yhead+H+Rhead2);
Image6->Canvas->Ellipse(Xpos-Rhead,Yhead,Xpos+Rhead, Yhead2*Rhead);
Image6->Canvas->Rectangle(Xpos-Rhead,Yhead-2*Rhead-1,
Xpos+Rhead,Yhead-2*Rhead-4); // shlyapa
break; case 1:
  Yhead = Ypos-L-Hmen;
  Image6->Canvas->MoveTo(Xpos,Ypos);
  Image6->Canvas->LineTo(Xpos,Yhead);
  Image6->Canvas->MoveTo(Xpos,Yhead=4);
  Image6->Canvas->LineTo(Xpos+revers*L,Yhead+4);
  Image6->Canvas->Ellipse(Xpos+revers*L-Rhead2,Yhead+4-
Rhead2, Xpos+revers*L+Rhead2,Yhead+4+Rhead2);
Image6->Canvas->Ellipse(Xpos-Rhead,Yhead,Xpos+Rhead,Yhead2*Rhead);
Image6->Canvas->Rectangle(Xpos−H / 2,Yhead−2*Rhead−1, Xpos+H / 2,
Yhead−2*Rhead−4);
    }
  }
```

5.3 Forma yaratish

```
void __fastcall TForm1::FormCreate(TObject *Sender)
{
Image6->Canvas->MoveTo(0,Ypos+3);
Image6->Canvas->Pen->Width = 4;
Image6->Canvas->LineTo(Image1->ClientWidth,Ypos+3); // "yer"
Image6->Canvas->Pen->Width = 1;
Image6->Canvas->Pen->Mode = pmNotXor;
Draw();
}
```

5.4 Timer1 kompanentasiga yoziladigan programma

Timer1 kompanentasi hossalari quydagicha o'rnatiladi:
Enabled=false, interval=500;

Programma matni*:*

```
void __fastcall TForm1::Timer1Timer(TObject *Sender)
{
Draw();
if ((Xpos >= Image6->Picture->Width-H)||(Xpos <= H))
revers = -revers;
 Xpos = Xpos + revers * H;
num = 1 − num;
Draw();
}
```

5.5 «Pusk/Stop***» tugmachasi ikki marta bosiladi va:***

```
void __fastcall TForm1::Button7Click(TObject *Sender)
{
```

Timer1->Enabled = ! Timer1->Enabled; }

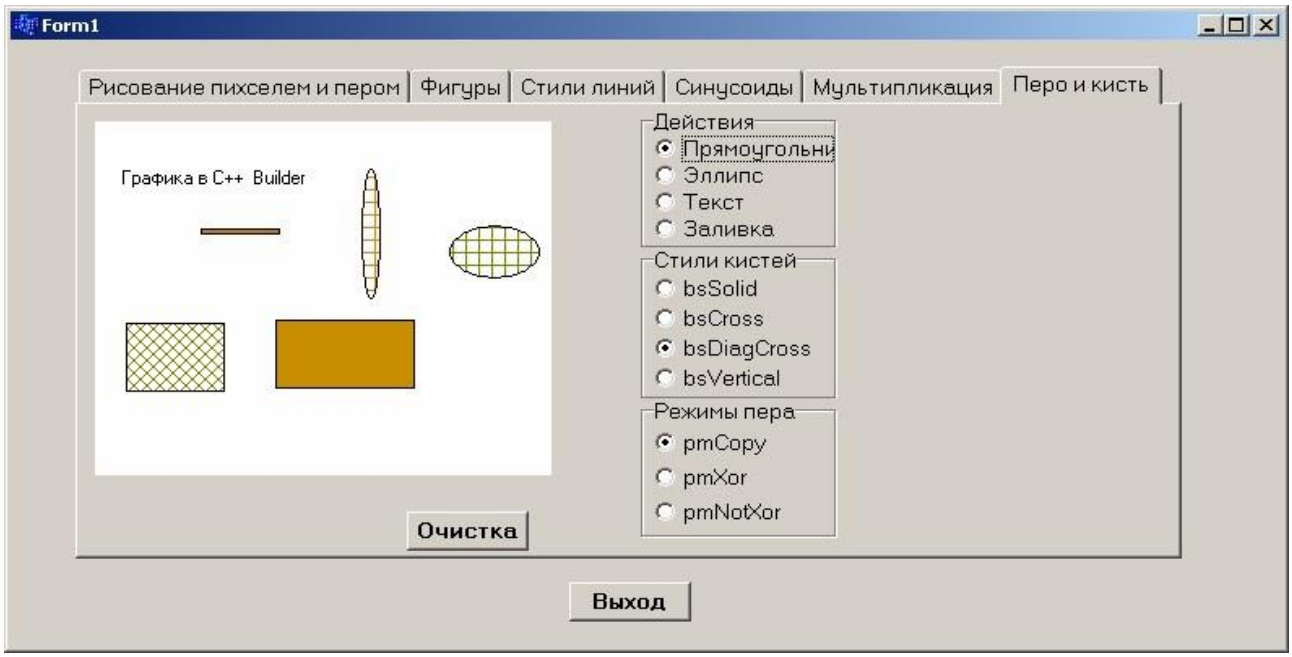
6. «Pero va mo'yqalam» bo’limi

6.1. Saxifa tavsifi

Bu sahifada sichqoncha tugmasi bosilishi bilan turli tanlangan shakil yoki yozuv chiqarish mumkin. Bunda mo’yqalam stili rangi tanlanadi. 1.6. programma matni izohlari bilan keltirilgan

6.1 *«*Tozalash*» **tugmasi***

```
void __fastcall TForm1::Button8Click(TObject *Sender)
{
  Image7->Canvas->Brush->Color=clWhite;
  Image7->Canvas->FillRect(Rect(0,0,Image7->Width,Image7->Height)); }
```



1.6 rasim − «Pero va mo'yqalam» saxifasi

6.2 Programma matni(sichqoncha tugmasini bosishda amalga oshiriladi):

```
void __fastcall TForm1::Image7MouseDown(TObject *Sender, TMouseButton
Button,TShiftState Shift,int X,int Y)
{ int sx,sy;
sx=random(100);
sy=random(100);
Image7->Canvas->Brush->Color=clGreen+sx*sy;
 switch (RG4->ItemIndex)
{  case 0: Image7->Canvas->Pen->Mode=pmCopy;
break;
case 1: Image7->Canvas->Pen->Mode=pmXor;
break;
case 2: Image7->Canvas->Pen->Mode=pmNotXor;
break;  }
switch (RG3->ItemIndex) {
 case 0: Image7->Canvas->Brush->Style=bsSolid;
break;
case 1: Image7->Canvas->Brush->Style=bsCross;
break;
case 2: Image7->Canvas->Brush->Style=bsDiagCross;
break;
case 3: Image7->Canvas->Brush->Style=bsVertical;
break;  }
switch (RG2->ItemIndex) {//to’g`ri to’trnurchak
 case 0: Image7->Canvas->Rectangle(X,Y,X+sx,Y+sy);
break;
 case 1: Image7->Canvas->Ellipse(X,Y,X+sx,Y+sy);
break;
//ellips
//matin yozamiz
case 2: Image7->Canvas->TextOut(X,Y, "Grafika C++  Builder");
break; case 3: //ixtiyotiy tanlangan qizil rang
Image7->Canvas->Brush->Color=clRed+sx*sy;
```

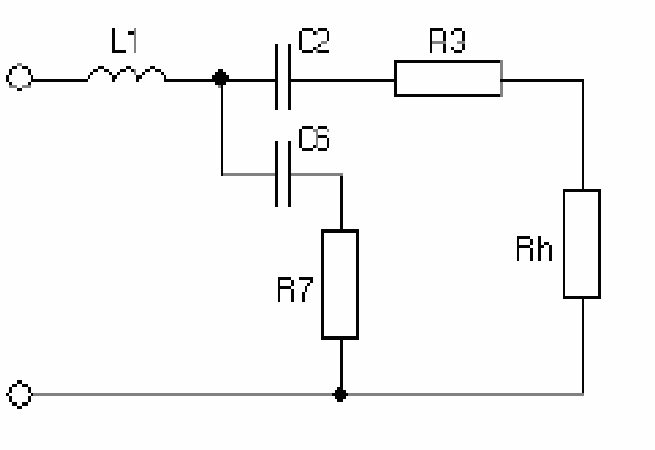
```
Image7->Canvas->FloodFill(X,Y,clWhite,fsSurface); //zalivka
break;  }  }
```

Labaratoriya ishi № 2

ELEKTOR SXEMASINI QURISH

Ish maqsadi. C++ dasturlash tilida Dastur ijrosi bo'yicha aniq bir misol bilan tanishish

Vazifa: 1. Quyida keltirilgan rasim C++ Builder muxitida chizilsin. 2.1. Sxema qurish uchun alohida tasvir yaratish funksiyalaridan foydalanilsin.



Pragramma matni:

```
#include <vcl.h>
#pragma hdrstop
#include "Unit1.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
TForm1 *Form1;

struct point // x,y koordinatalari strukturasi
{int x,y;};

point l; // struktura turidagi o’zgaruvchi

enum pos {h,v,left,right,up,down};  // chapga o’nga yuqoriga pasga yurish
```

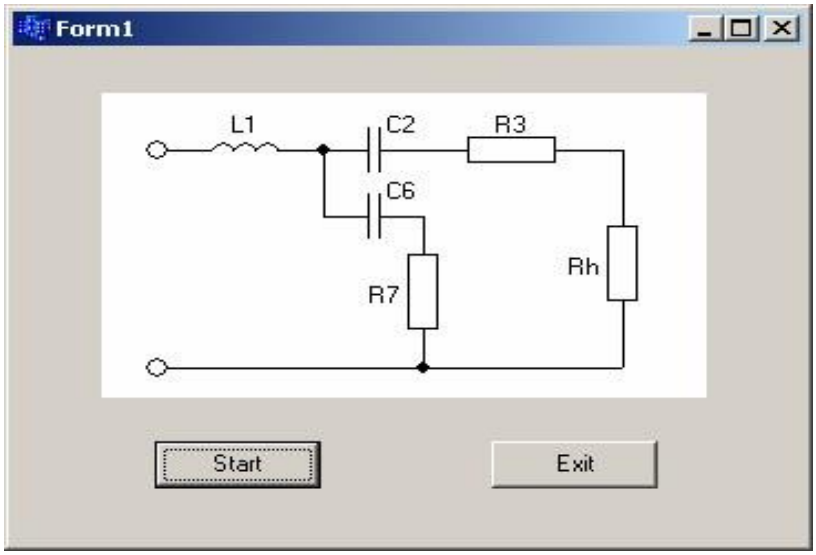
```
        TCanvas *t; // tekislik

        __fastcall TForm1::TForm1(TComponent* Owner)
              : TForm(Owner)
        {t=Image1->Canvas;   // Image1->Canvas }
        // l  struktura elementiga  x, y  koordinatalarini berish
         void set(int x, int y)
        { l.x = x;
        l.y = y; }
//  l  o'zgaruvchi elementlarini p strukurasiga berish
void set(point p)
        {
        l.x = p.x;
        l.y = p.y;
        }

        // aylana chizish
         void k(pos p)
        { switch(p)
        { case left: t->Ellipse(l.x-10,l.y-5,l.x,l.y+5);
        break;
        case right: t->Ellipse(l.x,l.y+5,l.x+10,l.y-5);
        break;
        case up: t->Ellipse(l.x-5,l.y-10,l.x+5,l.y);
        break;
        case down: t->Ellipse(l.x-5,l.y,l.x+5,l.y+10);
        break;
        }
        }

        //uopiq aylanalar chizish
         void kf(void)
        {t->Brush->Color=clBlack;
```

```
t->Ellipse(l.x-3,l.y-3,l.x+3,l.y+3);
t->FloodFill(l.x,l.y,clBlack,fsBorder);
t->Brush->Style=bsClear;
}
// Gorizantal va vertical to’g’ri chiziqlar chizish
void q(int len,pos q)
{
t->MoveTo(l.x,l.y); //boshlang’ich qiymatlar
 if(q){ t->LineTo(l.x, l.y+len);
// vertical chiziq
l.y+=len; }
else{ t->LineTo(l.x+len,l.y);
// gorizontal chiziq
l.x+=len;
}
}

// resistor
 // vertical gorizantal yozuv
void R(pos q,String s,pos p)
{ if(q)
{
t->Rectangle(l.x-7,l.y,l.x+7,l.y+40);  // vertical rezistor
switch(p){
case left:t->TextOutA(l.x-5-s.Length()*10,l.y+15,s);
break;
case right:t->TextOutA(l.x+10,l.y+15,s);
break;
}
l.y = l.y+40; }
else{ t->Rectangle(l.x,l.y-7,l.x+40,l.y+7);
// gorizantal rezistor
switch(p){
```

```
case up:t->TextOutA(l.x+12,l.y-20,s);
break;
case down:t->TextOutA(l.x+12,l.y+8,s);
break;

}
l.x+=+40;
}
}
// Kondensator
void C(pos q,String s,pos p){
if(q){                          // vertical kondensator
t->MoveTo(l.x-12,l.y);
t->LineTo(l.x+12,l.y);
t->MoveTo(l.x-12,l.y+5);
t->LineTo(l.x+12,l.y+5);
switch(p){
case left:t->TextOutA(l.x-5-s.Length()*10,l.y+10,s);
break;
case right:t->TextOutA(l.x+10,l.y+10,s);
break;
}
l.y Q= 5;
}
else { // gorizantal kondensator
t->MoveTo(l.x,l.y-12);
t->LineTo(l.x,l.y+12);
t->MoveTo(l.x+5,l.y-12);
t->LineTo(l.x+5,l.y+12);
switch(p){
case up:t->TextOutA(l.x+8,l.y-20,s);
break;
case down:t->TextOutA(l.x+8,l.y+6,s);
```

