O’ZBEKISTON RESPUBLIKASI

OLIY VA O’RTA-MAXSUS TA’LIM VAZIRLIGI

SAMARQAND DAVLAT UNIVERSITETI

MEXANIKA-MATEMATIKA FAKULTETI

AMALIY MATEMATIKA VA INFORMATIKA BO’LIMI

5110700-INFORMATIKA O’QITISH METODIKASI YO’NALISHI

„C++ TILIDA MATN MUXARRIRI YARATISH“

MAVZUSIDAN

KURS ISHI



206 guruh talabasi: Yusupov N

Kurs ishi rahbari: Nazarov F

SAMARQAND-2016

MUNDARIJA

Kirish;

I. C++ tilida matn muharrirlarini yaratishga doir nazariy tushunchalar:

1.1Turlar va C++ da o’zgaruvchilarni tavsiflash;

1.2.Malumotlar tipiva malumotlar tipiga keltirish;

II. C++ tilida yaratilgan matn muharririning dasturiy ta’minoti:

2.1. Ekran bilan matli rejimda ishlash;

2.2. Borland C++ da matn muharririni yaratish texnologiyasi va dasturiy ta’minoti;

Xulosa.

Foydalanilgan adabiyotlar ro’yhati.

Kirish.

Hozirgi kunda respublikamizdagi texnika oliy o‘quv yurtlarida “Informatikava axborot texnologiyalari” yo‘nalishi va mutaxassisliklariga turli xil dasturlashtillarini o‘rgatish mo‘ljallangan. Bizga ma’lumki, dasturlash tillarining yuzdan ortiq ko‘rinishlari mavjud, lekin qo‘llanilishi ko‘lamiga qarab C/C++ va C# dasturlash tillari yuqori dasturlash sinfiga mansubdir.Mutaxassislarning fikriga ko‘ra C++ dasturlash tili Assembler dasturlashtiliga eng yaqin bo‘lib, tezlik jihatidan 10 % ortda qolar ekan.Keyingi yillarda amaliy dasturchilarga juda ko‘p integratsion dastur tuzishmuhitlari taklif etilmoqda. Bu muhitlar u yoki bu imkoniyatlari bilan bir–biridanfarq qiladi. Aksariyat dasturlashtirish muhitlarining fundamental asosi C/C++ tiligaborib taqaladi.Ushbu kurs ishida hozirgi kunda kompyuterda berilgan funksiyalarnigrafigini va har xil ko‘rinishdagi shakllarni chizish va ularni aniq koordinatalarinianiqlash, ekranni grafik rejimga o‘tkazish va undagi mavjud piksel va ranglardanfoydalanish kabi vazifalarni o‘rganishga olib keladi.

Xozirgi vaqtga kelib komp`yuter olamida ko’plab dasturlash tillari mavjud. Paskal, C++ va boshqa dasturlash tillaridir. C++ dasturlash tili universal tildir. U UNIX sistemasi bilan bog`langan bo`lib, bu sistemada ishlatiladigan bir qancha dasturlar C++ tilida yozilgan. Paskal tili 1969 yil N. Virt tomonidan yaratilgan bo’lib, keyinchalik amerikaning Borland firmasi tomonidan qayta ishlandi va uni Turbo Pascal deb nomlangan. C++ Denis Ritchi tomonidan 1972 -yili UNIX tipidagi op erasion sistemalarini yaratish uchun loyihalashtirilgan.Turbo Pascal ni qayta ishlash natijasida ob'ektli dasturlash yo’lga qo’yildi va 1995 yilda Borland kompaniyasi guruxi dastur tuzuvchilari Chack va Denny tomonidan Windows uchun mo’ljallangan dasturlash muxiti Borland Delphi dasturlash vositasi yaratildi.Borland C++ va Delphi dasturlash tili Windows uchun mo’ljallangan bo’lib, uning birinchi versiyasi Windows operatsion sistema qobig’ida ishlagan. Borland C++ va Delphi dasturlash tili – bu dasturlarni qayta ishlash muxiti bo’lib, Windows operatsion sistemasida ishlaydi. Unda ob'ektli dasturlash tillari bo’lgan Object mujassamlashgan.Borland C++ va Delphi vizual proektlar, turli xolat

protseduralarini qayta ishlash va dasturlarni qayta ishlashda vaqtdan yutish va boshqalarni o’z ichiga oladi.

Men ham ushbu kurs ishim davomida C++ tilida matn muxarriri bilan ishlovchi obektlarga mo’ljallangan dasturni tuzib chiqmoqchiman. Ushbu dastur kichik dastur bo’lsada kelajakda boshqa dasturlar tuzishim uchun fundament vazifasini bajarishga harakat qilaman. Chunki hozirgi kunda matnlar bilan ishlash uchun katta dasturlar paketini o’rnatish, agar o’zida ishlaydigan dasturni ishlatsak o’zimizni sof tilda emas boshqa davlat tillarida ishlashi noqulayliliklar tugdiradi. Shuning uchun kichik matn muxarrirlari bilan ishlovchi dastur ishlab chiqaramiz

Bu dasturni tuzib nafaqat yozishni amalga oshirish balki kichik matnlar ustida taxrirlash va dasturni ishlashni bilmaydigan foydalanuvchi ham bir ko’rinishda tushinadigan tushinarli, bejirm ko’rinishga ega dastur tuzmoqchiman.

Men bu dasturni tuzib obektlar yaratishni va ular orqali shunga o’xshash yana boshqa obektlar yaratmoqchiman bu dasturda ishlashimga yana bir sabab boshqa obektlarga mo’ljallanmagan dasturlash tillarida faqat masalani matematik dasturi tuzib natija olinadi bunda yani C++ Builderda esa boshqa imkoniyatlar va kampanentalar bilan ham ishlashni o’rganmochimiz.

I. C++ tilida matn muharrirlarini yaratishga doir nazariy tushunchalar;

1.1Turlar va C++ da o’zgaruvchilarni tavsiflash

Har bir nom va har bir o’zgaruvchi ular ustida bajariluvchi amallar aniqlovchi turlarga ega bo’ladi. Masalan, int i; tavsiflash i o’zgaruvchi int turiga tegishli, ya’ni i butun o’zgaruvchi deb aniqlaydi. Tavsiflash - dasturga nom kirituvchi buyruqdir. Tavsiflash o’zgaruvchining turini aniqlaydi. Tur nom va ifodalardan to’g’ri foydalanishni aniqlaydi. Butun tur uchun quyidagi amallar aniqlangan: +, -, * va /.Asosiy turlar. Bevosita apparat ta’minotiga javob beradigan asosiy turlar quyidagilar: char; short; int; long; float; double. Birinchi to’rtta tur butun kattaliklarni, oxirgi ikkitasi suzuvchi nuqtali, ya’ni kasr sonlarni tasvirlash uchun ishlatiladi. char turidagi o’zgaruvchi mazkur kompyuterda belgilarni (odatda bayt) saqlash o’lchoviga ega, int turidagi o’zgaruvchi esa mazkur kompyuterdagi butun arifmetikaga mos o’lchovga ega (odatda so’z). Turlar bilan tasvirlangan butun sonlar diapazoni uning o’lchoviga bog’liq bo’ladi (uni sizeof buyrug’i yordamida hisoblash mumkin).C++ da o’lchovlar char turidagi kattaliklar o’lchovi birligida o’lchanadi. Asosiy turlar o’rtasidagi munosabatlarni quyidagicha yozish mumkin:

1 = sizeof(char) <= sizeof(short) <= sizeof(int) <= sizeof(long) = sizeof(float)

<= sizeof(double).

Umuman, asosiy turlar xususida yana boshqa narsalarni faraz qilish ma’nosiz. Xususan, ko’rsatgichlarni saqlash uchun butun tur etarli, degan xulosa barcha kompyuterlar uchun to’g’ri emas. Asosiy turlarga const so’zini qo’shib tavsiflash mumkin. Bu boshlang’ich turga shu turning o’zini beradi, faqat bu holatda const turidagi o’zgaruvchilarning qiymatlari initsializatsiyadan so’ng o’zgarishi mumkin emas.

```
const float pi = 3.14;
const char plus = '+';
```

Bittalik qo’shtirnoqqa olingan belgilar belgi o’zgarmaslar hisoblanadi.

Shunga e’tibor berish lozimki, bu usulda tavsiflangan o’zgarmaslar xotirada joy egallamaydi. uning qiymati talab qilingan joyda bevosita ishlatiladi. O’zgarmaslar initsializatsiya paytida tavsiflanishi shart. O’zgaruvchilar uchun initsializatsiya shartemas, ammo albatta tavsiya qilinadi. Lokal o’zgaruvchilarni initsializatsiyasiz kiritish asoslari juda ko’p.Bu turlarning ixtiyoriy kombinatsiyasiga quyidagi arifmetik amallar qo’llanilishi mumkin:

+ (plyus, unar va binar);

- (minus, unar va binar);

* (ko’paytirish);

/ (bo’lish).

Hamda taqqoslash amallari:

== (teng);

!= (teng emas);

< (kichik);

> (katta);

<= (kichik yoki teng);

>= (katta yoki teng).

Agar operandlar qo’yilgan shartni qanoatlantirsa , u holda taqqoslash amallari natijada 1 qiymatni beradi, aks holda esa 0 qiymatni beradi.Butunga bo’lish amali butun natijani beradi: 7/2 = 3. Butun kattaliklar ustida % - qoldiqni hisoblash amali bajariladi: 7%2 = 1.O’zlashtirishda va arifmetik amallarda C++ ularni guruhlash uchun asosiy turlar o’rtasida barcha ma’noli almashtirishlarni bajaradi:

```
double d = 1;
int i = 1;
d = d + i;
i = d + i;
```

Satriy turlar. C++ da belgilarning biron-bir ketma-ketligi (massivlar) dan iborat matn qatorlarini xotirada saqlash uchun maxsus AnsiString ma’lumotlar turi qo’llaniladi.

«Stroka» - «Satr» turidagi o’zgaruvchilar barcha boshqa o’zgaruvchilar kabi e’lon va initsializatsiya qilinadi.Kompilyatorga navbatdagi belgilar ketma-ketligi yangi o’zgaruvchining nomi emas, balki satr eka nnligini bildirish uchun satrlar bittalik qo’shtirnoq ichiga olinadi.Misol:

AnsiString st = ‘matn qatori’;

Satr turidagi o’zgaruvchilar ustida boshqa satr o’zgaruvchilar bilan qo’shish amali bajarilishi mumkin. Bu amal ikkita satrni ularning kelish tartibida birlashtirish deb tushuniladi.Misol:

AnsiString s1 = ‘qatori’;

AnsiString s2 = ‘ matn’;

AnsiString s = s2 + s1;

Natijada s o’zgaruvchi s1 va s2 o’zgaruvchilardan tashkil topgan ,matn qatori’ degan qiymatni qabul qiladi.Qo’shimcha turlarBorland C++ da butun qiymatli o’zgaruvchilarning turlarini qo’shimcha ajratish imkoni mavjud. Bu holda o’zgaruvchilarning barcha tur nomlari quyidagicha yoziladi - int X, bu erda X o’zgaruvchiining bitlardagi maydon o’lchami. X quyidagi qiymatlardan birini qabul qilishi mumkin: 8, 16, 32 va 64. Bu turdagi o’zgaruvchilardan foydalanish standart turda aniqlangan o’zgaruvchilardan foydalanishdan farq qilmaydi.

Quyidagi jadvalda bunday turlar bilan ishlash yaqqol ko’rsatilgan.

