O‘ZBEKISTON RESPUBLIKASI OLIY VA O‘RTA MAXSUS TA’LIM VAZIRLIGI
BUXORO DAVLAT UNIVERSITETI
Fizika –matematika fakulteti

“Amaliy matematika va axborot texnologiyalari” kafedrasi

Karimova Sarvinoz Hojiqurbonovna

TEZ TIBBIY YORDAM XIZMATINI QO`LLAB-QUVATLOVCHI INFORMATSION TIZIM MIJOZ QISMI.

5480100-“Amaliy matematika va informatika” ta’lim yo’nalishi bo’yicha bakalavr darajasini olish uchun

BITIRUV MALAKAVIY ISHI

“Ish ko‘rildi va himoya qilishga ruxsat berildi”
Kafedra mudiri. _____ dots T.B.Boltaev
«_____» _____2014y.

Ilmiy rahbar _____ . Z.Z.Jo`rayev
“_____” _____2014 y.
Taqrizchi _____ U. Narziyev
“_____” _____ 2014 y.

“Himoya qilishga ruxsat berildi”
Fakultet dekani _____ Sh.M.Mirzayev
“_____” _____ 2014 y.

Buxoro-2014

Mundarija.

Kirish.

O‘zbekiston Respublikasi mustaqillik odimlarini dadil qo'yayotgan hozirgi davrda, axborotlashgan jamiyat qurish masalasi mamlakatimiz uchun naqadar katta ahamiyat kasb etayotgani hech kimga sir emas.

Bugun kompyuter texnologiyalari dunyoda jadallik bilan rivojlanayotgan sohalaridan biriga aylanib bormoqda. Kompyuter asta-sekin insonlar hayotiga kirib, inson atrofida bo`layotgan giper axborot almashinuv tizimini osongina boshqarish imkonini beradi.

Har qanday ish yurutuvchi hamda kompyuter mutaxassisini faol ish yuritishida inavatsion texnologiyalarisiz tasavvur qilib bo’lmaydi. Kompyuter texnologiyalari inson hayot tarzida, kichik va katta korxonalarda, kompaniyalarda, o’quv dargohlarida, davlat muassasalarida behad ahamiyat kasb etmoqda.

Davlatimiz tomonidan zamonaviy kompyuter texnologiyalarini fan va meditsinaning barcha sohalariga keng joriy etish, xalqaro axborot tizimiga, shu jumladan, “Internet”ga kirib borishni kengaytirish, yuqori malakali vrachlarni tayyorlash, yagona milliy axborot tizimini tashkil etish masalasi davlat siyosati darajasiga ko’tarildi.

O`zbekiston Respublikasi Mustaqillikka erishganidan so’ng yurtimizda kompyuter texnologiyalari kundan kunga rivojlanib bormoqda.Respublikamiz Prezidenti I.A.Karimov O’zbekiston Respublikasi Oliy Majlisining IX sessiyasida ta’kidlaganidek, “Buyuk ma’naviyatimizni tiklash va yanada yuksaltirish, milliy ta’lim-tarbiya tizimini takomillashtirish, uning milliy zaminini mustahkamlash,sog`lom avlodlarni dunyoga keltirish va tarbiyalash zarurdir”. Xususan bu borada davlatimiz tomonidan ishlab chiqilgan dastur asossida bosqichma bosqich olib borilayotgan tashkiliy ishlar ,Tibbiyot xodimlari tomonidan ko`rsatilayotgan tibbiy xizmatlarni yevropa standartlariga asosan tashkil etish amalgam oshirilmoqda.

Mamlakatimiz o’z mustaqilligiga erishgach, meditsina tizimida tub o`zgarishlar sodir bo`ldi. Jamiyat taraqqiyotining yoshlarga sog`ligiga ularning

hayotida ishlab chiqarish milliy hamda umuminsoniy munosabatlarda zamonaviy qarash imkoniyati kengaydi.

Bitiruv malakaviy ish mavzusining dolzarbligi. Hozirga qadar mamlakatimizda tez tibbiy yordam faoliyatini tashkil etishda yagona baza asosida axborot almashinuvini boshqarish tashkil etilmagan. Bu esa tezkor tibbiy xizmat ko`rsatishni kechikishi, samarali tashkil etilmayotganiga sabab bo`lmoqda.

Tez tibbiy yordam xizmatini qo`llab-quvatlovchi informatsion tizim mijoz qismi dasturi. Tez tibbiy yordam xizmatini samarali tashkil etish uchu zarur bo`lgan ma`lumotlar bu bemorlar haqidagi ma`lumotlar, tez tibbiy xizmat ko`rsatish guruhlar ma`lumotlari, hudud nomlari haqidagi ma`lumotlar bilan tezkor ishlash imkonini beradi. Bemorga yordam ko'rsatilgan jarayoning aniq faktli ma'lumatlarga tayangan holda ish olib borish. Tez tibbiy xizmatini jarayonini tahlil qilish oson va tez amalgam oshirishga erishish. Bundan tashqari iqtisodiy tejamkorlikga erishiladi. Tiz tibbiy xizmati hisobotlarini tez va aniq olish imkoniyati yaratiladi.

Bitiruv malakaviy ishining tadqiqot obyekti. Aholi puktlarida yashovcho aholining tibbiy xizmatga muhtoj bo`lgan bemorlariga o`z vaqtida tibbiy xizmat ko`rsatish.Tibbiyot xodimlarini ishini samarali tashkil etish. Bemor kasallik tarixi ma`lumotlarini elektron saqlash.

Malakaviy bitiruv ishining maqsad va vazifalari. Ushbu ishning asosiy maqsadi tez tibbiy xizmatini qo`llab quvatlovchi tizim mijoz qismini yaratishdan iborat. Bu tez tibbiy xizmatida ishlaydigan xodimlarni, bemorlarni, ishchi guruhlarni, tez tibbiy yordan punktlariga keladigan dorilarni ro'yxatga olish va ularning bemorga xavsizligi, yaroqliligi, summasi, qaysi tashkilotda ishlab chiqarilganligi haqidagi ma`lumotlarni o`zida saqlaydi. Ushbu ishning mijoz qismini Delphi muhitida yaratilgan bo`lib yagona baza asosida ishlaydi.

Bitiruv malakaviy ishidan olingan natijalar asosida tez tibbiy xizmatini ishini yanada yaxshilash kabi inavatsion imkoniyatlarni yaratish imkonini beradi.Ushbu dasturiy vositasini yaratishdan maqsad tez tibbiy xizmat ishlashini qo`llab-

quvatlovchi ma’lumotlar bazasi ustida mijoz uchun informatsion dastur yaratishdan iborat.

Malakaviy bitiruv ishning tadqiqot usuli va uslubiyoti. Ushbu malakaviy bitiruv ishining usuli SQL serverda yaratilgan tez tibbiy ma`lumotlar ustida Delphe da mizoz qismini yaratishdan iborat bo’lib, bunday dasto’rni yaratish uchun Delphe muhitida ishlashni chuqur bilishimiz, baza bilan bog’lanish va SQL serverda yaratilgan jadvallarni bog’lanishlarni hosil qilishdan iborat.

Olingan asosiy natijalar: tez tibbiy xizmatini qo`llab-quvatlovchi informatsion tizim mijoz uchun dastur qism yaratilganligi.

Bitiruv malakaviy ishidan olingan natijalarning yangiligi va amaliy ahamiyati: Yangi kompyuter va dasturiy texnalogiyalar asosida tez tibbiy xizmatini ishini aniqligi va uni qo`llab quvatlovchi informatsion dasto’rni yaratish, buning natijasida xatoliklarni oldini olishga erishildi.

Tadbiq etish darajasi va iqtisodiy samaradorligi qo`llash sohasi. Ushbu bitiruv malakaviy ishi davomida tez tibbiy xizmatida hamma jarayonlar kompyuterlashtirilgani natijasida ortiqcha sarf xarajatlar kamayishiga va opertorlar va dispecherlarning vaqti tejalishiga erishildi.

Malakaviy bitiruv ishinig tuzilishi: : Malakaviy bitiruv ishi kirish, ikki bob, xulosa, foydalanilgan adabiyotlar ro’yxati iborat.

I-BOB. Delphi qobiq dasturi.

1.1. Ob’yektga mo’ljallangan dasturlash tillari.

Delphi dasturlash muhiti Borland kompaniyasi tomonidan ishlab chiqarilgan bo’lib, u Borland Pascal tilining keyingi versiyasi Borland Object Pascal tili asosida qurilgan. (1.1-rasm).

1.1-rasm. Delphidasturlash tili.

Ob’yektga mo’ljallangan dasturlash tilllarida dastur tuzish ancha oson va ishchi vaqt tejaladi. Dasturchi ob’yektlarni qurib o’tirmaydida tayyor ob’yektlarni qo’yib olaveradi va dastur asosiy qismiga bosh qotiradi xolos.

O’zbekistonga BorlandDelphi1998 yillardan keyin kirib kelgan. Yildan yilga o’tib yangi versiyalari kirib kelgan. Xozir komppyuter va internet texnologiyasi rivojlangan O’zbekistonda ham xar qanday dasto’rning oxirgi versiyasini topish mumkin. Delphida dastur tuzish uchun 80% vaqtingiz ketadi. Windows oynaga

(Delphi da “forma” deb ataladi) kerakli komponentlarni qo’yishingiz, bemalol oyna bo’ylab surishingiz mumkin va ularning xususiyatlarini maxsus (ObjectIsrector) oyna yordamida o’zgartirishimiz mumkin. U yordamida komponentlarga xodisalarni (tugmani bosilishi, sichqoncha xolati va x.k.) boғlashimiz mumkin. Delphi kuchli xatolarni bartaraf qilish (Debuger) sistemasiga va qulay yordamchi sistemasiga egadir.
Siz MicrosoftIDL yordamisiz ActiveX komponentlar tuzishingiz, Amaliy HTML, XMLyoki ASP tillarni bilmagan xolda ҳam web-serverlar imkoniyatini kengaytirishingiz mumkin. Keng qo’lanilayotgan SOM va CORBA asosidagi dasturlarni yaratish, Internet va Intranet dasturlar, BDE(BorlandDataBaseEngine), ODBC – drayver, MicrosoftADO ma’lumotlar bazasiga murojaat qilish imkoniyatiga ega bo’lamiz. Delphi 3 dan boshlab yangi ko’ptarmoqli texnologiya qo’llanila boshlangan.

Delphi tili to’liq ob’yektli-boshqarish tiliga mos keladi. Klasslar asosida nasllar yaratish mumkin. S++ dagi overload (qayta yuklash) va excertions (isklyuchitelnых situastiya) metodlarini qo’llab quvvatlaydi. WideChar va AnsiChar formatidagi uzun qatorlarni qo’llash imkoniyatiga ega.

Delphi ning yana bir xususiyati u o’zini o’zi rivojlantiradi. Siz o’zingizni komponetlarinigizni yaratishingiz, OCX – komponentlarni qo’llashingiz, loyixalar uchun shablonlar yaratishingiz mumkin.Delphi ning integrallashgan muhiti (IDE) yordamida foydalanuvchi o’zining dasturini tashqi dasturlar bilan boғlash imkoniyatiga ega.

Delphi da OrenGL va DirectX texnologiyalaridan foydalangan xolda 3 o’lchovli loyixalarni yaratishimiz mumkin.

Yuqorida aytib o’tdikki, Delphi bu – Pascal tilini kompilyatori. Delphi kompilyatorlari Pascal kompilyatorini evolyustiyasi natijasida rivojlanib kelgan, yani bu til o’zini-o’zi yaratib kelgan.

Endi Delphi versiyalaridan Delphi7 ning qisqacha tavsiflari bilan tanishib chiqamiz.

Delphi7 2002 yilda ishlab chiqarilgan ushbu versiya o’zida ko’plab o’zgarishlarni mujassamlashtirgan.

Bu versiyaning asosiy o’zgarishlari quyidagicha:

- bu versiya Windows 2k va XP OS larni qo’llab quvvatlaydi va o’zida XP.MANIFEST (vizual stillar) ni ta’minlovchi resusrlari mavjud;

- 6 versiya komponentlari va resusrlari bilan mos tushadi va fayllarini qshllab quvvatlaydi;

- yangi Nevrona firmasi tomonidan ishlab chiqarilgan RaveReports 5.0 xisobotlar generatori qo’shildi, u uch qismdan iborat:

- xisobot generatori yadrosi xisobotlarni tayyorlashni boshqaradi, oldindan qo’rishni va printerdan bosmaga chiqarishni ta’minlaydi;

- RaveReports ning vizual muhiti Delphi ga utilita sifatida kiritilgan bo’lib, turli xildagi xisobotlarni tayyorlash uchun xizmat qiladi. U xisobot listlari qo’shishni, unga matn hamda grafik tasvirlar qo’yish, xisobotga ma’lumotlar bazalarini boғlashni boshqaradi. Xisobot fayllari .rav ko’rinishida saqlanadi;

- RaveReports komponentlar Delphi komponentlar palitrasining Rave sahifasida joylashgan. Ular dastur yordamida xisobotlar tayyorlashni ta’minlaydi.

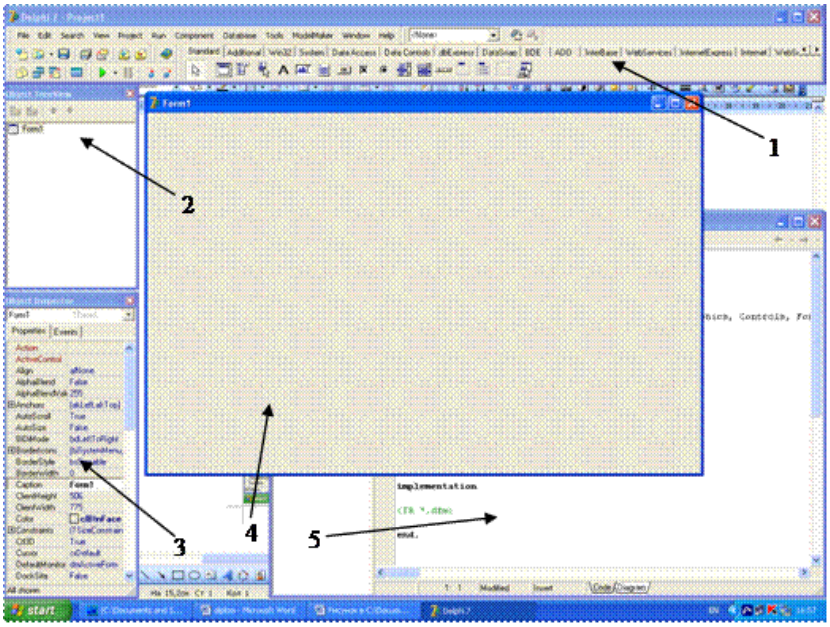
- Xisobotlar generatori \Delphi7\Rave5 papkaga o’rnatiladi, ammo dastur kodi keltirilmagan (*.pas).

- internet hamda intranet serverlar uchun mo’ljallangan IntraWeb komponentlar paketi qo’shildi. U yordamida Web sahifa dasturlari osongina ObjectPascal ko’rinishida tuzilaveradi. Kerakli komponentlar IntraWeb sahifasida joylashgan.

Delphi yordamida tuzish uchun 2-3 soat ketadigan oddiy loyixalar, o’n va yuzdan ortiq foydalanuvchilar foydalanadigan katta loyixalarni ham yaratish mumkin. Bularni yaratish uchun juda ozvaqt va oz kuch sarflanadi.

Delphi juda qulay interfeysga ega. Uni ishga tushirganimizda 1.3-rasm ko’rinishidagi oynalar hosil bo’ladi.

Delphi quyidagi asosiy oynalarga bo’linadi: 1-asosiy boshqarish oynasi, 2-ob’yektlar joylashish strukturasi (ObjectTreeView), 3-ob’yekt xususiyatlarini o’zgartirish oynasi (ObjectInrector), 4-dastur ishchi oynasi (forma – yaratilayotgan dastur oynasi), 5-dastur kodini kiritish oynasi.



1.3 - chizma. Delphidasturlash muhitining umumiy ko’rinishi.

Asosiy boshqarish oynasida (1) dasto’rni va interfeysni boshqarish uchun asosiy menyular, boshqarish tugmalari va komponetlar paneli joylashgan. Komponetlar paneli sahifalar ko’rinishida bo’lib, xar bir komponent mavzusiga mos bo’limda joylashgan (1.3-rasm.).