```
break;
}
l.x += 5;
}
}

// induktivlik:
 void L(pos q,String s,pos p)
{if(q){
for (int i=0;i<3;i++)
t->Arc(l.x-4,l.y+i*10,l.x+4,l.y+i*10+10,l.x,l.y+i*10+10,l.x,l.y+i*10);
switch(p){
case left:t->TextOutA(l.x-2-s.Length()*8,l.y+10,s);
break;
case right:t->TextOutA(l.x+8,l.y+10,s);break;
}
l.y += 30;
}
else{              //vertikal
 for (int i=0;i<3;i++)
t->Arc(l.x+i*10,l.y-4,l.x+i*10+10,l.y+5,l.x+i*10+10,l.y,l.x+i*10,l.y);
switch(p){
case up:t->TextOutA(l.x+8,l.y-20,s);
break;
case down:t->TextOutA(l.x+8,l.y+6,s);
break;
}
l.x += 30; }
}
//------- Start tugmasi ----------------------------------------------------
void __fastcall TForm1::Button1Click(TObject *Sender)
{point s0,s1;
set(30,30);k(left); // sxema boshi
 q(20,h); // gorizontal chiziq
```

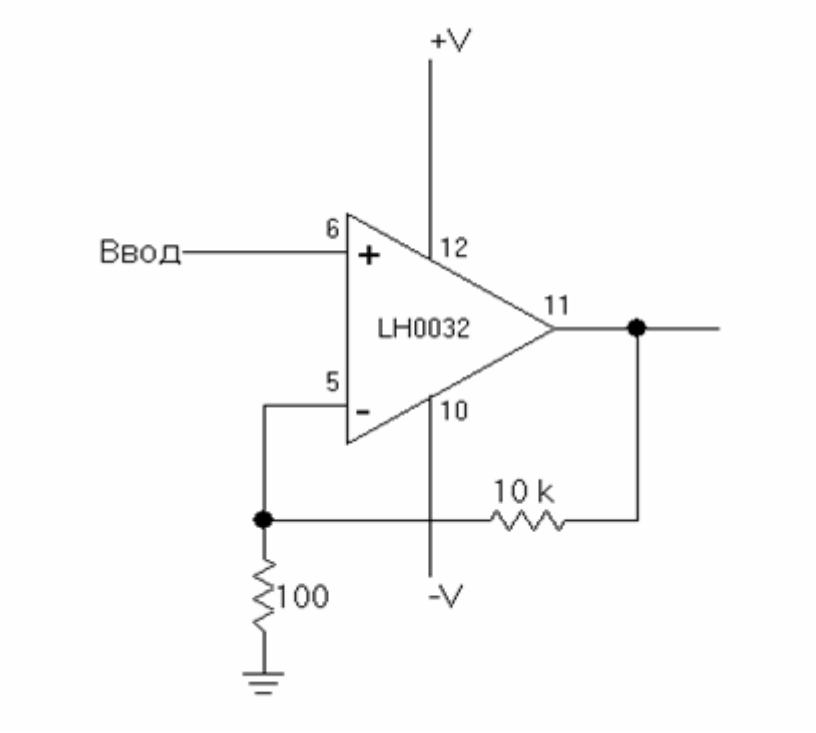
```
L(h,"L1",up); // L1 tepada
q(20,h);kf();s0ql; // s0 nuqta tutashuvi
q(20,h);
C(h,"C2",up); //
q(40,h);
R(h,"R3",up);//
q(30,h);q(40,v); //
R(v,"Rh",left); //
q(35,v);s1=l;
set(s0);q(35,v);
q(20,h);
C(h,"C6",up);//
q(20,h);q(20,v); //
R(v,"R7",left); //
q(s1.y-l.y,v); // vertical chiziq
s0=l; set(30,s1.y);  k(left); //
q(s0.x-l.x,h); //
kf();//
q(s1.x-l.x,h); //
}
void __fastcall TForm1::Button2Click(TObject *Sender)
{
Close();
}
```



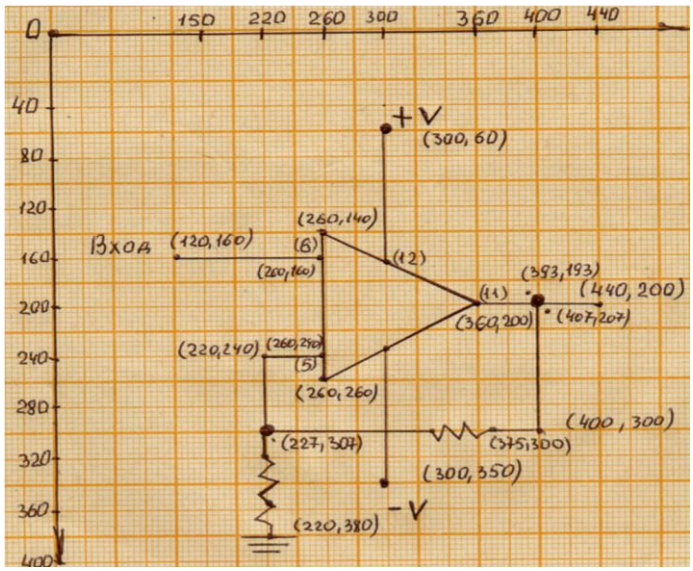
2.2 rasim – Programma natijasi

Vazifa 2:

2.3 sxemani c++ gravika funksiyalaridan foydalangan holda yaratilsin



2.3 rasim



2.4 rasim

Programma matni:

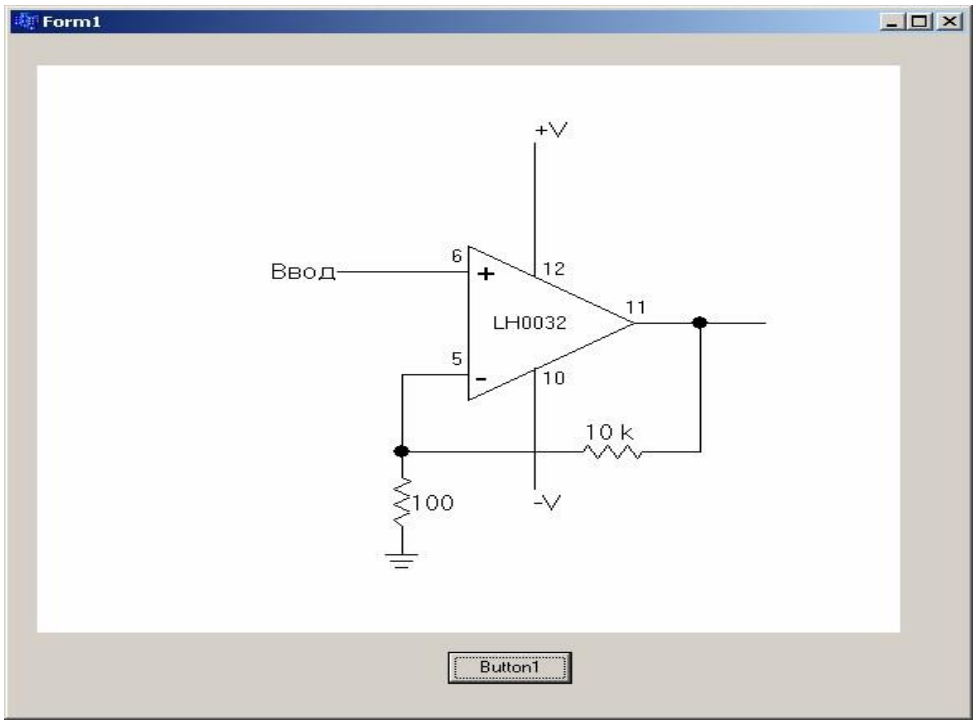
```
void __fastcall TForm1::Button1Click(TObject *Sender)
{//Zalivkaprozrachnaya (bezfona)
Image1->Canvas->Brush->Style=bsSolid;
Image1->Canvas->MoveTo(300,50);
Image1->Canvas->LineTo(300,165);
Image1->Canvas->MoveTo(300,235);
Image1->Canvas->LineTo(300,330);
// uchburchak
Image1->Canvas->MoveTo(260,140);
Image1->Canvas->LineTo(360,200);
Image1->Canvas->LineTo(260,260);
Image1->Canvas->LineTo(260,140);
//gorizontal chiziq
Image1->Canvas->MoveTo(360,200);
Image1->Canvas->LineTo(440,200);
// chiziq (chap)
Image1->Canvas->MoveTo(400,200);
Image1->Canvas->LineTo(400,300);
Image1->Canvas->LineTo(365,300);
//«Zubchiklar»
//TPoint foydalanamiz
 points[8];
points[0].x=365;
points[0].y=300;
points[1].x=362;
points[1].y=305;
points[2].x=356;
points[2].y=295;
points[3].x=350;
points[3].y=305;
points[4].x=344;
```

```
points[4].y=295;
points[5].x=338;
points[5].y=305;
points[6].x=332;
points[6].y=295;
points[7].x=329;
points[7].y=300;
 Image1->Canvas->Polyline(points,7);
//
Image1->Canvas->MoveTo(329,300);
//gorizantal chiziq chap
Image1->Canvas->LineTo(220,300);
// uchburchak tepa chap qismida joylashgan chiziq
Image1->Canvas->MoveTo(260,160);
Image1->Canvas->LineTo(180,160);
// uchburchakdan pastda  joylashgan chiziq
Image1->Canvas->MoveTo(260,240);
Image1->Canvas->LineTo(220,240);
//
Image1->Canvas->LineTo(220,320);
//«Zubchiklar»
s[0].x=220;
points[0].y=320;
points[1].x=225;
points[1].y=323;
points[2].x=215;
points[2].y=329;
points[3].x=225;
points[3].y=335;
points[4].x=215;
points[4].y=341;
points[5].x=225;
points[5].y=347;
```

```
points[6].x=215;
points[6].y=353;
points[7].x=220;
points[7].y=358;
Image1->Canvas->Polyline(points,7);
Image1->Canvas->MoveTo(220,358);

Image1->Canvas->LineTo(220,380);
//
int x=10; int y=380;
for (int i=0; i<3; i++){
Image1->Canvas->MoveTo(220-x,y);
Image1->Canvas->LineTo(220+x,y);
 x-=3; y+=5;
}
Image1->Canvas->Brush->Color=clBlack; //

Image1->Canvas->Ellipse(393,193,407,207); //

Image1->Canvas->FloodFill(400, 200, clBlack, fsBorder);
Image1->Canvas->Ellipse(213,293,227,307); //

Image1->Canvas->FloodFill(220, 300, clBlack, fsBorder);
Image1->Canvas->Brush->Color= clWhite; //
Image1->Canvas->Font->Color = clBlack; //
Image1->Canvas->Font->Size=12;
Image1->Canvas->TextOut(140,150,"Kiritish:");
Image1->Canvas->TextOut(300,40,"+V");
Image1->Canvas->TextOut(300,330,"-V");
Image1->Canvas->TextOut(330,275,"10 k");
Image1->Canvas->TextOut(225,330,"100");
//Razmer shrifta 10
Image1->Canvas->Font->Size=10;
```

```
Image1->Canvas->TextOut(305,150,"12");
Image1->Canvas->TextOut(305,235,"10");
Image1->Canvas->TextOut(355,180,"11");
Image1->Canvas->TextOut(250,140,"6");
Image1->Canvas->TextOut(250,220,"5");
Image1->Canvas->TextOut(275,192,"LH0032");
//Razmer shrifta 14
Image1->Canvas->Font->Size=14;
Image1->Canvas->TextOut(265,150,"+");
Image1->Canvas->TextOut(265,230,"-");  }
```

C++ Builderdagi natija:

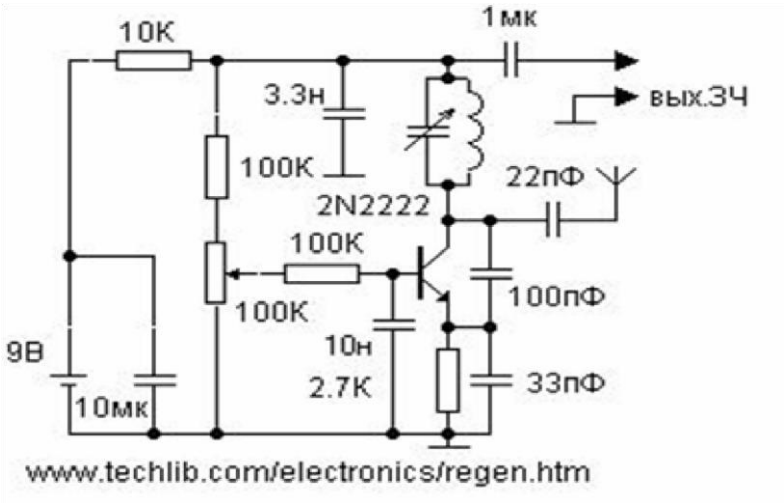


2.5 **rasim**

Laboratornaya ishi № 3

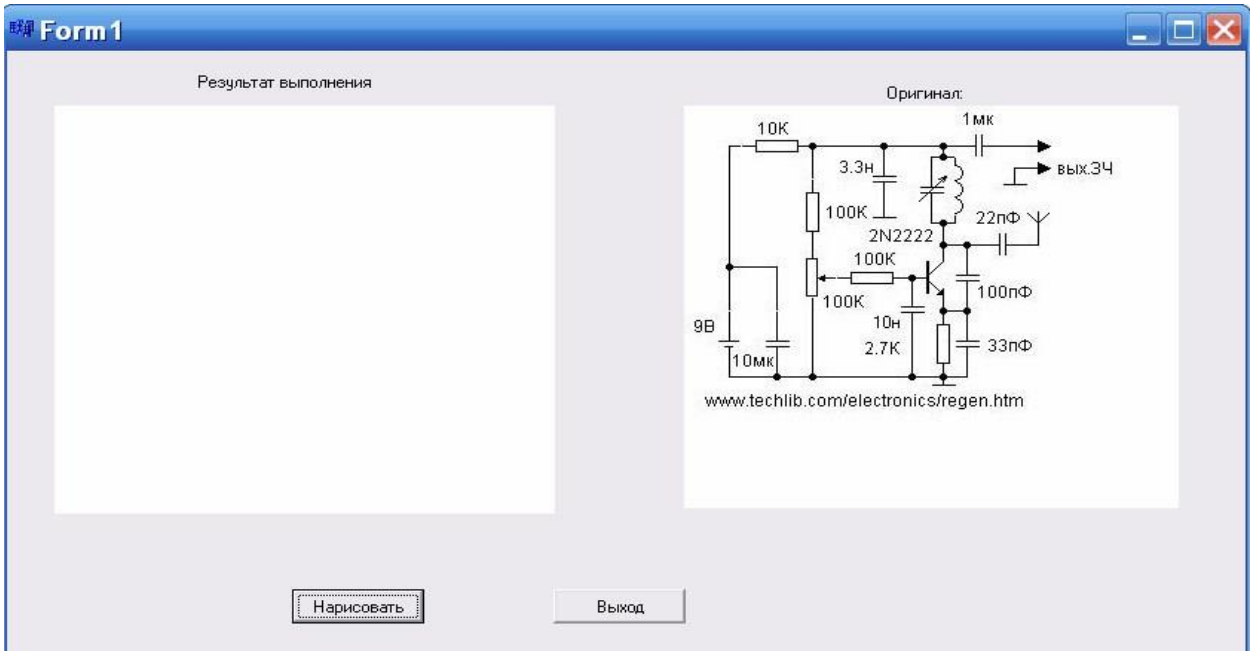
Grafika – sxemalar qurish

Vazifa 1: C++ grafik imkoniyatlaridan foydalanib grafika chizish 3.1 rasim



3.1 rasim

Forma ko’rinishi



3.2 rasim

Dastur matni:

```
#include <vcl.h>
#pragma hdrstop
#include "Unit1.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
TForm1 *Form1; struct
point
{int x,y;};

point l;
enum pos {h,v,left,right,up,down};
TCanvas *t;