Tur nomi misol O’lcham

int8 c = 128; 8 bit

int16 s = 32767; 16 bit

int32 i = 123456789; 32 bit

int64 big = 12345654321; 64 bit

111Turlarni o’zgartirish protseduralari. Standart turlarni o’zgartirishC++ ning ma’lumotlarning turlari ustida qattiq nazorati tufayli imkoni boricha qiymatlarni saqlovchi, turlarni o’zgartirish amallari kiritilgan.Boshqa o’zgaruvchidan ma’lum bir tur qiymatlarini olish uchun quyidagi konstruksiya ishlatiladi: (yangi tur)o’zgaruvchi.Misol:

short S = 100;

```
int I = (int)S;
```

Bu misol ortiqcha buyruqlarga ega. C++ da ko’pgina tur o’zgaruvchilarining to’g’ridan-to’g’ri o’zlashtirilishi nazarda tutilgan, ammo ba’zi hollarda bu buyruqlar majburiy hisoblanadi (masalan, o’zgaruvchining qiymatini biror funksiyaga uzatishda).

Sonli qiymatlarni satrga almashtirish. C++ turlarning to’g’ridan-to’g’ri almashtirishda o’zgaruvchini uning o’nlik ko’rinishidan belgilar qatori ko’rinishiga yo’l qo’ymaydi, chunonchi, ular shakllarning ko’gina komponentalarida ishlatiladi. To’g’ridan-to’g’ri almashtirish faqatgina asosiy va qo’shimcha turlar uchun amalga oshiriladi. Massiv hisoblanadigan satr kattaliklar hosilaviy tur bo’lganligi sababli bunday almashtirishga yo’l qo’yilmaydi.Bunday almashtirishlar uchun quyidagi standart almashtirish funksiyalari ishlatiladi: IntToStr, StrToInt, FloatToStr va boshqalar. Ko’pchilik ma’lumotlar turlari uchun shu kabi satrga va teskari o’tkazish funksiyalari mavjud.Misol:

```
char S[10];  // belgilar massivi
int I = 100;  // butun qiymatli o’zgaruvchi
S = IntToStr(I);  // o’tkazish
```

Shartli buyruq. Dasturda tarmoqlanishni amalga oshirish, ya’ni ba’zi faktorlarga bog’liq holda turli amallar bajarilishi uchun if buyrug’i ishlatiladi.Buyruq quyidagi formatga ega:

```
if (ifoda){ 1 - operator;} [else { 2 - operator;}]
```

if-buyrug’ining bajarilishi ifodaning qiymatini hisoblashdan boshlanadi. So’ngra ish quyidagi sxema asosida amalga oshiriladi:

agar ifoda rost bo’lsa (ya’ni 0 dan farqli), u holda 1 - operator bajariladi.

agar ifoda yolg’on bo’lsa (ya’ni 0 ga teng), u holda 2 - operator bajariladi.

agar ifoda yolg’on va 2 - operator yo’q bo’lsa (kvadrat qavsga zarur bo’lmagan konstruktsiya kiritiladi), u holda if dan keyingi buyruq bajariladi. Misol:

```
if (i < j)
{
i++;
}
```

```
else
{
j = i-3;
i++;
}
```

Bu misol 1 - operatorning o’rnida ham, 2 - operatorning o’rnida ham murakkab konstruktsiya qatnashishi mumkinligini bildiradi. Ichma-ich if buyrug’ini ishlatish imkoniyati ham mavjud. if buyrug’i boshqa if buyrug’ining if yoki else konstruktsiyalari ichida qatnashishi ham mumkin.

1.2. Malumotlar tipi va malumotlar tipiga keltirish.

Shu paytgacha ma'lumotlar tipi deganda butun son va kasrli son bor deb kelgan edik. Lekin bu bo'limda maylumotlar tipi tushunchasini yahshiroq ko'rib chiqish kerak bo'ladi. Chunki funksiyalar bilan ishlaganda argument kiritish va qiymat qaytarishga to'g'ri keladi. Agar boshidan boshlaydigan bo'lsak, kompyterda hamma turdagi ma'lumotlar 0 va 1 yordamida kodlanadi. Buning sababi shuki, elektr uskunalar uchun ikki holat tabiyi-dir, tok oqimi bor yoki yo'q, kondensatorda zaryad bor yoki yo'q va hakozo. Demak biz bu holatlarni oladigan jihozlarni bir quti deb faraz qilsak, quti ichida yo narsa bo'ladi, yo narsa bo'lmaydi. Mantiqan buni biz bir yoki nol deb belgilaymiz. Bu kabi faqat ikki holatga ega bo'lishi mumkin bo'lgan maylumot birligiga biz BIT deymiz. Bu birlik kichik bo'lgani uchun kompyuterda bitlar guruhi qo'llaniladi. Bittan keyingi birlik bu BAYT (byte). Baytni sakkizta bit tashkil etadi. Demak bir bayt yordamida biz 256 ta holatni kodlashimiz mumkin bo'ladi. 256 soni ikkining sakkizinchi darajasiga tengdir. Bitimiz ikki holatga ega bo'lgani uchun biz kompyuterni ikkili arifmetikaga asoslangan deymiz. Ammo agar kerak bo'lsa, boshqa sistemaga asoslangan mashinalarni ham qo'llash mumkin. Masalan uchli sanoq sistemasiga asoslangan kompyuterlar bor. Informatika faniga ko'ra esa, hisoblash mashinasi uchun eng optimal sanoq sistemasi e ga teng bo'lar ekan. Demak amaldagi sistemalar ham shu songa iloji borisha yaqin bo'lishi kerakdir. C/C++ da baytga asoslangan tip char dir. char tipi butun son tipida bo'lib, chegaraviy qiymatlari -128 dan +127 gachadir. Bu tip lotin alifbosi harflarini va y ana qo'shimcha bir guruh simvollarni kodlashga qulay bo'lgan. Lekin hozirda milliy alifbelarni kodlash uchun 16 bitlik UNICODE qo'llanilmoqda. U yordamida 65536 ta simvolni ko'rsatish mumkin. char tipida o'zgaruvchi e'lon qilish uchun dasturda

char g, h = 3, s;

kabi yozish kerak. O'zgaruvchilar vergul bilan ayriladi. E'lon bilan bir vaqtning o'zida boshlang'ich qiymat ham berish imkoni bor. Mashina ichida baytdan tashkil topgan boshqa kattaliklar ham bor. Ikki baytdan tuzilgan kattalik so'z (word) deyiladi, unda 16 bit bo'ladi. 4 ta bayt guruhi esa ikkili so'z (double word) bo'ladi. Bu birlik 32 bitli mashimalarda qo'llaniladi. Hozirda qo'llanilmoqda bo'lgan mashinalar asosan 32 bitlidir, masalan Pentium I/II/III sistemalari. C++ da butun sonlarning ikki tipi bor. Biri char - uni ko'rib chiqdik. Ikkinchisi int dir. Mashinalarning arhitekturasi qanday kattalikda bo'lsa, int tipining ham kattakigi huddi shunday bo'ladi. 16 bitlik mashinalarda int 16 bit edi. Hozirda esa int ning uzunligi 32 bitdir. int (integer - butun son) tipi charga o'hshaydi. Farqi bir baytdan kattaligidadir. 16 bitli int ning sig'imi -32768 dan +32767 gachadir. 32 bitli int esa -2 147 483 648 dan +2 147 483 647 gacha o'rin egallaydi. Bu ikki butun son tipidan tashqari C++ da ikki tur vergulli, (nuqtali) yani haqiqiy son tipi

mavjud. Bulardan biri float, hotirada 4 bayt joy egallaydi. Ikkinchisi esa double, 8 baytkattalikka ega. Bularning harakteristikalari quyidagi jadvalda berilgan. Ushbu tiplar bilan ishlaganda unsigned(ishorasiz, +/- siz), signed (ishorali) long (uzun) va short (qisqa) sifatlarini qo'llasa bo'ladi. unsigned va signed ni faqat butun son tiplari bilan qo'llasa bo'ladi. unsigned qo'llanganda sonning ishorat biti bo'lmaydi, ishorat biti sonning kattaligini bildirish uchun qo'llaniladi. Masalan char tipida 8 chi, eng katta bir odatda ishorat bitidir. Biz unsigned char ch; desak, ch o'zgaruvchimizga faqat 0 va musbat qiymatlarni berishimiz mumkin. Lekin oddiy char [-128;127] ichida bo'lsa, unsigned char [0;255] orasidagi qiymatlarni oladi, chunki biz ishorat biti ham qo'llamoqdamiz. Huddi shunday unsigned int da (4 baytli) qiymatlar [0;4 294 467 296] orasida yotadi.

signed ni ishlatib esa biz ochiqchasiga butun sonimizning ishorati bo'lishi kerakligini bildiramiz. Normalda agar signed yoki unsigned qo'yilmasa, tipimizning ishorasi bo'ladi. **long int** bilan qo'llanilganda 16 bitli int 32 ga aylanadi. Bu agar mashina 16 bitli bo'lsa, mashina 32 bitli arhitekturaga ega bo'lsa, int ning kattaligi 4 bayligicha qolaveradi. long double tipi esa 10 bayt joy oladi. Short sifati int bilan qo'llanilganda 32 bit emas, 16 bit joy egallashga boshlaydi. Tezlikni oshirish maqsadida kam joy egallaydigan ma'lumot tiplarini qo'llash maqsadga muofiqdir. Agar tipning nomi yozilmagan bo'lsa, o'zgaruvchi int tipiga ega deb qabul qilinadi.

Ma'lumot tiplarining nomlari	Sinonimlar	Keng tarqalgan harakteristikalari
long double		10 bayt, +/-3.4e-4932...+/-3.4e4932
double		8 bayt, +/-1.7e-308...+/-1.7e308
float		4 bayt, +/-3.4e-38...+/-3.4e38
unsigned long int	unsigned long	
long int	long	
unsigned int	unsigned int	
unsigned short int	unsigned short	
short int	short	

unsigned char

short

char

char va **int** dan tashqari C++ da yana bir necha integral tiplar mavjud. Bulardan biri bool tipidir. bool tipi faqat ikki farqli qiymat olishi mumkin. Bittasi true (to'g'ri) ikkinchisi false (noto'g'ri). bool tipi mantiqiy arifmetika amallarini bajarganda juda qo'l keladi. bool tipi boshqa bir integral tipga asoslangan bo'lishiga qaramasdan (int yoki char), yuqoridagi ikki qiymatdan tashqari boshqa qiymat ololmaydi. bool tipi o'zgaruvchilari to'g'ri shaklda initsalizatsiya qilinmagan taqdirda, ularning qiymati hato ravishda na true va na false bo'lishi mumkin.Yana boshqa bir integral tip bu **wchar_t** dir (wide char type - keng simvol tipi). U ham ko'pincha boshqa bir butun son tipiga asoslanadi - bir baytdan kattaroq bo'lishi kerakligi uchun short int qo'llaniladi.**wchar_t** simvollar kodlanishida qo'llaniladi. Masalan C++ da UNICODE ni odatda wchar_t bilan kodlaymiz. Hozirda wchar_t ning kattaligi 16 bit, lekin yuqori kattaligi necha bit bo'lishi kerakligi standartda belgilanmagan. Butun sonlarni C++ da bir necha asosda berish mumkin. Hech qanday belgi qo'yilmasdan yozilgan son o'nlik asosda (decimal) deb qabul qilinadi.

Sakkizli asosdagi (octal) sonni berish uchun sondan oldin 0o yoki 0O belgilarini qo'yish kerak.O'n oltilik sistemadagi (hexadecimal) sonlar oldiga 0x yoki 0X lar yoziladi. Sakkizli sistemada qo'llaniladin raqamlar to'plami 0,1,2,3,4,5,6 va 7 dir. O'n oltilik asosda 0 dan 9 gacha sonlar, 10 - a, 11 - b, 12 - c, 13 - d, 14 - e va 15 uchun f qo'llaniladi. Harflar katta bo'lishi ham mumkin. Harflarning registorining (katta-kichikligi) farqi yo'q. Misol beraylik:

```
char d = 10, j = 0o11; // d 10 ga teng, j 9 ga teng.

int f = 0X10;          // f 16 ga teng
```

Butun son va kasr son tiplaridan tashqari C++ da void (bo'sh, hech narsa) tipi ham mavjud. Bu tipning oladigan qiymatlari bo'sh to'plamga tengdir. Void tipidagi ma'lumot chala tugallangan hisoblanadi. Boshqa turdagi ma'lumotni void ga keltirish mumkindir. Bu tip bilan ishlashni dasturlarimizda ko'rib chiqamiz.