ObjectTreeView oynasida (2) foydalanilgan komponetlarni dasturda joylashish tartibini ko’rish mumkin.

ObjectInsrector oynasida (3) joriy ob’yektni xusisiyatlari va xodisalarini ko’rish hamda o’zgantirishimiz mumkin.

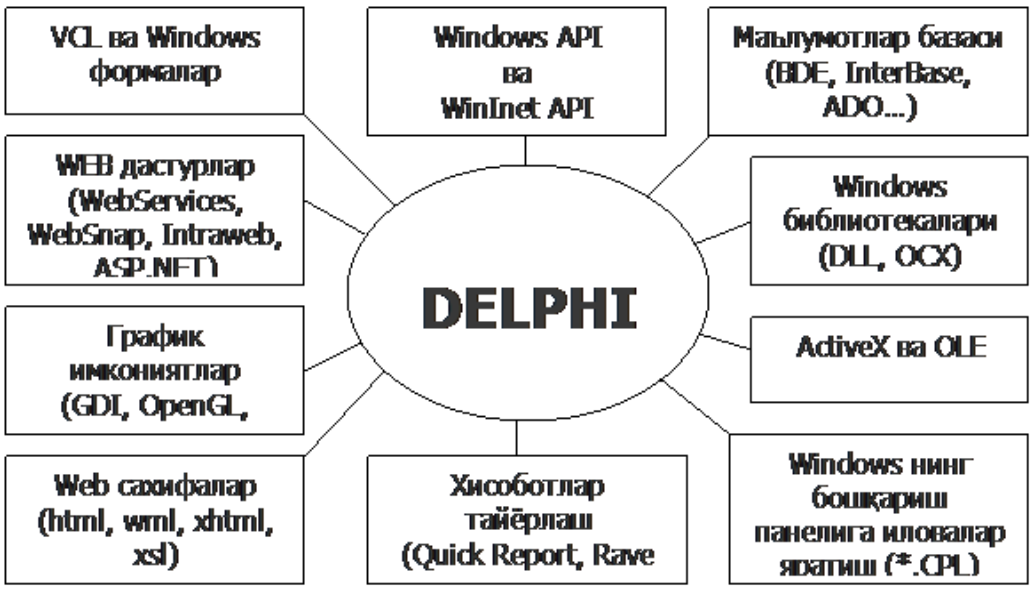
Yaratilayotgan dastur elementlari dastur formasi(4) ga joylashtiriladi. Foydalanuvchiga qulay bo’lish maqsadida forma kataklarga ajratilgan.

Dastur kodini dasturlash oynasi(5)ga kiritiladi. Bu oynani qulayliklari ko’p, masalan egaki xar bir so’zni turiga mos holda ranga ajratadi. Kursor turgan joyda [CTRL]+[SPACE] tugmalarini bosilsa yoki ob’yekt nomidan so’ng “.” Belgisi

bosilsa ishlatish mumkin bo’lgan prostedura va funkstiyalar ro’yxati paydo bo’ladi. Kerakli so’z tanlanib [ENTER] tugmasi bosilsa kursor turgan joyga ushbu so’zni avtomatik yozib beradi. Yana bir qulaylik prostedura yoki funkstiya nomini yozib “(“ belgisi yozilsa kerakli parametrlar xaqidagi ma’lumot paydo bo’ladi.

Tools menyusida interfeysni sozlash bo’limlari va qo’shimcha yordamchi dasturlar joylashgan bo’lib, bajarilayotgan ishni osonlashtiradi.

Delphi ning qo’llanilish soxasi universal bo’lib, operastion sistema resurslaridan foydalangan holda har qanday loyihani yaratish imkoniyatiga ega.Yuqorida Delphi7 ning versiyasini ko’rib chiqishimiz davomida shunga amin bo’ldikki versiya yangilangan sari uning qo’llanilish sohasi ham ortib borgan. 1.4–rasmda Delphi da yaratilishi hamda qo’llash imkoniyati mavjud bo’lgan loyixa turlari keltirilgan.



1.4- chizma. Delphi yaratilishi mumkin bo’lgan loyihalar keltirilgan.

Windows muhitida dastur yaratilishidan oldin siz yangi loyiha yaratishingiz yoki mavjud bulgan loyihangizni ochishingiz kerak. Yangi loyiha yaratishdan oldin ochilgan loyiha va uning formalarini yopishingiz shart. Loyiha va uning formalarini yopish uchun asosiy menyudan (File\Close All) buyrug’ini tanlashingiz kerak. Yangi loyiha yaratish uchun FileYNew Application (File\New\Application)

buyrug’ini berishingiz kerak. Loyihangizni xotiraga saqlashingiz uchun (File\Save_All yoki Shift+Ctrl+S) buyrug’ini tanlang. Saqlashda muloqot oyna chiqganda ikkita faylni saqlashingiz kerak. Iloji boricha har bir loyihangizni alohida aloxida papkalarda saqlang. Loyiha faylini (Project.dpr) uning kengaytmasi *.DPR (Delphi Project) bo’ladi va *.dfm (Delphi Form) Forma faylini nomini nima berishingizdan qat’iy nazar shunaqa nomli *.pas faylini ham saqlaydi, bu fayl shu formaning dastur matni fayli (modul). Faylni saqlagandan keyin dasturingizni ishga tushirsangiz loyihangiz saqlanayotgan papka ichida mustaqil hech qanday Delphi ga bog’liqmas dastur fayli yaratiladi va loyiha nomi *.exe kengaytmali fayl paydo bo’ladi. Undan tashqari shu papka ichida (*.~pas, *.~dfm) kengaytmali fayllar ham paydo bo’lishi mumkin. Bu fayllar saqlashdan oldingi fayllar. (*.ddp, *.dcu, *.dof, *.res, *.cfg) kengaytmali fayllar ham paydo bo’ladi.

Delphi ixtiyoriy loyihasi kamida oltita fayl yaratadi. Ulardan uchtasi loyiha tomonidan yaratiladi va dasturchilar ularning tarkibini ko’pincha o’zgartirmaydilar. Ular quyidagilar:

- Loyihaning bosh fayli, dastlab Projectl.dpr deb nomlanadi.
- Dasto’rning dastlabki moduli /unit/, u ishning boshida avtomatik ravishda yaratiladi. Fayl Unitl.pas deb nomlanadi va uning nomini ixtiyoriy nom bilan almashtirish mumkin. Masalan, Main.pas.
- Bosh forma fayli Unitl.dfm deb nomlanadi, u bosh formaning tashqi ko’rinishi haqidagi ma’lumotlarni saqdaydi.
- Projectl.res loyiha uchun belgini saqlaydi.
- Projectl.opt fayli matnli fayl hisoblanib, siz kompilyatorda kullagan o’rnatmalaringizni saqlaydi.
- Projectl.dsk fayli ishchi soha holati haqidagi ma’lumotlarni saqdaydi.

Albatta loyiha nomini o’zgartirilganda RES, OPT va DSK kengaytmali fayllarning nomlari ham o’zgaradi. Dastur kompilyatsiyasidan keyin quyidagi kengaytmali fayllar hosil bo’ladi:

DCU - tayyor modullar.

EXE - ishchi fayl.

DSM - dasto’rning muhitda ishga tushishi uchun xizmat qiluvchi fayl, uning hajmi juda katta va ish yakunlangach uni o’chirish tavsiya qilinadi.

~PA, ~DP - backoup. - muharrirning vaqtinchalik fayllari. Dastur matni faylining tuzilishi Yangi loyiha yaratganingizdan keyin ekranda bo’sh forma hosil bo’ladi - bu esa dasturingizni asosiy formasi hisoblanadi. Dastur matni muhariri oynasida Unit1.pas fayli ochiladi. Modulingiz ichida formangizda nima ishlar bajarilishi saqlanadi. Formangiz Form1 deb nomlanadi. Modulingiz ichida quyidagi dastur matni bo’ladi:

unit Unit 1; interface uses Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs; type TForm1 = class(TForm)

private

{ Private declarations }

Public

{ Public declarations }

end;

var

Form1: TForm1;

implementation {$R *.dfm}

end.

unit Unit1; // Modulingizni nomi Unit1 interface // Modulingizni interfeys qismi (bu erda boshqa modullardan chaqirilishi mumkin bo’lgan qismi). Uses Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs; Modulingizni interfeys qismida standart modullarning bog’lanishi (chaqirilishi) bajariladi.

Undan so’ng formangiz tipi e’lon qilinadi. Loyihangiz yaratilishi vaqtida, Delphi ning sinflari asosida yangi tip e’lon qilinadi va ko’rsatilgan sinfni barcha imkoniyatlarini merosga oladi.

Type

TForm1 = class(TForm)

Private

{ Private declarations }

Public

{ Public declarations }

end;

Formaning barcha modullardan chaqirilishi mumkin bo’lgan qismi public da e’lon qilinadi private da esa chaqirilishi mumkin bo’lmagan qismi elon qilinadi.

var Form1: TForm1; // Bundan keyin TForm1 tipiga tegishli bitta o’zgaruvchi e’lon qilinadi.

Delphi ning barcha komponentalari, vizual komponentalar bibliotekasida saqlanadi Visual Component Library (VCL). Har qanday komponent hatto forma ham, o’ziga mos komponenta nomi oldiga T harfi qo’shiladi. Masalan: Forma, TForm sinfini barcha imkoniyatlarini merosga oladi. Nishon (Metka, Label) komponenti - TLabel sinfini va hokazo.

implementation // Modulingizni asosiy dastur qismi va bu erda barcha modullardan chaqirilishi mumkin bo’lmagan qismi.

Bundan so’ng bitta kompilyator direktivasi mavjud {$R *.dfm}.bunda mos ravishda bu modul bilan ishlaydigan forma ko’rsatilgan. Forma va formada joylashgan barcha komponentalar, Delphi da *.dfm kengaytmali faylda saqlanadi va bu fayllar matn fayli ko’rinishida bo’ladi.

Loyiha matni faylining tuzilishi.

Loyiha fayli, ayrim qo’shimcha imkoniyatlari ham mavjud bo’lgan Pascal tilidagi matni ko’rinishidagi fayldir. Dasturingizni asosiy qismi shu faylda joylashgan. Dasturingizni asosiy ishlarini (vazifalarini) Delphi da modullarga bajariladi, asosiy faylda (loyiha faylida) emas. Loyiha faylini matn muhariri oynasida ochish uchun Project\View_source buyrug’ini bering. Yangi loyiha faylini matni quyidagicha:

program Project1;

uses Forms,Unit1 in 'Unit1.pas' {Form1}; Unit2 in 'Unit2.pas' {Form2};

{$R *.res}

Begin

Application.Initialize

Application.CreateForm(TForm1, Form1);

Application.CreateForm(TForm2, Form2);

Application.Run;

end.

program Project1; // Programma nomi yoki loyihani nomi

uses Forms // Loyihada standart modullarning bog'lanishi (chaqirilishi) bajariladi.

Unitl in 'Unitl.pas' {Forml}; // Forml ning modulini bog’lanishi (chaqirilishi) bajariladi.

Unit2 in 'Unit2.pas' {Form2}; // Form2 ning modulini bog’lanishi (chaqirilishi) bajariladi.

Bundan so’ng bitta kompilyator direktivasi mavjud {$R *.res}. Unda loyihangizdagi resurslar fayli ko’rsatiladi. (*.res kengaytmali fayl).

begin // Loyihangizni asosiy qismi bo’lib bu erda dasturni bajarilish ketma-ketligi beriladi.

Application.Initialize;//JloftHhara3HH (dasturingizni) yaratilishini tashkil etadi

Application.CreateForm(TForml, Forml); // Loyihangizda Forma ni yaratilishi beriladi

Application. CreateForm(TForm2, Form2); // Loyihangizda Form2 ni yaratilishi beriladi va hokazo. Qavs ichida formani nomi, uning tipi (sinf tipi) beriladi. Loyihada bir nechta forma yaratish mumkin lekin birinchi bo’lib yaratilgan forma asosiy forma hisoblanadi va u yopilganda loyihangiz (dasturingiz) yopiladi.

Application.Run; // Loyihani (Dasto’rni) ishga tushirish va birinchi bo’lib yaratilgan forma chiqadi.

end. // dasturingizni tugagan qismi.

Delphi da ixtiyoriy komponentalarida ko’rinishi jihatidan uchga bulinadi.

Komponent konteyner - bunda bu komponent uzi ichiga yana boshqa komponentni olishi mumkin. (Masalan: Form, Panel, GroupBox vahokazo).

Komponent vizual - bunda bu komponent o'zi ko'rinmaydi uni chaqirish orqali ko'rish mumkin. (Masalan: Faylni ochish, saklash yoki rang tanlash muloqot oynasi komponentlari).

Komponenta - bu komponent oddiy komponenta.

Delphi da ixtiyoriy komponentalarda (ob'yektlarda) uch xil buyruqlar bo'ladi.

1 Xususiyat (Properties) - Bunda komponentalarni ko'rinishi xossalari kiradi (Masalan: Color, Width, Height,Caption va hokazo).

2 Hodisa (Events) - Bunda komponentalarni ustida sodir bo'ladigan jarayonlar kiradi (Masalan: OnClick, OnDblClick, OnKeyPress va hokazo).

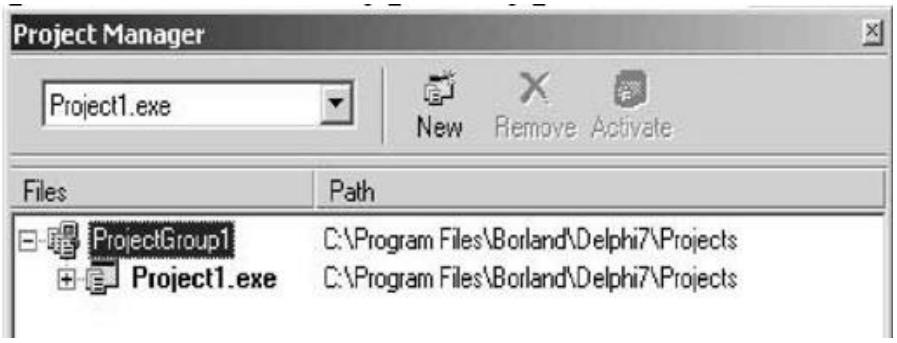
3 Metod - Bunda komponent ustida bajariladigan amallar kiradi (Masalan Memo komponentiga LoadFromFile metodi orqali biror faylni ochilishi va hokazo).

Delphi dasturining menyu bo'limlari. Menyuning "File" bo'limi Loyihaning fayllari bilan ishlash uchun "File" menyusini tanlang. "File" menyusi quyidagicha tuzilgan: Bu erda beshta bo'lim mavjud:

- Birinchi bo'lim loyiha bilan tupik ishlashga muljallangan.
- Ikkinchi bo'lim loyihaning formalari, modullari va qismlari ustidan nazoratga mo'ljallangan.
- Uchinchi bo'lim loyihaga fayllarni qo'shish va olib tashlash uchun xizmat qiladi.
- To'rtinchi bo'lim chop kilishni boshqaradi.
- Beshinchi bo'lim Delphi dan chiqish.

"File" menyusining har bir bo'limi yordam tizimida izohlangan bo'lganligi uchun ularga to'xtab o'tirmaymiz. Bularni ko'rish uchun "File" menyusini tanlab, F1 tugmasini bosing.

Loyihani boshqarish. Endi loyihalar menedjeri sharxiga o'tsak bo'ladi. Loyihalar Menedjeri ikki qismdan iborat bo'lib, birinchi qism boshqaruvchi tugmalar va undan pastdagi ikkinchi qismda loyiha tarkibiga kiruvchi modullar nomi va ularning diskdagi o'rni ko'rsatilgan.



1.5- chizma. Loyihalar menedjeri.

Siz boshqaruvchi tugmalar yordamida loyihaga yangi fayllarni qo'shishingiz yoki olib tashlashingiz mumkin. Bu o'zgarishlar dastlabki faylga o'zgartirishlar kiritadi, ya'ni loyihaga yangi modul qo'shilsa unga yo'llanma DPR kengaytmali faylda avtomatik holda yoziladi(1.5- chizma).

Menyuning "File" bo'limini yuqoda kurib o'tdik. Quyida menyuning - "Edit", "Search", "View" va "Compile" bo'limlarini qisqacha ko'ramiz.