__fastcall TForm1::TForm1(TComponent* Owner)
        : TForm(Owner)
{  t=Image1->Canvas;  }

void set(int x, int y)
{ l.x = x;  l.y = y; }

void set(point p)
{ l.x = p.x; l.y = p.y; }

void kf(void)
{
t->Brush->Color=clBlack;
t->Ellipse(l.x-3,l.y-3,l.x+3,l.y+3);
t->FloodFill(l.x,l.y,clBlack,fsBorder);
t->Brush->Style=bsClear;  }

void q(int len,pos q)
{ t->MoveTo(l.x,l.y);
```

```
 if(q){
t->LineTo(l.x,l.y+len);
l.y+=len;
}
else  {  t->LineTo(l.x+len,l.y);  l.x+=len;
} }

void R(pos q,String s,pos p)
{
if(q){
t->Rectangle(l.x-4,l.y,l.x+4,l.y+28);
 switch(p){
case left: t->TextOutA(l.x+5-s.Length()*10,l.y,s);
break;
case right: t->TextOutA(l.x+5,l.y+18,s);
break;    }
l.y = l.y+28;
       }
else
{ t->Rectangle(l.x,l.y-4,l.x+28,l.y+4);
switch(p) {
case up:t->TextOutA(l.x+5,l.y-20,s);
break;
case down : t->TextOutA(l.x+12,l.y+8,s);
break; }
 l.x+=+28; }
}

void C(pos q,String s,pos p){
if(q)
{ t->MoveTo(l.x-9,l.y);
t->LineTo(l.x+9,l.y);
t->MoveTo(l.x-9,l.y+4);
x+10-s.Length()*10,l.y-13,s);
```

```
break;
case right: t->TextOutA(l.x+15,l.y-3,s);
break;
}
l.y += 5; }
else{  t->MoveTo(l.x,l.y-
9);
t->LineTo(l.x,l.y+9);
t->MoveTo(l.x+4,l.y-9);
t->LineTo(l.x+4,l.y+9);
switch(p){
case up:t->TextOutA(l.x-8,l.y-25,s);
break;
case down:t->TextOutA(l.x+8,l.y+6,s);
break;
}
l.x += 5; }
}

void L(pos q,String s,pos p){
if(q){
for (int i=0;i<3;i++)
t->Arc(l.x-5,l.y+i*11,l.x+5,l.y+i*11+11,l.x,l.y+i*11+11,l.x,l.y+i*11);
switch(p){
case left: t->TextOutA(l.x-2-s.Length()*8,l.y+10,s);
break;
case right:t->TextOutA(l.x+8,l.y+10,s);
break;
}
l.y += 33; }
else{
for (int i=0;i<3;i++)
t->Arc(l.x+i*10,l.y-4,l.x+i*10+10,l.y+5,l.x+i*10+10,l.y,l.x+i*10,l.y);
switch(p){
```

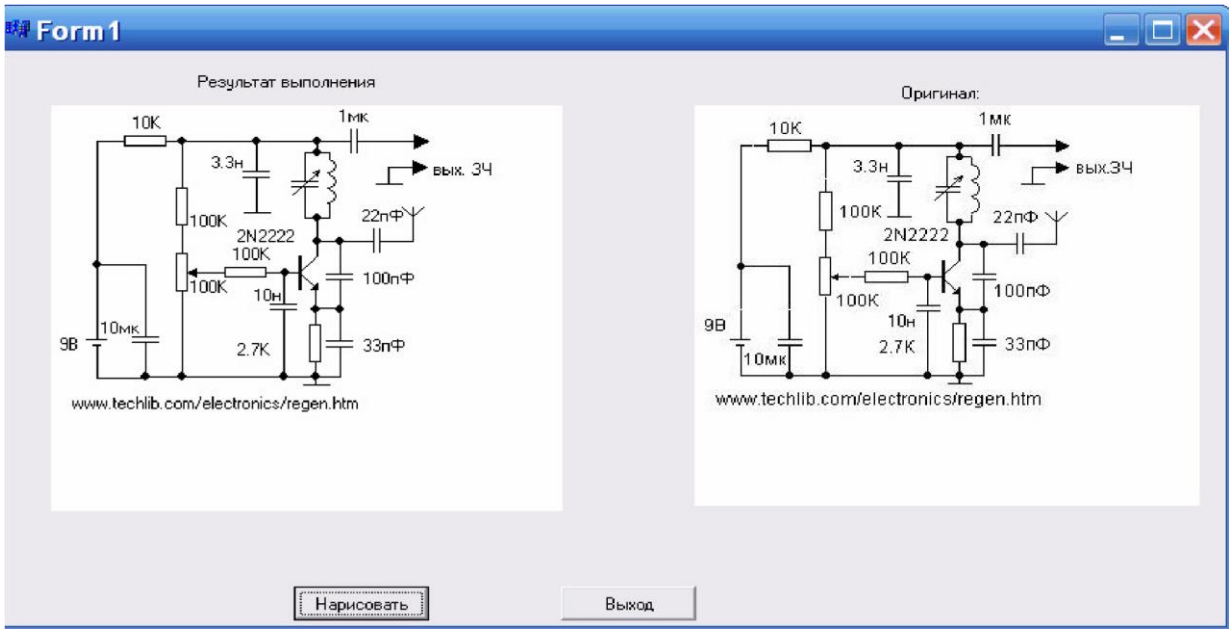
```
case up: t->TextOutA(l.x+8,l.y-20,s);
break;
case down:t->TextOutA(l.x+8,l.y+6,s);
break;
}
l.x += 33; } }
void H(pos q)
{if(q==v){t->MoveTo(l.x-7,l.y-6);
t->LineTo(l.x,l.y+1);
t->MoveTo(l.x+7,l.y-6);
t->LineTo(l.x,l.y+1);
t->MoveTo(l.x,l.y-6);
t->LineTo(l.x,l.y+1);}
else
{t->MoveTo(l.x+6,l.y-7);
t->LineTo(l.x-1,l.y);
t->MoveTo(l.x+6,l.y+7);
t->LineTo(l.x-1,l.y);
t->MoveTo(l.x+6,l.y);
t->LineTo(l.x-1,l.y);}}

void E(pos q,String s,pos p){
if(q){
t->MoveTo(l.x-7,l.y);
t->LineTo(l.x+7,l.y);
t->MoveTo(l.x-3,l.y+4);
t->LineTo(l.x+3,l.y+4);
switch(p){
case left:t->TextOutA(l.x-5-s.Length()*10,l.y-3,s);
break;
case right:t->TextOutA(l.x+15,l.y-3,s);break;  }
l.y += 5; }
else {
t->MoveTo(l.x,l.y-7);
```

```
t->LineTo(l.x,l.y+7);
t->MoveTo(l.x+5,l.y-4);
t->LineTo(l.x+5,l.y+4);
switch(p){
case up: t->TextOutA(l.x-13,l.y-28,s);
break;
case down: t->TextOutA(l.x+8,l.y+6,s);
break; } l.x += 5;  }  }
 void  sr()
 { t->Brush->Style=bsSolid;
t->Brush->Color=clBlack;
  TPoint p[4];
  p[0].x=l.x+9;
  p[0].y=l.y;
  p[1].x=l.x;
  p[1].y=l.y-5;
  p[2].x=l.x;
  p[2].y=l.y+5;
  p[3].x=l.x+9;
  p[3].y=l.y;
  t->Polygon(p,3);
  t->Brush->Color=clWhite;
  }

   void  sl()
 { t->Brush->Style=bsSolid;
t->Brush->Color=clBlack;
TPoint p[4];
p[0].x=l.x-5;p[0].y=l.y;
p[1].x=l.x;p[1].y=l.y-3;
p[2].x=l.x;p[2].y=l.y+3;
p[3].x=l.x-5;p[3].y=l.y;
t->Polygon(p,3);
t->Brush->Color=clWhite;
```

```
  }

void ze()
  {t->MoveTo(l.x+9,l.y);
    t->LineTo(l.x-9,l.y);
    t->MoveTo(l.x,l.y);
}
  void Cs( )
{t->MoveTo(l.x-9,l.y);
t->LineTo(l.x+9,l.y);
t->MoveTo(l.x-9,l.y+4);
t->LineTo(l.x+9,l.y+4);
t->MoveTo(l.x-8,l.y+10);
t->LineTo(l.x+10,l.y-8);
t->MoveTo(l.x+5,l.y-6);
t->LineTo(l.x+10,l.y-8);
t->MoveTo(l.x+8,l.y-4);
t->LineTo(l.x+10,l.y-8);
l.y+=5; }

void  Tr( )
{   TPoint p[4];
t->Brush->Color=clBlack;
p[0].x=l.x;p[0].y=l.y+24;
p[1].x=l.x;p[1].y=l.y+20;
p[2].x=l.x-4;p[2].y=l.y+24;
p[3].x=l.x;p[3].y=l.y+24;
t->LineTo(l.x-12,l.y+12);
t->LineTo(l.x-2,l.y+22);
t->Polygon(p,3);
t->Brush->Color=clWhite;
t->Pen->Width=2;
t->MoveTo(l.x-11,l.y);
t->LineTo(l.x-11,l.y+24);
```

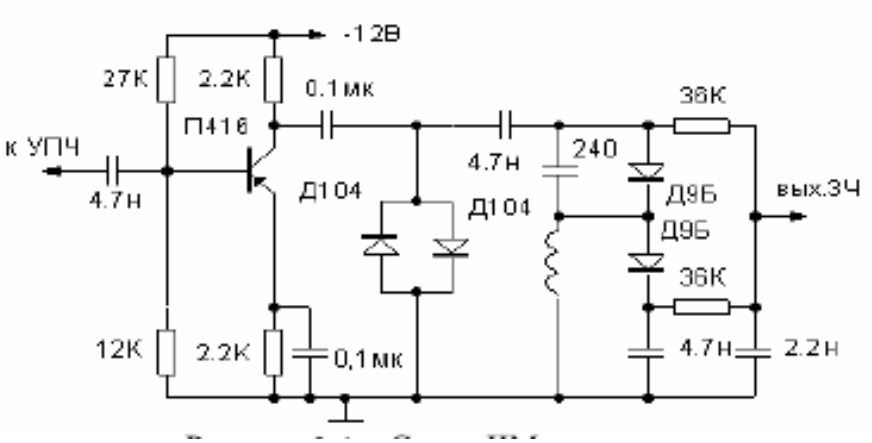
```
        t->Pen->Width=1;
        l.x-=11; l.y+=12;
        }


        void __fastcall TForm1::BitBtn2Click(TObject *Sender)
        {
        Close();
        }

void __fastcall TForm1::BitBtn1Click(TObject *Sender)
{  set(30,26);  q(90,v);  kf(); //30,116
q(32,h);  q(53,v); C(v,"10мк",left); q(24,v);  kf();  //62,198
q(24,h); kf(); // 86,198
q(67,h); kf(); //153,198
q(21,h); kf();// 174,198
q(16,h); q(-22,v); set(190,172); C(v,"33пФ",right);
 set(190,172);  q(-23,v); kf(); // 190, 148
q(-23,v); set(190,123); C(v,"100пФ",right);
set(190,123); q(-24,v); kf(); // 190,99
q(22,h); C(h,"22пФ",up); q(21,h); q(-19,v);
H(v); set(30,116); q(55,v); E(v,"9В",left);
q(23,v); q(33,h); set(30,26);
q(18,h); R(h,"10К",up); q(10,h);  kf();// 86,26
q(34,v); R(v,"100К",right); q(20,v); R(v,"100К",right);
q(59,v); set(86,26); q(49,h); kf();//135,26
q(40,h); kf();//175,26
q(22,h); C(h,"1мк",up); q(37,h); sr(); set(135,26);
q(22,v); C(v,"3.3н",left); q(25,v); ze();  set(175,26);
q(8,v);  kf();//175,34
q(-8,h);  q(21,v);  Cs();  q(22,v);  q(8,h);  kf();//175,82
q(17,v);  kf();//175,98
q(16,h);  set(175,34);  q(8,h);  q(5,v);  L(v,"",right);
q(10,v);  q(-8,h);  set(175,98);
q(12,v);  Tr();  q(-10,h);  kf();//xz
q(-13,h);   set(l.x-27,l.y);   R(h,"100К",up);
set(l.x-28,l.y);  q(-19,h);  sl();
```

```
set(153,124);  q(21,v);  C(v,"10н",left);
q(47,v);  set(174,198);  q(-11,v);
set(174,160);  R(v,"",left);    set(174,160);
q(-11,v);    kf();     q(-15,v);
set(190,148);  q(-16,h);  set(174,198);
q(6,v);  ze();   t->TextOutA(122,172,"2.7K");
t->TextOutA(122,89,"2N2222");
t->TextOutA(13,211,"www.techlib.com/electronics/regen.htm");
set(222,45);  q(16,h);    sr(); set(222,45);
q(12,v);   ze(); set(251,42);
t->TextOutA(251,40,"вых. ЗЧ");
}
```



3.3 rasim

Vazifa 2 : C++ da chizilsin



Programma matni:

```
#include <vcl.h>
#pragma hdrstop
#include "Unit1.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
TForm1 *Form1;

struct point //
{int x,y;};
point l; //
enum pos {h,v,left,right,up,down};//

TCanvas *t; //
//---------------------------------------------------------------------------
__fastcall TForm1::TForm1(TComponent* Owner)
        : TForm(Owner)
{ t=Image1->Canvas;   }
//---------------------------------------------------------------------------
void set(int x, int y)
{ l.x = x;  l.y = y; }
//---------------------------------------------------------------------------
void set(point p)
{ l.x = p.x;
l.y = p.y;  }
//---------------------------------------------------------------------------
void q(int len,pos q)
{
```

```
t->MoveTo(l.x,l.y); //
if(q){
t->LineTo(l.x,l.y+len); //
l.y+=len;
}else{
t->LineTo(l.x+len,l.y); //
l.x+=len;  }
}
//
void L()
{ for (int i=0;i<3;i++)
t->Arc(l.x-4,l.y+i*10,l.x+4,l.y+i*10+10,l.x,l.y+i*10+10,l.x,l.y+i*10);
l.y+=30;  }

void kf(void)
{
t->Brush->Color=clBlack;
t->Ellipse(l.x-3,l.y-3,l.x+3,l.y+3);
t->FloodFill(l.x,l.y,clBlack,fsBorder);
t->Brush->Style=bsClear;
}
void R(pos q,String s,pos p){t->Pen->Width=1;
if(q){//
t->Rectangle(l.x-7,l.y,l.x+7,l.y+40);
switch(p){
case left:t->TextOutA(l.x-30,l.y+10,s);break; //
case right:t->TextOutA(l.x+10,l.y+15,s);break; //
}
l.y+=40;
}else{
t->Rectangle(l.x,l.y-7,l.x+40,l.y+7); //
switch(p){
case up:t->TextOutA(l.x+12,l.y-20,s);break;  //
case down:t->TextOutA(l.x+12,l.y+8,s);break; //
}
l.x+=40;  }
}
//---------------------------------------------------------------------------
```

