O'ZBEKISTON ALOQA, AXBOROTLASHTIRISH VA

TELEKOMMUNIKATSIYA

TEXNOLOGIYALARI DAVLAT QO'MITASI

TOSHKENT AXBOROT TEXNOLOGIYALAR UNIVERSITETI
NUKUS FILIALI

Kompyuter Injiniring fakulteti
3b Informatsion texnologiyalar yo'nalishi talabasi

Sadiqov Temurning
C++ dasturlash tili
fanidan

Mustaqil ishi

Mavzu: C++ builder muhitida matn muharririni yaratish.

Bajargan: Sadiqov T.
Qabul qilgan: Tleuov K.

Nukus-2014

Referat mazmuni:

Kirish

Hozirgi kunda respublikamizdagi texnika oliy o‘quv yurtlarida “Informatika va axborot texnologiyalari” yo‘nalishi va mutaxassisliklariga turli xil dasturlash tillarini 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 dasturlash tiliga eng yaqin bo‘lib, tezlik jihatidan 10 % ortda qolar ekan.

Keyingi yillarda amaliy dasturchilarga juda ko‘p integratsion dastur tuzish muhitlari taklif etilmoqda. Bu muhitlar u yoki bu imkoniyatlari bilan bir–biridan farq qiladi. Aksariyat dasturlashtirish muhitlarining fundamental asosi C/C++ tiliga borib taqaladi.

Ushbu mustaqil ishida hozirgi kunda kompyuterda berilgan funksiyalarni grafigini va har xil ko‘rinishdagi shakllarni chizish va ularni aniq koordinatalarini aniqlash, ekranni grafik rejimga o‘tkazish va undagi mavjud piksel va ranglardan foydalanish 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 operasion 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.

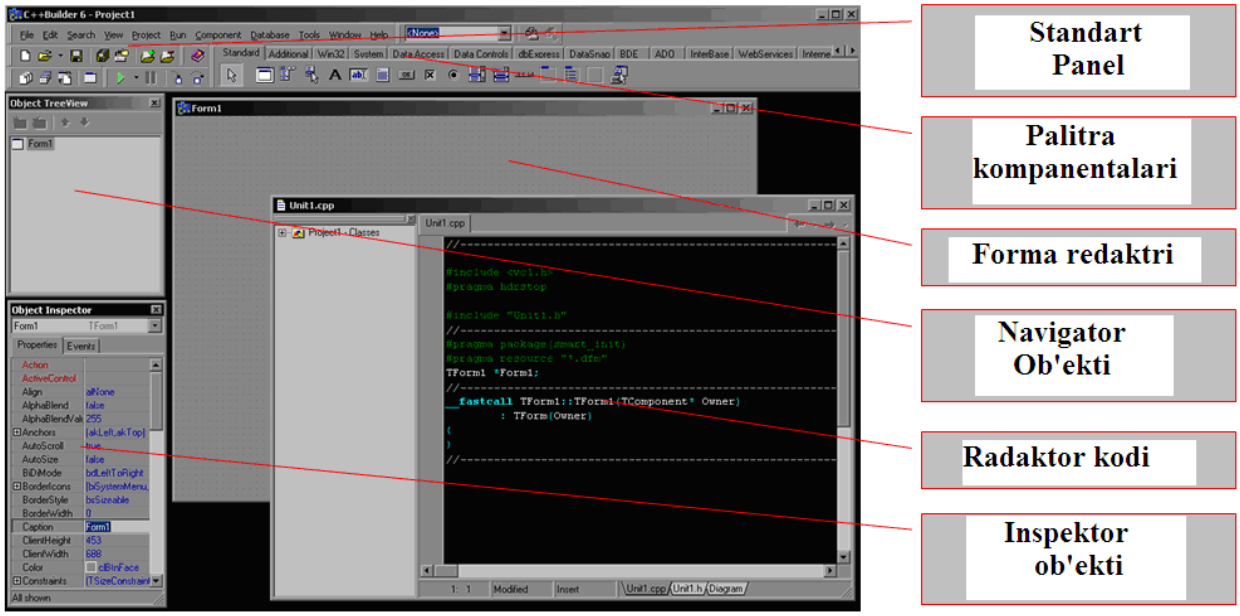
Borland C++ va Delphi dasturlash tili Windows uchun mo’ljallangan bo’lib, uning birinchi versiyasi Windows operatsion sistema qobig’ida ishlagan.