"Edit" menyusi "Undo" va "Redo" buyruqlarini saqlaydi, ular muharrirda ishlayotganda behosdan qilingan xatolardan (masalan, matnnnig kerakli qismi uchirib yuborildi) voz kechishga muljallangan.

"Cut", "Copy", "Paste" va "Delete" buyruqdari Windowsning boshqa dasturlaridagi kabi, lekin ularni faqat matnlarga emas, balki, ob'yektlarga ham qo'llash mumkin.

"Bring To Front", "Send To Back", "Align" va "Size" ob'yektlarning formada turishi va ularning o'lchamlarini boshqarishga mo'ljallangan.

"Search" menyusida "Find Error" (xatoliklarni qidirish) buyrug'i mavjud bo'lib, u dastur bajarilish vaqtidagi xatoliklarni topishga mo'ljallangan. Xatolik haqidagi xabarda chiqarilgan raqamni *"Find Error"* muloqot oyasiga kiritilsa va muhit dasturdagi xatolik sodir bulgan qismni kursatadi.

"View" menyusining tarkibi quyidagicha:

- Project Manager (Loyiha Menedjeri).
- Project Source -muxarrirga loyihaning bosh fayli (DPR) ni yuklaydi.
- Ekranda Object Inspector oynasini ko'rsatish yoki yo'qligini bildiruvchi o'rnatma.

- Ekranda Alignment Palette kursatish kerak yoki yukdigini bildiruvchi o’rnatma.

- Browser - dasturdagi obektlar ierarxiyasini namoyish qiluvchi vositani chaqirish.

- Watch, Breakpoint va Call Stack - dasto’rni boshqarish bilan bog’lik va ularga keyinroq to’xtalamiz.

- Component List - Komponentlar to’plami. Komponentlarni nomi bo’yicha qidirishda yoki sichqoncha yo’qligida qo’llaniladi.

- Window List - Delphi muhitida ochilgan oynalar ruyxati.

- Toggle Form/Unit, Units, Forms - forma va moe modullar orasida utish, forma va modullar ruyxatdan tanlanadi.

- New Edit Window - muxarrirning kushimcha oynasini ochadi. Bu ko’pincha bir faylning ikki nushasini bilan ishlaganda qo’laniladi yaratadi.

- SpeedBar i Component Palette -ekranda namoyish qilish o’rnatmalari.

Menyuning “Compile” bo’limida loyihani ishga tayyorlash (compile) yoki qayta ishlash (build) mumkin. Agar Compile yoki Run tanlansa, Delphi faqat oxirgi kompilyastiyadan keyin o’zgartirilgan modullarni tayyorlaydi. Build all esa, loyiha tarkibidagi hamma modullar va dastur matnlarini tayyorlaydi. Syntax Check buyrugi esa, faqat dastur matni buyruqlari to’g’ri yozilganligini tekshiradi.

Eng pastki - Information buyrug’i dastur haqidagi umumiy ma’lumotlarni chikdradi.

“Run” bo’limidan dasto’rni ishga tayyorlash va dastur uchuy buyruq satri parametrlarini berish maksadida foydalanish mumkin.

“Options” tizim menyusining ancha murakkab qismidir. Siz bu erdan loyiha va Delphining butun ishchi muhit uchun o’rnatmalarni o’zgartirishingiz mumkin. “Options”HHHr etti buyrugi mavjud: Project - Environment - Tools - Gallery - Open Library - Install Components - Rebuild Library .

Dastlabki to’rt buyruq muloqot oynalarini chaqiradi. quyida menyuning “Options” bo’limiga qisqacha sharh keltirib o’tilgan:

Project-joriy loyihaga to’g’ridan-to’g’ri ta’sir qiluvchi parametrlarni o’rnatish.

Environment - dasturlash muhitining tuzilishini boshqarish (IDE). Masalan, bu erda muharrirda ishlatiladigan ranglarni o’zgartirish mumkin.

Tools - bosh menyuning Tools bo’limidagi tashki dasturlarni yuklash va ulardan voz kechish. Masalan, siz biror moduldan ko’p foydalansangiz, uni ro’yhatga qo’shib qo’yish mumkin.

Gallery-Forma va loyihalar eksperti uchuy spestifik o’rnatmalarni qo’llaydi. Oxirgi uchta buyruq; komponentlar to’plamini tuzishga yordam beradi.

Options | Project muloqot oynasi beshta asosiy sahifadan iborat:

Forms sahifasida loyihada ishlatiladigan barcha formalar keltirilgan;

Application sahifasida siz dastur uchuy kerak bo’ladigan sarlavha, yordam fayli va belgilarni aniqlashingiz mumkin.

Compiler sahifasi dastur matnidagi xatoliklar, dastur bajarilish vakti atoliklari, uni tayyorlash kabi vazifalar bilan shugullanadi.

Linker sahifasida dasto’rni boglash sharoitlarini aniklash mumkin.

Form, Forma va uning xususiyatlari va holatlari (komponent konteyner) Forma Delphi ning asosiy komponenti xisoblanadi va bunda loyihangizdagi barcha bajarilishi kerak bulgan loyihalashtirish va dasturingizni bajarilishi kerak bulgan ishlari bajariladi. Delphida tuziladigan dasturlar biror forma asosida qilinadi. Delphi da har bir yangi hosil qilingan formaga unga mos bo’lgan modul avtomatik tashkil qilinib tuziladi.

Bu esa dasturchi uchun juda qulay imkoniyat, ya’ni uning ishini tezlashtirishga yordam beradi. Bu formaning shaklini tanlash, unda kompanentalarni joylashtirish bizning ixtiyorimizda bo’ladi. Formaga biror kompanentani qo’ymoqchi bo’lsak, shu kompanentaning ustida sichqonchaning chap tugmasini ikki marta bosiladi va komponenta formaning urtasiga joylashadi yoki komponent ustida sichqonchaning chap tugmasini bir marta bosib va formani ixtiyoriy joyida sichqonchaning chap tugmasini

bosib komponentni joylashtirish mumkin. Biz uni xohlagan joyimizga surib kuchirishimiz mumkin.

Form, Forma komponentining holatlari.

OnActivate- Forma oynasi faol (aktiv) bo'lish jarayonida.

OnDeActivate- Forma oynasi faolligini (aktivligini) yukrtganda.

OnClose- Forma oynasi yopilayotganda

OnCloseQuery- Forma oynasi yopilguncha bajariladi. Forma oynasini yopilishi uchun surov yuboriladi, shuning uchun bu holatdan forma oynasi yopilishini tasdiklash mumkin.

OnCreate- Forma yaratilayotganda

OnCloseQuery- Forma oynasi yopilguncha bajariladi. Forma oynasini yopilishi uchun surov yuboriladi, shuning uchun bu holatdan forma oynasi yopilishini tasdiklash mumkin.

OnCreate- Forma yaratilayotganda

OnDestroy- Forma oynasi (unichtojaetsya) yukrtilayotganda

OnHide- Forma oynasi ko'rinmas holatiga utayotganda (formaning Visible xususiyat false bulganda)

OnShow- Forma oynasi ko'rinadigan holatiga utayotganda (formaning Visible xususiyat true bulganda)

OnClick- Forma oynasida sichqoncha bir marta bosilganda

OnDblClick- Forma oynasida sichqoncha ikki marta tez bosilganda

OnKeyDown- Klaviatura tugmasi bosilganda, bajariladigan holat

OnMouseMove- Sichqoncha tugmasi bu komponenta ustida xarakatlanayotganda bajariladigan holat.

Ko'proq foydalaniladigan Komponentlar Standart bo'limida joylashadi. (1.6-chizma). Formaga Komponentni o'rnatish uchun, komponentlar palitrasidan kerakli komponentni tanlab, uning piktogrammasi ustida sichqonchaning chap tugmasini 2 marta bosish kerak. Shundan sung komponentni xohlagan joyga joylashtirib, uning burchaklaridan sichqoncha tugmasi yordamida o'lchamlarini o'zgartiramiz. Natijada formada komponent standart o'lchamda hosil bo'ladi.

1.6- chizma. Standart bo’limida

Standart bo’limi o’z ichiga tez-tez foydalaniladigan komponentlarni oladi. Komponent o’lchamlarini uni formaga o’rnatish jarayonida berish mumkin. Bu uchun komponentni palitradan ajratib olgach, komponentning chap yuqori burchagi formaning qaysi nuqtasida joylashishi lozim bo’lsa, sichqoncha belgisini o’sha joyga ko’chirib, sichqonchaning chap tugmasi bosiladi, uni bosib turgan holatda kursorni komponentning quyi o’ng nuqtasi formaning kayerida joylashishi kerak bo’lsa o’sha joyga ko’chirib, keyin tugma qo’yib yuboriladi. Formada kerakli o’lchamdagi komponent hosil bo’ladi.

De lphi dagi har bir komponent nomlanishiga va tartib raqamiga ega. Masalan, formaga 2 ta Edit komponentini urnatsak, ular Edit1 va Edit2 kabi nomlanadi. Dasturchi Name xususiyati qiymatini o’zgartirish yo’li bilan komponent nomini o’zgartiradi. Oddiy dasturlarda komponent nomi odatda o’zgartirilmaydi.

Har bir komponent ustida 3 xil amal bajarish mumkin, ya’ni uni xususiyatini o’zgartirish mumkin, u ustida qandaydir hodisa bajarish mumkin va u ustida qandaydir usul qo’llash mumkin.

Delphi da biror komponentga biror amalni boglamokchi bo’lsak Delphi o’zi shu komponent nomiga tanlangan hodisa nomini birlashtirib qism dastur (prostedura) qilib beradi. Yaratilgan qism dasturga bajariladigan amallar ketma-ketligini kiritamiz. Quyida misolning qism dasturi keltirilgan.

```
procedure TForm1.Button1Click(Sender: TObject);
begin
Edit3.Text:=inttostr(strtoint(Edit1.Text)+
strtoint(Edit2.Text)) ;
end;
```

Edit komponentiga ma’lumotlar satr kurinishda saqlanadi.

Matnni (namoyish etuvchi) chikaruvchi komponentalar.

Bu kompanentalar ma’lumotlarni kiritish, saqlash, tahrir qilish va turli xabarlar berish vositasi sifatida foydalaniladi.

-TLabel (belgi) kompanentasi. (Standart komponentlar to’plamidan)

Belgi sizga mo’ljallashtirishda va bundan tashqari foydalanuvchiga **o’**z nomidan tashqari axborot berishda hamda oyna va belgilarni tavsiflashda yordam beradi. Yozuvlarni namoyish etish uchun ko’pincha metka (Label) komponentidan foydalanamiz. Bu komponenta oddiy matnni ishlatadi va dasturni ishlash jarayonida foydalanuvchi bu matnni o’zgartira olmaydi.

L abel komponentini matnini Caption xususiyati orqali o’zgartiramiz va komponenta, o’lchamini mos ravishda o’zgartirishi uchun uning AutoSize xususiyatidan foydalanamiz. Uning tipi mantikiy bo’lib, chin qiymat berishimiz kerak.

Label komponenti ichidagi matnni tekislash uchun Alignment xususiyatidan foydalanamiz. U Alignment tipli bo’lib quyidagi qiymatlardan birini olishi mumkin:

- taLefUustify - matnni chap tomonidan tekislash
- taCenter - matnni markazlashtirish
- taRightJustify - matnni ung tomonidan tekislash

Agar AutoSize xususiyatini qiymati chin (true) bo’lsa unda Alignment ta’sir etmaydi (ishlamaydi) .

Matnni uzunligi bo’yicha sig’maydigan so’zlarni avtomatik tarzda boshqa satrda o’tishi uchun Wordwrap xususiyatidan foydalanamiz. Uning tipi mantikiy bo’lib, chin qiymat berishimiz kerak. Agar AutoSize xususiyatini qiymati chin (true) bo’lsa unda

Wordwrap ta’sir etmaydi (ishlamaydi). Rang bilan tuldirilish rangini Color xususiyati orqali beramiz. Label rang bilan to’ldirilgan yoki rangsiz bo’lishini Transparent xususiyati orqali aniklanadi. Uning tipi mantikiy bo’lib, chin qiymat berishimiz kerak. Qachonki Label rasm ustida o’rnatilgan bo’lsa u holda rangsiz

bo’lishi kerak. Transparent xususiyatini qiymatini chin qilamiz. Masalan rasm geografik karta bo’lsa. Komponentdagi yozuvlarni shriftini Font xususiyati orqali o’zgartirish mumkin.

Label ni *Left* va *Top* xususiyatlari orqali bu komponentani, komponent konteynerdagi joylashgan o’rnini beramiz. Unda *Left* chap tomondan necha piksel *Top* esa yuqoridan.

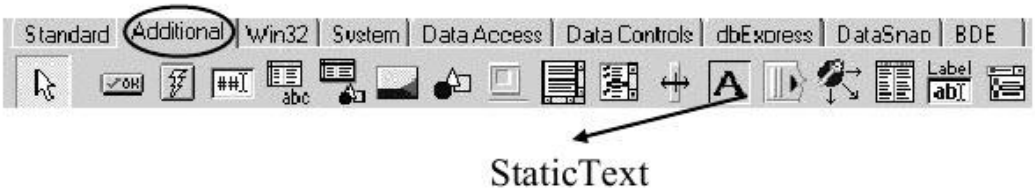
Label o’zi aktiv (faol) bo’la olmaydi, lekin tugmalar kombinastiyasi yordamida uni tanlash orqali boshqa komponentaga faolligini berishi mumkin. FocusControl xususiyati shu uchun ishlatiladi, uni tipi TWinControl. Masalan Label 1, Editl ni izoxi sifatida ishlatsak u holda Labell.FocusControl := Editl berish o’rinli.

FocusControl ni faqat tugmalar kombinastiyasi orqali ishlatish mumkin. Label 1 ni sichqoncha orqali tanlanganda Editl komponenti faol (aktiv) bo’lishi uchun Label 1 ni OnClick holatiga quyidagini yozamiz:

```
procedure TForml.Label lClick(Sender: TObject);
begin
if Editl.CanFocus then Editl.SetFocus;
end;
```

Bunda tekshiriliyapti agar Editl faol bo’lmasa u holda faol (aktiv) lashtir.

Label matnni chiqaruvchi komponenti bo’lib, matnini tahrirlab bo’lmasligi uchun uni statik matn deb ham aytadilar.



1.7- chizma. Additional bo’limi.

Lekin statik matn nomli komponent ham mavjud (StaticText). Bu ikkita komponentani ishlash vazifasi va maqsadi bir xildir. StaticText, TWinControl

sinfining merosxo'ri (vorisi) bo'lib, TWinControl sinfi asosida yaratilgan boshqa oynali komponentalar bilan boglash mumkin Undan tashqari StaticText o'zining atrofida ramka (chizik) chizishi mumkin buni BorderStyle xususiyati orqali bajaramiz. Bu xususiyatni tipi TStaticBorderStyle uchta qiymatdan iborat

- sbsNone - Ramka yo'q
- sbsSingle - Ramka chiziqli bor
- sbsSunken - Ramka chiziqsiz bor (ichkariga kirilgan kurinishda).

Matnini tahrirlab bo'lmaydigan komponentlar uchun Edit dan ham foydalanish mumkin. Agar uni Readonly xususiyatini qiymati true (chin) bo'lsa. Masalan:

```
Editl.Readonly := true;
Editl.Color := clBtnFace;
Editl.CtBD := false;
Editl.BorderStyle := bsNone;
Editl.Text := 'Matn! faqat ko rish uchun';
```

Editl ni hozirgi holatida StaticText va Label dan farq qilmaydi.

Matnni kirituvchi komponentalar.

Ma'lumotlarni kiritish va tahrirlash uchun Delphi da maxsus turli xil komponentlar berilgan. Masalan Edit, MaskEdit, Memo, RichEdit. Bulardan tashqari Delphi7 da yangi LabelEdit komponenti ham yaratilgan. Bu komponentlar orqali nafaqat ma'lumotlar chiqarish, xatto foydalanuvchi bu ma'lumotlarni o'zgartira olishi ham mumkin.

Bir satrda matnni kirituvchi komponentalar.