```
void C(pos q,String s,pos p){
if(q){      //
t->MoveTo(l.x-12,l.y);
t->LineTo(l.x+12,l.y);
t->MoveTo(l.x-12,l.y+5);
t->LineTo(l.x+12,l.y+5);
switch(p){
case left:t->TextOutA(l.x-5-s.Length()*10,l.y+10,s);break; //
case right:t->TextOutA(l.x+10,l.y+10,s);break; //
}
l.y += 5;
}else{    //
t->MoveTo(l.x,l.y-12);
t->LineTo(l.x,l.y+12);
t->MoveTo(l.x+5,l.y-12);
t->LineTo(l.x+5,l.y+12);
switch(p){
case up:t->TextOutA(l.x+4,l.y-25,s);break; //
case down:t->TextOutA(l.x-5,l.y+15,s);break; //
}
l.x += 5; }
}
//---------------------------------------------------------------------------
void ST(pos q,String s,pos p)
{  //влево
if(q){
t->MoveTo(l.x,l.y);
t->LineTo(l.x+6,l.y+3);
t->LineTo(l.x+6,l.y-3);
t->LineTo(l.x,l.y);
switch(p){
case up:t->TextOutA(l.x-15,l.y-20,s);break; //
case right:t->TextOutA(l.x+8,l.y-8,s);break; //
}
l.x+=6;
}
else
{      //
```

```
t->MoveTo(l.x,l.y);
t->LineTo(l.x-6,l.y-3);
t->LineTo(l.x-6,l.y+3);
t->LineTo(l.x,l.y);
switch(p){
case up:t->TextOutA(l.x-15,l.y-20,s);break; //
case right:t->TextOutA(l.x+10,l.y-8,s);break; //
}l.x-=6;  }
}
//---------------------------------------------------------------------
//
void T1(pos q)
{
if(q){
t->Pen->Width=2;
t->MoveTo(l.x-16,l.y);
t->LineTo(l.x+16,l.y);
t->Pen->Width=1;
t->MoveTo(l.x-3,l.y);
t->LineTo(l.x-16,l.y+13);
t->MoveTo(l.x+3,l.y);
t->LineTo(l.x+16,l.y+13);
//
t->Pen->Width=2;
t->MoveTo(l.x-9,l.y+3);
t->LineTo(l.x-1,l.y-1);
t->MoveTo(l.x-1,l.y-1);
t->LineTo(l.x-7,l.y+6);
}else {
t->Pen->Width=2;
t->MoveTo(l.x,l.y-16);
t->LineTo(l.x,l.y+16);
t->Pen->Width=1;
t->MoveTo(l.x,l.y-3);
t->LineTo(l.x+13,l.y-16);
t->MoveTo(l.x,l.y+3);
t->LineTo(l.x+13,l.y+16);
//
```

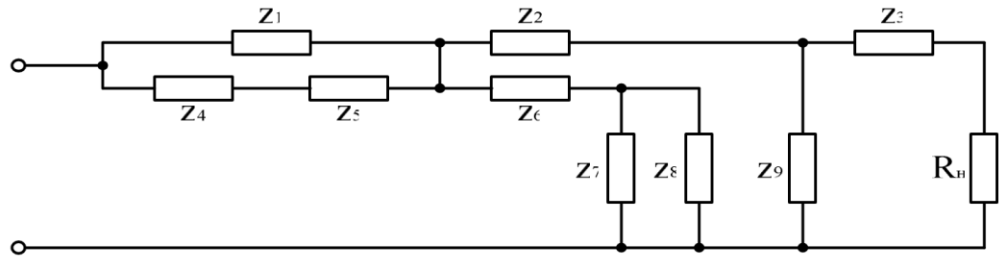
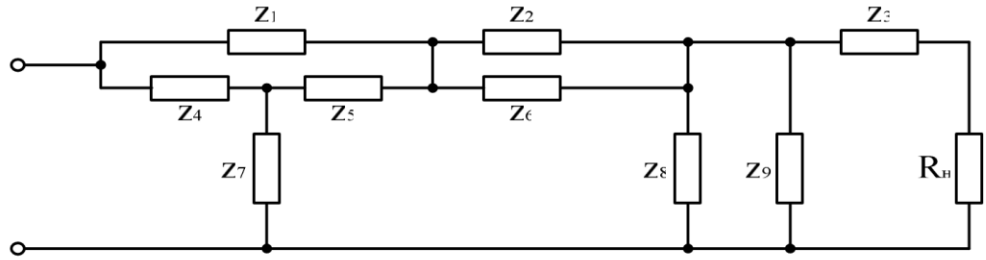
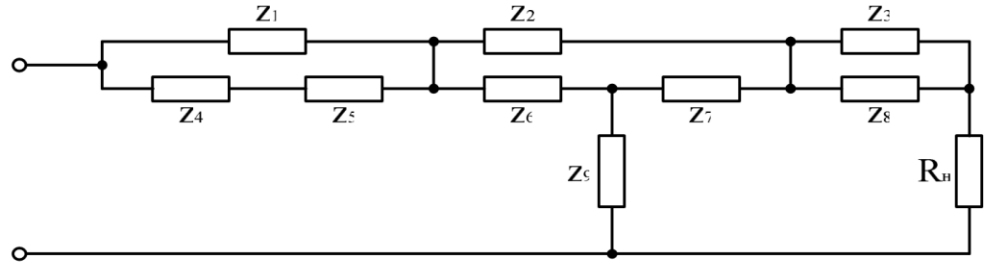
```
t->Pen->Width=2;
t->MoveTo(l.x,l.y+5);
t->LineTo(l.x+3,l.y+10);
t->MoveTo(l.x,l.y+4);
t->LineTo(l.x+6,l.y+7);
t->Pen->Width=4;
t->LineTo(l.x+3,l.y+10);  }
}
void Z()
{
t->MoveTo(l.x-9,l.y);
t->LineTo(l.x+9,l.y);
}
 //
void VD(pos q,String S,pos p)
{ if(q) {
t->Pen->Width=1;
t->MoveTo(l.x,l.y);
t->LineTo(l.x+10,l.y);
t->MoveTo(l.x,l.y);
t->LineTo(l.x-10,l.y);
t->LineTo(l.x,l.y+10);
t->MoveTo(l.x+10,l.y);
t->LineTo(l.x,l.y+10);
t->MoveTo(l.x+10,l.y+10);
t->LineTo(l.x-10,l.y+10);
l.x;  l.y+=10;}
else {
t->MoveTo(l.x,l.y);
t->LineTo(l.x-10,l.y+10);
t->MoveTo(l.x,l.y);
t->LineTo(l.x+10,l.y+10);
t->LineTo(l.x-10,l.y+10);
t->MoveTo(l.x+10,l.y);
t->LineTo(l.x-10,l.y);
}
switch (p){
case left:t->TextOutA(l.x-35,l.y-20,S);break;
```

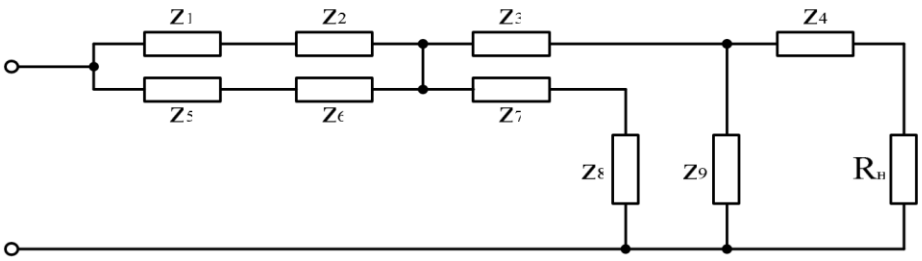
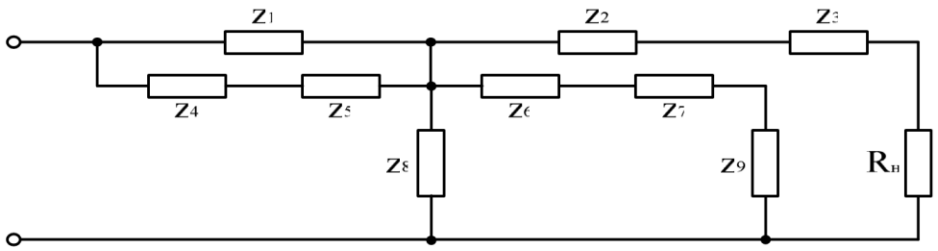
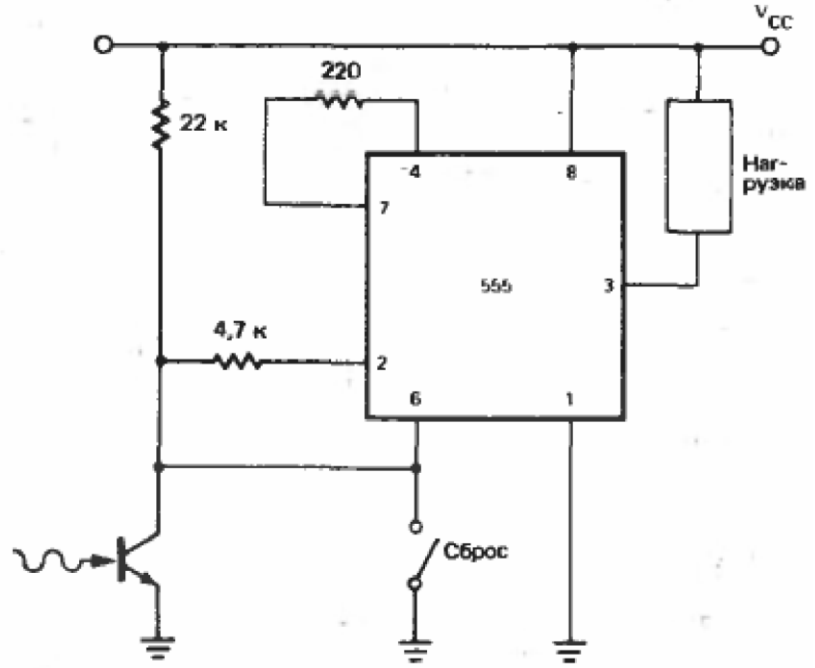
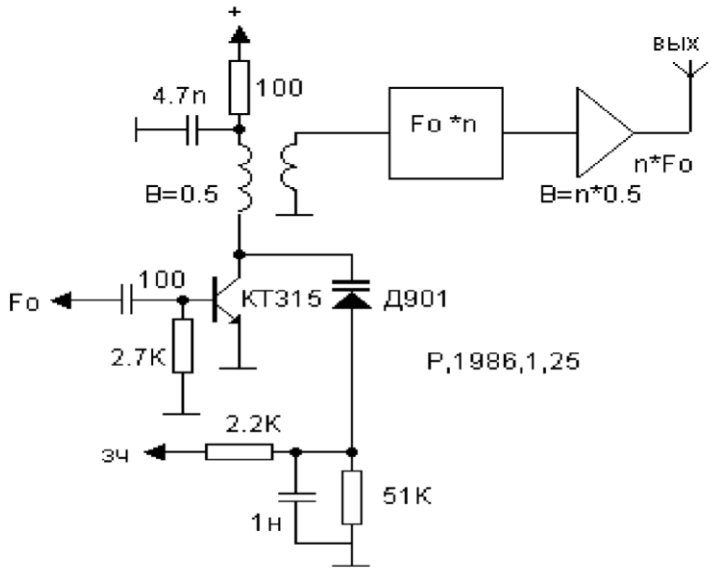
```
case right:t->TextOutA(l.x+10,l.y-24,S);break;
} }

void __fastcall TForm1::Image1DblClick(TObject *Sender)
{
Close();
}

void __fastcall TForm1::FormCreate(TObject *Sender)
{
BitBtn1Click(0);
}

void __fastcall TForm1::BitBtn1Click(TObject *Sender)
{
t->Pen->Width=1;
point s0,s1,s2,s3,s4,s5,s6,s7,s8,s9,s10,s11,s12,s13,s14,s15,s16,s17,s18,s19;
set(170,30); ST(h,"-12B",right);
q(-20,h); kf();s0=l;q(-60,h);q(10,v);R(v,"27K",left);
q(40,v);kf();s1=l;q(-24,h);set(30,120);
ST(v,"к УПЧ",up);q(20,h);
C(h,"4.7н",down); set(s1);q(100,v);
R(v,"12K",left);q(20,v);
q(60,h);kf();s2=l;q(20,h);kf();
s3=l;q(20,h);kf();s4=l;q(40,h);kf();
s5=l;q(65,h);kf();s6=l;q(35,h);kf();s7=l;
set(s1);q(48,h); T1(h);
t->Pen->Width=1; set(s0);q(10,v);
R(v,"2.2K",left);q(10,v);kf();s8=l;q(15,v);
set(144,135);q(70,v);kf();s9=l;q(15,v);
R(v,"2.2K",left);q(20,v); set(s9);q(20,h);q(35,v);
C(v,"0,1мк",right);q(35,v);
set(s4);q(15,v);Z(); set(s8);q(20,h);
C(h,"0.1мк",up);q(55,h);kf();s10=l;
q(60,v);kf();s11=l;q(15,h);q(25,v);
VD(v,"Д104",right);q(25,v);q(-15,h);kf();s12=l;q(-15,h);q(-25,v);
set(s11);q(-15,h);q(25,v);
VD(h,"Д104",left);set(s12);q(70,v);
```

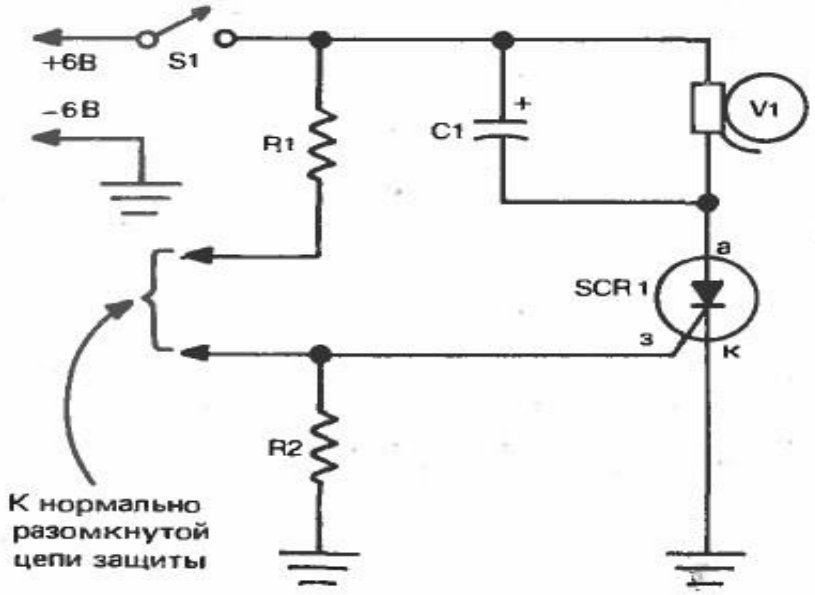
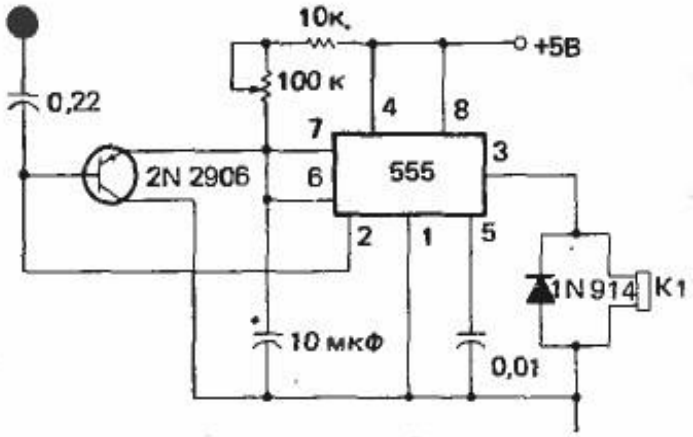
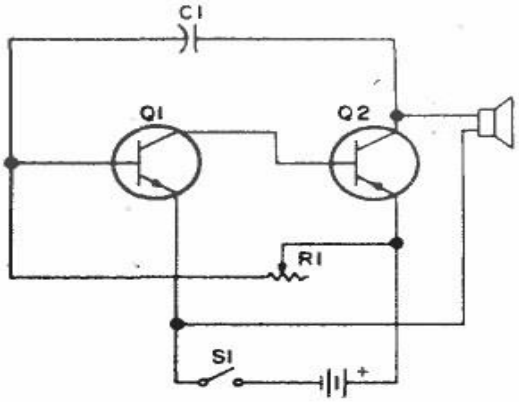
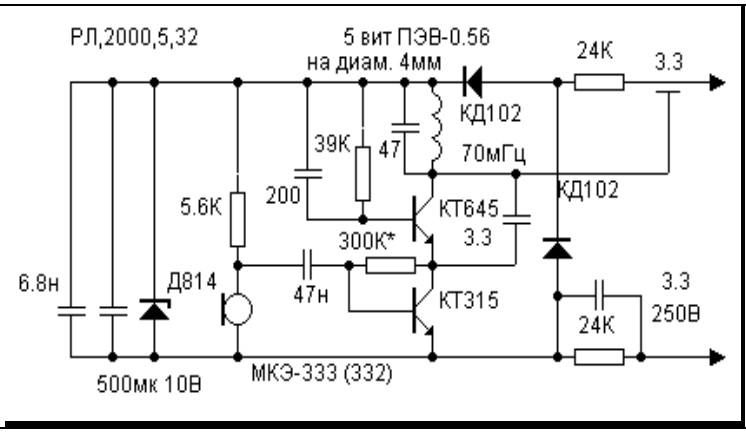
```
set(s10);q(30,h);
C(h,"4.7н",down);q(30,h);kf();s13=l;
q(20,v);C(v,"240",right);q(20,v);kf();s14=l;q(20,v);
L();q(s6.y-l.y,v); set(s14);q(35,h);kf();
set(s13);q(35,h);kf(); s15=l;q(20,v);
VD(v,"Д9Б",right); q(50,v);
VD(v,"Д9Б",right);q(30,v); kf(); s16=l; q(30,v);
C(v,"4.7н",right);q(35,v); set(s15); q(20,h);
R(h,"36К",up);q(10,h);q(45,v);kf(); s17=l;q(s16.y-l.y,v); kf();
q(30,v);C(v,"2.2н",right);q(35,v); q(s16.x-l.x,h);
set(s16);q(15,h);
R(h,"36К",up);q(15,h);  set(s17); q(25,h);
ST(h,"вых. ЗЧ",up);
}