MA'LUMOTLAR TIPINI KELTIRISH (DATA CASTING)

Gohida bir turdagi o'zgaruvchining qiymatini boshqa tipdagi o'zgaruvchiga berish kerak bo'ladi. Bu amal ma'lumot tipini keltirish (data type casting) deyiladi. Ko'p hollarda bu amal avtomatik ravishda, kompilyator tarafidan bajariladi. Masalan ushbu parchani ko'raylik:

```
char c = 33;
int k;
k = c;
```

Bu yerda k ning sig'imi c nikidan kattaroqdir. Shuning uchun c ning qiymatini k ga berishda hech qanday muammo paydo bo'lmaydi. Quyidagi misolni ko'raylik:

```
int i = 5;
float f = 9.77;
float result;
result = f + i;
```

C++ ning kompilyatori ikki turdagi o'zgaruvchilar bilan ishlay olmaydi. Shu sababli ifodadagi sig'imi kichik bo'lgan o'zgaruvchilar ifodadagi qatnashgan eng katta sig'imga o'tqaziladi. Bu ham avtomatik tarzda bajariladi. i o'zgaruvchimiz qiymati vaqtinchalik **float** tipidagi o'zgaruvchiga beriladi. Bu vaqtinchalik o'zgaruvchi esa f ga qo'shiladi. Chiqqan javob result ga beriladi. Yuqorida ko'rib chiqqanlarimiz kompilyator tarafidan bajariladi. Bu kabi tip o'zgarishlarini avtomatik konversiya(implicit conversion) deymiz. Lekin gohida to'g'ri kelmaydigan tiplarni birga qo'llashga to'g'ri keladi. Masalan **float** tipiga double tipni o'tqazish, char ga int va hokazo. Bu hollarda ochiq konversiya (explicit conversion) amalini bajarishimiz kerak. Buni bajarishning ikki uslubi bor. Birinchisi C da qo'llaniladigan yo'l, ikkinchisi C++ uslubi. C da tipni keltirish uchun o'zgaruvchi oldiga kerakli tipni () qavslar ichida yozamiz.

```
int k = 100;
char s;
s = (char)k;
```

Yuqorida k ning qiymatini char tipidagi vaqtinchalik o'zgaruvchiga berildi, keyin s ga ushbu o'zgaruvchi qiymatini berildi.Bu yerda etibor berilishi kerak bo'lgan narsa shuki, 100 char ga ham to'g'ri keladi. Agar k ning qiymati char oladigan qiymattan kattaroq/kichikroq bo'ladigan bo'lsa, bu hato olib keladi. Shu sababli C dagi tip keltirish nisbatan havfli hisoblanadi. Lekin albatta bilib qo'llanilsa katta yordam beradi. C++ da ma'lumotlar tipini keltirish uchun mahsus operatorlar kiritildi. C uslubidagi keltirish hamma sharoitda qo'llanilar edi. C++ ning keltirish operatorlari esa faqat o'ziga ajratilgan funksiyalarni bajaradi. Undan tashqari ular C dagi keltirishlardan ko'ra kuchsizroqdir. Shu sababli hato ehtimoli kamaytirildi. Yana bir afzallik tarafi shundaki, yangi stildagi

keltirish operatorlari tip tekshirishlarini bajarishadi, agar noto'g'ri keltirish bajarilsa, bu sintaktik hatoga olib keladi.

Ular quyida berilgan:

static_cast

dynamic_cast

const_cast

reinterpret_cast

static_cast ni ko'rib chiqaylik.

```
int k = 200;
char h;
h = static_cast<char>(k);
```

static_cast dan keyin kerakli tip nomi <> qavslar ichida beriladi, va tipi o'zgarishi kerak bo'lgan o'zgaruvchi () qavslar ichida parametr sifatida beriladi. Static_cast kompilyatsiya davrida tip keltirishlarida qo'llaniladi.**dynamic_cast** esa dastur ishlash davrida tip keltirishlari uchun qo'llaniladi.

const_cast esa o'zgaruvchilardan const (o'zgarmas) va volatile (o'zgaruvchan, uchuvchan) sifatlarini olib tashlashda qo'llaniladi. Odatda const o'zgaruvchining qiymatini o'zgartirib bo'lmaydi. Ushbu holda const_cast

qo'llaniladi. **reinterpret_cast** odatiy bo'lmagan keltirishlarni bajarishda qo'llaniladi (masalan void* ni int ga). reinterpret_cast o'zgaruvchining bitlarini boshqa ma'noda qo'llashga imkon beredi. Bu operator bilib

ishlatilinishi kerak. Ushbu operatorlar bilan keyinroq yanada yaqin tanishamiz.

II. C++ tilida yaratilgan matn muharririning dasturiy ta’minoti;

2.1. Ekran bilan matli rejimda ishlash

Matnli rejimda ekranni boshqaruvchi biblioteka. Kompyuter ekrani har doim matnli rejimda ishlaydi va uning o‘lchami80x25 (80 ta belgi va 25 ta satr). **conio bibliotek**asi yordamida ekranni matnli rejimda boshqarish uchunquyidagi funksiyalarni taqsimlash mumkin:

- matndagi belgilarni va fon rangini o‘rnatish;
- matnni ekranni kerakli joyidan chiqarish;
- ekran koordinatalarini o‘rnatish;
- ekranda kursor holatini o‘zgartirish;
- displey ekraniga oynani moslash.

Sgets Konsol bilan satr kiritish

Clreol Kursor turgan joydan boshlab matnni satr oxirigacha o‘chirish

Clrscr Ekranni tozalash

Cprintf Ekranga formatli chiqarish

Cputs Ekranga satrni chiqarish

Cscanf Klaviaturadan kiritish

Getch Tugmacha bosilmaguncha Dastur bajarilishini ushlab turish

Getche Ekranga exoli klaviatura yordamida belgi kiritish

MovetextEkranni kerakli nuqtatsidan matnli so‘zlarnii chiqarish

Putch Ekranga belgini chiqarish

Puttext Ekranni kerakli nuqtatsidan boshlab matn chiqarish.

C++ tilida quyidagi ranglar majmuasi bor:

O‘zgarmaslarQiymatlar Rangi

BLACK 0Qora

BLUE 1Ciniy

GREEN 2 yashil

RED 4Qizil

....

WHITE 15 Oq

Textcolor funksiyasi. Funksiya tanasi void textcolor(int newcolor);bu erda newcolor- rang qiymati (yuqoridagi jadvalda) textcolor ni chaqirish: textcolor(rang); Bu erda rang- 0 dan 15 gacha qiymat yoki rang nomlari.

Textbackground funksiyasi –ekran va oyna fon rangini o‘rnatish. Funksiya ko‘rinishi: void textbackground(int newcolor);

Macalan: textbackground(4); fon rangi qizilsimvol (belgi) larni rangini o‘rnatish datsturi:

```
# include<conio.h>
# include<stdio.h>
void main(void)
{ int i;
textbackground(0);
textcolor(15); clrscr();
for(i=1;i<16; i++)  // rang nomerini tanlash
{ textcolor(i);  // simvolni rangini o‘rnatish
cprintf(“ rang=%i”,i);  // chop qilish
If (i%5 ==0) cprintf(“\r\n”);
}
printf(“  \n”);
getch();
}
```

Matndagi simvolni aniqligini past, yuqori va normal o‘rnatish funksiyalari. Lowvideo,normvideo, highvideo. Bu holatni o‘rnatish uchun ularni quyidagicha aniqlash mumkin. Masalan: normvideoni highvideo orasidagi 0 va 7 ranglar 8 va 15 ranglarbilan almashtiriladi. Simvol yarkostlarini aniqlash dasturi:

```
# include<conio.h>
# include<stdio.h>
```

```
void main()
{ clrscr();
normvideo();  // normal yarkost
cprintf(“ normal yarkotst  \r\n”);
highvideo();  // ortiqcha yarkotst
cprintf(“ ortiqcha yarkotst \r\n”)
Lowvideo();  // pasaygan yarkotst
cprintf(“ pasaygan yarkotst \r\n”)
}
printf(“  \n”);
getch();
}
```

Ekranni berilgan nuqtasida matn chiqarish uchun kursorni joriy ekrannikerakli joyiga o‘rnatish mumkin. Buning uchun gotoxy funksiyasidanfoydalanamiz.

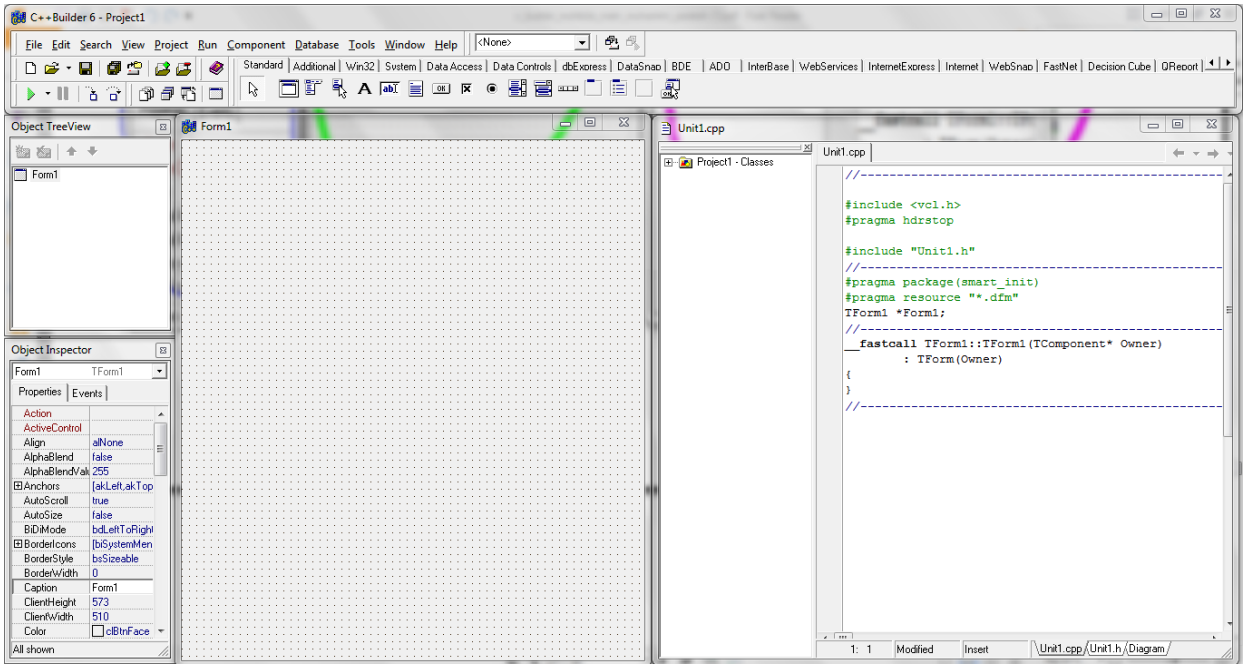
void gotoxy(int x, int y);

Bu erda x va y – joriy ekran koordinatalari, x- ekrandagi satrning pozisiyaraqami, gorizontal kordinata. x=1 ÷ 80; y- ekran oynasidagi satr raqami y=1÷25 ; Ekranning chap yuqori koordinatasi (1, 1), o‘ng pastki koordinatasi 80,25 (x=80, y=25);Funksiyani chaqirish gotoxy(10,50). Ekrandagi kursorni wherex va wherey funksiyalari yordamida uni gorizontalva vertikal holatini aniqlash.Funksiya ko‘rinishi: int wherex(void); Int wherey(void); delline va insline funksiyalari:Bu funksiyalar ekrandagi satrni o‘chirishda ishlatiladi. Satrni o‘chirishkursorni o‘rnatilishiga bog‘liq. Funksiya kbhit dastur bajarilishini to‘xtatmaydi. Bukerakli tugmachani bosish orqali amalga oshadi. Masalan: 0-agarda birorta xam tugmacha bosilmasa!0- faqat nol emas, boshqa ixtiyoriy tugmacha. Ushbu tugmachalardantashqari: ctrl, alt, caps lock, numlock, print screen va pause.