Bir satrda matnni kirituvchi komponentalarda ma'lumotni bir satrda chiqaradi va tahrirlashga ruxsat beradi. Delphi da bir nechta shunday komponentlar bor, bulardan ko'p foydalaniladiganlardan Edit komponentidir. Bu komponentga ma'lumotlarni kiritish yoki undan foydalanish uchun Text xususiyatidan foydalanamiz. Atrofidagi ramkalarni o'zgartirish, ranglash, shriftlarni sozlash barcha komponentlar uchun bir xil.

Satrda belgilarni registrini o’zgartirishi uchun CharCase xususiyatidan foydalanamiz va uning tipi TEditCharCase bo’lib uchta qiymatdan birini qabul qiladi.

- ecLowerCase - Belgilar kichik registrga o’tkaziladi (kichik harflarga).

- ecNormal - Belgilar registri o’zgartirilmaydi (kiritilgan belgi o’zligicha koladi).

- ecUpperCase- Belgilar katta registrga o’tkaziladi (katta harflarga).

Edit ni parol kiritishda ham foydalanish mumkin. Unda PasswordChar xususiyatidan foydalanamiz va uning tipi Char. Ma’lumot kiritish vaqtida, kiritilayotgan belgi o’rnida bu

xususiyatda berilgan belgi chiqaveradi. Masalan: Editl.PasswordChar:= ‘*’; Edit 1.Text:=’Parol’; Edit da ‘*****’ ko’rinishda chiqadi va uning Text xususiyatida ’Parol’ qiymati turadi.

Mask Edit komponenti. Bu komponent ham Edit komponentiga o’xshaydi MaskEdit ko’p hollarda ma’lumotlarni yashirish holda yoki biror bir kodlash orqali kiritishga to’gri kelib qoladi. Shunday holda bu komponentadan foydalanish mumkin. Masalan parol kiritishda va hokazo. Edit komponentidan farqli MaskEdit komponentida EditMask xususiyati mavjud. Bu xususiyat MaskEdit komponentida ma’lumotlarni qanday ko'rinishda chiqishini ko’rsatadi.

Ko’p satrli matnni kirituvchi komponentalar.

Memo komponenti sodda matn muhariri bo’lib, u ko’p satrli ma’lumotlarni kiritish va chiqarish uchun ishlatiladi. Bu komponent mantlar ustida amallar bajarish uchun ishlatiladi. Bu matn muhaririga Memo1.Lines[ ] xususiyati orqali ma’lumotlarni olish yoki qo’shish mumkin.

PasswordChar xossasi. Ushbu xossa matnni kiritishdan himoya uchun foydalaniladi. Parol kiritish kerak paytida foydalaniladi. Ushbu xossadan siz maxfiylikning qo’shimcha darajasini ta’minlash uchun va foydalanuvchi ismini yashirish uchun foydalanish mumkin.

ReadOnly xossasi. Ushbu xossa ob’ektning qiymatini tahrirlash imkoniyatini boshqaradi. Siz uning qiymatini False yoki True qilib qo’yishingiz mumkin.

Agarda xossaning qiymati True bo’lsa foydalanuvchi ushbu ob’yektdan foydalanishga, ya’ni ixtiyoriy matnni belgilash va nusxasini buferga olish uchun birmuncha ruxsat beradi.

MaxLength xossasi. Ushbu xossadan uchala ob’ektga kiritiladigan simvollarni cheklash uchun foydalaniladi.

RichEdit - komponenti sodda matn muhariri bo’lib, u ko’p satrli ma’lumotlarni kiritish va chiqarish uchun ishlatiladi. Bu komponent ham **Memo** komponentiga o’xshaydi faqat unga nisbatan imkoniyati yuqoriroq. Bu komponent **Win32** komponentlar to’plamida joylashgan.

1.2. Delphi dasturida ma’lumotlar bazasini bog’lash.

Deiphi 7 da oldingi versiyalarida bo’lmagan Microsoft, Active, Data Objects bilan ishlash imkoniyati yaratildi. ADO – bu Microsoft ma’lumotlar ombori bilan ishlaydigan standart texnologiya. Bu texnologiya BDE bilan deyarli bir xil va imkoniyatlari ham yaqin.

1.8- chizma. ADO kompanent ko’rinishi.

ADO bo’linmasi quydagi asosiy oltita komponentni mavjud: TADOConnection, TADOCommand, TADODataSet, TADOTable, TADOQuery, TADOStoredProc.

TADOConnection- BDE Database komponenti bilan bir xil bo’lib ma’lumotlar bazasini ko’rsatishda ishlatiladi.

TADOTable- ADOConnection da ko’rsatilgan bazaning jadvaliga muroojat.

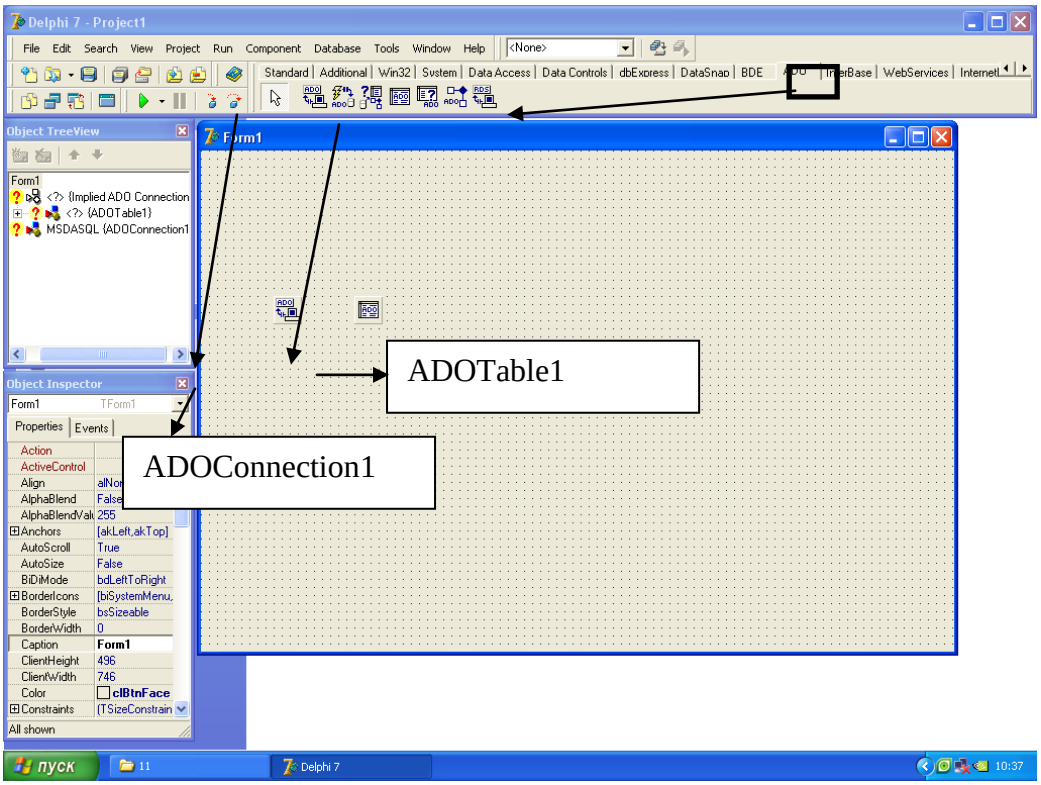
TADOQuery- ma’lumotlar bazasiga so’rovlar bo’lib, so’rov natijasida ma’lumot qaytarish(SELECT)

TADOStoredProc – Yaratilgan protsedura chaqiruvi. BDE va InterBase ga saqlanadigan jarayonlardan farqli o'laroq ADO da ma'lumotlarni qaytarish mumkin, shu sababli bu tur komponenti TDATASET avlodi sanaladi va TDATASource ma'lumotlar manbasi bo'lishi mumkin.

TADOCommand va TADODataSet ADO ning umumiy komponentlari hisoblanadi lekin ular bilan ishlash qiyinroq. Ikkala component ham tip bilan bog'liq ma'lumotlarni qaytarishga yordaam beradi.

Ular farqi shundaki TADODataSet amalga oshiriladigan topshiriqlar ma'lumotlarni qaytaradi va Delphi bilan ishlashga yordam beradi.

ADO bo'limidan ADOConnection komponenti ustiga sichqonchaning chap tugmasini birmarta bosamiz va sichqonchani yaratilgan Form ustiga keltirib chap tugmasini bir marta bosamiz. Shunday yo'l bilan ADOTable ham olamiz.

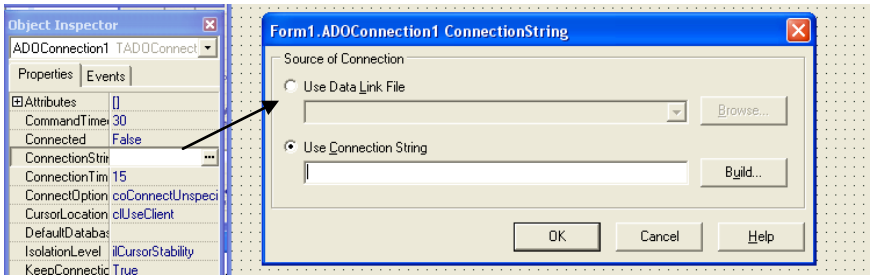


1.9- chizma. Foam1 ga ADO Connection va ADO Table o'rnatilinishi.

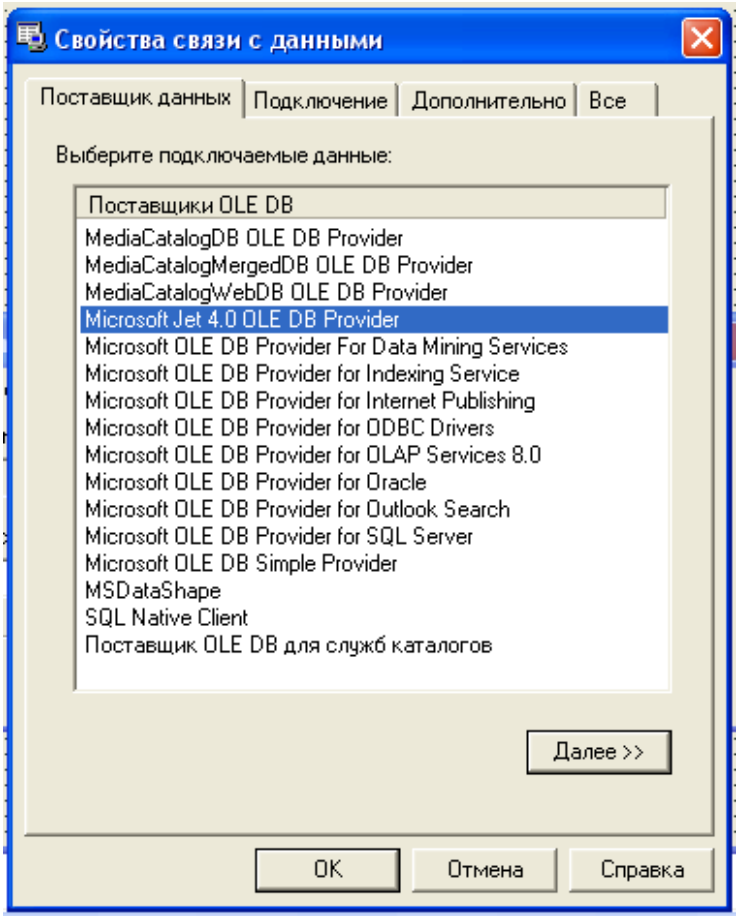
ADOConnection1 – komponenti ma'lumotlar bazasini bog'lash uchun ishlatiladi. Uning ConnectionString – xususiyati orqali ma'lumotlar bazasi bog'lanadi. Quyidagicha: ConnectionString xususiyatining o'ng tomonidagi

tugmani bosamiz ConnectionString oynasi chiqadi. Oynada Use Data Ling File va Use Connection String qisimlaridan Use Connection String qismini tanlab “Buld” tugmasini bosamiz.

Bog’lash bo’limida baza qaysi malumotlar omboriga qurulgan bo’lsa oyna aynan o’sha ma’lumotlar omborini ko’rsatamiz. Jet 4.0 ni tanlab “dalee” tugmasini bosamiz. Agar ma’lumotlar bazasi SQL serverda yoki Accses da tuzulgan bo’lsa o’sh ma’lumotlar ombori tanlanadi.



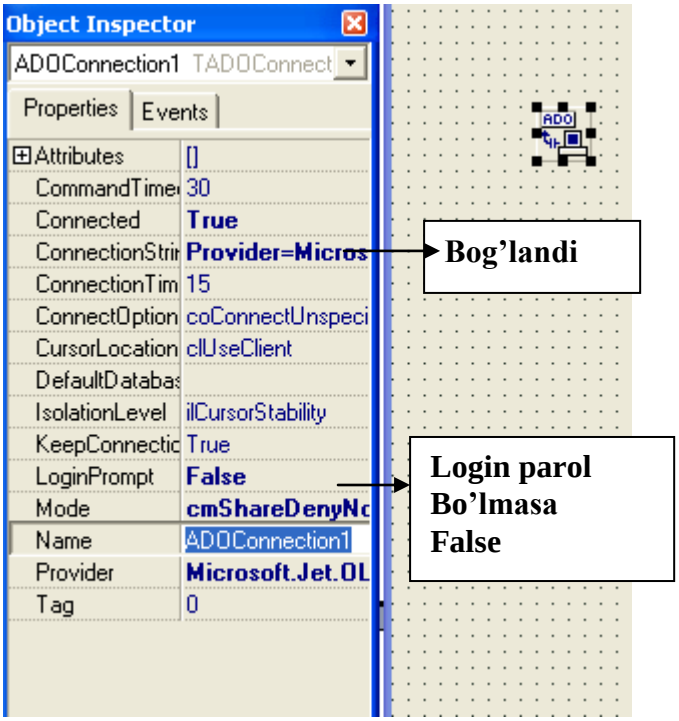
1.10- chizma. Connection String oynasi.



1.11- chizma. Baza faylini bog’lash.

Bazani faylini ko’rsatib parolni bo’sh deb belgilab “Проверить подкючение” tugmasini bosamiz. Bog’lanish ko’g’riligini tekshirgach OK- tugmasini bosamiz.

Connection String xususiyati bog’lanishni saqlaydi. LoginPrompt xususiyati False qiymat beramiz. Connected ga True qiymat beramiz(1.12- chizma). ADOTable1 – komponenti orqali ma’lumotlar bazasidagi jadvallar bog’lanadi.

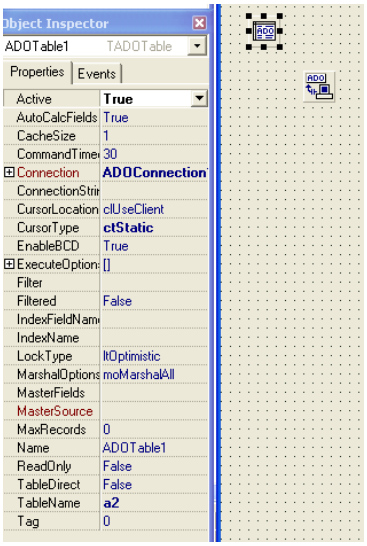


1.12- chizma. Bog’lanish.

ADO Table1 ning Connection xususiyatiga ADOConnection1 bog’lanadi. TaleName bog’langan ma’lumotlar bazasi jadvali bog’lanadi. Agarda yaratilayotgan loyihamimizda bir nechta jadvallardan foydalansak o’zimizga tushunarli bo’lishi uchun Name xususiyatiga jadval nomini qo’ysa bo’ladi. Uchinchi qiladigan ishimiz “Active” xususiyatiga True qiymatida bog’lagan jadvalimizni faol qiladi.

Endi jadvaldagi ma’lumotlarni namoish qilish uchun DBGrid1 va DataSource1 komponentlaridan foydalanamiz. Menyu panelining DataAccess bolimidagi DataSource1– komponenti joylashtiramiz. DataSource1– komponentining DataSet xususiyatiga ADOTable1 bog’lanadi. DBGrid1 – komponentining DataSource xususiyatiga DataSource1 bog’lanadi. Jadvaldagi

ma’lumotlar DBGrid1 – komponentiga namoyish etiladi ma’lumotlarni bemalol ko’rishingiz mumkin.



1.13- chizma. ADO Table1 ADO connection 1 ga bog’lanishi.