void __fastcall TForm1::BitBtn2Click(TObject *Sender)
{
Close();
}
```

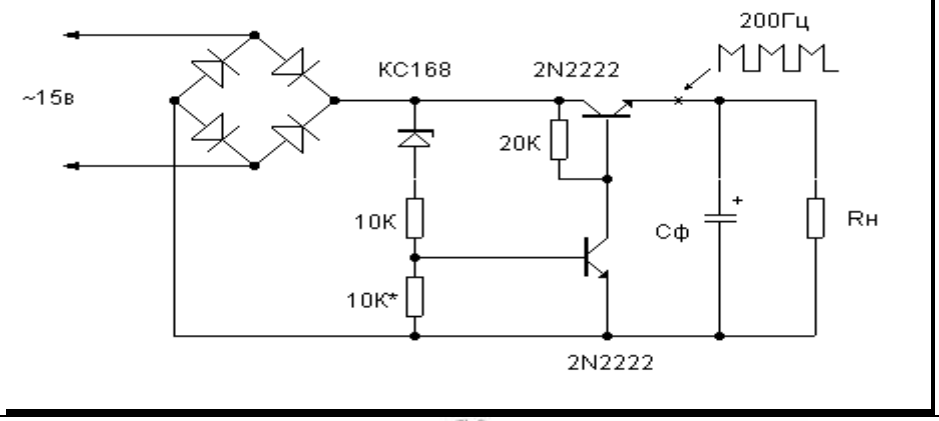
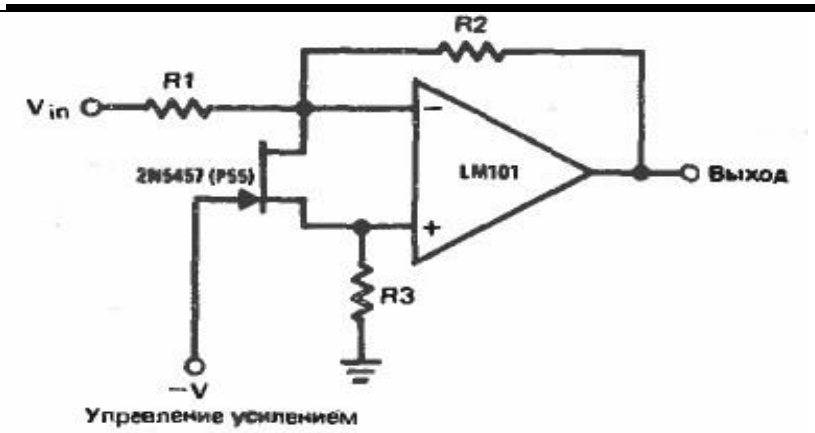
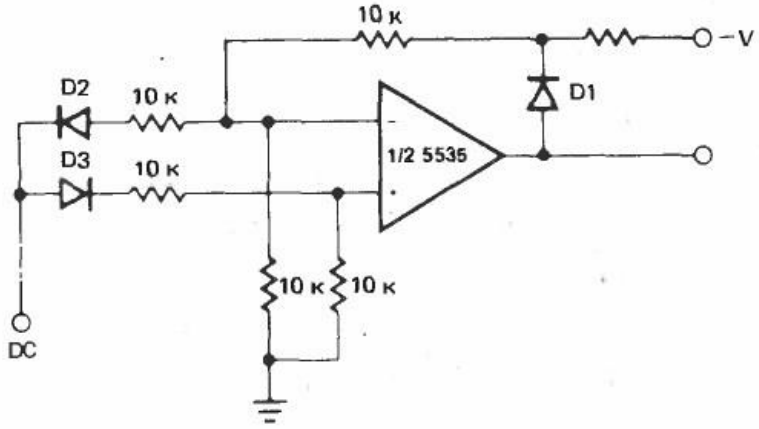
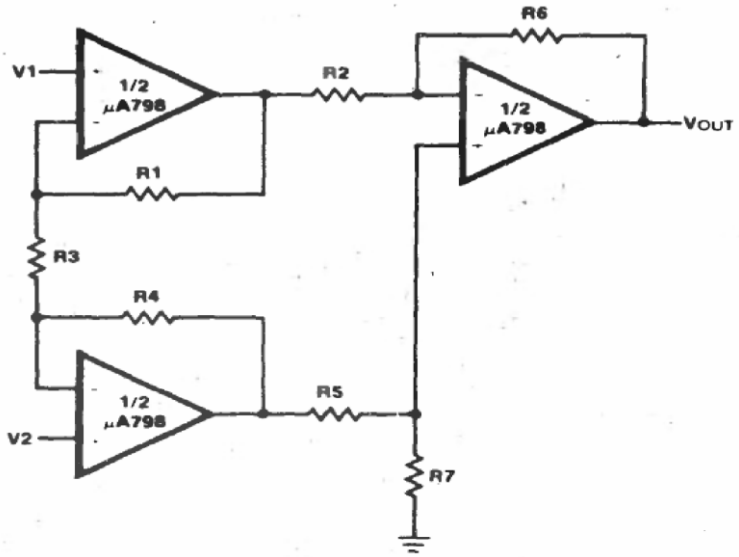
3.5 rasim

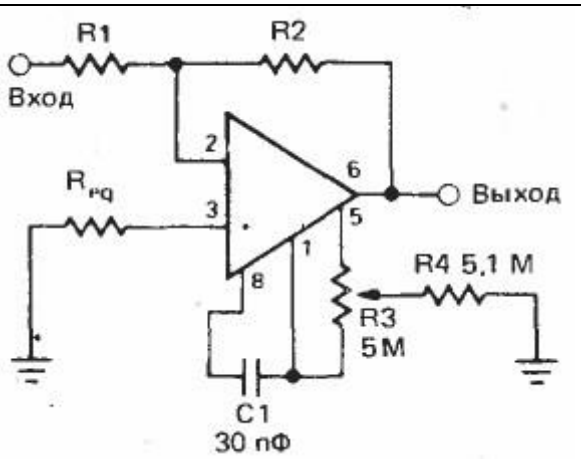
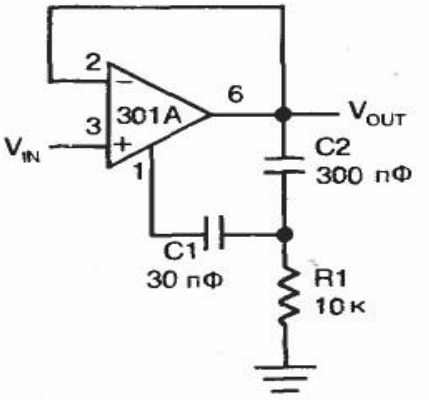
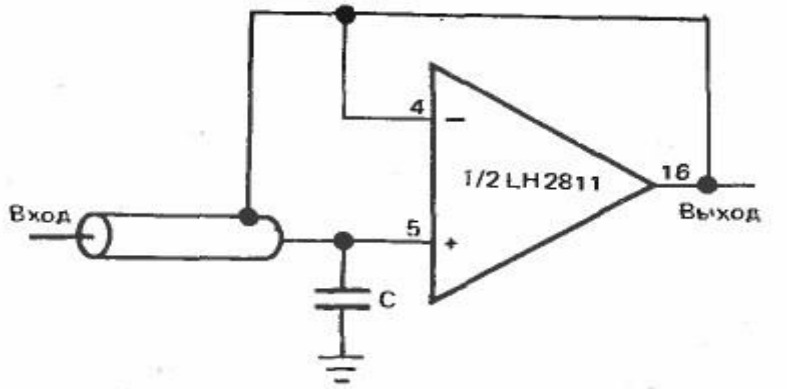
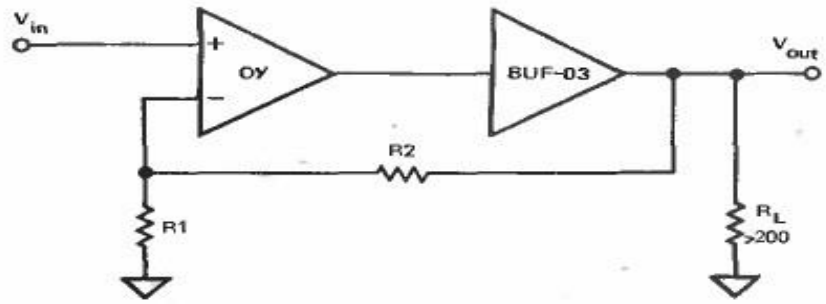
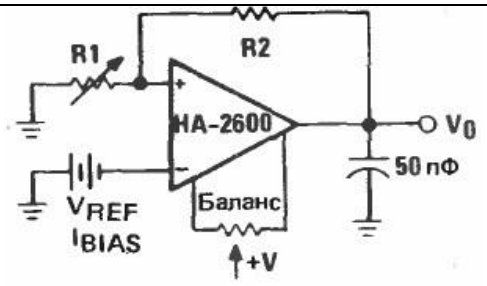
Vazifalar:

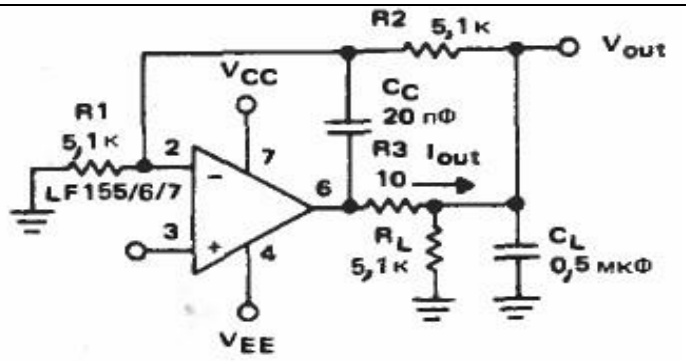
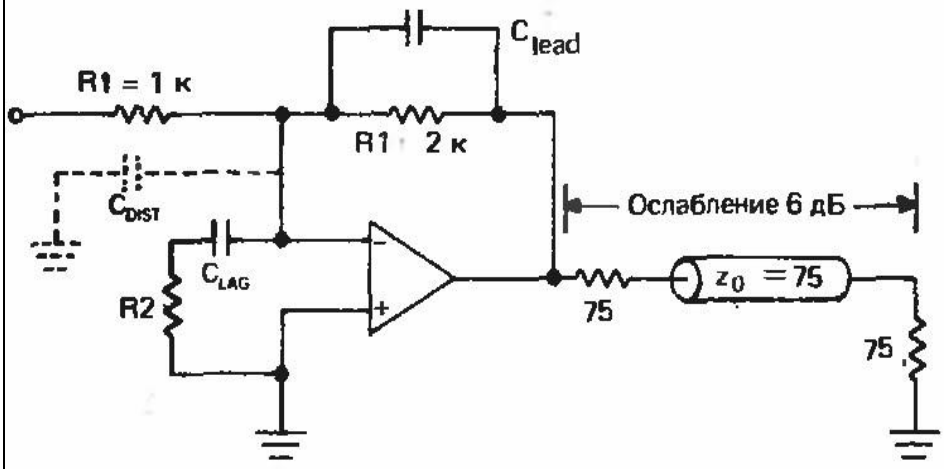
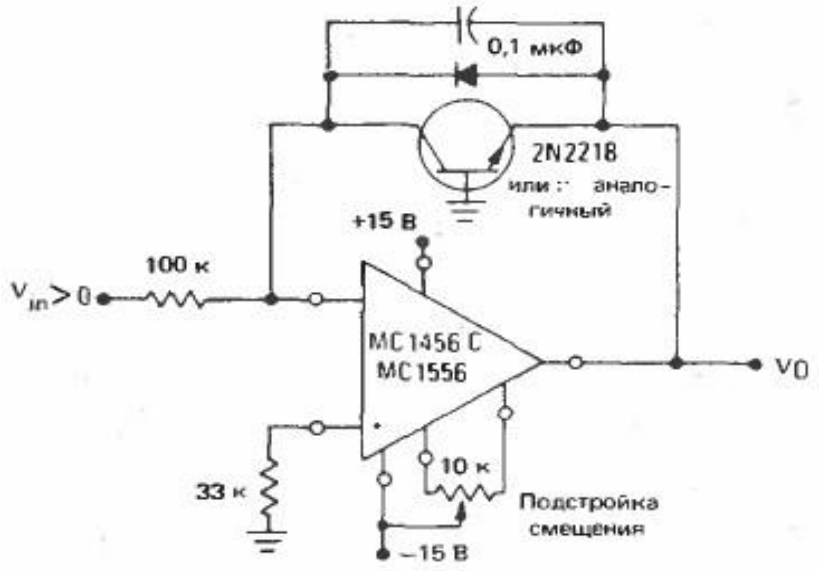
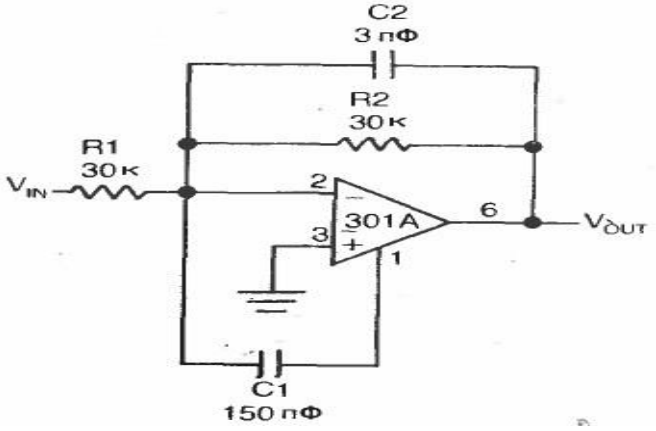
Variant 1	
Variant 2	
Variant 3	

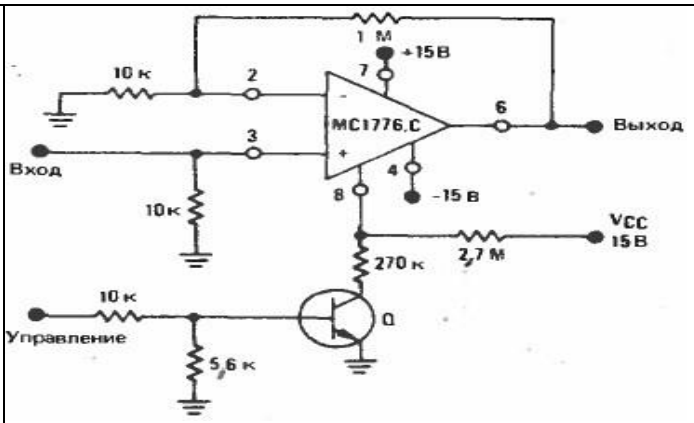
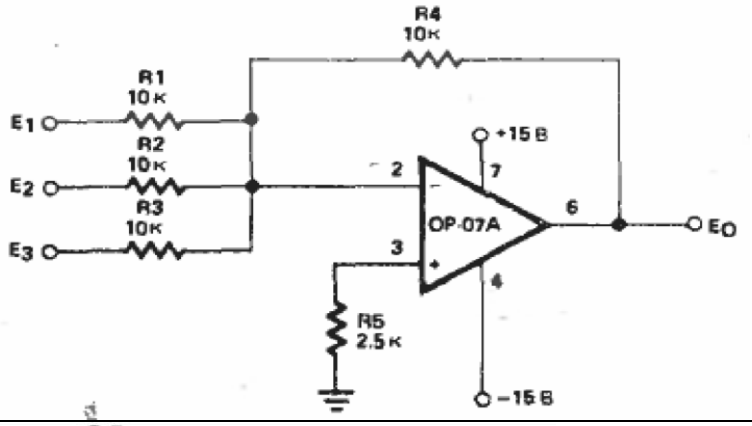
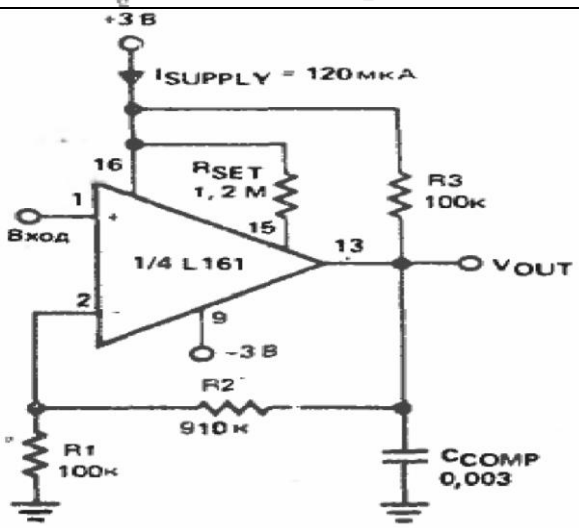
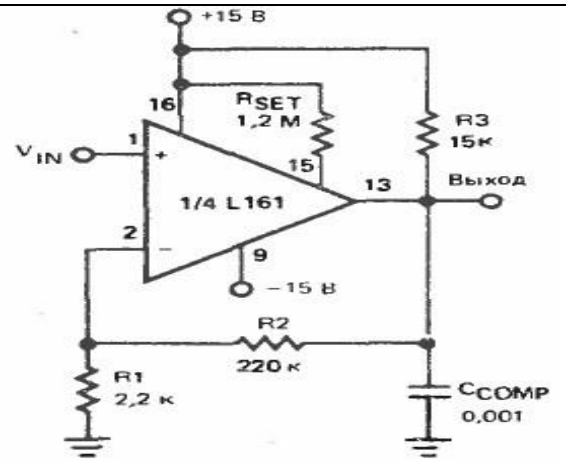
Variant 4	
Variant 5	
Variant 6	
Variant 7	

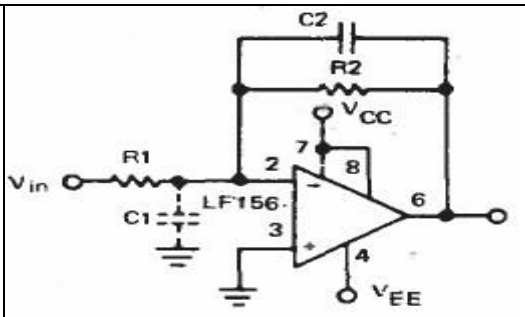
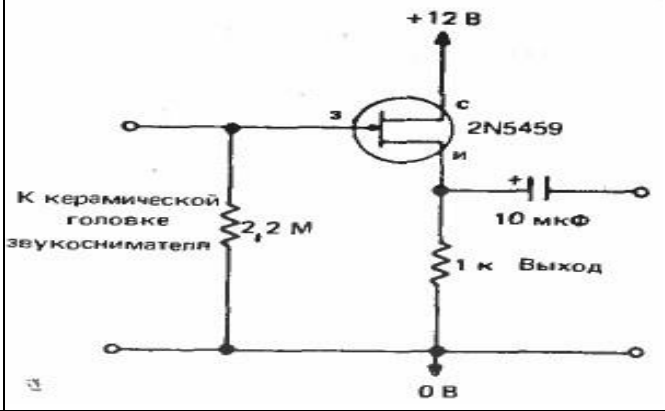
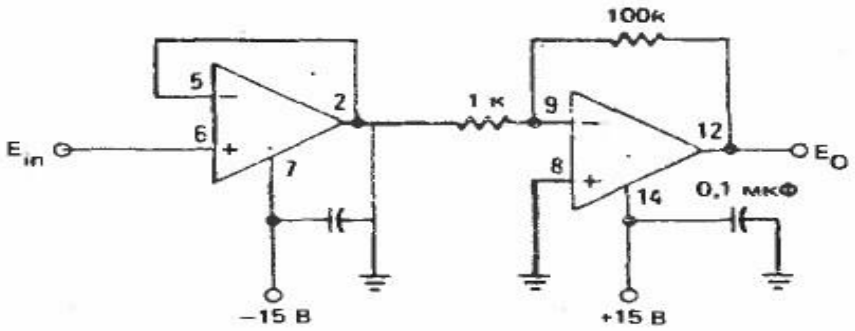
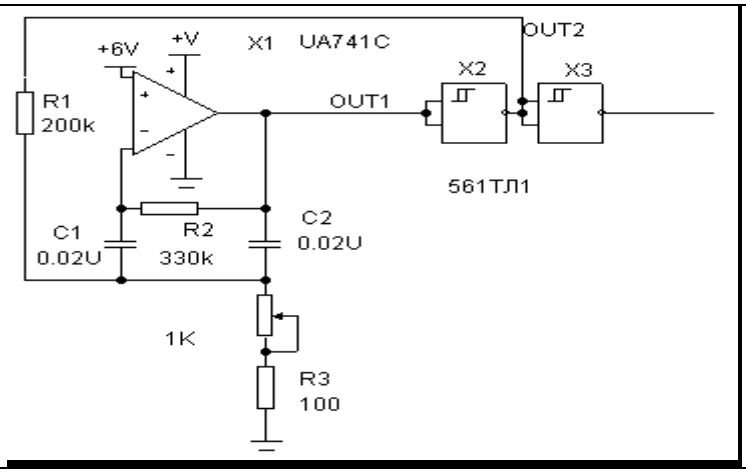
Variant 8	
Variant 9	
Variant 10	
Variant 11	

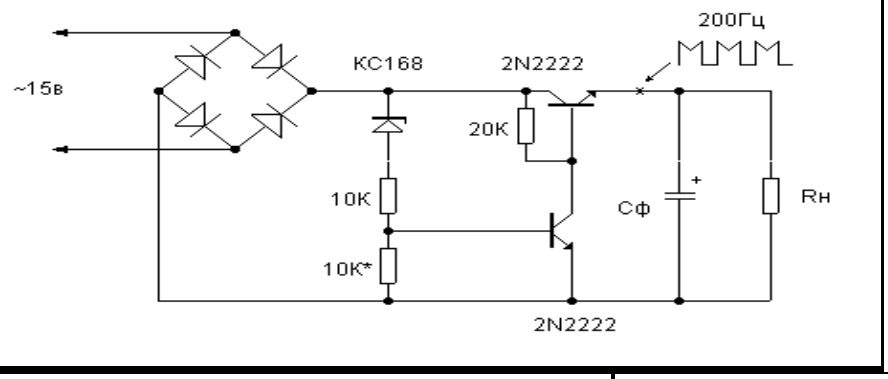
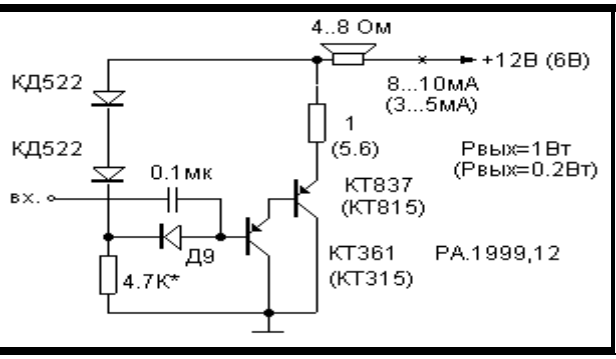
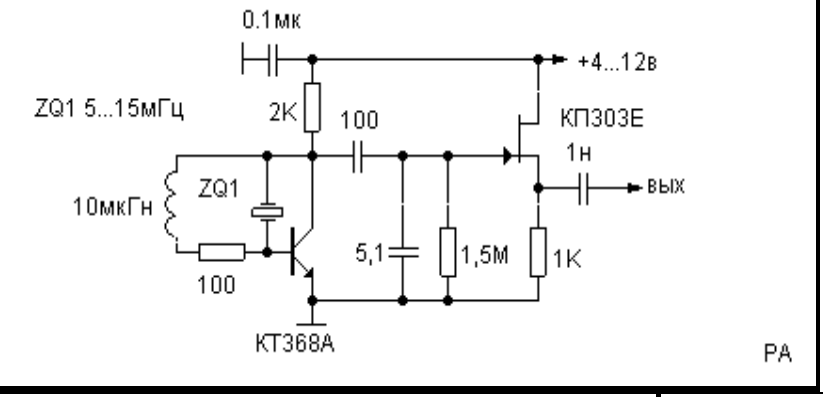
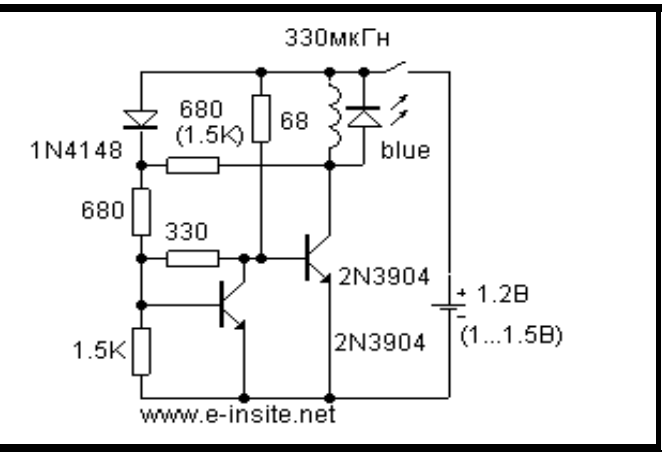
Variant 12	
Variant 13	
Variant 14	
Variant 15	

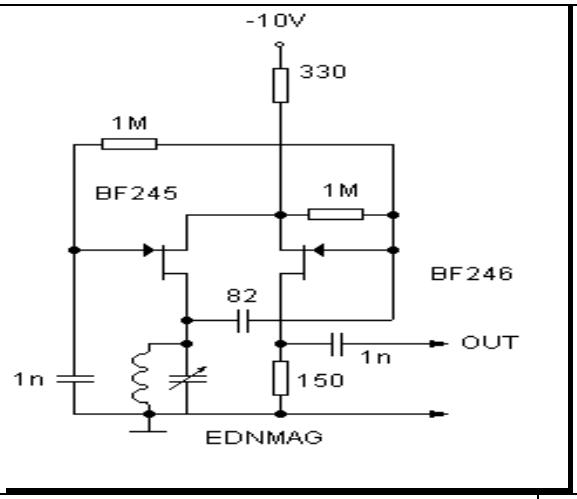