2.2. Borland C++ da matn muharririni yaratish texnologiyasi va dasturiy ta’minoti;

Vazifa: Matn fayllarini tashkil qiuvchi va o’zgartiruvchi dastur tuzilsin. Diskdagi ayllarni ochish va kiritilgan o’zgartirishlarni saqlash imkoniyatlari amalga oshirilsin. Fayllarni izlash, hamda fayllarni saqlash joyini tanlash uchun fayllarni ochish/saqlash standart muloqatlardan foydalanilsin. Faylning matni Memo maydonida aks ettirilsin.

Dastur tuzish muhiti: Dasturni C++ Builder muhitida tuzishni ko’rib chiqamiz. Biz ishlaydigan muhitning boshlang’ich ko’rinishida bosh menyu, Forma, Unit qismi, obyektlar sxemasini, ko’rsatish qismi va obyekt tahrilagich bo’limchalaridaniborat bo’ladi. Yani quyidagi oynalar bo’ladi:



Yechish :

Mazkur dasturni yozishda shaklda mos ravishda fayllarni ochadigan va saqlaydigan tugmachalarni joylashtirishga to’g’ri keladi. Yana mos muloqat oynalarini ham o’rnatish talab etiladi. Tugmachalarni bosish xodisalarini qayta ishlovchiga muloqat oynalarini chaqirish o’rnatiladi: SaveDialog1 ->Execute. Muloqat oynasining OnCanClose xodisalarni qayta ishlovchisiga esa fayllar

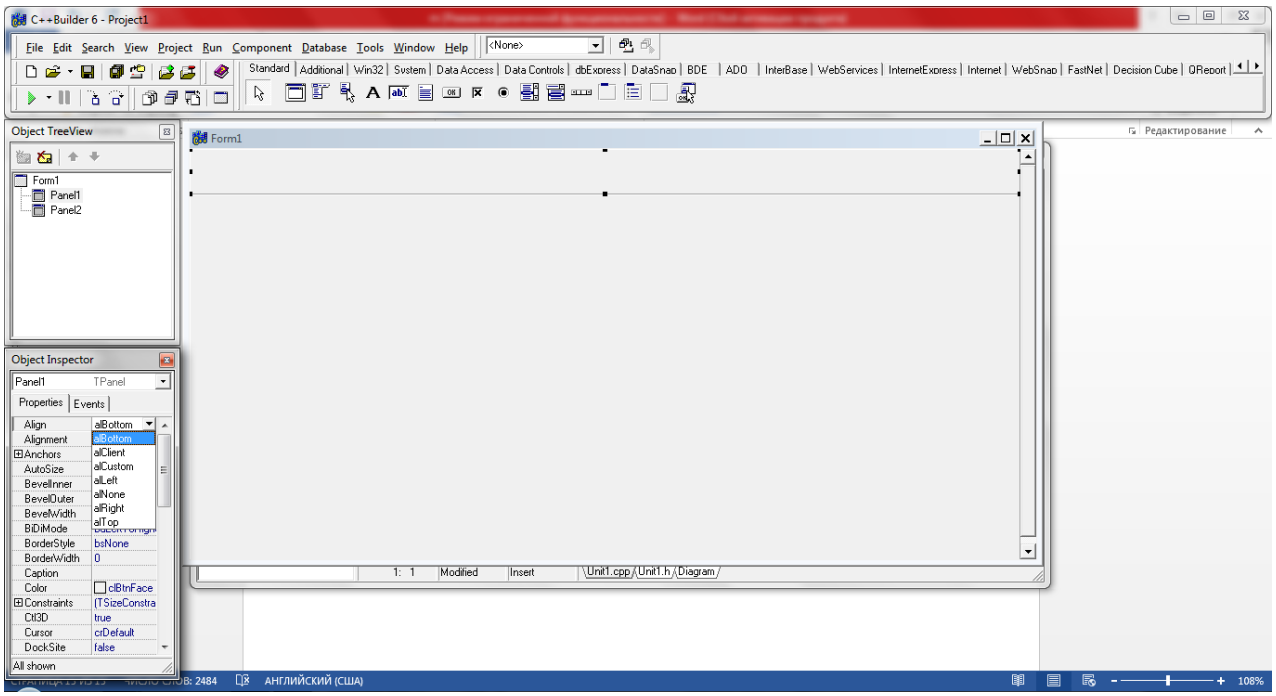
bilanishlashni amalga oshiruvchi dastur kodi o'rnatiladi. Fayllar bilan ishlash muloqat oynasining OnCanClose xodisasi sodir bo'lganda mos muloqat oynasining FileName xossasida tanlangan faylning nomi paydo bo'ladi. Aynan shu fayl bilan ishlash kerak bo'ladi.Ifstream sinf ob'ektining satriga yozilgan fayl to'g'risidagi ma'lumotlarni uzatish uchun satrni belgilar massiviga almashtirishga to'g'ri keladi. Bu ishni massivning birinchi elementiga murojatni va bevosita satrni uzatish mo'ljallangan protsedura yaratib osongina amalga oshirish mumkin. Bu protsedura belgilarni satrdan olib, massivni ketma-ket, elementma-element to'ldiradi. Bu protseduraning yordamida barcha kerakli almashtirishlarni osongina bajarish mumkin. Faylning mazmunini Memo1 maydoniga yozish uchun satrni ifstream sinfining getline() funksiyasi yordamida ketma-ket o'qish va uning qism ob'ekti Lines (Memo1->Lines->Add(stroka);) ning Add() funksiyasi yordamida Memo1 maydoniga yozish kerak.

Ma'lumotlarni faylda saqlash uchun faylga Memo1 ob'ektini satrini belgima-belgi, satr oxiri belgisi (g'n) ni qo'shib va yangi satrdan boshlab yozish zarur.

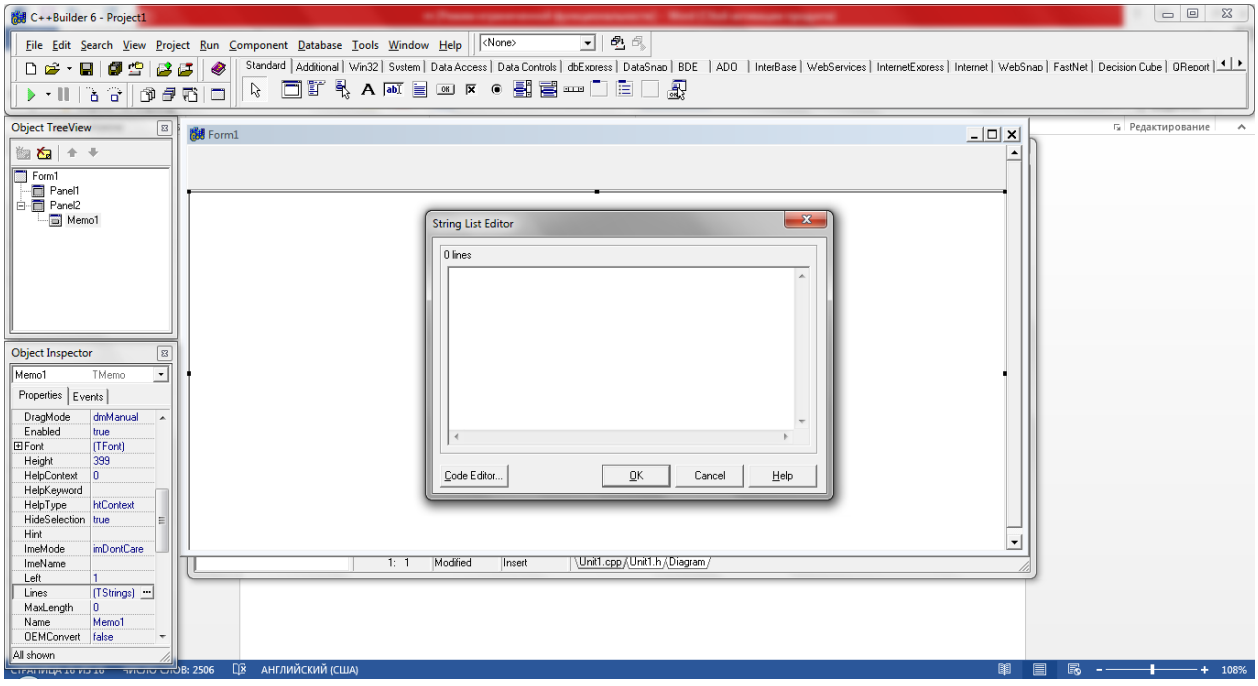
Dasturni tuzish jarayoni.

Dastlab C++ Builder dasturlash tilini ishga tayyorlab olamiz. Bu dasturni uchun quyidagi ketma- ketlikda C++ Builder 6 dasturiga kiramiz: **Пуск ->Всепрограммы -> Borland C++ Builder 6 -> C++ Builder 6 .**

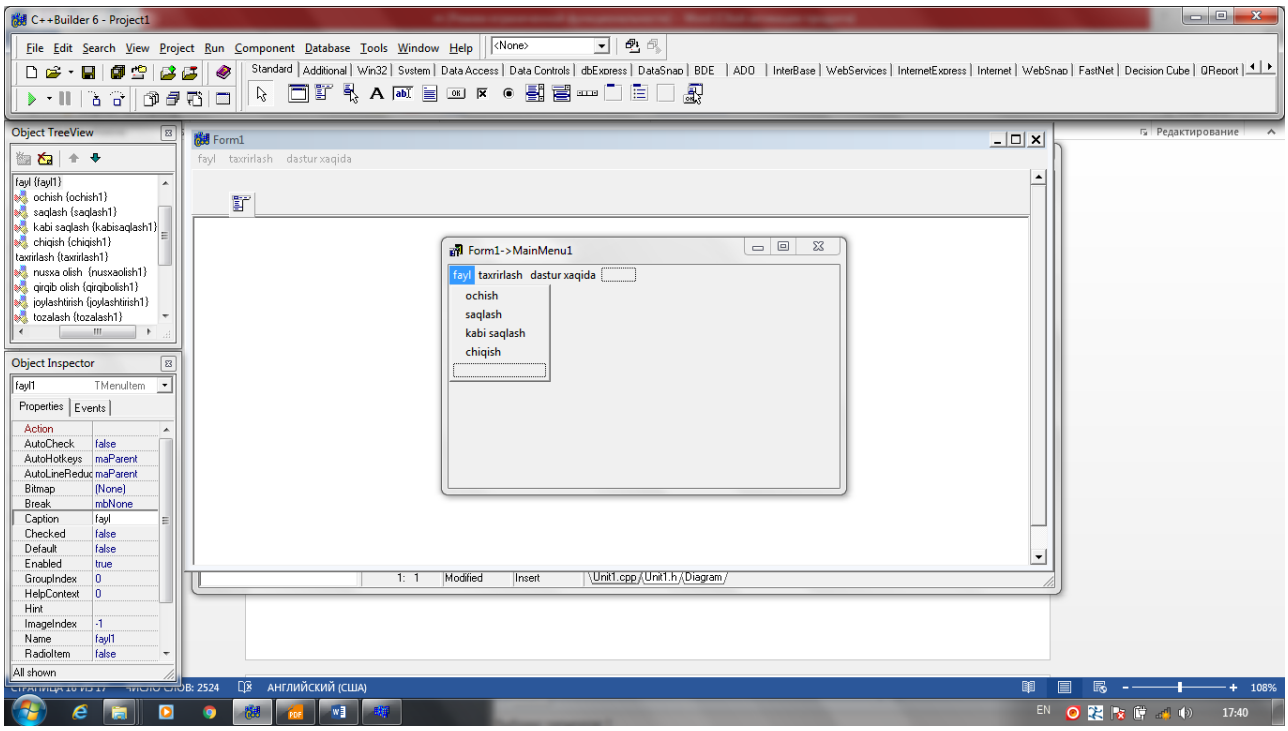
Forma ochib Formaga Standart komponentalar palitrasidan 2 ta Panel komponentasini qo'yamiz va '**object inspector'** dagi "**aling**" xossasini "**alBottom**"ga o'zgartiramiz. Shundan so'ng shu oynadagi "**Caption** " xossasini ham tozalaymiz. Shunda bizga uyidagicha oyna xosil bo'ladi:



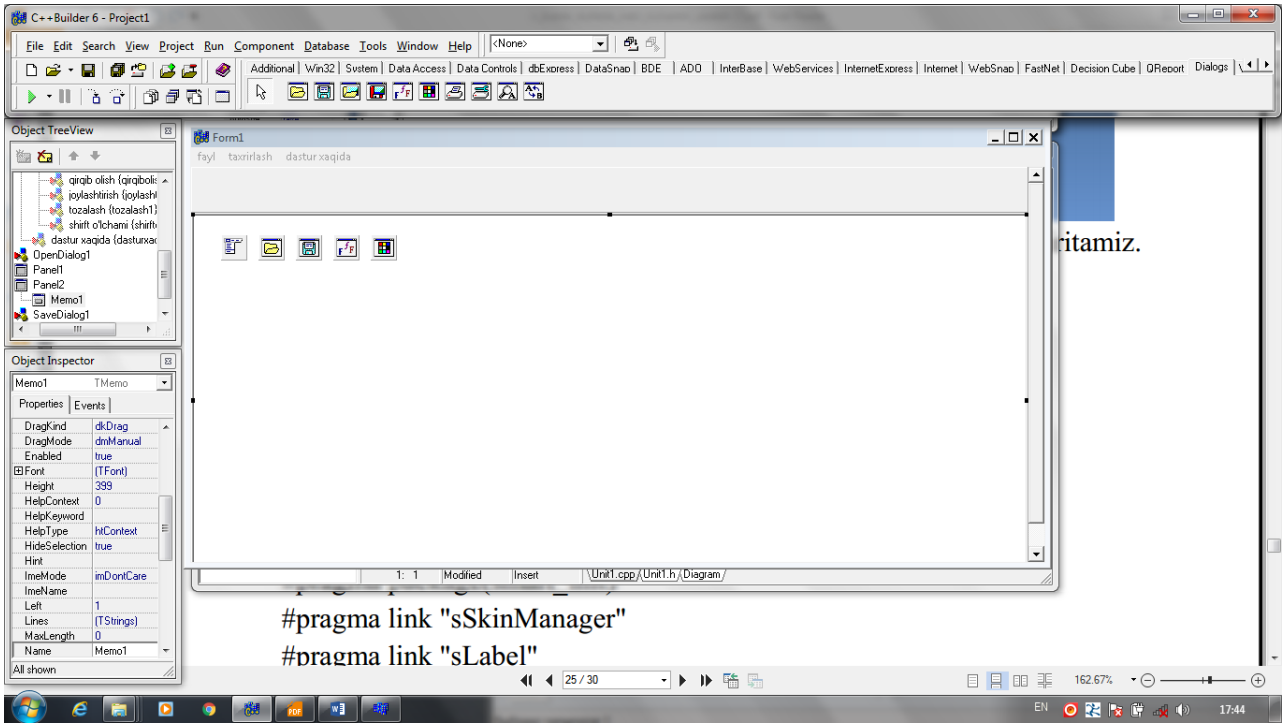
Bu o’garishlarni bajarganimizdan so’ng **Panel2** ga Standart komponentalar palitrasidan Memo joylashtiramiz. **Memo1** ning “Lines ” xossasini tozalab, “**aling**” xossasini “**alClient**”ga o’zgartiramiz.



Undan keyin esa Standart komponentalar palitrasidan Mainmenu ni joylashtiramiz. Mainmenu ning items xususiyati orqali kerakli menyuni yaratib olamiz.



Hosil qilgan menyuimizning har bir xususiyatiga mos dialoglarni olamiz. Bu dialoglar dialogs komponentalar palitrasidan joylashgan.



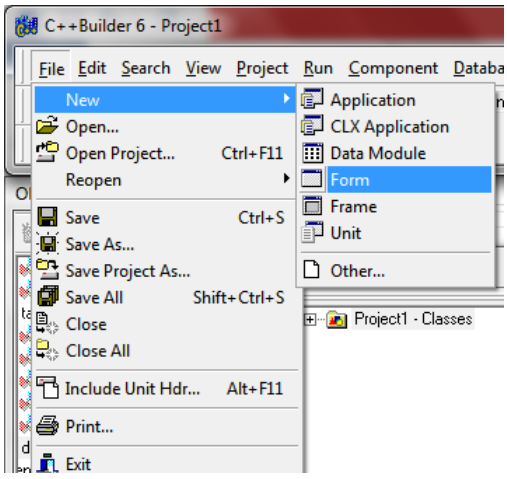
Endilikda Mainmenu ga sichqonchani chap tugmasinin 2 marta bosib buyrilarga mos ravishda quyidagi kodni kiritamiz:

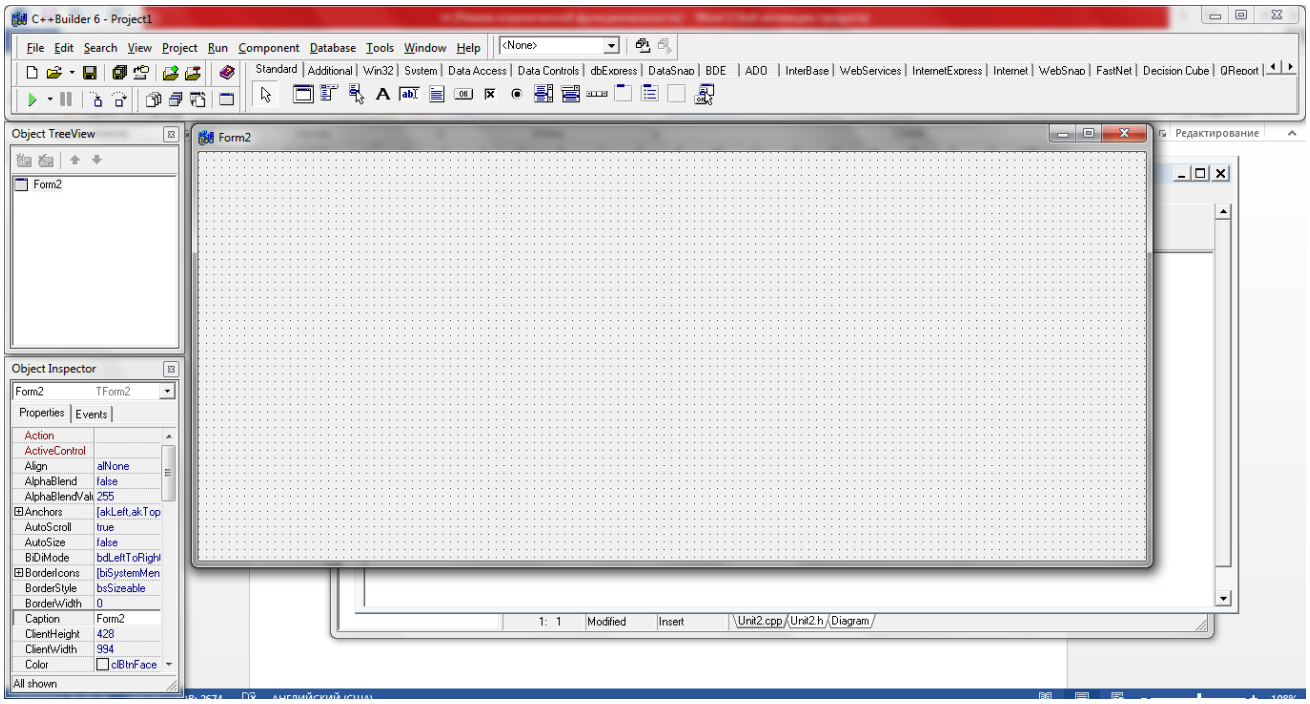
```
void __fastcall TForm1::ochish1Click(TObject *Sender)
{
```

```
if(OpenDialog1->Execute())
Memo1->Lines->LoadFromFile(OpenDialog1->FileName);

}
//---------------------------------------------------------------------------
void __fastcall TForm1::saqlash1Click(TObject *Sender)
{
  Memo1->Lines->SaveToFile(SaveDialog1->FileName + ".txt");
}
//---------------------------------------------------------------------------
void __fastcall TForm1::kabisaqlash1Click(TObject *Sender)
{
if(SaveDialog1->Execute())
Memo1->Lines->SaveToFile(SaveDialog1->FileName + ".Doc");

}
//---------------------------------------------------------------------------
void __fastcall TForm1::chiqish1Click(TObject *Sender)
{
ShowMessage("dasturdan chiqyapsiz !!!");
Close();

}
//---------------------------------------------------------------------------
void __fastcall TForm1::nusxaolish1Click(TObject *Sender)
{
   Memo1->CopyToClipboard();
}
//---------------------------------------------------------------------------
void __fastcall TForm1::qirqibolish1Click(TObject *Sender)
{

Memo1->CutToClipboard();

}
```

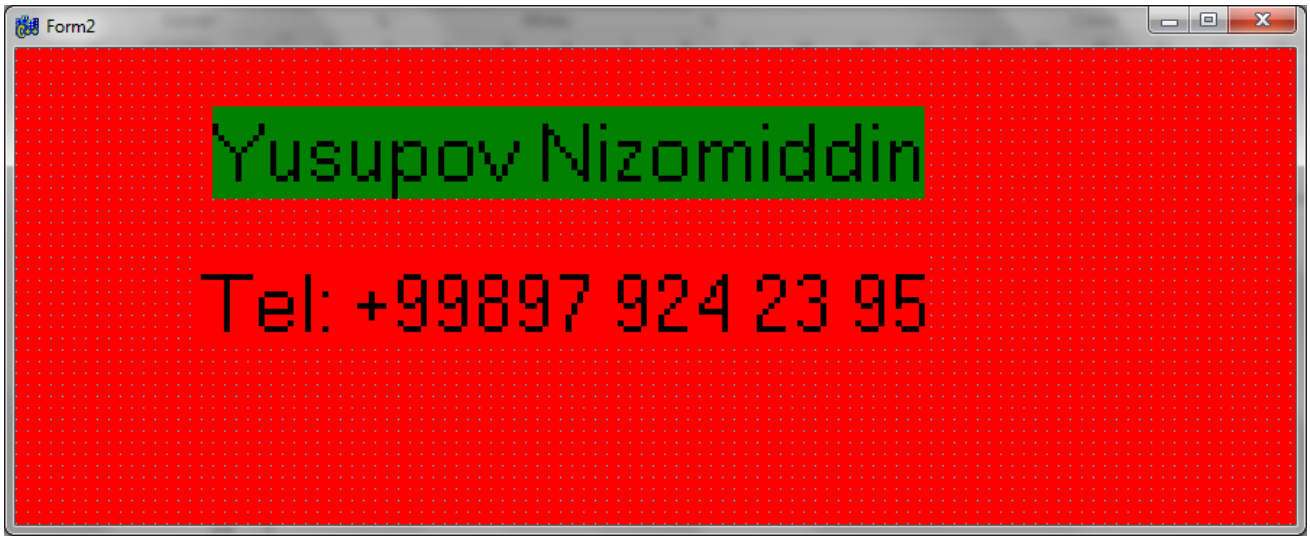
```
//---------------------------------------------------------------------------
void __fastcall TForm1::joylashtirish1Click(TObject *Sender)
{
Memo1->PasteFromClipboard();
}
//---------------------------------------------------------------------------
void __fastcall TForm1::tozalash1Click(TObject *Sender)
{
Memo1->Clear();
}
//---------------------------------------------------------------------------
void __fastcall TForm1::shirftolchami1Click(TObject *Sender)
{
if (FontDialog1->Execute())
{ Memo1->Font->Size=FontDialog1->Font->Size;
  Memo1->Font->Color=FontDialog1->Font->Color;
  Memo1->Font->Style=FontDialog1->Font->Style;
}

}
//-----------------------------------------
```

Oxirgi “dastur xaqida ” deb nomlangan menuni ishlatish uchun bitta form yaratib olishimiz kerak. Buning uchun **C++ Builder** ning **File** menusidagi **New** dan **Form**ni tanlaymiz.