Xulosa: Delphi dasturlash muhiti Borland kompaniyasi tomonidan ishlab chiqarilgan bo’lib, u Borland Pascal tilining keyingi versiyasi Borland Object Pascal tili asosida qurilgan. Ob’yektga mo’ljallangan dasturlash tilllarida dastur tuzish ancha oson va ishchi vaqt tejaladi. Dasturchi ob’yektlarni qurib o’tirmaydida tayyor ob’yektlarni qo’yib olaveradi va dastur asosiy qismiga bosh qotiradi xolos. Delphi kuchli xatolarni bartaraf qilish (Debuger) sistemasiga va qulay yordamchi sistemasiga egadir. Delphi juda qulay interfeysga ega.

Deiphi 7 da oldingi versiyalarida bo’lmagan Microsoft, Active, Data Objects bilan ishlash imkoniyati yaratildi. ADO – bu Microsoft ma’lumotlar ombori bilan ishlaydigan standart texnologiya. Bu texnologiya BDE bilan deyarli bir xil va imkoniyatlari ham yaqin.

II BOB. Tez tibbiy yordami xizmatini qo’llab quvvatlovchi informatsion tizim mijoz qismi.

2.1. Sozlovchi qismi.

Tez tibbiy yordami xizmatini qo’llab quvvatlovchi informatsion tizim dasturini yaratilishiga sabab shundaki tez tibbiy xizmatni faol xizmat ko’rsatish tezligini yanada oshirish buning natijasida ish unumdorligi ko’paytirish. Aniq ma’lumotlarga ega bo’lish. Sozlovchi qismidagi birgina hodimlar oynasi orqali Hodimlarning turar joyi, Ishga qabul qilinganligi to’grisidagi buyrug’i, ishga qabul qilingan sanasi, hissasi, pasport serya raqami, tulg’ulgan kuni, xodim haqida qo’shimcha ma’lumot, uy telefoni, malumotlar birta jadvalga jamlangan. Xodim yangi kelsa yoki ayrim bir ma’lumotlarini o’zgartirish kerak bo’lsa bu judaham oson ma’lumotlar chisqa holda to’ldiriladi saqlab qoyiladi, o’chiriladi dastur shukabi qulayliklar va ko’pgina imkoniyatlarga ega.

Tez tibbiy yordami xizmatini qo’llab quvvatlovchi informatsion tizim dasturini asosiy oyna bosh menyu qisimlari “Chiqish”, ”Spravochnik”, ”Asosiy”, ”Operator” lardan iborat. Spravochnik darchasida ”Brigada turlari” , ”Diagnoz turlari” , ”Kasallik belgilari”, ”Dori darmonlar”, ”Shtatlar”, ”Manzillar”, ”Markazlar”, ”Kasalxonalar” menyusidan iborat(2.1- rasm).

2.1- chizma. Asosiy oyna bosh menyu qisimlari

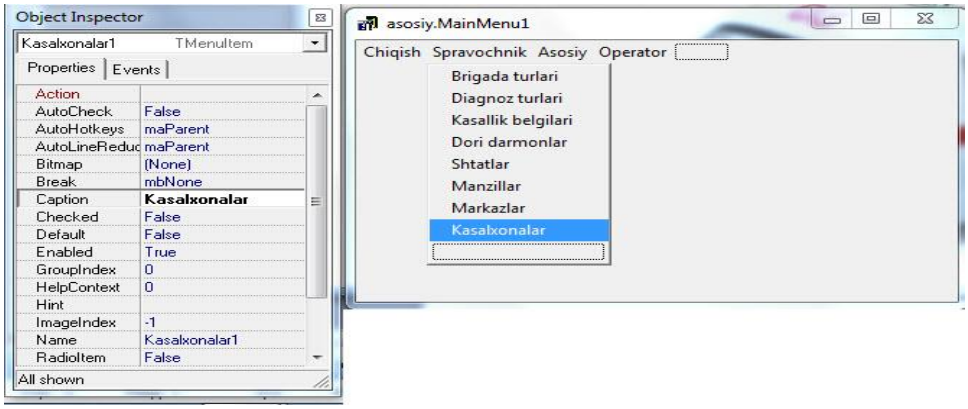
Asosiy oynani yaratamiz. Ob’yektlar inspektorining Caption xususiyatiga Form1 ga Asosiy oyna deb nom beramiz. Ob’yektlar inspektorining FormStyle

xususiyatiga fsMDIForm qiymatni beramiz. Asosiy oyna keyingi yaratiladigan form larni o’z ichiga oladi. Asosiy oyna xususiyatlari uning ”bola form” (fsMDIForm) sanalmish forimlarga o’tadi. Dastur kodi:

```
program Project1;
uses
  Forms,
  Unit1 in 'Unit1.pas' {asosiy},
{$R *.res}
begin
  Application.Initialize;
  Application.CreateForm(Tasosiy, asosiy);
   Application.Run;
end.
```

Asosiy (Form1) oyna yuqori qismiga Menyu bo’limini joylashtiramiz. Bosh menyu komponenti Standart komponentlar to’plamida joylashgan. TmainMenu-dasturga bosh menyu qo’shish imkoniyatini beradi. MainMenu1 ni Asosiy formaga o’rnatib. Komponent ustiga sichqonchaning chap tugmasini ikki marta bosamiz yoki Ob’yektlar inspektorining Items xususiyati yordamida Menyu dizaynerini chaqiramiz. Menyu dizaynerida menyu qisimlarini kiritiladi. Menyu bo’limlari nomlarini menyu dizayneri chiqgandan keyin uning Caption xususiyati orqali kiritiladi(2.1- chizma). Dastur kodi:

```
unit Unit1;
interface
uses Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms, Dialogs, Menus, DBActns, ActnList, ToolWin, ActnMan, ActnCtrls,
XPStyleActnCtrls, StdCtrls, jpeg, ExtCtrls;
```



2.2-chizma. MainMenu1.

Type

Asosiy1: TMenuItem;

Apteka1: TMenuItem;

Brigadayaratish1: TMenuItem;

Xodimlar1: TMenuItem;

Shtatlar1: TMenuItem;

Manzillar1: TMenuItem;

Markazlar1: TMenuItem;

Kasalxonalar1: TMenuItem;

Avtomashinalar1: TMenuItem;

Operator1: TMenuItem;

Tadbir1: TMenuItem;

Kartaochish1: TMenuItem;

Asosiy Formga ActionManager1 yaratilish dastur kodi.

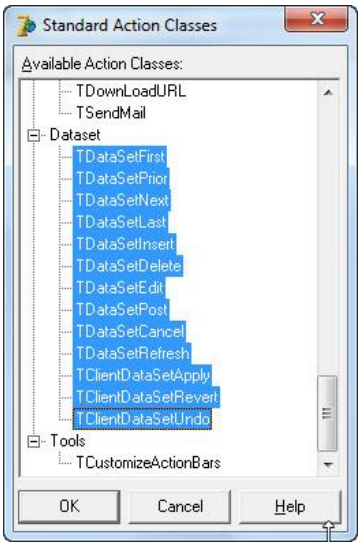
unit Unit1;

interface

uses Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms, Dialogs, Menus, DBActns, ActnList, ToolWin, ActnMan, ActnCtrls, XPStyleActnCtrls, StdCtrls, jpeg, ExtCtrls;

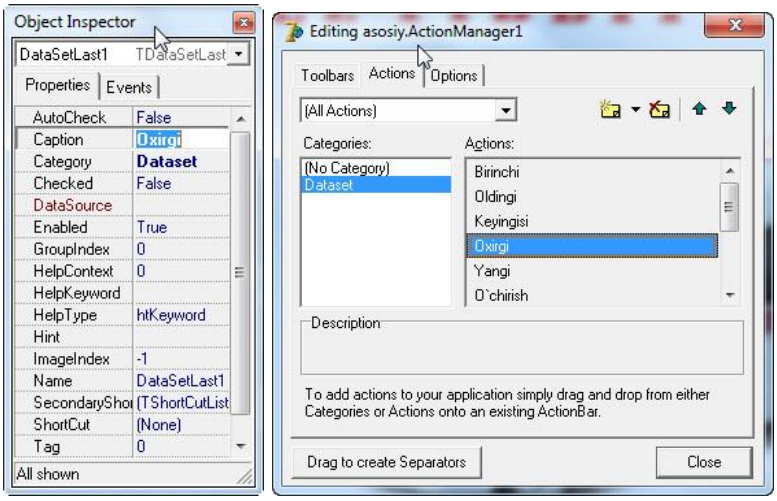
Type

ActionManager1: TActionManager



2.3- chizma. DataSet qismi.

Ob’yektlar inspektorining Caption xususiyatiga tushunarli qilib izoxlab chiqiladi.(2.4- chizma).



2.4- chizma. Caption xususiyati.

ActionToolBar1: TActionToolBar;

DataSetFirst1: TDataSetFirst;

DataSetPrior1: TDataSetPrior;

DataSetNext1: TDataSetNext;

DataSetLast1: TDataSetLast;

DataSetInsert1: TDataSetInsert;

DataSetDelete1: TDataSetDelete;

DataSetEdit1: TDataSetEdit;

DataSetPost1: TDataSetPost;

DataSetCancel1: TDataSetCancel;

DataSetRefresh1: TDataSetRefresh;

Asosiy oynaning yuqori qismi quydagi ko’rinishda hosil bo’ladi. (2.5 – chizma).

2.5 – chizma. ActionManager asosiy oynaning yuqorisida.

Bosh menyuda joylashgan ”Brigada turi” tanlanganda yangi form Brigada turi oynasi ochilsin va jadvalni ko’rsatish talabini qo’yamiz. Buning uchun yangi form yaratib Ob’yektlar inspektorining Caption xususiyatiga Brigada turi deb kiritaniz. Ob’yektlar inspektorining Name xususiyatiga br_turi deb oynamiz nomiga moslab qo’yiladi. FormStyle xususiyatiga fsMDIChild qiymatini beramiz. Endi Brigada turi oyna Asosiy oynaning ichida yaraladi va Asosiy oyna xususiyatlarini meros bo’lib o’tadi. Forma yopish tugmasi bosilganda yopilish uchun quydagi kodni kiritamiz:

unit Unit3;

interface

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

type

Tbr_turi = class(TForm)

```
        DBGrid1: TDBGrid;
        procedure FormClose(Sender: TObject; var Action: TCloseAction);
    private
        { Private declarations }
    public
        { Public declarations }
    end;
    var
    br_turi: Tbr_turi;
    implementation
    uses Unit1, Unit2;
    {$R *.dfm}
    procedure Tbr_turi.FormClose(Sender: TObject; var Action: TCloseAction);
    begin
    free;
    end;
    end.
```

Bosh menyuda joylashgan ”Brigada turi” tanlanganda Brigada turi oynasi ochilishi uchun Unit1 ga quydagi kodlarni qo’shib qo’yiladi.

```
    procedure Brigadaturlari1Click(Sender: TObject);
    procedure Tasosiy.Brigadaturlari1Click(Sender: TObject);
    begin
     Application.CreateForm(Tbr_turi, br_turi);
    end;
```

Brigada turi oynasini brigada turi jadvalini ko’rsatishi uchun Data Controls bo’limidagi DBGgid ni oynaga kerakli kaattalikka joylashtiramiz. Baza bilan bog’langanda DBGrid jadvalni ko’rsatadi.

Tez tibbiy yordami xizmatini qo’llab quvvatlovchi ma’lumotlar ombori SQL Serverda yaratilgan bazani TADOConnection orqali Delphi ga bog’lanadi.

DataModule2 yatamiz. Loyihada ko’p jadvallardan va jadvallarning bog’lanishlaridan foydalanishga to’g’ri keladi. Jadvalga murojat ADOConnection, ADOTable1, DataSource larni DataModule1 ga joylashtiriladi.

ADO bo’limidan ADOConnection1 ni DataModule1 ga joylashtiramiz. Ob’yektlar inspektorining Name xususiyatiga “con” nomini beramiz. ADOConnection1 ning Ob’yektlar inspektorining ConnectionString – xususiyati orqali SQL Serverda yaratilgan Tezyordam ma’lumotlar bazasi bog’laymiz. Quyidagicha: ConnectionString xususiyatining o’ng tomonidagi tugmani bosamiz ConnectionString oynasi chiqadi. Oynada Use Data Ling File va Use Connection String qisimlaridan Use Connection String qismini tanlab “Buld” tugmasini bosamiz. Baza SQL Sever da qurulganligi uchun SQL Server ni tanlaymiz. Bazani faylini ko’rsatib parolni bo’sh deb belgilab”Проверить подкючение”

Tugmasini bosamiz. Bog’lanish to’g’riligini tekshirgach Ok – tugmasini bosamiz.

ConnectionString xususiyati bog’lanishni salaydi. LoginPrompt xususiyati Yolg’on(False) qiymat beramiz. Connected ga True qiymat beramiz.

ADO bo’limidan ADOTable1 ni DataModule1 ga joylashtiramiz.

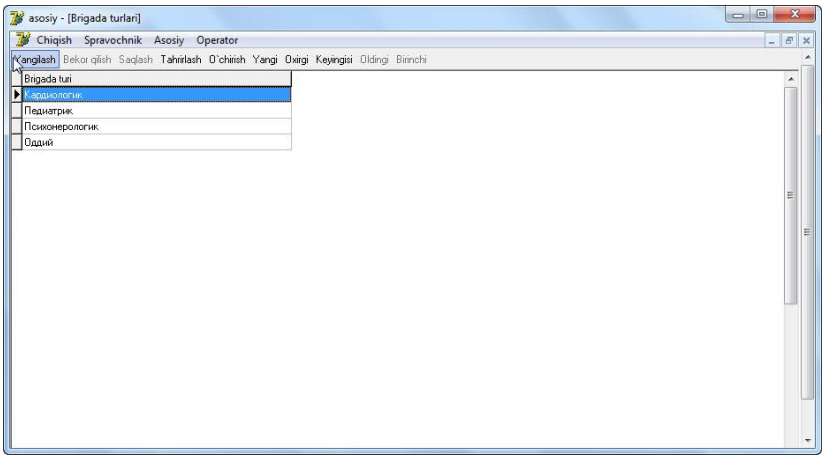
ADOTable1 – komponenti orqali ma’lumotlar bazasidagi jadvallar bog’lanadi. Quyidagicha: ADO Table1 Ob’yektlar inspektorining Connection xususiyatiga ”con” bog’lanadi. Ob’yektlar inspektorining TaleName xususiyatiga Tezyordam bazasidagi kerakli bo’lgan ”Brigada turi” jadvalni tanlaymiz bog’lanadi. Active” xususiyatiga True qiymat berganimizda bog’lagan ”Brigada turi” jadvalimiz faol bo’ladi.

DataAccess bo’limidan DataSource1 ni DataModule2 ga joylashtiramiz. Ob’yektlar inspektorining DataSet xususiyatiga birgadaturi tanlaymiz. DataSource1 ADOTable1(Birigada turi) ga bog’landi. Ob’yektlar inspektorining Name xususiyatiga ds_birgadaturi nom beramiz.

DBGrid ning DataSource xususiyatiga DataModule2.ds_birgadaturi qiymatni beramiz. DBGrid jadvalni ko’rsatadi. Jadval birta brigada turi ustinidan

iborat. Bu jadval ustiniga yangi satir kiritishimiz, saqlab qo’yishimiz yoki o’chirishimiz mumkin(2.6- chizma).

Bosh menyuda joylashgan ”Diagnoz turlari” tanlanganda yangi form Brigada turi oynasi ochilsin va jadvalni ko’rsatish talabini qo’yamiz. Buning uchun yangi form yaratib Ob’yektlar inspektorining Caption xususiyatiga Diagnoz turlari deb kiritaniz.



2.6-chizma. Brigada turi oynasi.