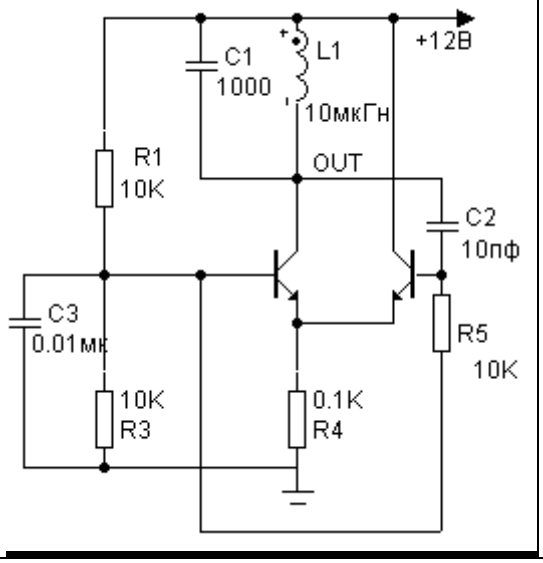
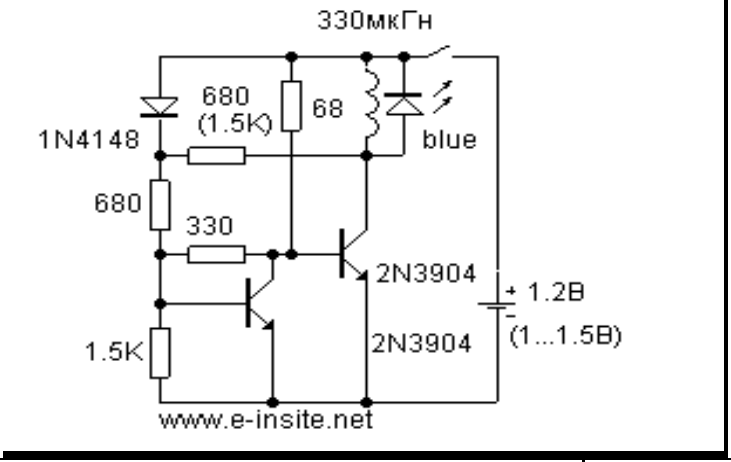
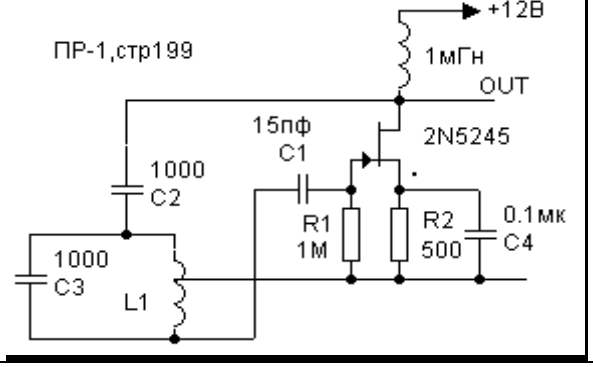
Variant 16	
Variant 17	
Variant 18	
Variant 19	
Variant 20	

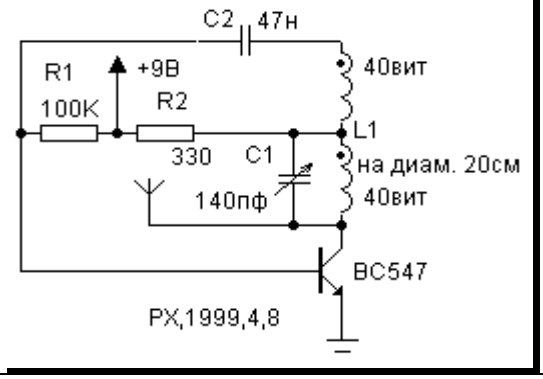
Variant 21	
Variant 22	
Variant 23	
Variant 24	

Variant 25	
Variant 26	
Variant 27	
Variant 28	

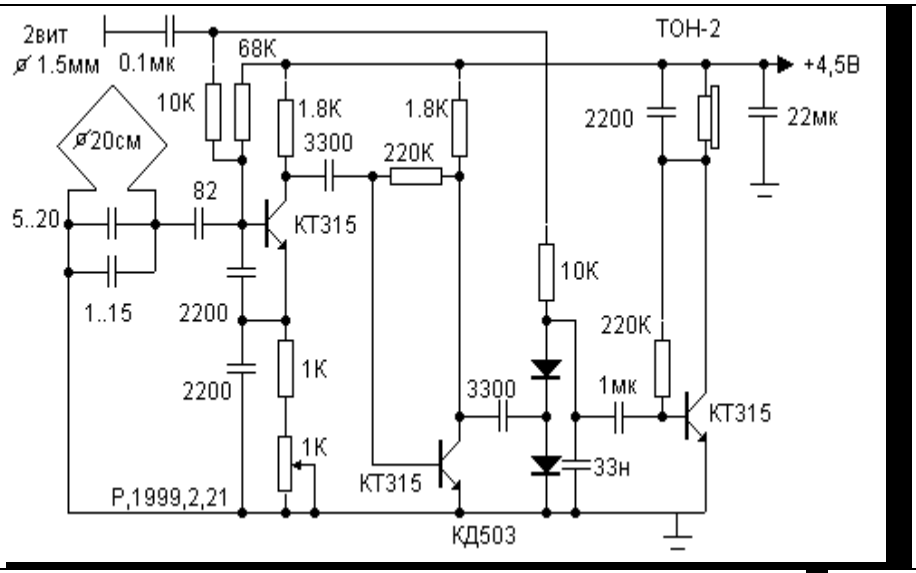
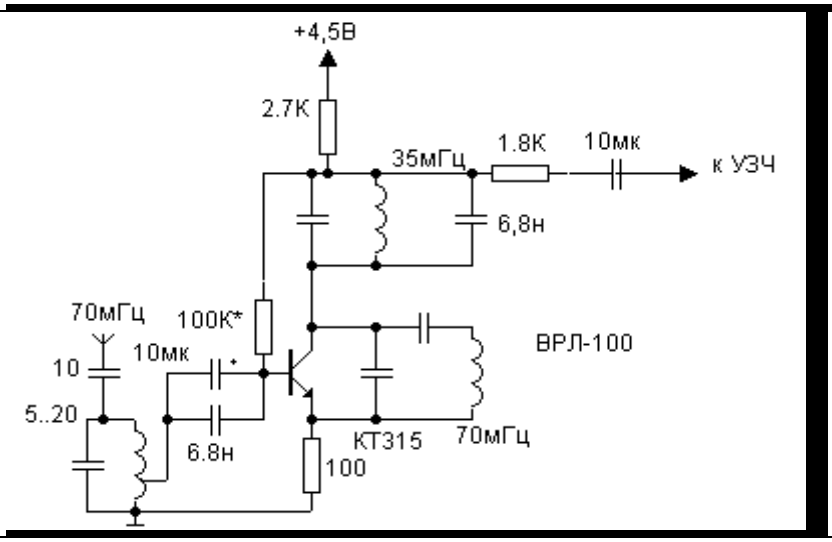
Variant 29	
Variant 30	
Variant 31	
Variant 32	

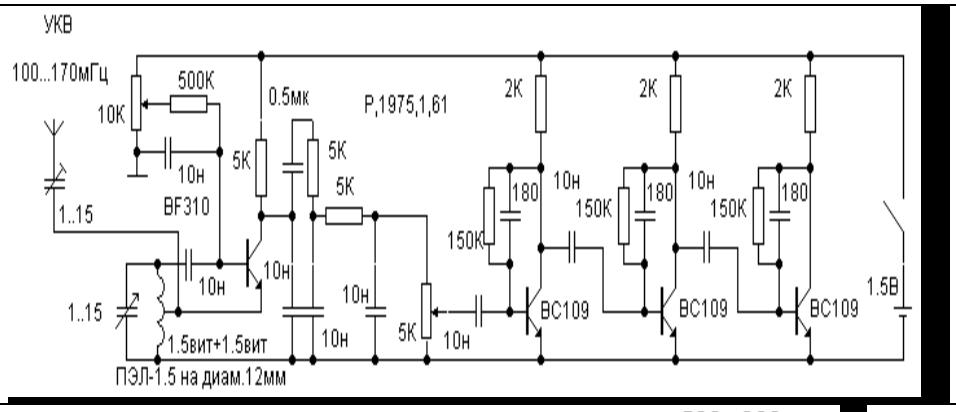
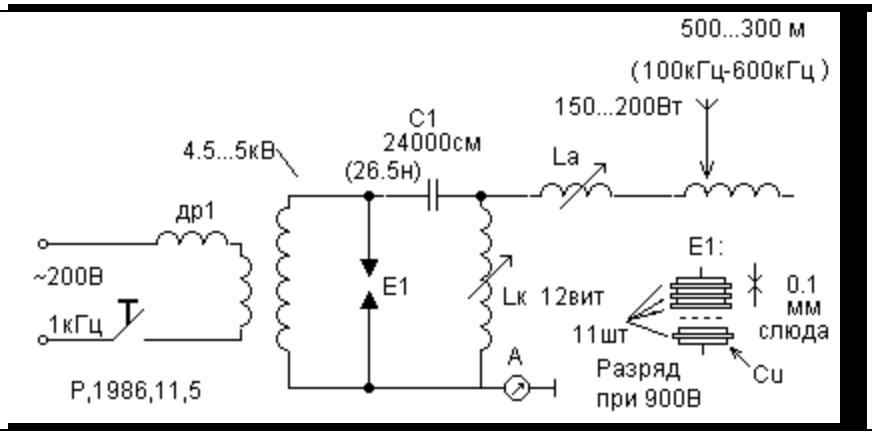
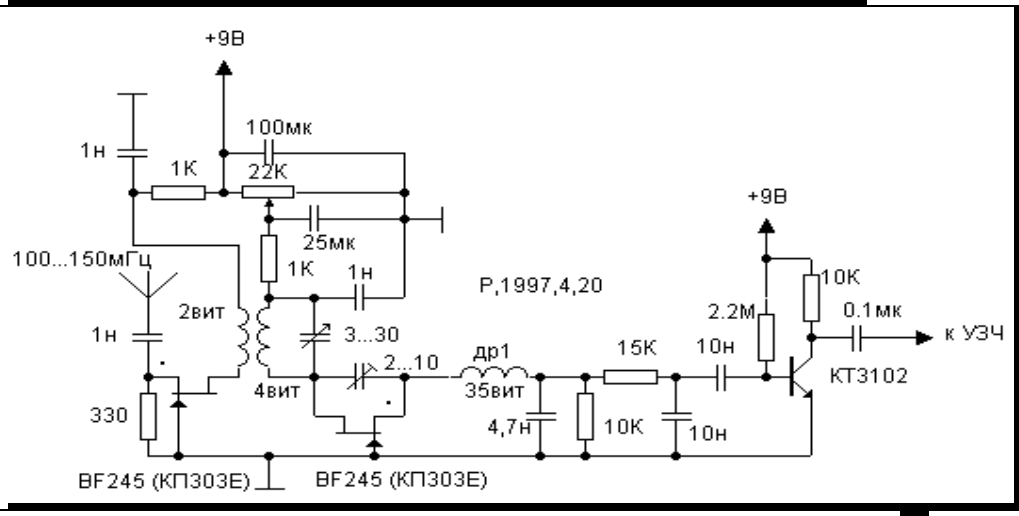
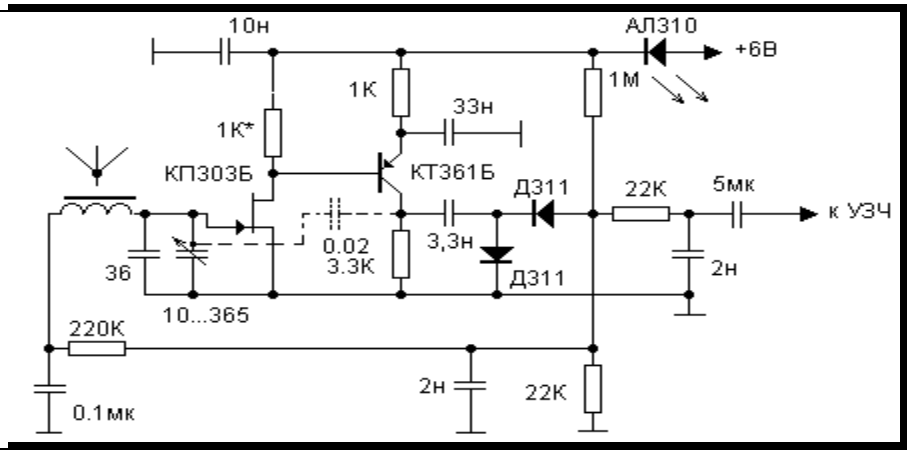
Variant 33		
Variant 34		
Variant 35		
Variant 36		

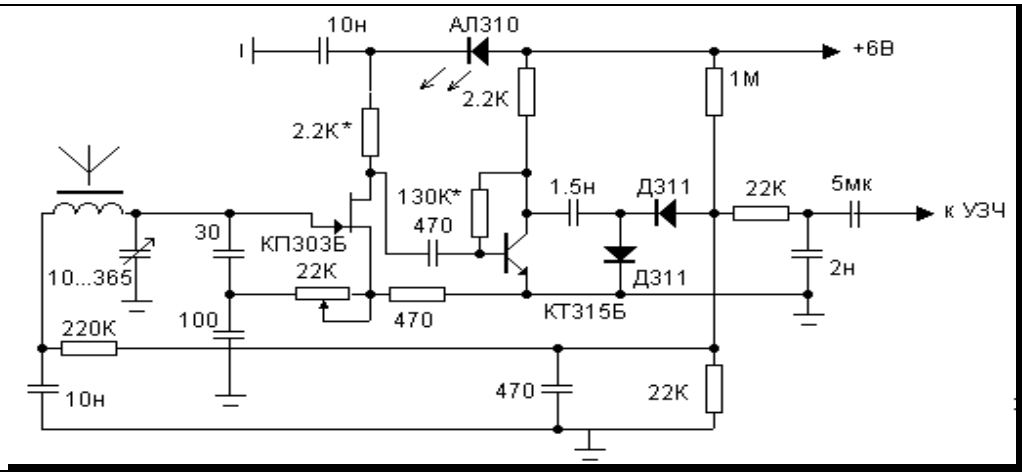
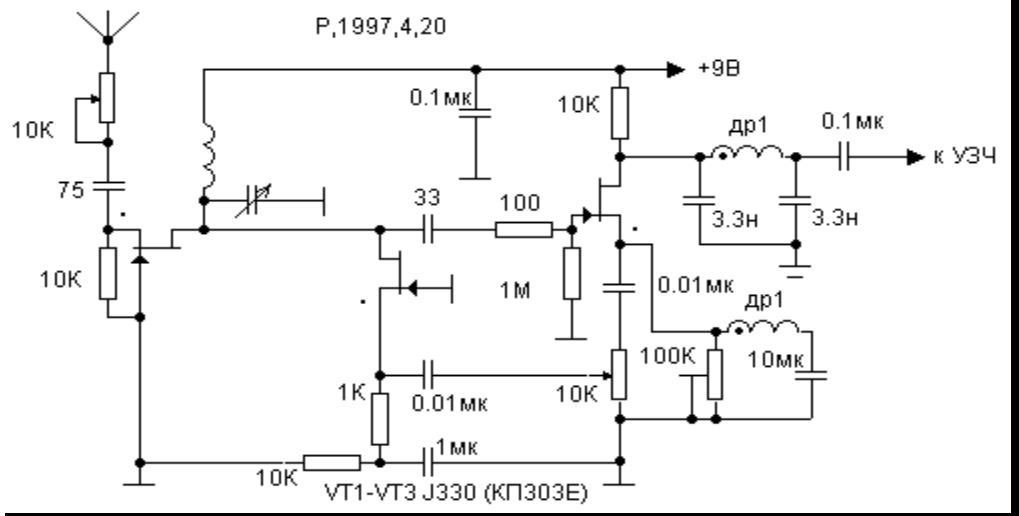
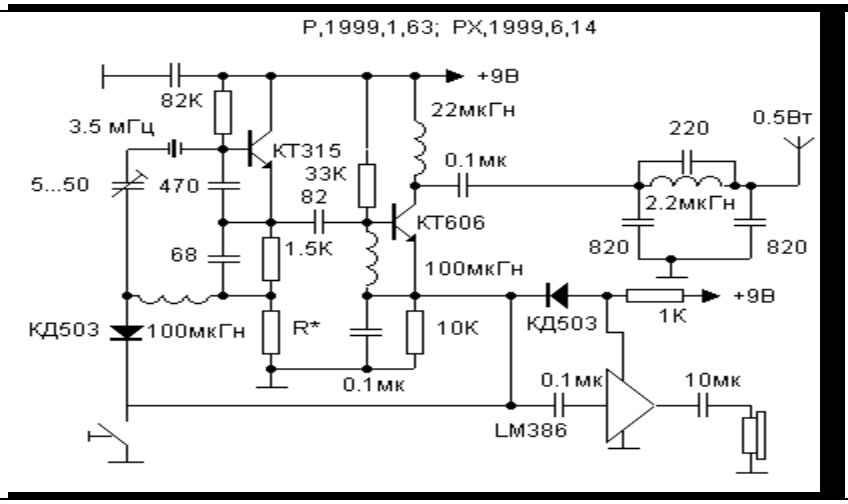
Variant 37		
Variant 38		
Variant 39		
Variant 40		

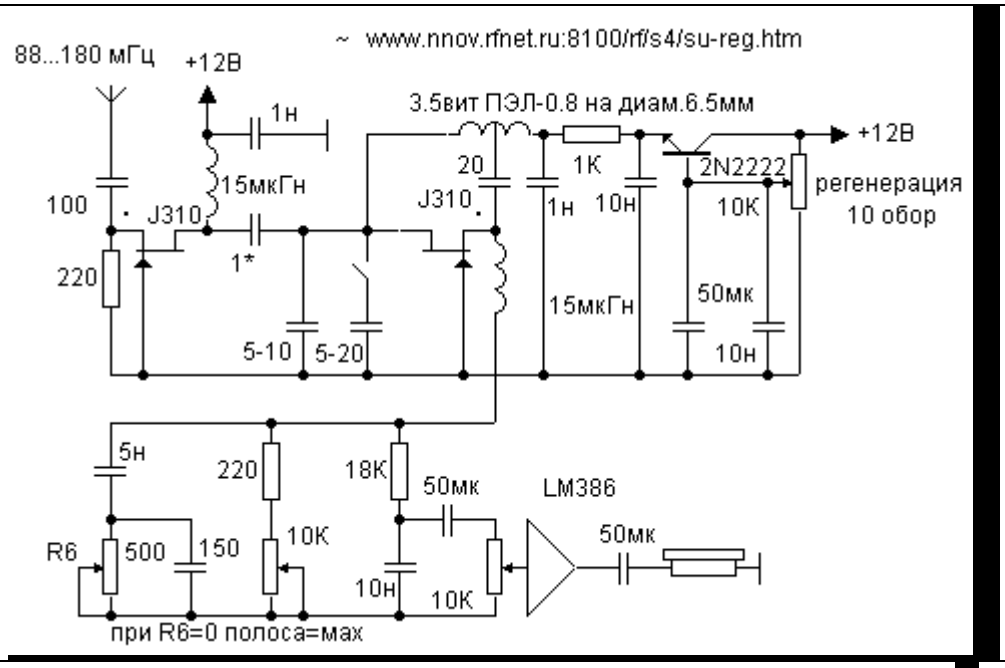
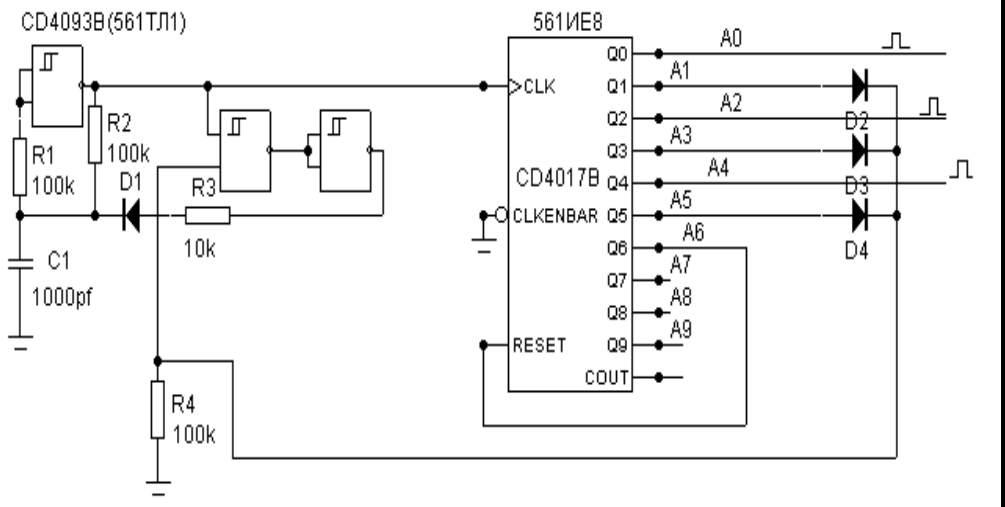
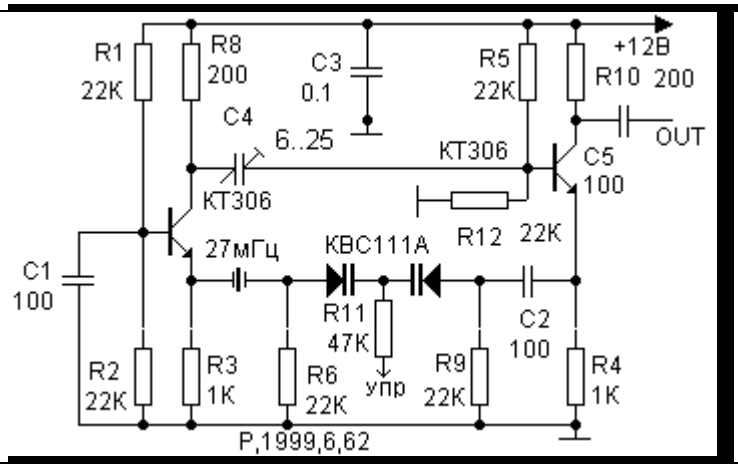
Variant 41		

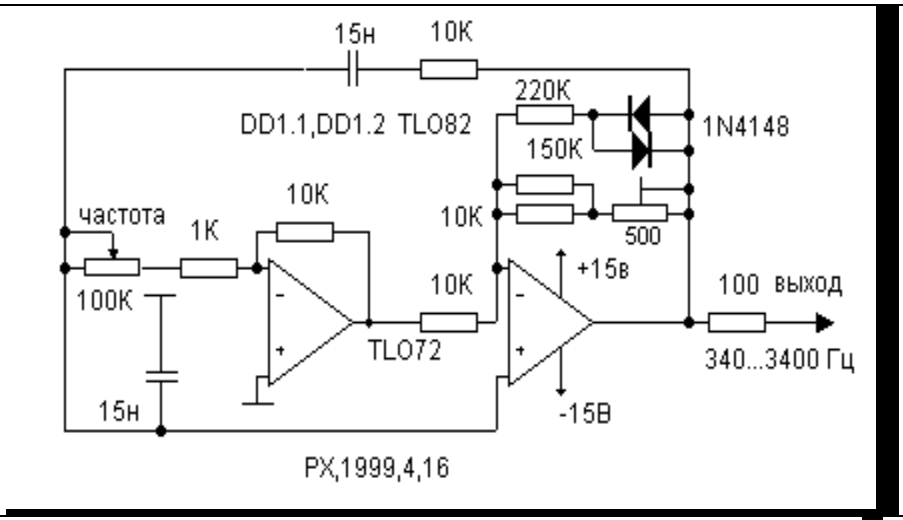
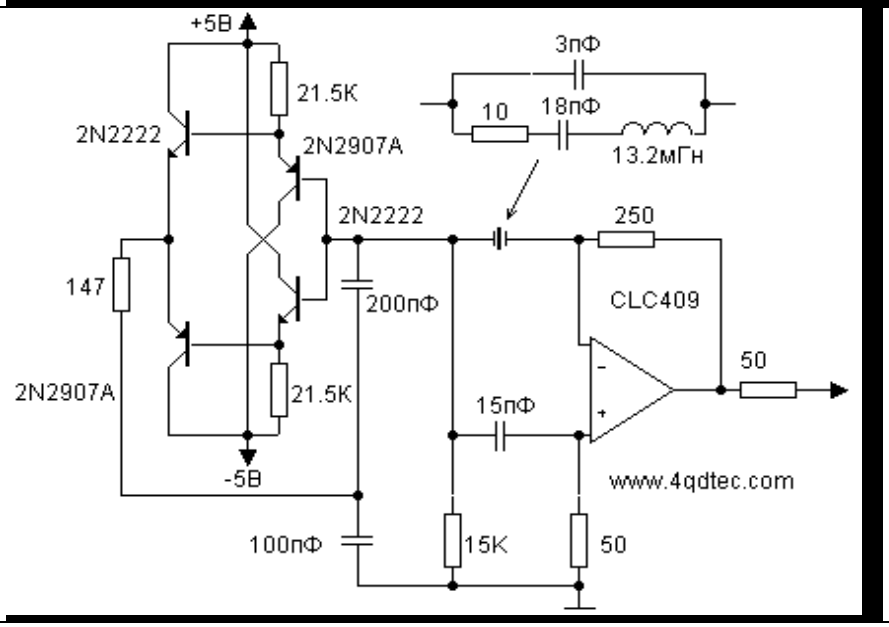
Murakkab sxemalarni tuzish bo’yicha variantlar

Variant 42		
Variant 43		

Variant 44	
Variant 45	
Variant 46	
Variant 47	

Variant 48	
Variant 49	
Variant 50	

Variant 51	
Variant 52	
Variant 53	

Variant 54	
Variant 55	

1- Sxemada ko’rsatilgan variantlar uchun Z-o’zgaruvchining qiymatlari jadvali