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.

Dastur yaratish muhiti

Dastur yaratish umumlashgan muhiti Redaktor form – Shakllar muharriri, Inspektor ob'ektov – Ob'ektlar inspektori, Palitra komponentov – Komponentlar palitrasi, Administrator proekta – Proekt administratori va to'la umumlashgan Redaktor koda – Kodlar muharriri hamda kodlar va resurslar ustidan to'liq nazoratni ta'minlaydigan , dastur ilovalarini tezkor yaratadigan Otladchik - instrumentov - Sozlash-instrumentlari kabilarni birlashtiradi.



Komponentlar

Komponentalarni shaklga o'rnatish uchun komponentlar palitrasidagi kerakli piktogramma tanlanadi, so'ngra shaklning komponenta joylanishi kerak bo'lgan joyi tanlanadi. Shundan so'ng komponentalar xossalarini ob'ektlar inspektori yordamida tahrirlash mumkin. Properties bandida komponentalar xossalarining ro'yxati (chapda) va bu xossalarning qiymatlar ro'yxati (o'nggda) joylashgan.

Komponentalar ko'rinadigan (vizual) va ko'rinmaydigan (vizual bo'lmagan) larga bo'linadi. Vizual komponentalar bajarilish paytida proektlash paytidagidek paydo bo'ladi. Bunga knopkalar va tahrirlanuvchi maydonlar misol bo'la oladi. Vizual bo'lmagan komponentalar proektlan vaqtida shakldagi piktogramma ko'rinishida paydo bo'ladi. Ular bajarilish paytida hech qachon ko'rinmaydi, ammo ma'lum funksionallikga ega bo'ladi (masalan, berilganlarga murojatni ta'minlaydi, Windowsning standart muloqatlarini chaqiradi).

Xossalar

Xossalar komponentalarning tashqi ko’rinishi va tabiatini aniqlovchi atributlar hisoblanadi. Xossalar ustunidagi ko’p xossalar komponentalari oldindan o’rnatilgan (po umolchaniyu) qiymatlarga ega bo’ladi (masalan, knopkalar balandligi). Komponentalar xossalari xossalar varag’i (Properties) da aks ettiriladi. Ob’ektlar inspektori komponentalarning nashr etilgan (published) xossalarini aks ettiriladi. published-xossalardan tashqari komponentalar umumiy (public), faqat ilovalarning bajarilish paytidagina murojat qilish mumkin bo’lgan nashr qilingan xossalarga ega bo’ladi. Xossalar ro’yxati ob’ektlar inspektori xossalar varag’ida joylahadi. Xossalarni proektlash paytida aniqlash mumkin yoki ilovalarning bajarilish paytida ko’rinishini o’zgartirish uchun kod yozish mumkin. Komponenta xossalarini proektlash paytida aniqlash uchun shakldagi komponenta tanlanadi, ob’ektlar inspektori xossalari varag’i ochiladi, aniqlanadigan xossa tanlanadi va zurur bo’lsa xossalar muharriri yordamida o’zgartiriladi (bu kiritish uchun oddiy maydon yoki son, osilib tushuvchi ro’yxat, ochiluvchi ro’yxat, muloqat paneli va boshqalar bo’lishi mumkin).

Biror komponentaning xossalarini dasturning bajarilish paytida o’zgartirish uchun «Imya Komponenta» –> «Nazvanie svoystva» tavsifiga o’zaruvchidek murojat qilish kerak, ya’ni qiymatlarni o’zimiz hohlagandek o’qishimiz yoki almashtirishimiz mumkin.