Ob’yektlar inspektorining Name xususiyatiga diagnoz_turi deb oynamiz nomiga moslab qo’yiladi. FormStyle xususiyatiga fsMDIChild qiymatini beramiz. Endi Diagnoz turlari oyna Asosiy oynaning ichida yaraladi va Asosiy oyna xususiyatlarini meros bo’lib o’tadi. Forma yopish tugmasi bosilganda yopilish uchun quydagi kodni kiritamiz:

```
procedure Tdiagnoz_turi.FormClose(Sender: TObject;
var Action: TCloseAction);
begin
free;
end;
```

Oynaning dizaynini oshirish maqsadida beshta Panel va uchta DBGrid dan foydalanayiz. Dastur kodi:

```
unit Unit4;
interface
```

```
uses Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls,
Forms, Dialogs, Grids, DBGrids, ExtCtrls, StdCtrls, DBCtrls;
type
Tdiagnoz_turi = class(TForm)
DBGrid1: TDBGrid;
Panel1: TPanel;
Panel2: TPanel;
DBGrid2: TDBGrid;
Panel3: TPanel;
Panel4: TPanel;
DBGrid3: TDBGrid;
Panel5: TPanel;
DBLookupComboBox1: TDBLookupComboBox;
Label1: TLabel;
procedure FormClose(Sender: TObject; var Action: TCloseAction);
procedure Panel1Click(Sender: TObject);
private
{ Private declarations }
public
{ Public declarations }
end;
var
diagnoz_turi: Tdiagnoz_turi;
implementation
uses Unit1, Unit2, Unit3;
{$R *.dfm}
procedure Tdiagnoz_turi.FormClose(Sender: TObject;
var Action: TCloseAction);
begin
free;
```

end;

end.

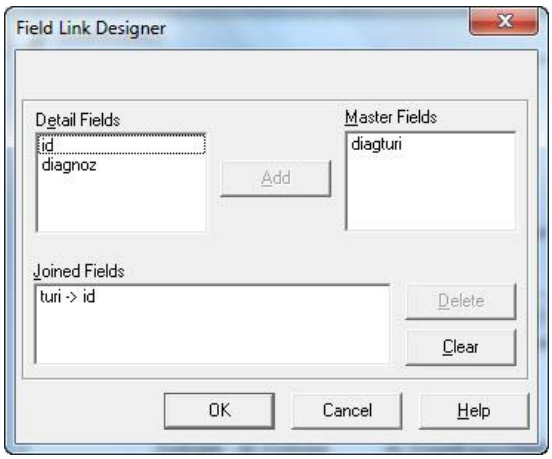
Diagnoz turlari oynasi Diagnozturi, Diagnozlar, V_diagnoz_belgi jadvallari ma’lumotlaridan foydalanib yaratamiz. ADOTable2 ni DataModule2 ga joylashtiramiz. ADO Table1 Ob’yektlar inspektorining Connection xususiyatiga ”con” bog’lanadi. Ob’yektlar inspektorining TaleName xususiyatiga ” Diagnozturi” jadvalni tanlaymiz. Active xususiyatiga True qiymat berganimizda bog’lagan ” Diagnozturi” jadvalimiz faol bo’ladi.

DataAccess bo’limidan DataSource2 ni DataModule2 ga joylashtiramiz. Ob’yektlar inspektorining DataSet xususiyatiga ” Diagnozturi” tanlaymiz.

Keyingi ikkita jadvallarimizni ham xuddu shutartibda bog’laymiz.

Diagnozturi, Diagnozlar, V_diagnoz_belgi ADOTable lar o’zaro boglanadi.

Diagnozlar ADOTable si MasterSource xususiyati orqali Diagnozturi ADOTable bilan bog’lanadi. MasterFields xususiyatiga Diagnozlar jadvalining “turi” ustuni bilan Diagnoz turi jadvalining “id” ustuni bog’lab Add tug’masini bosamiz va OK tugmasi bosiladi. (2.7-chizma).

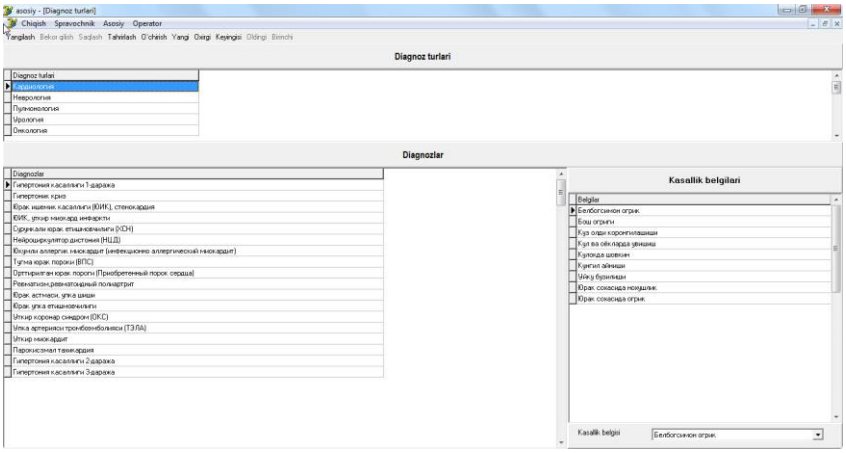


2.7- chizma. Diagnozlar jadvalining “turi” ustuni bilan Diagnoz turi jadvalining “id” ustuni bog’lanishi.

V_diagnoz_belgi ADOTable si MasterSource xususiyati orqali Diagnozlar ADOTable bilan bog’lanadi. MasterFields xususiyatiga V_diagnoz_belgilari

jadvalining “Diagnoz” ustuni bilan Diagnozlar jadvalining “id” ustuni bog’lab Add tug’masini bosamiz va OK tugmasi bosiladi.

DBGrid ning DataSource xususiyatiga DataModule2.ds_diagnozturi qiymatni beramiz. DBGrid jadvalni ko’rsatadi. Jadval birta Diagnoz turlari ustinidan iborat. Bu jadvalda tanlangan satriga mos keluvchi Diagnozlar jadvalining diagnozlar ustuni o’zgaradi. Qolgan DBGrid larga ham shunday bo’lanadi.(2.8-chizma).



2.8- chizma. Diagnoz turlari oynasi.

Bosh menyuda joylashgan ”Kasallik belgilari” tanlanganda yangi form Kasallik belgilari oynasi ochilsin va jadvalni ko’rsatish talabini qo’yamiz. Buning uchun yangi form yaratib Ob’yektlar inspektorining Caption xususiyatiga Kasallik belgilari deb kiritaniz. Ob’yektlar inspektorining Name xususiyatiga Kasallik belgilari deb oynamiz nomiga moslab qo’yiladi. FormStyle xususiyatiga fsMDIChild qiymatini beramiz. Endi Kasallik belgilari oyna Asosiy oynaning ichida yaraladi va Asosiy oyna xususiyatlari meros bo’lib o’tadi. Forma yopish tugmasi bosilganda yopilish uchun quydagi kodni kiritamiz:

```
unit Unit5;
interface
uses Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms, Dialogs, Grids, DBGrids, ExtCtrls;
type
```

```
  TKasallik_belgilari = class(TForm)
      Panel1: TPanel;
      DBGrid1: TDBGrid;
      procedure FormClose(Sender: TObject; var Action: TCloseAction);
  private
      { Private declarations }
  public
      { Public declarations }
  end;
var
  Kasallik_belgilari: TKasallik_belgilari;
  implementation
  uses Unit1, Unit2, Unit3, Unit4;
  {$R *.dfm}
  procedure TKasallik_belgilari.FormClose(Sender: TObject; var Action:
TCloseAction);
  begin
  free;
  end;
  end.
```

ADOTable ni DataModule1 ga joylashtiramiz. ADO Table1 Ob'yektlar inspektorining Connection xususiyatiga "con" bog'lanadi. Ob'yektlar inspektorining TaleName xususiyatiga " Kasallik belgilari" jadvalni tanlaymiz. Active xususiyatiga True qiymat berganimizda bog'lagan " Kasallik belgilari" jadvalimiz faol bo'ladi.

DataSource ni DataModule2 ga joylashtiramiz. Ob'yektlar inspektorining DataSet xususiyatiga " Kasallik belgilari" tanlaymiz.

DBGrid ning DataSource xususiyatiga DataModule2.ds_Kasallik_belgilari qiymatni beramiz. DBGrid jadvalni ko'rsatadi. Jadval birta Kasallik belgilari

ustinidan iborat. Bu jadval ustiniga yangi satir kiritishimiz, saqlab qo’yishimiz yoki o’chirishimiz mumkin(2.9- chizma).

Bosh menyuda joylashgan ”Dori darmonlar” tanlanganda yangi form Dori darmonlar oynasi ochilsin va jadvalni ko’rsatish talabini qo’yamiz. Buning uchun yangi form yaratib Ob’yektlar inspektorining Caption xususiyatiga Dori darmonlar deb kiritaniz. Ob’yektlar inspektorining Name xususiyatiga Dori_darmonlar deb oynamiz nomiga moslab qo’yamiz.

2.9- chizma. Kasallik belgilari oynasi.

FormStyle xususiyatiga fsMDIChild qiymatini beramiz. Endi Dori darmonlar oynasi Asosiy oynaning ichida yaraladi va Asosiy oyna xususiyatlari meros bo’lib o’tadi. Forma yopish tugmasi bosilganda yopilish uchun quydagi kodni kiritamiz:

```
unit Unit6;
interface
uses Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms, Dialogs, DBCtrls, Grids, DBGrids, ExtCtrls;
type
Tdoridarmon = class(TForm)
    Panel1: TPanel;
    DBGrid1: TDBGrid;
```

```
procedure CloseForm(Sender: TObject; var Action: TCloseAction);
private
  { Private declarations }
public
{ Public declarations }
end;
var
doridarmon: Tdoridarmon;
implementation
uses Unit5, Unit1, Unit2, Unit3, Unit4;
{$R *.dfm}
procedure Tdoridarmon.CloseForm(Sender: TObject; var Action: TCloseAction);
begin
free;
end;
end.
```

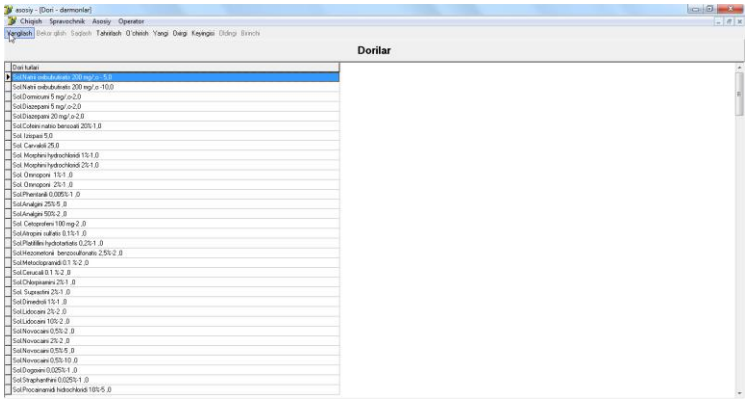
ADOTable ni DataModule2 ga joylashtiramiz. ADO Table Ob'yektlar inspektorining Connection xususiyatiga "con" bog'lanadi. Ob'yektlar inspektorining TaleName xususiyatiga " Dori turlari" jadvalni tanlaymiz. Active xususiyatiga True qiymat berganimizda bog'lagan " Dori turlari" jadvalimiz faol bo'ladi.

DataSource ni DataModule2 ga joylashtiramiz. Ob'yektlar inspektorining DataSet xususiyatiga " Dori turlari" tanlaymiz.

DBGrid ning DataSource xususiyatiga DataModule2.ds_Doriturlari qiymatni beramiz. Jadval birta Dori turlari ustinidan iborat. Bu jadval ustiniga yangi satir kiritishimiz, saqlab qo'yishimiz yoki o'chirishimiz mumkin(2.10- rasm).

Bosh menyuda joylashgan "Shtatlar" tanlanganda yangi form Dori darmonlar oynasi ochilsin va jadvalni ko'rsatish talabini qo'yamiz. Buning uchun yangi form yaratib Ob'yektlar inspektorining Caption xususiyatiga Shtatlar deb

kiritaniz. Ob’yektlar inspektorining Name xususiyatiga Shtatlar deb oynamiz nomiga moslab qo’yamiz.



2.10- chizma. Dori darmon oynasi.

FormStyle xususiyatiga fsMDIChild qiymatini beramiz. Endi Shtatlar oynasi Asosiy oynaning ichida yaraladi va Asosiy oyna xususiyatlari meros bo’lib o’tadi. Forma yopish tugmasi bosilganda yopilish uchun quydagi kodni kiritamiz:

```
unit Unit9;
interface
uses Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms, Dialogs, DBCtrls, Grids, DBGrids, ExtCtrls, StdCtrls, Mask;
type
Tshtatlar = class(TForm)
        Panel1: TPanel;
        Label1: TLabel;
        DBLookupComboBox1: TDBLookupComboBox;
        DBGrid1: TDBGrid;
        Panel2: TPanel;
Label2: TLabel;
        DBLookupComboBox2: TDBLookupComboBox;
        Label3: TLabel;
        DBEdit1: TDBEdit;
        procedure FormClose(Sender: TObject; var Action: TCloseAction);
private
```

```
{ Private declarations }
public
{ Public declarations }
end;
var
shtatlar: Tshtatlar;
implementation
uses Unit1, Unit2, Unit3, Unit4, Unit5, Unit6, Unit7, Unit8, Unit10;
{$R *.dfm}
procedure Tshtatlar.FormClose(Sender: TObject; var Action: TCloseAction);
begin
free;
end;
end.
```

Shtatlar oynasiga Markazlar jadvalini bog'laymiz sababi har bir markazning bo'lim mudiri, hodimlar shatlarini alohida kiritish kerak.

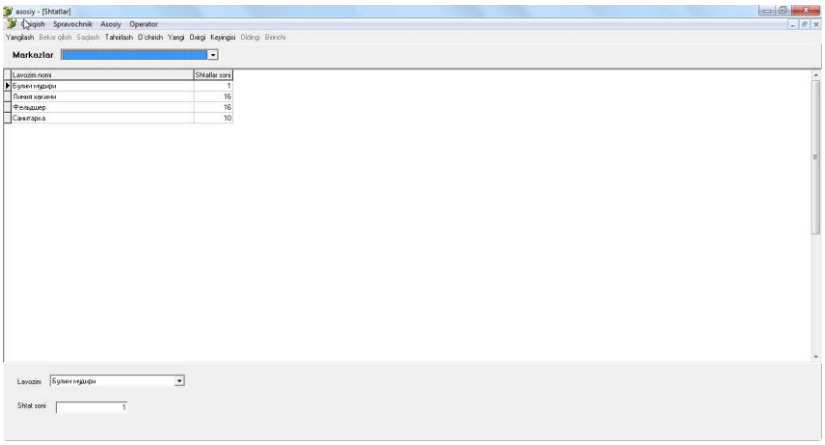
ADOTable ni DataModule2 ga joylashtiramiz.

ADO Table Ob'yektlar inspektorining Connection xususiyatiga "con" bog'lanadi. Ob'yektlar inspektorining TaleName xususiyatiga " v_shtatlar" jadvalni tanlaymiz. Active xususiyatiga True qiymat berganimizda bog'lagan " Statlar" jadvalimiz faol bo'ladi.

Yana bir ADOTable ni DataModule2 ga joylashtiramiz. ADO Table Ob'yektlar inspektorining Connection xususiyatiga "con" bog'lanadi. Ob'yektlar inspektorining TaleName xususiyatiga " markazlar" jadvalni tanlaymiz. Active xususiyatiga True qiymat berganimizda bog'lagan " markazlar" jadvalimiz faol bo'ladi. Shtatlar ADO Table ning MasterSource xususiyati orqali markazlar ADOTable bilan bog'lanadi. MasterFields xususiyatiga Shtatlar jadvalining "markaz" ustuni bilan Mrkazlar jadvalining "id" ustuni bog'lab Add tug'masini bosamiz va OK tugmasi bosiladi.

DataSource ni DataModule2 ga joylashtiramiz. Ob’yektlar inspektorining DataSet xususiyatiga ”shtatlar” tanlaymiz.

DBGrid ning DataSource xususiyatiga DataModule2.ds_shtatlar kiritamiz. Jadval ikkita Lavozimlar, shtatlar soni ustinidan iborat. Bu jadval ustiniga yangi satir kiritishimiz, saqlab qo’yishimiz yoki o’chirishimiz mumkin. Buning uchun DBLookupComboBox1 ga lavozimlar jadvali bog’langan, DBEdit ga kiritgan qiymatimiz Shtatlar jadvalining shatlar soni ustiniga bog’langan. Markaz tanlanganda Shtatlar jadvali tanlangan markazga qanday lavozimlar va nechta shtatlari borligini ko’rsatadi (2.11- chizma).



2.11- chizma. Shtatlar oynasi.