Variant	Nomersxemo’	Elemento’sxemo’								
		Z1	Z2	Z3	Z4	Z5	Z6	Z7	Z8	Z9
1.	1	∞	C2	L3	R4	L5	C6	R7	∞	∞
2.	1	∞	L2	C3	L4	R5	C6	R7	∞	∞
3.	1	∞	C2	L3	C4	R5	L6	∞	R8	∞
4.	1	∞	L2	C3	R4	C5	R6	∞	L8	∞
5.	1	L1	L2	R3	∞	∞	R6	C7	∞	∞
6.	1	∞	R2	L_3	C_4	L5	∞	∞	∞	C_9

7. 8. 9. 10. 11. 12.	2 2 2 2 2 2	∞ L1 ∞ ∞ ∞	∞ C2 C2 R2 ∞ C_2	L3 R3 L3 L3 L3 L_3	L4 ∞ R4 L4 C4 L_4	C5 C5 L5 C5 L5 R5	R6 C6 C6 ∞ R6 C_6	∞ R7 ∞ ∞ ∞	C8 ∞ ∞ C8 C8 R_8	∞ ∞ R9 C9 C9 ∞
13. 14. 15. 16. 17. 18.	3 3 3 3 3 3	∞ ∞ ∞ L1 C1 ∞	∞ ∞ ∞ R2 L2 ∞	L3 C3 L3 R3 C3 ∞	R4 L4 C4 ∞ ∞ R_4	L5 R5 R5 ∞ ∞ L_5	C6 C6 C6 ∞ ∞ C_6	0 0 0 L7 L7 L_7	C8 L8 C8 C8 R8 C_8	R9 R9 L9 C9 R9 R_9
19. 20. 21. 22. 23. 24.	4 4 4 4 4 4	L1 C1 ∞ L1 C1 ∞	R2 R2 ∞ R2 L2 ∞	C3 L3 L3 C3 L3 R_3	L4 C4 C4 R4 R4 C_4	∞ ∞ R5 ∞ ∞ L_5	∞ ∞ L6 ∞ ∞ R_6	C7 L7 R7 C7 C7 ∞	R8 R8 C8 L8 R8 ∞	∞ ∞ ∞ ∞ ∞ C_9
25. 26. 27. 28. 29. 30.	5 5 5 5 5 5	L1 C1 ∞ ∞ ∞ ∞	C2 R2 C2 R2 0 C2	R3 L3 L3 L3 R3 L3	∞ ∞ C4 C4 L4 L4	∞ ∞ R5 L5 C5 C5	L6 L6 L6 C6 L6 ∞	C7 C7 R7 R7 C7 ∞	∞ ∞ ∞ ∞ ∞ R8	R9 R9 0 0 R9 ∞

Adabiyotlar:

1 И.Г. Швайко. Информатика. Модул 4.Программирование задач со списками и файлами. Объектно-ориентированное программирование. /И.Г. Швайко, Ю.В. Прокоп, Л.Л. Леоненко, М.В. Северин. − Одеса: ОНАЗ,

2 И.А. Ещенко. Графика в языке Турбо Паскал. / И.А. Ещенко, А.И. Ещенко − Одеса: УДАЗ, 1997. –50 с.

3 С++. Основы программирования. Теория и практика: пидручник [/О.Г. Трофименко, Ю.В. Прокоп, таин.] − Одесса: Феникс, 2010. – 544 с.