Xodisalar

Ob’ektlar inspektorining xodisalar varag’i (Events) komponentalar tomonidan taniladigan xodisalar ro’yxatini ko’rsatadi. Har bir komnonenta o’zining shaxsiy xodisalarni qayta ishlovchi naborga ega bo’ladi. C++ Builder da xodisalarni qayta ishlovchi funksiyalarni yozish va xodisalarni bu funksiya bilan bog’lashga to’g’ri keladi. Biror bir xodisaga qayta ishlovchi yozib, siz dasturga bu xodisa ro’y berganda yozilgan funksiyaning bajarilishini topshirasiz.

Xodisani qayta ishlovchini qo’shish uchun shaklda xodisani qayta ishlovchi komponenta tanlanadi. So’ngra xodisalar varag’ida ob’ektlar inspektori ochilib (Event bandi) xodisaning qatoridagi qiymatlar ustunida sichqonning chap tugmasi ikki marta bosiladi. Bu bilan C++ Builder ni xodisalarni qayta ishlash prototipini generatsiya qilishga va uni kodlar muharririda ko’rinishiga majbur qiladi. Bu holda bo’sh funksiya nomi generatsiya qilinadi va muharrir kod kiritilishi zarur bo’lgan joyda ochiladi. Kursor buyruqlar qavslari ichiga joylashadi { … }. So’ngra xodisa sodir bo’lganda bajarilishi kerak bo’lgan kod kiritiladi. Xodisalarni qayta ishlovchi funksiya nomidan keyin ko’rsatiladigan parametrlarga ega bo’lishi mumkin.

Quyida xodisalarni qayta ishlovchi protseduraning shunday bo’sh karkasi ko’rsatilgan:

```
void __fastcall TForm1::Button2Click(TObject *Sender)
```

```
{

}
```

Turlar 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 = s1 + s2;

Natijada s o’zgaruvchi s1 va s2 o’zgaruvchilardan tashkil topgan ‘stroka teksta’ degan qiymatni qabul qiladi.

Qo’shimcha turlar

Borland 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	O’zgaruvchini tavsiflashga misol	O’lcham
__int8	__int8 c = 128;	8 bit
__int16	__int16 s = 32767;	16 bit
__int32	__int32 i = 123456789;	32 bit
__int64	__int64 big = 12345654321;	64 bit
unsigned __int64	unsigned __int64 huge = 1234567887654321;	64 bit

Turlarni o’zgartirish protseduralari

Standart turlarni o’zgartirish

C++ 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.

Misollar:

```
int t = 2;
int b = 7;
int r = 3;
if (t>b)
{
 if (b < r)
 {
  r = b;
 }
}
else
{
 r = t;
 return (0);
}
```

Bu dastur bajarilganda r ning qiymati 2 ga teng bo’ladi.

Ekran bilan matnli rejimda ishlash

Matnli rejimda ekranni boshqaruvchi biblioteka

Kompyuter ekrani har doim matnli rejimda ishlaydi va uning o‘lchami 80x25 (80 ta belgi va 25 ta satr).

conio bibliotekasi yordamida ekranni matnli rejimda boshqarish uchun quyidagi 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.

sonio bibliotekasi

Nomlanishi	Funksiya maqsadi
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
Movetext	Ekranni kerakli nuqtatsidan matnli so‘zlarnii chiqarish
Putch	Ekranga belgini chiqarish
Puttext	Ekranni kerakli nuqtatsidan boshlab matn chiqarish

sprintf funksiyasi

Matnni kerakli ranglar otstida ekranga chiqarish funksiyasi cprintf.

Uning strukturatsi: int cprintf (const char * format [, argument,….]);

O‘zgarmatslar	Qiymatlar	Rangi
BLACK	0	Qora
BLUE	1	Ciniy

GREEN	2	YAshil
RED	4	Qizil
....	.....	....
WHITE	15	Oq
BLINK	128	Belgini yonib-o‘chishi

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 qizil

simvol (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.