Bosh menyuda joylashgan ”Markazlar” tanlanganda yangi form Markazlar oynasi ochilsin va jadvalni ko’rsatish talabini qo’yamiz. Buning uchun yangi form yaratib Ob’yektlar inspektorining Caption xususiyatiga Markazlar deb kiritaniz. Ob’yektlar inspektorining Name xususiyatiga Markazlar deb oynamiz nomiga moslab qo’yamiz. FormStyle xususiyatiga fsMDIChild qiymatini beramiz. Endi Markazlar oynasi Asosiy oynaning ichida yaraladi va Asosiy oyna xususiyatlari meros bo’lib o’tadi. Forma yopish tugmasi bosilganda yopilish uchun quydagi kodni kiritamiz:

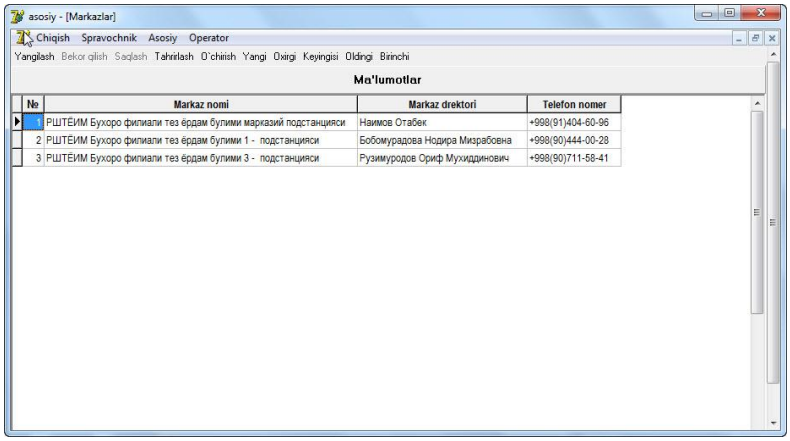
```
unit Unit12;
interface
```

```
uses Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls,
Forms, Dialogs, Grids, DBGrids, ExtCtrls;
type
TMarkazlar = class(TForm)
 Panel1: TPanel;
DBGrid1: TDBGrid;
procedure FormClose(Sender: TObject; var Action: TCloseAction);
private
 { Private declarations }
public
 { Public declarations }
end;
var
Markazlar: TMarkazlar;
implementation
uses Unit1, Unit10, Unit11, Unit2, Unit3, Unit4, Unit5, Unit6, Unit7,
Unit8, Unit9;
{$R *.dfm}
procedure TMarkazlar.FormClose(Sender: TObject; var Action:
TCloseAction);
begin
free;
end;
end.
```

Endi ishni ADOTable ni DataModule2 ga joylashtirishdan boshlamaymiz negaki Markazlar jadvalimizni con ga bog'lab, DataSource ni yartganmiz.

DBGrid ning DataSource xususiyatiga DataModule2.ds_markazlar kitamiz .Jadval uchta Markaz nomi, Markaz direktori, Telefon nomeri ustinidan iborat. Bu jadval ustiniga yangi satir kiritishimiz, saqlab qo'yishimiz yoki o'chirishimiz mumkin(2.12- chizma).

Bosh menyuda joylashgan ” Kasalxonalar” tanlanganda yangi form Kasalxonalar oynasi ochilsin va jadvalni ko’rsatish talabini qo’yamiz. Buning uchun yangi form yaratib Ob’yektlar inspektorining Caption xususiyatiga Kasalxonalar deb kiritaniz.



2.12-chizma. Markazlar oynasi.

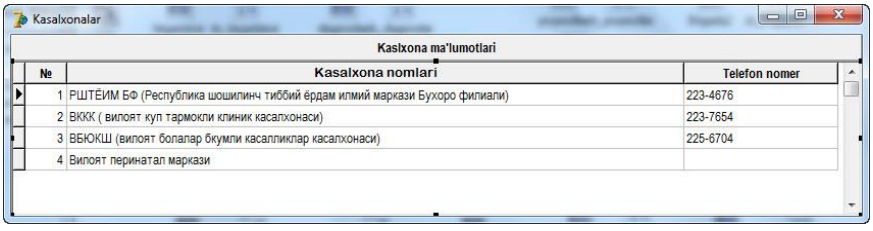
Ob’yektlar inspektorining Name xususiyatiga Kasalxonalar deb oynamiz nomiga moslab qo’yamiz. FormStyle xususiyatiga fsMDIChild qiymatini beramiz. Endi Kasalxonalar oynasi Asosiy oynaning ichida yaraladi va Asosiy oyna xususiyatlari meros bo’lib o’tadi. Form yopish tugmasi bosilganda yopilish uchun quydagi kodni kiritamiz:

```
unit Unit13;
interface
uses Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms, Dialogs, Grids, DBGrids, ExtCtrls;
type
Tkaslxonalar = class(TForm)
Panel1: TPanel;
 DBGrid1: TDBGrid;
 procedure FormClose(Sender: TObject; var Action: TCloseAction);
private
{ Private declarations }
public
 { Public declarations }
```

```
end;
var
kaslxonalar: Tkaslxonalar;
implementation
uses Unit1, Unit10, Unit11, Unit12, Unit2, Unit3, Unit4, Unit5, Unit6, Unit7, Unit8, Unit9;
{$R *.dfm}
procedure Tkaslxonalar.FormClose(Sender: TObject;
var Action: TCloseAction);
begin
free;
end;
end.
```

ADOTable ni DataModule2 ga joylashtiramiz. ADO Table Ob'yektlar inspektorining Connection xususiyatiga "con" bog'lanadi. Ob'yektlar inspektorining TaleName xususiyatiga " Kaslxonalar" jadvalni tanlaymiz. Active xususiyatiga True qiymat berganimizda bog'lagan " Kaslxonalar" jadvalimiz faol bo'ladi. DataSource ni DataModule2 ga joylashtiramiz. Ob'yektlar inspektorining DataSet xususiyatiga Kasalxonalar" tanlaymiz.

DBGrid ning DataSource xususiyatiga DataModule2.ds_Kasalxonalar kitamiz. Jadval uchta Kasalxona nomi, Telefon nomeri ustinidan iborat. Bu jadval ustiniga yangi satir kiritishimiz, saqlab qo'yishimiz yoki o'chirishimiz mumkin (2.13- chizma).



2.13- chizma. Kasalxonalar oynasi.

Bosh menyuda joylashgan ”Apteka” tanlanganda yangi form Apteka oynasi ochilsin va jadvalni ko’rsatish talabini qo’yamiz. Buning uchun yangi form yaratib Ob’yektlar inspektorining Caption xususiyatiga Apteka deb kiritaniz. Ob’yektlar inspektorining Name xususiyatiga Apteka deb oynamiz nomiga moslab qo’yamiz. FormStyle xususiyatiga fsMDIChild qiymatini beramiz. Endi Apteka oynasi Asosiy oynaning ichida yaraladi va Asosiy oyna xususiyatlari meros bo’lib o’tadi. Forma yopish tugmasi bosilganda yopilish uchun quydagi kodni kiritamiz:

```
unit Unit7;
interface
uses Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms, Dialogs, Grids, DBGrids, ExtCtrls, StdCtrls, Mask, DBCtrls;
type
TApteka = class(TForm)
 Panel1: TPanel;
 DBGrid1: TDBGrid;
 Label1: TLabel;
 DBLookupComboBox1: TDBLookupComboBox;
 Label2: TLabel;
 DBLookupComboBox2: TDBLookupComboBox;
 Label3: TLabel;
 DBEdit1: TDBEdit;
 Label4: TLabel;
 DBEdit2: TDBEdit;
 Label5: TLabel;
 DBEdit3: TDBEdit;
 Label6: TLabel;
 DBEdit4: TDBEdit;
 Label7: TLabel;
 DBLookupComboBox3: TDBLookupComboBox;
 procedure FormClose(Sender: TObject; var Action: TCloseAction);
```

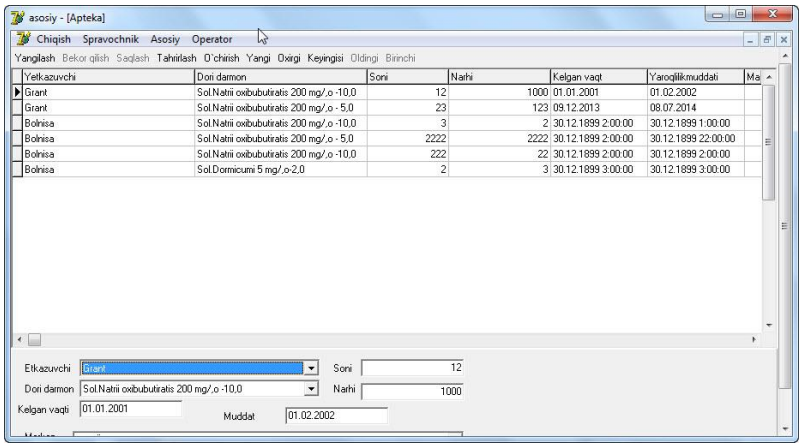
```
 private
 { Private declarations }
 public
 { Public declarations }
 end;
var
Apteka: TApteka;
implementation
uses Unit1, Unit2, Unit3, Unit4, Unit5, Unit6;
{$R *.dfm}
procedure TApteka.FormClose(Sender: TObject; var Action:TCloseAction);
begin
free;
end;
end.
```

ADOTable ni DataModule2 ga joylashtiramiz. ADO Table Ob'yektlar inspektorining Connection xususiyatiga "con" bog'lanadi. Ob'yektlar inspektorining TaleName xususiyatiga " vdorilar" jadvalni tanlaymiz. Active xususiyatiga True qiymat berganimizda bog'lagan " Dorilar" jadvalimiz faol bo'ladi. Yana bir ADOTable ni DataModule2 ga joylashtiramiz. ADO Table Ob'yektlar inspektorining Connection xususiyatiga "con" bog'lanadi. Ob'yektlar inspektorining TaleName xususiyatiga " doriyetkazuvchilar" jadvalni tanlaymiz. Active xususiyatiga True qiymat berganimizda bog'lagan " dori yetkazuvchilar" jadvalimiz faol bo'ladi.

DataSource ni DataModule2 ga joylashtiramiz. Ob'yektlar inspektorining DataSet xususiyatiga " doriyetkazuvchilar" tanlaymiz.

DBGrid ning DataSource xususiyatiga DataModule2.ds_Dorilar kiritamiz. Jadval yettita Yetkazuvchi, Dori darmon, Soni, Narhi, Kelgan vaqt, Yaroqlilikmuddati, Markaz ustinlari bor. Bu jadval ustiniga yangi satir kiritishimiz, saqlab qo'yishimiz yoki o'chirishimiz mumkin. Buning uchun

DBLookupComboBox1 ga doriyetkazuvchilar jadvali bog’langan, DBEdit larga kiritgan qiymatimiz Dorilar jadvalining ustunlariga bog’langan (2.14- chizma).



2.14- chizma. Apteka oynasi.

2.2. Operator.

Sozlovchi qismidan alohida ob’yekt Operator qismini yaratamiz. Operator karta va tadbir kartasini to’ldiradi.

```
program Operator;
uses
Forms,
Unit1 in 'Unit1.pas' {Form1},
Unit2 in 'Unit2.pas' {DataModule2: TDataModule},
Unit3 in 'Unit3.pas' {Form3},
Unit4 in 'Unit4.pas' {Form4};
{$R *.res}
Begin
Application.Initialize;
Application.CreateForm(TForm1, Form1);
Application.CreateForm(TDataModule2, DataModule2);
Application.CreateForm(TForm4, Form4);
Application.Run;
end.
```

Operator qismining birinchi oynasida operator tizimga kiriga servirdan ruxsat so’raladi. Tizimga kirishda porol va login bilan kiriladi. Operatot qo’ngiroq bo’lganda ”karta” tugmasi bosilganidan Kartaga ma’lumot -larini to’ldirib bo’lguniga qadar audio zapisga yoziladi va vaqtni servirda olib soatni bazaga saqlab qo’yadi. Shunday imkoniyatlarni bergan dastur kodi;

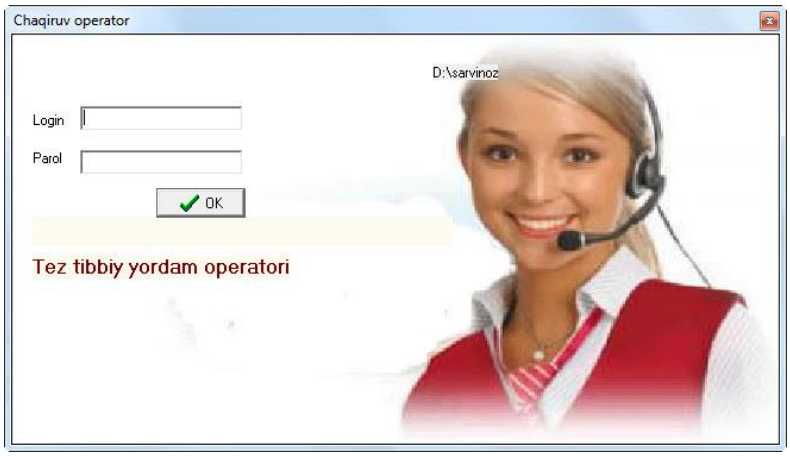
```
unit Unit1;
interface
uses Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms, Dialogs, StdCtrls, Buttons, ExtCtrls, ComCtrls, jpeg, DBCtrls;
type
TForm1 = class(TForm)
ProgressBar1: TProgressBar;
Timer1: TTimer;
Image1: TImage;
Label1: TLabel;
Edit1: TEdit;
Label2: TLabel;
Edit2: TEdit;
BitBtn1: TBitBtn;
Button1: TButton;
Button2: TButton;
Timer2: TTimer;
DBText1: TDBText;
Label3: TLabel;
zapisser: TLabel;
procedure BitBtn1Click(Sender: TObject);
procedure Timer1Timer(Sender: TObject);
procedure Button1Click(Sender: TObject);
procedure Button2Click(Sender: TObject);
procedure Timer2Timer(Sender: TObject);
```

```
procedure FormCreate(Sender: TObject);
procedure Button3Click(Sender: TObject);
private
{ Private declarations }
Public
{ Public declarations }
end;
var
Form1: TForm1;
implementation
uses Unit2, Unit3, Math, Unit4;
{$R *.dfm}
procedure TForm1.BitBtn1Click(Sender: TObject);
begin
ProgressBar1.Position:=0;
ProgressBar1.Visible:=true;
If (Edit1.Text<>'') and (Edit2.Text<>'')  then
begin
DataModule2.ADOStoredProc1.Close;
DataModule2.ADOStoredProc1.Parameters.ParamByName('@login').Value:
=Edit1.Text;
DataModule2.ADOStoredProc1.Parameters.ParamByName('@parol').Value:
=Edit2.Text;
DataModule2.ADOStoredProc1.Open();
Timer1.Enabled:=True;
end;
end;
procedure TForm1.Timer1Timer(Sender: TObject);
begin
If ProgressBar1.Position<100 then
```

ProgressBar1.Position:=ProgressBar1.Position+4

else

begin



2.15- chizma Chaqiruv operator oynasi.