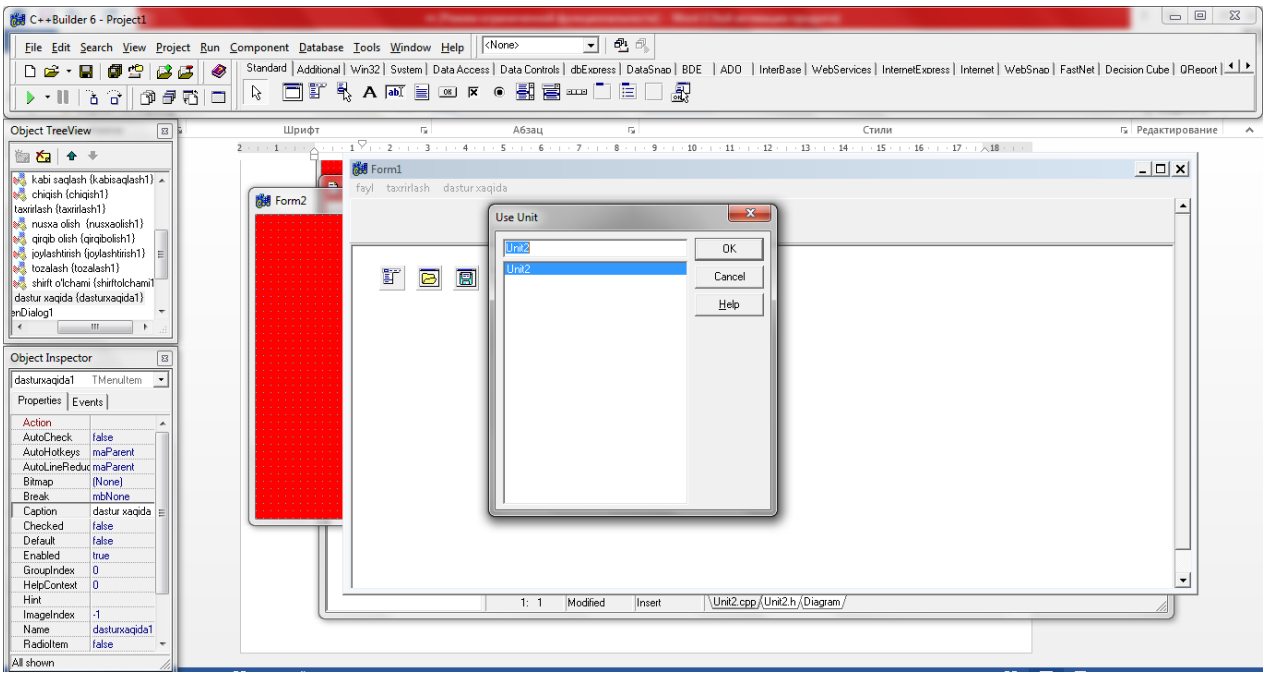




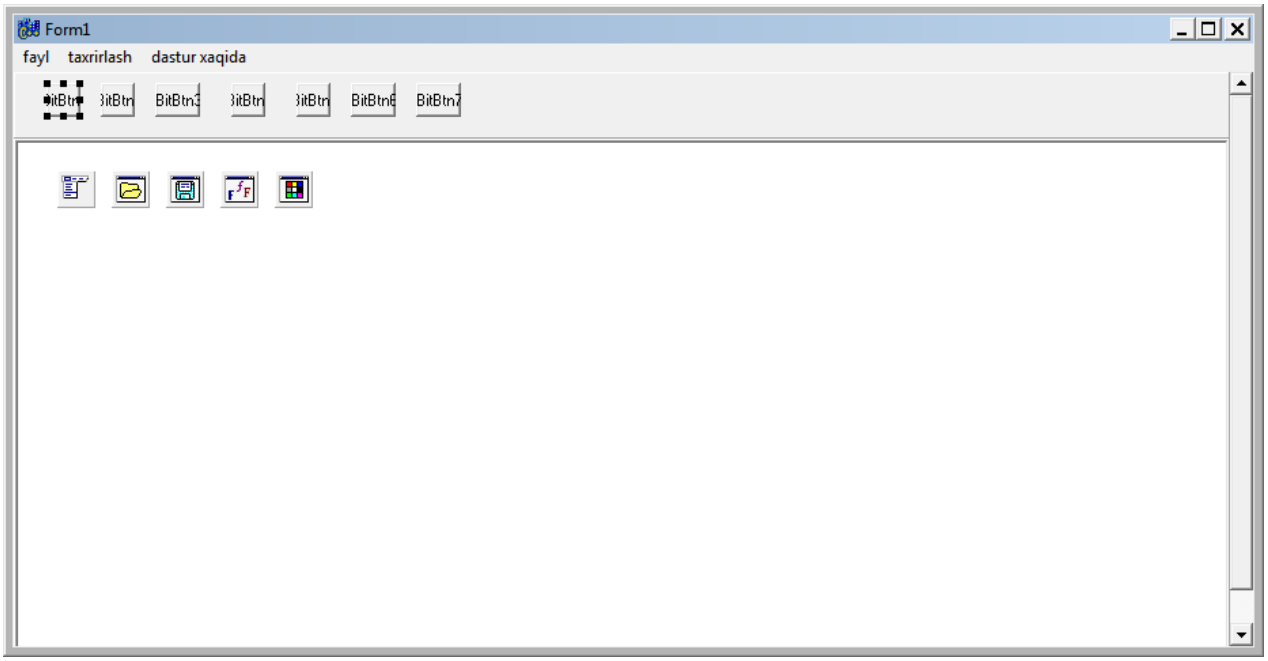
Form2 oynasiga dastur yaki dasturchi xaqida ma’lumotlarni Standart komponentalar palitrasidan Label Komponentasini olib Caption xossasiga kiritamiz. Form2 ning rangini ‘**object inspector’** dagi “**color**” xossasini o’zgartirib tanlaymiz.



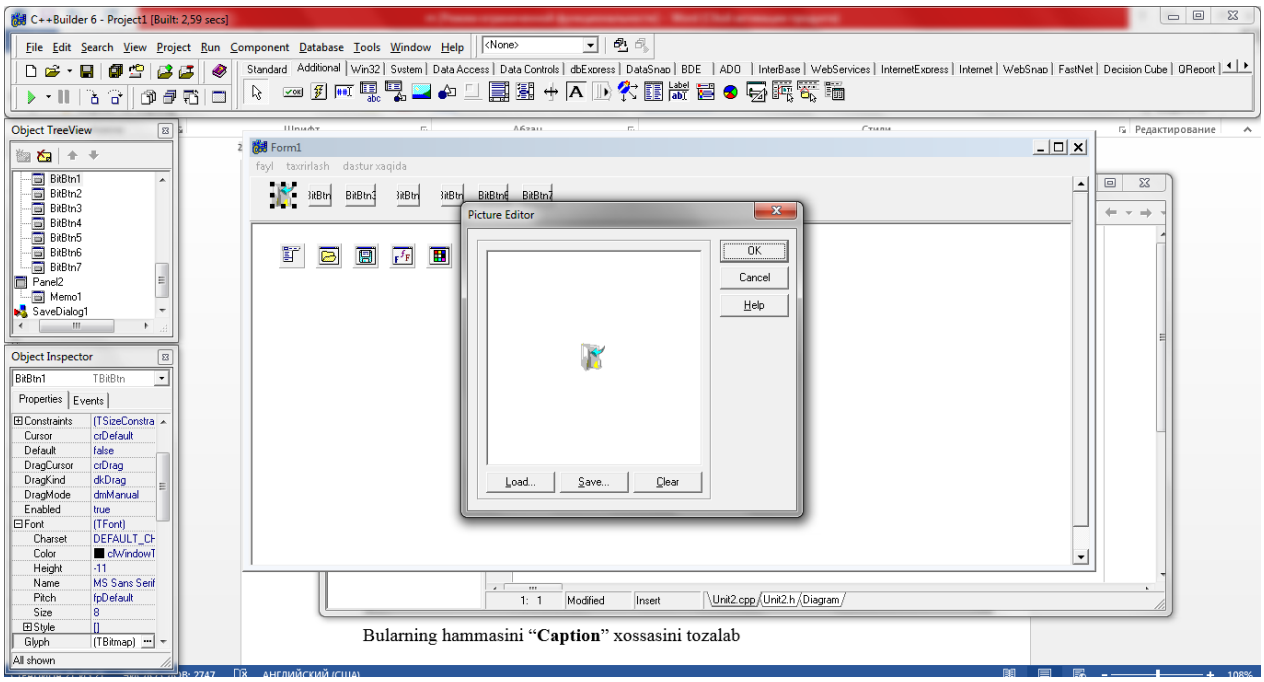
Form1 ga o’tib **C++ Builder** ning **File** menusidagi **Include Unit Hdr**buyrug’ini yoki klaviaturadan **Alt+F11** klavishalarini birgalikda bosamiz va xosil bo’lgan oynadan Unit2 ni tanlab Ok tugmasini bosamiz.



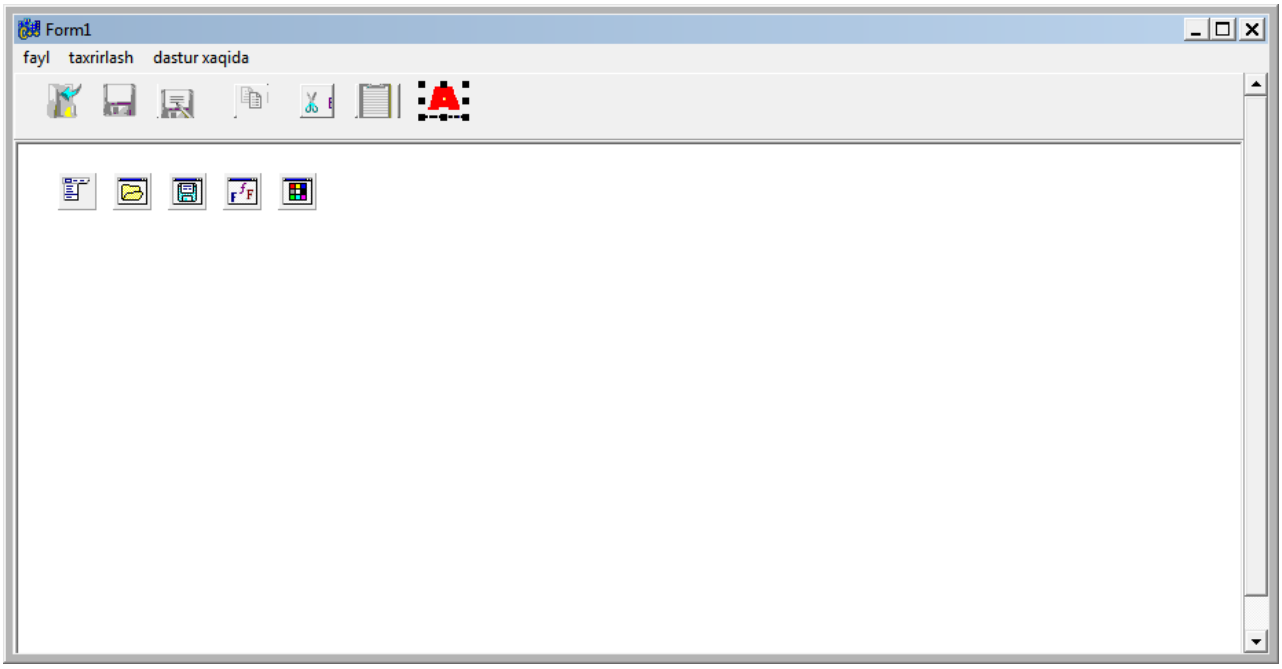
Endi Panel1 ga Standart buyruqlar komponentasini joylashtirish qoldi. Buning uchun Additional komponentalar palitrasidan BitBtn komponentasini Panel1 ga 7 tani ketma ket joylashtiramiz.