Matsalan: normvideo ni highvideo oratsidagi 0 va 7 ranglar 8 va 15 ranglar bilan 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 ekranni kerakli joyiga o‘rnatish mumkin. Buning uchun **gotoxy** funksiyasidan foydalanamiz.

void gotoxy(int x, int y);

Bu erda x va u – joriy ekran koordinatalari, x- ekrandagi satrning pozisiya raqami, gorizontal kordinata. x=1 ÷ 80;

u- 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 gorizontal va 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‘chirish kursorni o‘rnatilishiga bog‘liq. Funksiya kbhit dastur bajarilishini to‘xtatmaydi. Bu kerakli tugmachani bosish orqali amalga oshadi.

Matsalan: 0-agarda birorta xam tugmacha bosilmasa

!0- faqat nol emas, boshqa ixtiyoriy tugmacha. Ushbu tugmachalardan tashqari: ctrl, alt, caps lock, numlock, print screen va pause.

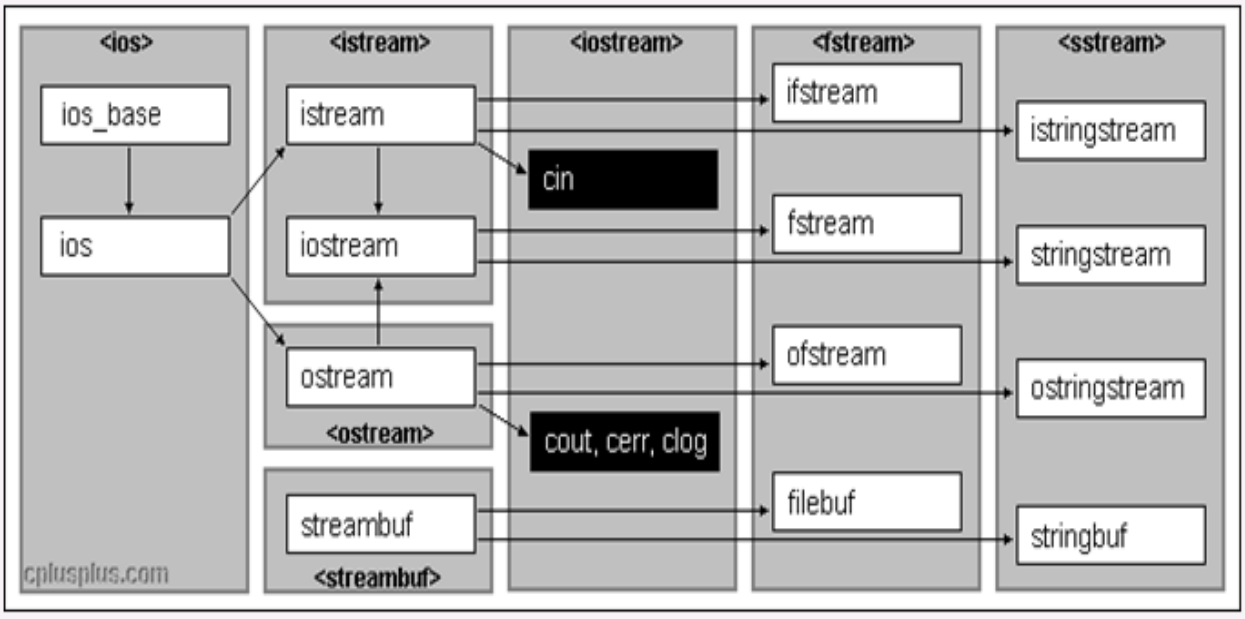
Matnli fayllar bilan ishlashdagi amallar

Matnli fayllar bilan ishlash binar fayllar bilan ishlashdan bir oz **farq** qiladi. Matnli fayllarda ma`lumotlar satrlarda saqlanadi. Matnli fayl elementilari har xil uzunlikdagi satrlardir. Bu satrlar bir biridan satr oxiri belgisi bilan ajratiladi. Matnli fayl elementlari indekslanmagan bo`lganligi uchun, faylning istalgan elementiga bevosita murojaat qilib bo`lmaydi.