4 Седжвик Р. Фундаментальные алгоритмы на С++. / СеджвикР. −М. Диа-Софт., 2001. – 688 с.

5 Топп У. Структуры данных в С++. / Топп У., Форд У.– М.: Бином., 2000. – 816 с.

6 Вирт Н. Алгоритмы иструктуры данных. / Вирт Н. – М.: Мир, 1989. – 896 с.

7 Архангельская. Я. Программирование в С++ Builder 6. 2006. / Архангельская Я., ТагинМ.А. – М.: Бином-Пресс, 2007. – 1184 с.

8 Примеры выполнения лабораторных работ по алгоритмам компьютерной графики: Метод. указания. /Сост.: Хайдаров Г.Г., АлексеевС.Ю. − СПб., СПбГТИ(ТУ), 2005. – 30 с.

9 Шикин А.В. Компьютерная графика. Динамика, реалистические изображения. / Шикин А.В., Боресков А.В. – М.: ДИАЛОГ-МИФИ, 1996. – 288 с.

10 Порев В.Н. Компютерная графика./ Порев В.Н. - СПб.: БХВ- Петербург, 2002. – 432 с.

11 Культин Н. С++ Builder в задачах и примерах ./ Култин Н. - СПб.: БХВ-Петербург, 2005. -328 с.

Mundarija

«DASTURLASh ASOSLARI» kursi laboratoriya ishlari uchun C++**Builder**da grafika kompleks masalalarni bajarish uchun va laboratoriya ishlariga uslubiy ko’rsatmalar.

TATU ilmiy-metodik kengashining 2016 yil
12 maydagi 9(90) sonly majlisida ko’rib
chiqildi va nashr etishga tavsiya etildi.

Mualliflar: dots.M.Yu. Xaydarova, ass. Nazirova M.X.

korrektor : Dadenova G.K..

Format 60x84 1/16 _____ sonly buyurtma, adadi_____