Bularning hammasini “**Caption**” xossasini tozalab, “**Glyph**” xossasini ochib Load buyrug;ini tanlab kerakli bo’lgan rasmni olib ok tugmasini bosamiz.



Shu jarayonni 7 marta takrorlaymiz.



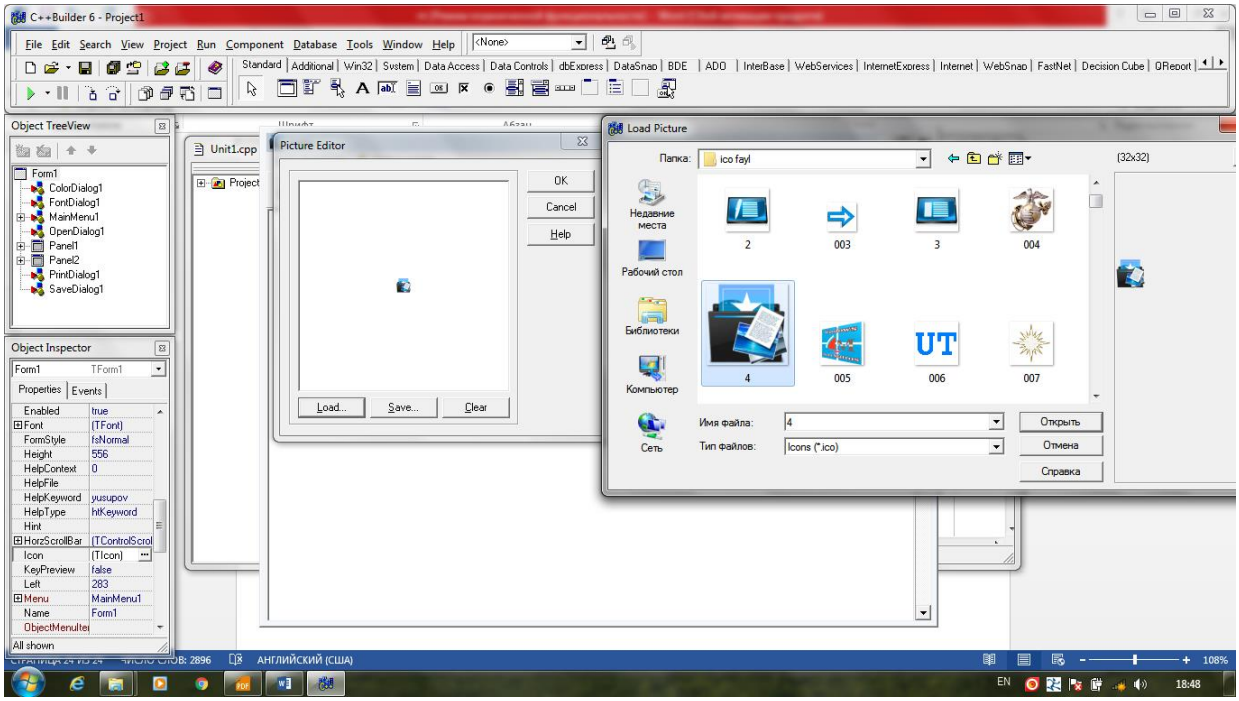
Shu ishlarni bajarib bo'lganimizdan so'ng "**BitBtn**"larning har birining onClick xossasiga mos ravishda quyidagi kodni kiritamiz.

```
void __fastcall TForm1::BitBtn1Click(TObject *Sender)
{
if(OpenDialog1->Execute())
Memo1->Lines->LoadFromFile(OpenDialog1->FileName);

}
//---------------------------------------------------------------------------
```

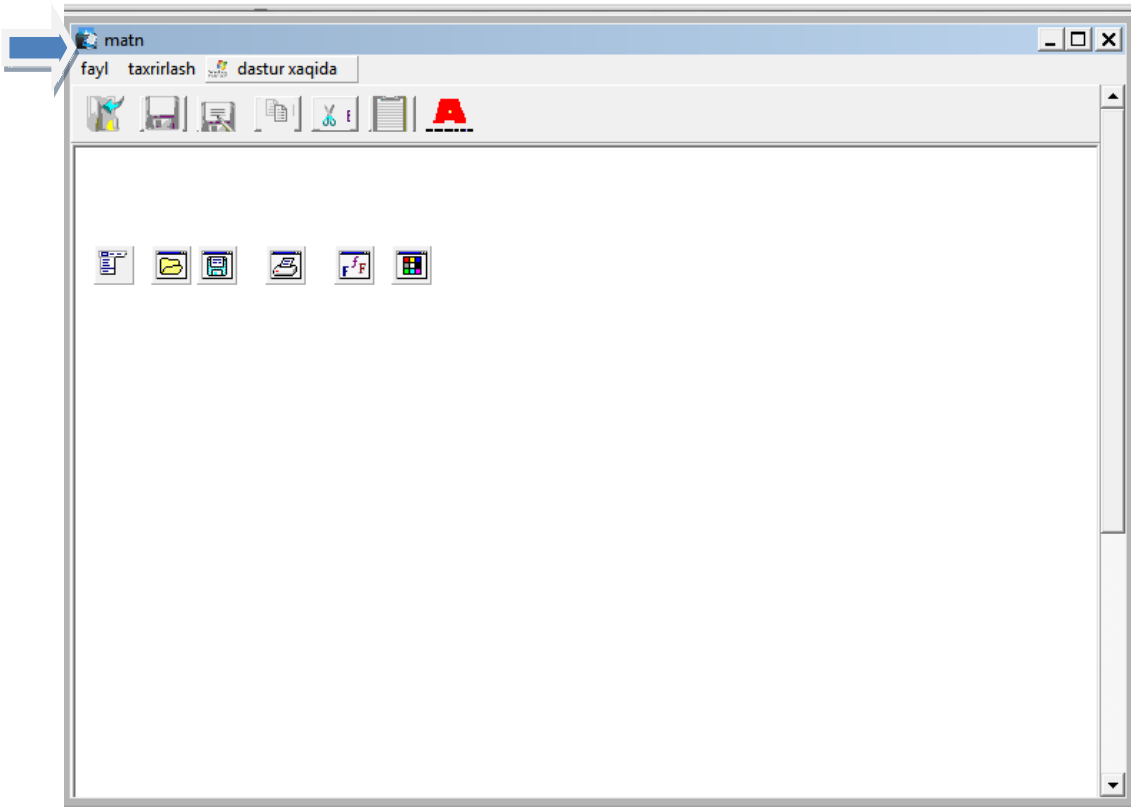
```
void __fastcall TForm1::BitBtn2Click(TObject *Sender)
{
if(SaveDialog1->Execute())
Memo1->Lines->SaveToFile(SaveDialog1->FileName + ".txt");

}
//---------------------------------------------------------------------------
void __fastcall TForm1::BitBtn3Click(TObject *Sender)
{
if(SaveDialog1->Execute())
Memo1->Lines->SaveToFile(SaveDialog1->FileName + ".Doc");
}
//---------------------------------------------------------------------------
void __fastcall TForm1::BitBtn4Click(TObject *Sender)
{
Memo1->CopyToClipboard();
}
//---------------------------------------------------------------------------
void __fastcall TForm1::BitBtn5Click(TObject *Sender)
{
Memo1->CutToClipboard();
}
//---------------------------------------------------------------------------
void __fastcall TForm1::BitBtn6Click(TObject *Sender)
{
Memo1->PasteFromClipboard();
}
//---------------------------------------------------------------------------
void __fastcall TForm1::BitBtn7Click(TObject *Sender)
{
if (FontDialog1->Execute())
{ Memo1->Font->Size=FontDialog1->Font->Size;
  Memo1->Font->Color=FontDialog1->Font->Color;
  Memo1->Font->Style=FontDialog1->Font->Style;
}
```

Mana kichkina matn muxariri kichik matnlar ustida amal bajarishga tayyor. Endi bu dasturga tushinarliligi oshishi uchun bir nechta o’zgartirishlar kiritamiz.

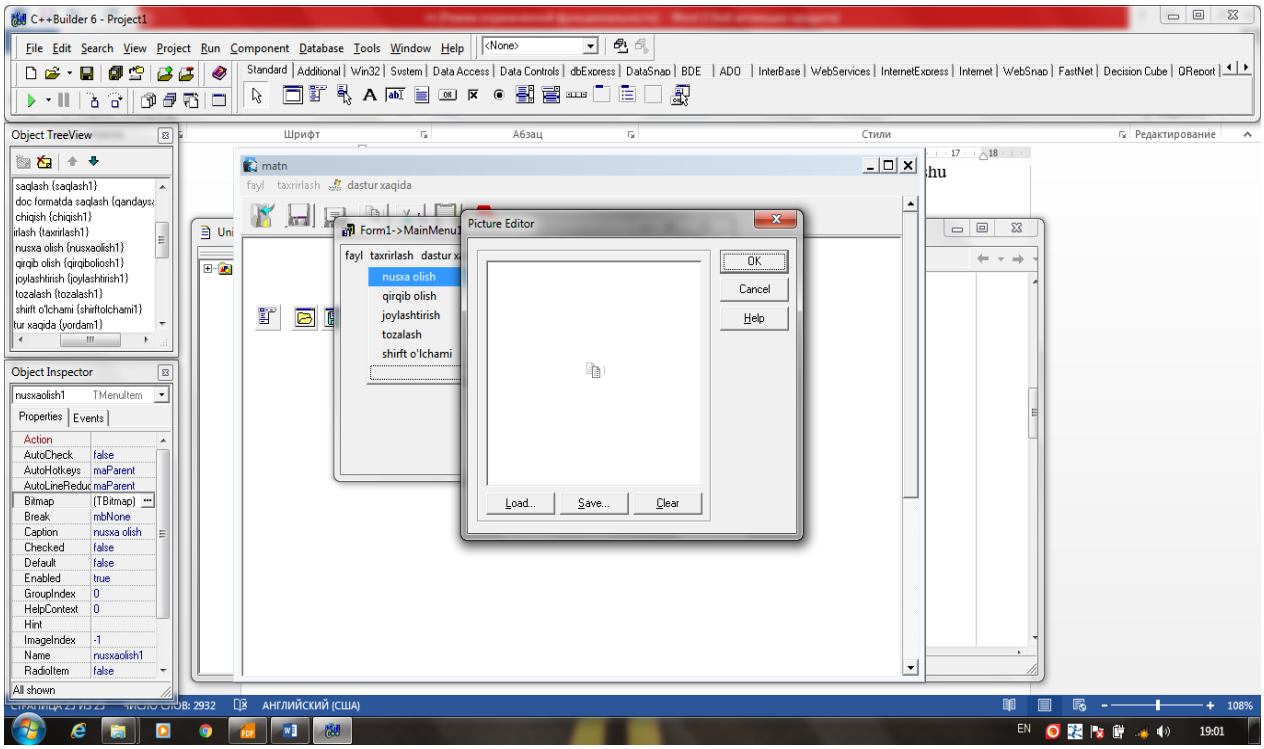
Birinchi navbatda form1 ning fizik nomini o’zgartiramiz. Buning uchun Form1 ni tanlab “**Caption**”xossasiga “matn muxarriri” deb yozamiz. Shu joyda “**Icon**”xossasini tanlab kerakli rasm olib dastur ustidagi emblemani o’zgartirishimiz mumkin.