C++ da matnli yoki binar fayllar bilan ishlash uchun keng imkoniyatlar berilgan. Matnli fayllar bilan ishlashda oddiy C ning funksiyalaridan ham foydalanish mumkin. Masalan, formatli o'qish va yozish funksiyalari yoki oldingi mavzudagi funksiyalardan foydalanishimiz mumkin. Matnli fayllar bilan ishlashning bunday usuli kitoblarda keng yoritilgan. Ularni mustaqil o'qib - o'rganishingiz mumkin.

Bu mavzu fayllar bilan ishlovchi oqimlarni qisqacha o'rganamiz va buni matnli fayl misolida ko'ramiz.

Standart kiritish / chiqarish kutubxonasi sinflari quyidagicha shajaraga ega:



Fayllar bilan ishlash uchun quyidagi sifnlar ob'ektlari hosil qilinadi:

- **ofstream** - faylga ma'lumot yozish uchun
- **ifstream** - fayldan ma'lumot o'qish uchun
- **fstream** - fayldan ma'lumot o'qish uchun va yozish uchun

Bu sinflarni dasturda ishlatish uchun **<fstream>** sarlavha faylini qo'shish kerak bo'ladi. Bundan keyin programmada aniq fayllar oqimini aniqlash mumkin. Masala:

```
ofstream yozish; // faylga yozish oqimini e'lon qilish
ifstream oqish; // fayldan o'qish oqimini e'lon qilish
fstream yoz_oqi; // faylga yozish va o'qish oqimini e'lon qilish
```

Keyin faylni ochish kerak bo'ladi. Faylni ochish deganda, uning ustida nima amal qilinishi haqida amaliyot tizimiga xabar berish tushuniladi.

void **open** (const char * filename, ios_base::openmode mode = ios_base::out);

mode parametri quyidagicha qiymatlarni qabul qilishi mumkin:

ios::in	faqat ma'lumot o'qish uchun
ios::out	faqat ma'lumot yozish uchun
ios::ate	faylni ochishda fayl ko'rsatkichini fayl oxiriga qo'yish
ios::app	fayl oxiriga ma'lumotlarni yozish uchun
ios::trunc	bor bo'lgan faylning ustidan yangi faylni yozish
ios::binary	binar holda ma'lumotlarni almashish uchun

Har bir sinf uchun mode parametrining odatiy qiymatlari mavjud:

class	default mode parameter
ofstream	ios::out
ifstream	ios::in
fstream	ios::in | ios::out

Fayl ustida o'qish yoki yozish amalini bajarib bo'lgandan song, faylni yopish kerak bo'ladi. Faylni yopish uchun close funksiyadi ishlatiladi. Masalan:

```
yozish.close();
oqish.close();
```

Matnli faylga ma'lumot yozish

```
#include <iostream>
#include <fstream>

using namespace std;

int main ()
{
    ofstream yozish; // faylga yozish oqimini hosil qilish

    yozish.open("namuna.txt");
    // yangi namuna.txt nomli fayl hosil qilinadi.
    // agar namuna.txt fayli oldindan bo'lsa,
    // uning eski qiymatlari o'chiriladi
    // va yangi fayl hosil qilinadi

    yozish << "Matnli faylga ma'lumot yozish" << endl;
    yozish << "Juda oson!" << endl;

    yozish.close(); // faylni yopish

    return 0;
}
```