Timer1.Enabled:=False;

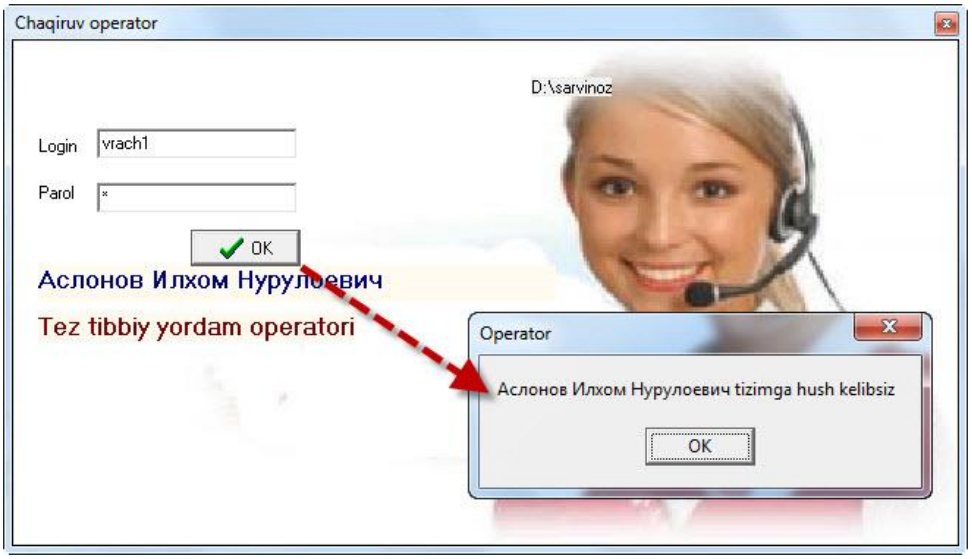
ProgressBar1.Visible:=False;

If DataModule2.ADOStoredProc1.RecordCount>0 then

begin

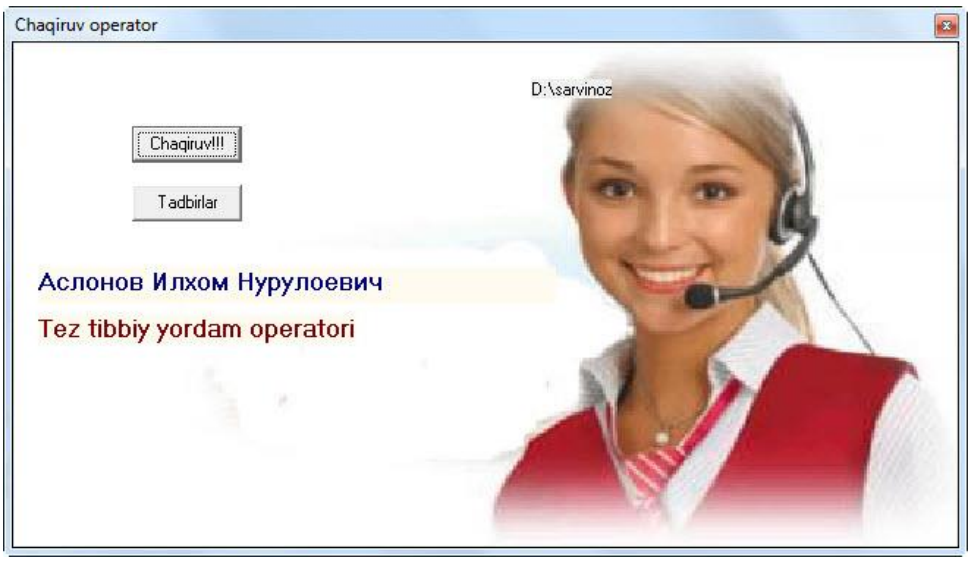
Timer2.Enabled:=True;

ShowMessage(DataModule2.ADOStoredProc1fio.Value+ ' tizimga hush kelibsiz');



2.16- chizma. Muloqot oynasi.

```
Edit1.Visible:=False;
Edit2.Visible:=False;
Label1.Visible:=False;
Label2.Visible:=False;
BitBtn1.Visible:=False;
Button1.Visible:=True;
Button2.Visible:=True;
End;
Else
ShowMessage('Iltimos parol va loginni to`gri kriting');
end;
end;
procedure TForm1.Button1Click(Sender: TObject);
begin
DataModule2.serversoat.Close;
DataModule2.serversoat.Open;
DataModule2.karta.Insert;
Application.CreateForm(TForm3, Form3);
Form3.Show;
Form3.DBEdit1.Text:=DataModule2.serversoatservervaqt.AsString;
end;
procedure TForm1.Button2Click(Sender: TObject);
begin
Form4.Show;
end;
procedure TForm1.Timer2Timer(Sender: TObject);
begin
DataModule2.aktiv_tadbir.Close;
DataModule2.aktiv_tadbir.Open;
```



2.17- chizma. Operator tizimga kirish qismi.

```
if DataModule2.aktiv_tadbir.RecordCount>0 then
begin
Form4.Show;
end;
end;
procedure TForm1.FormCreate(Sender: TObject);
var
f:TextFile;
s:String;
begin
AssignFile(f,'zapis.txt');
Reset(f);
readln(f,s);
zapisser.Caption:=s;
CloseFile(f);
end;
procedure TForm1.Button3Click(Sender: TObject);
begin
DataModule2.serversoat.Close;
```

DataModule2.serversoat.Open;

DataModule2.karta.Insert;

Form3.DBEdit1.Text:=DataModule2.serversoatservervaqt.AsString;

Form3.Show;

end;

end.

Operator Karta to’ldirish uchun ”Karta” tumasi bosilganda Form 3 ochiladi.

unit Unit3;

interface

uses Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms, Dialogs, ExtCtrls, StdCtrls, Buttons, DBCtrls, Mask, MPlayer, MMSystem;

type

TForm3 = class(TForm)

Panel2: TPanel;

Label1: TLabel;

Label2: TLabel;

Label3: TLabel;

Label4: TLabel;

Label5: TLabel;

Label6: TLabel;

Label7: TLabel;

Label8: TLabel;

Label9: TLabel;

Label10: TLabel;

Label13: TLabel;

Label27: TLabel;

DBEdit1: TDBEdit;

DBLookupComboBox1: TDBLookupComboBox;

DBEdit2: TDBEdit;

DBEdit3: TDBEdit;

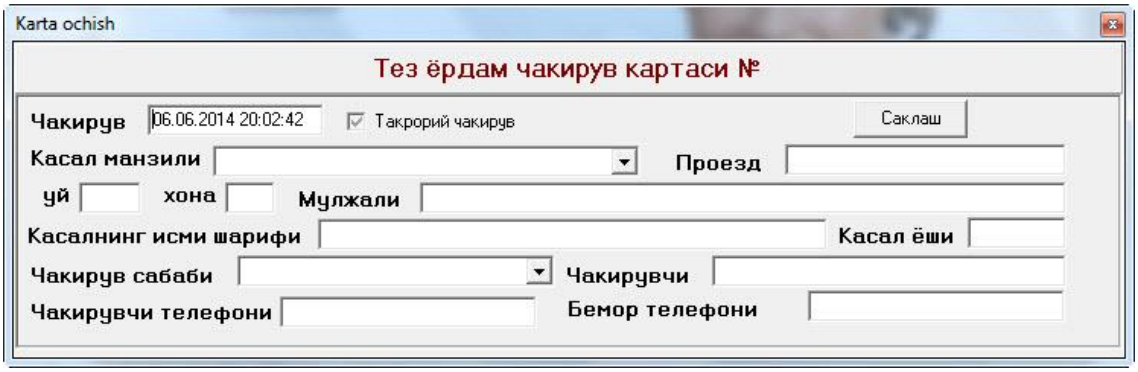
```
DBEdit4: TDBEdit;
DBEdit5: TDBEdit;
DBEdit6: TDBEdit;
DBLookupComboBox2: TDBLookupComboBox;
DBEdit7: TDBEdit;
DBEdit8: TDBEdit;
DBEdit9: TDBEdit;
Button3: TButton;
DBEdit21: TDBEdit;
DBCheckBox1: TDBCheckBox;
Panel1: TPanel;
BCheckBox2: TDBCheckBox;
procedure Button3Click(Sender: TObject);
procedure FormClose(Sender: TObject; var Action: TCloseAction);
procedure FormCreate(Sender: TObject);
procedure AppException(Sender: TObject; E: Exception);
private
FDeviceID: Word;
Handle: HWnd;
Public
end;
var
Form3: TForm3;
Implementation
uses Unit1, Unit2;
var
MyError, Flags: Longint;
procedure OpenMedia;
var
MyOpenParms: TMCI_Open_Parms;
```

```
        MyPChar: PChar;
        TextLen: Longint;
        Begin
        Flags := mci_Wait or mci_Open_Element or mci_Open_Type;
        with MyOpenParms do
        begin
        dwCallback :=form3.Handle; // TForm1.Handle
        lpstrDeviceType := PChar('WaveAudio');
        lpstrElementName := PChar('');
        end;
        MyError := mciSendCommand(0, mci_Open, Flags,
Longint(@MyOpenParms));
        if MyError = 0 then
        FORM3.FDeviceID := MyOpenParms.wDeviceID;
        end;
        procedure RecordMedia;
        var
        MyRecordParms: TMCI_Record_Parms;
        TextLen: Longint;
        Begin
        Flags := mci_Notify;
        with MyRecordParms do
        begin
        dwCallback := FORM3.Handle; // TForm1.Handle
        dwFrom := 0;
        dwTo := 10000;
        end;
        MyError := mciSendCommand(FORM3.FDeviceID, mci_Record, Flags,
Longint(@MyRecordParms));
        end;
```

```
procedure StopMedia;
var
MyGenParms: TMCI_Generic_Parms;
Begin
if FORM3.FDeviceID <> 0 then
begin
Flags := mci_Wait;
MyGenParms.dwCallback := form3.Handle; // TForm1.Handle
MyError := mciSendCommand(form3.FDeviceID, mci_Stop, Flags,
Longint(@MyGenParms));
end;
end;
procedure SaveMedia(f:String);
type // íå ðåàëèçîâàíî â Delphi
PMCI_Save_Parms = ^TMCI_Save_Parms;
TMCI_Save_Parms = record
dwCallback: DWord;
lpstrFileName: PAnsiChar;
end;
var
MySaveParms: TMCI_Save_Parms;
Begin
if form3.FDeviceID <> 0 then
begin
Flags := mci_Save_File or mci_Wait;
with MySaveParms do
begin
dwCallback := form3.Handle;
lpstrFileName := PChar(f);
end;
```

```
MyError := mciSendCommand(form3.FDeviceID, mci_Save, Flags,
Longint(@MySaveParms));
end;
end;
procedure CloseMedia;
var
MyGenParms: TMCI_Generic_Parms;
Begin
if form3.FDeviceID <> 0 then
begin
Flags := 0;
MyGenParms.dwCallback := form3.Handle; // TForm1.Handle
MyError := mciSendCommand(form3.FDeviceID, mci_Close, Flags,
Longint(@MyGenParms));
if MyError = 0 then
form3.FDeviceID := 0;
end;
end;
{$R *.dfm}
procedure TForm3.Button3Click(Sender: TObject);
var
s:string;
begin
DataModule2.kartaoperator.Value:=DataModule2.ADOStoredProc1id.Value
DataModule2.karta.Post;
s:=Form1.zapisser.Caption+DataModule2.kartaid.AsString+'.wav';
StopMedia;
SaveMedia(s);
CloseMedia;
end;
```

```
procedure TForm3.FormClose(Sender: TObject; var Action: TCloseAction);
begin
DataModule2.karta.Cancel;
StopMedia;
CloseMedia;
Free;
end;
procedure TForm3.FormCreate(Sender: TObject);
begin
Application.OnException := form3.AppException;
OpenMedia;
RecordMedia;
end;
procedure TForm3.AppException(Sender: TObject; E: Exception);
begin
StopMedia;
CloseMedia;
end;
end;
```

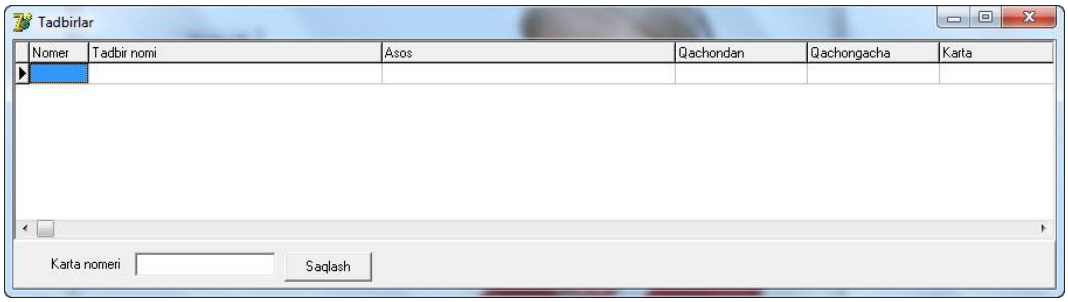


2.18- chizma. Operator chaqiruv kartasi.

Operatot tadbir kartasini to’ldirish uchun ”Tadbir” tugmasini bosganda Form 4 ochiladi.

```
uses Unit1, Unit2, Unit3;
{$R *.dfm}
```

```
procedure TForm4.Button1Click(Sender: TObject);
begin
DataModule2.aktiv_tadbir.Edit;
DataModule2.aktiv_tadbiroperator.Value:=DataModule2.ADOStoredProc1id.
Value;
DataModule2.aktiv_tadbir.Post;
DataModule2.aktiv_tadbir.Close;
DataModule2.aktiv_tadbir.Open;
ShowMessage('Ma`lumotlar saqlandi');
end;
procedure TForm4.FormShow(Sender: TObject);
begin
if DataModule2.aktiv_tadbir.RecordCount>0 then
MediaPlayer1.Play;
end;
end.
```



2.19- chizma. Tadbir oynasi.

Xulosa: Tez tibbiy yordami xizmatini qo’llab quvvatlovchi informatsion tizim dasturini yaratilishiga sabab shundaki tez tibbiy xizmatni faol xizmat ko’rsatish tezligini yanada oshirish buning natijasida ish unumdorligi ko’paytirish. Aniq ma’lumotlarga ega bo’lish. Sozlovchi qismidagi birgina hodimlar oynasi orqali Hodimlarning turar joyi, Ishga qabul qilinganligi to’grisidagi buyrug’i, ishga qabul qilingan sanasi, hissasi, pasport serya raqami, tulg’ulgan kuni, xodim haqida qo’shimcha ma’lumot, uy telefoni, malumotlar birta jadvalga jamlangan.

Xodim yangi kelsa yoki ayrim bir ma’lumotlarini o’zgartirish kerak bo’lsa bu judaham oson ma’lumotlar chisqa holda to’ldiriladi saqlab qoyiladi, o’chiriladi dastur shukabi qulayliklar va ko’pgina imkoniyatlarga ega.

Xotima.

Delphi dasturlash muhiti Borland kompaniyasi tomonidan ishlab chiqarilgan bo'lib, u Borland Pascal tilining keyingi versiyasi Borland Object Pascal tili asosida qurilgan. Ob'yektga mo'ljallangan dasturlash tilllarida dastur tuzish ancha oson va ishchi vaqt tejaladi. Dasturchi ob'yektlarni qurib o'tirmaydida tayyor ob'yektlarni qo'yib olaveradi va dastur asosiy qismiga bosh qotiradi xolos. Delphi kuchli xatolarni bartaraf qilish (Debuger) sistemasiga va qulay yordamchi sistemasiga egadir. Delphi juda qulay interfeysga ega.

Deiphi 7 da oldingi versiyalarida bo'lmagan Microsoft, Active, Data Objects bilan ishlash imkoniyati yaratildi. ADO – bu Microsoft ma'lumotlar ombori bilan ishlaydigan standart texnologiya. Bu texnologiya BDE bilan deyarli bir xil va imkoniyatlari ham yaqin.

Tez tibbiy yordami xizmatini qo'llab quvvatlovchi informatsion tizim dasturini yaratilishiga sabab shundaki tez tibbiy xizmatni faol xizmat ko'rsatish tezligini yanada oshirish buning natijasida ish unumdorligi ko'paytirish. Aniq ma'lumotlarga ega bo'lish. Sozlovchi qismidagi birgina hodimlar oynasi orqali Hodimlarning turar joyi, Ishga qabul qilinganligi to'grisidagi buyrug'i, ishga qabul qilingan sanasi, hissasi, pasport serya raqami, tulg'ulgan kuni, xodim haqida qo'shimcha ma'lumot, uy telefoni, malumotlar birta jadvalga jamlangan. Xodim yangi kelsa yoki ayrim bir ma'lumotlarini o'zgartirish kerak bo'lsa bu judaham oson ma'lumotlar chisqa holda to'ldiriladi saqlab qoyiladi, o'chiriladi dastur shukabi qulayliklar va ko'pgina imkoniyatlarga ega.

Adabiyotlar.

1. Абрамов В.Г., Трифонов Н. П., Трифона Г. Н. Введение в язык паскал. М. Наука, 1988- 320 с.

2. Гофман В.Э., Хомоненко А.Д. Deiphi 5.-СПб. БХВ-Санкт-Петербург 2000.-800с.

3. Пильшиков В.Н. Упражнения по язык паскаль.-М. МГУ 1986.

4. Поляков Д.Б. Круглов И.Ю. Программирование в среде Турбо-паскаль. (версия 5.5). М.МАИ 1992-576с.

5. Elov. B.B. Berilgan ma’lumotlar tuzulmasi. Buxoro. 2004y 88 bet.