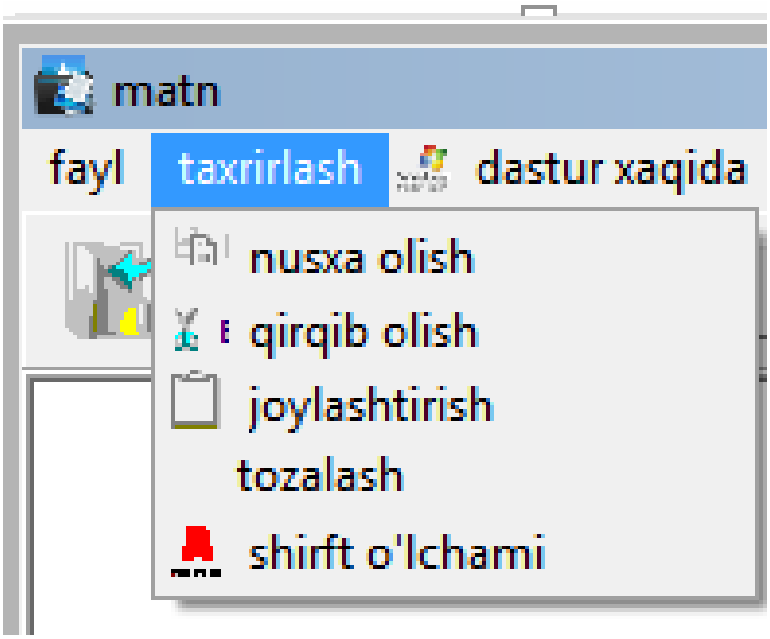
Yuqoridagi oynadan rasm tanlab OK tugmasini bosilsa dastur ustidagi rasmcha shu tanlagan rasmga o’zgarganini ko’rishimiz mumkin.



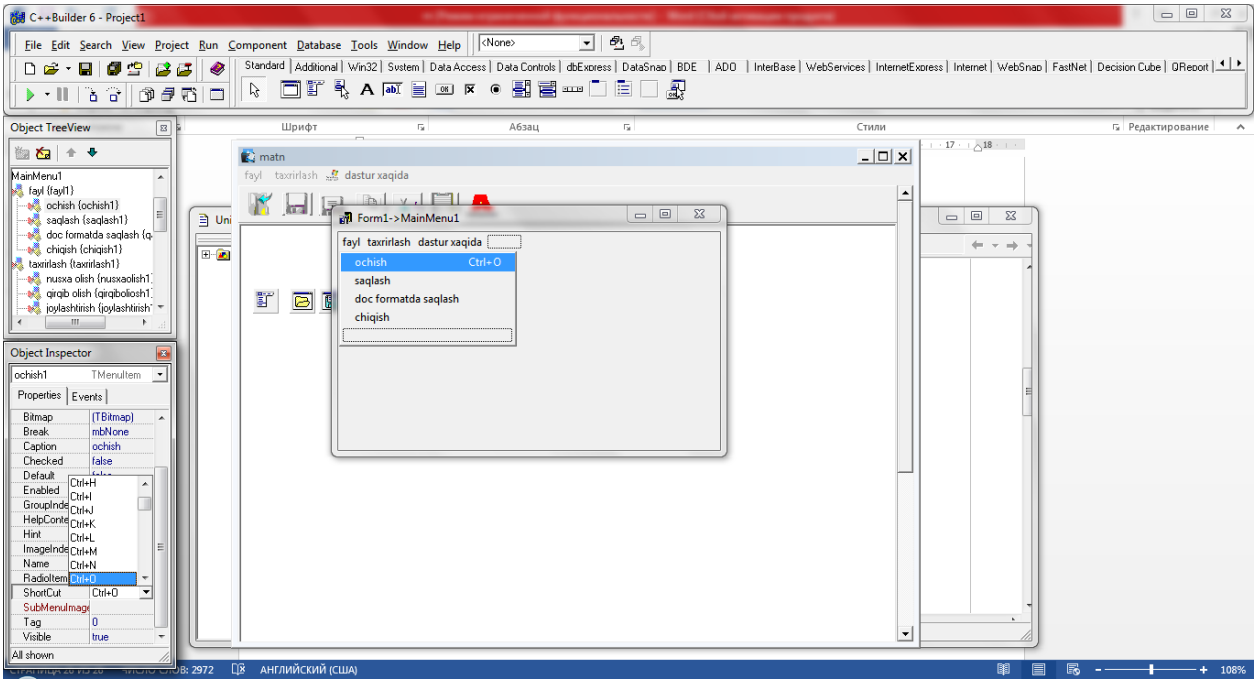
Bundan tashqari yaratilgan dasturning menulariga ham rasmlar joylashtirishimiz, shu menularni biror klavishlar birikmasiga bog’lashimiz mumkin. Buning uchun shu menuni tanlab “**Bitmap**” xossasini tanlaymiz va kerakli rasmni tanlab Ok tugmasini bosamiz.



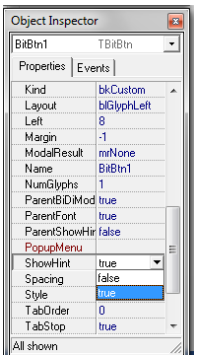
Shu tariqa hamma menuning buyriqlariga rasmlar o’rnatishimiz mumkin.

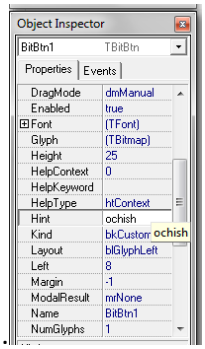


Endi menuning buyruqlariga klavishlar birikmasini biriktiramiz. Buning uchun shu menuning kerakli buyrug'ini tanlab "ShortCut" xossasini tanlab kerakli klavishlar birikmasini tanlab qo'yishimiz mumkin.

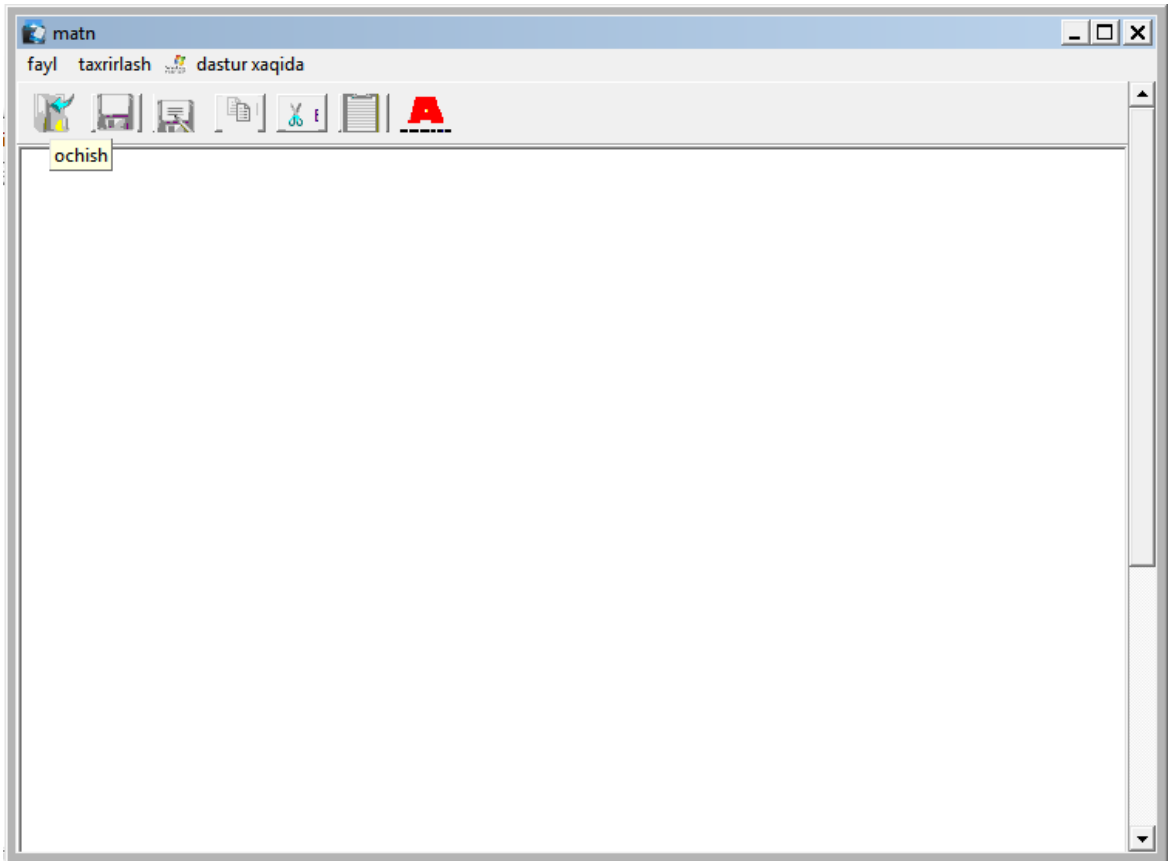


Bundan tashqari yaratgan dasturimizning Standart asboblar panelini ham ustiga sichonchani olib borilsa u xaqida malumot beradigan ham ilishimiz mumkin. Buning uchun yaratilgan dasturning qaysi piktogrammasiga qo'ymoqchi bo'lsak uni tanlab "**ShowHind**" xossasini "true" ga o'zgartiramiz va "**Hind** " xossasiga qanday ma'lumot chiqarmochi bo'lsak shu matnni kiritamiz.





Shu tariq kerakli bo’lgan buyruqlarimizga izoh qo’shishimiz mumkin.



Xulosa.

Biz ushbu “**kurs ishi**” tayyorlash davomida quyidagi xulosaga keldik, matn muharririni yaratish uchun C++ Builder muhiti haqida yetarli ma’lumotga ega bo’lishimiz kerak. Formaga matnlar bilan ishlashda kerak bo’ladigan komponentalarni joylashimiz va har bir komponentaning tegishli kodlarini kiritib chiqishni o’rgandik . Dasturni kompilyatsiya qilishdan avval unga tegishli barcha fayllarni saqlab qo’yishni ham o’rgandik. Har bir dastur bir emas bir nechta fayllardan iborat bo’ladi. Ularni saqlab qo’yayotganda har bir dasturni alohida kataloglarda maxsus nom ostida saqlash maqsadga muvofiqligi haqida ko’nikmaga ega bo’ldik. Chunki dasturni boshqa joyga ko’chirishga to’g’ri kelib qolgan vaqtda dasturlarning o’xshash fayllarini adashtirib yuborish mumkinligi, bunday holatda dasturda xatolik aniqlanib dastur ishlamay qolishi, Dasturni to’liq saqlash uchun faqat save ni emas balki save all ni bosib hamma fayllarni saqlashni organdik.

Bu dasturni tuzib nafaqat yozishni amalga oshirish balki kichik matnlar ustida taxrirlash va dasturni ishlashni bilmaydigan foydalanuvchi ham bir ko’rinishda tushinadigan tushinarli, bejirm ko’rinishga ega dastur tuzdik.

Biz bu dasturni tuzib obektlar yaratishni va ular orqali shunga o’xshash yana boshqa obektlar yaratdik bu dasturda ishlashimga yana bir sabab boshqa obektlarga mo’ljallanmagan dasturlash tillarida faqat masalani matematik dasturi tuzib natija olinadi bunda yani C++ Builderda esa boshqa imkoniyatlar va kampanentalar bilan ham ishlashni o’rgandik.

Foydalanilgan adabiyotlar ro’yhati

1. Страуструп Б. Язык программирования C++. Третье издание, М.: Бином, 1999.

2. Шмидский Я.К.Программирование на языке C++: Самоучитель. Учебное пособие. Диалектика. 361 стр, 2004 г.

3. Q.Abduraximov C++ Dasturlash asoslari 2014

4. Ашарина Н.А. Основы программирования на языках Си,C++. Учебный курс.М.: 2002 г.

5. Подбельский В.В. Язык C++ М.: Финансы и статистика, 1996.

6. www.tammi.uz

7. www.ziyonet.uz

8. www.google.uz qidiruv sayti