Matnli fayldan o'qish

```
#include <iostream>
#include <fstream>
#include <string>

using namespace std;

int main ()
{
    ifstream oqish; // fayldan o'qish oqimini hosil qilish
    string satr;

    oqish.open("namuna.txt");

    // faylni ochishda xatolik sodir bo'lsa
    if (!oqish.is_open())
```

```
{
    cout << "Faylni ochishda xatolik sodir bo'ldi." << endl;
    exit(1); // dasturni tugatish
}

while (!oqish.eof())
{
    // fayldan o'qish
    getline(oqish, satr);
    // ekranga chiqarish
    cout << satr << endl;
}

// namuna.txt fayli bilan oqish oqimi aloqasini uzish
oqish.close();

return 0;
```

}

istream sinfi funksiyalari

```
istream& seekg ( streampos pos );
istream& seekg ( streamoff off, ios_base::seekdir dir );;
```

oqish oqimi ko'rsatkichini o'rnatish (siljitish).

pos - oqim buferining yangi pozitsiyasi.
dir parametri quyidagilardan birini qabul qilishi mumkin:

Qiymat	Izoh
ios::beg	oqimning boshlanishi
ios::cur	oqimning joriy xolari
ios::end	oqim oxiri

```
long tellg();
```

o'qish oqimining joriy xolatini (pozitsiyasi) aniqlash.

ostream sinfi funksiyalari

```
ostream& seekp ( streampos pos );
ostream& seekp ( streamoff off, ios_base::seekdir dir );
```

yozish oqimi o'rnini (pozitsiyasini) o'rnatish.
pos - oqim buferining yangi pozitsiyasi
dir parametri beg, cur, end qiymatlaridan birini qabul qilishi mumkin.

```
long tellp() - yozish oqimining kelgan joyini aniqlash.
```

Fayldan nusxa olish

```
//ushubu dastur orqali ixtiyoriy fayldan nusxa olish mumkin
#include <iostream>
#include <fstream>
using namespace std;

int main ()
{
    int length;
    char * buffer, fayl[] = "matn.txt", yangi[]="yangi_fayl.txt";
    // fayl - nusxalanadigan fayl nomi
    // yangi - yangi nusxalangan fayl nomi

    // o'qish oqimi
    ifstream fromfile(fayl, ios::binary );
    if (!fromfile.is_open())
    {
        cout << "faylni o'qishda xatolik sodir bo'ldi\n";
        exit(1);
    }

    // yozish oqimi
    ofstream tofile(yangi, ios::binary );

    // fayl xajmini aniqlash:
    fromfile.seekg (0, ios::end); // fayl oxiriga o'tish
    length = fromfile.tellg();
    fromfile.seekg (0, ios::beg); // fayl boshiga o'tish

    // xotira ajratish:
    buffer = new char [length];

    // blokka ma'lumotlarni o'qish:
```

```
    fromfile.read (buffer, length);

    fromfile.close();

    // nusxalanishi kerak bo'lgan faylga yozish
    tofile.write (buffer, length);

    // xotirani bo'shatish
    delete[] buffer;

    cout << "fayl nusxalandi\n";

    return 0;
}
```

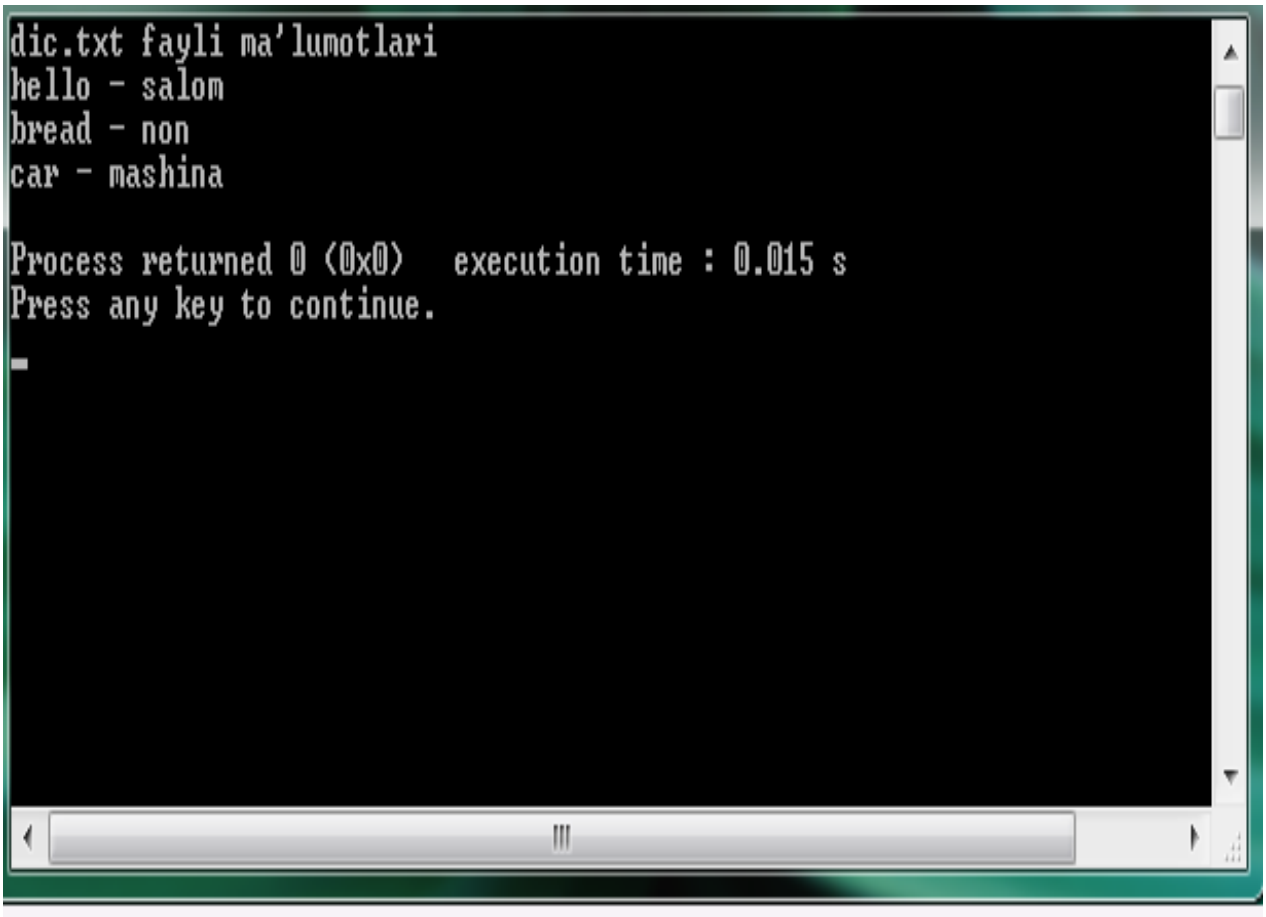
dic.txt nomli fayl berilgan . Faylning har bir satrida inglizcha va o`zbekcha so`zlar "-" belgisi bilan ajratilgan. Inglizcha so`zlarni english.txt fayliga, o`zbekcha so`zlarni uzbek.txt fayliga o`tkazuvchi programma tuzilsin.

dic.txt fayli quyidagicha bo'ladi:
hello - salom
bread - non
car - mashina

```
#include <iostream>
#include <fstream>
#include <string>

using namespace std;

int main ()
{
    ifstream dic("dic.txt");
    ofstream uzbek("uzbek.txt");
    ofstream english("english.txt");

    if (!dic.is_open())
    {
        cout << "dic.txt - fayli topilmadi\n";
        exit(1);
    }

    string s, uzb, eng;
    int p;

    cout << "dic.txt fayli ma'lumotlari\n";
    while (!dic.eof())
```

```
    {
        getline(dic, s);
        p = s.find("-");
        eng.assign(s, 0, p - 1);
        uzb.assign(s, p + 1, s.length() - (p + 1));

        uzbek << uzb << endl;
        english << eng << endl;

        cout << s << endl;
    }

    dic.close();
    uzbek.close();
    english.close();

    return 0;
}
```

Dastur natijadi quyidagicha bo’ladi:



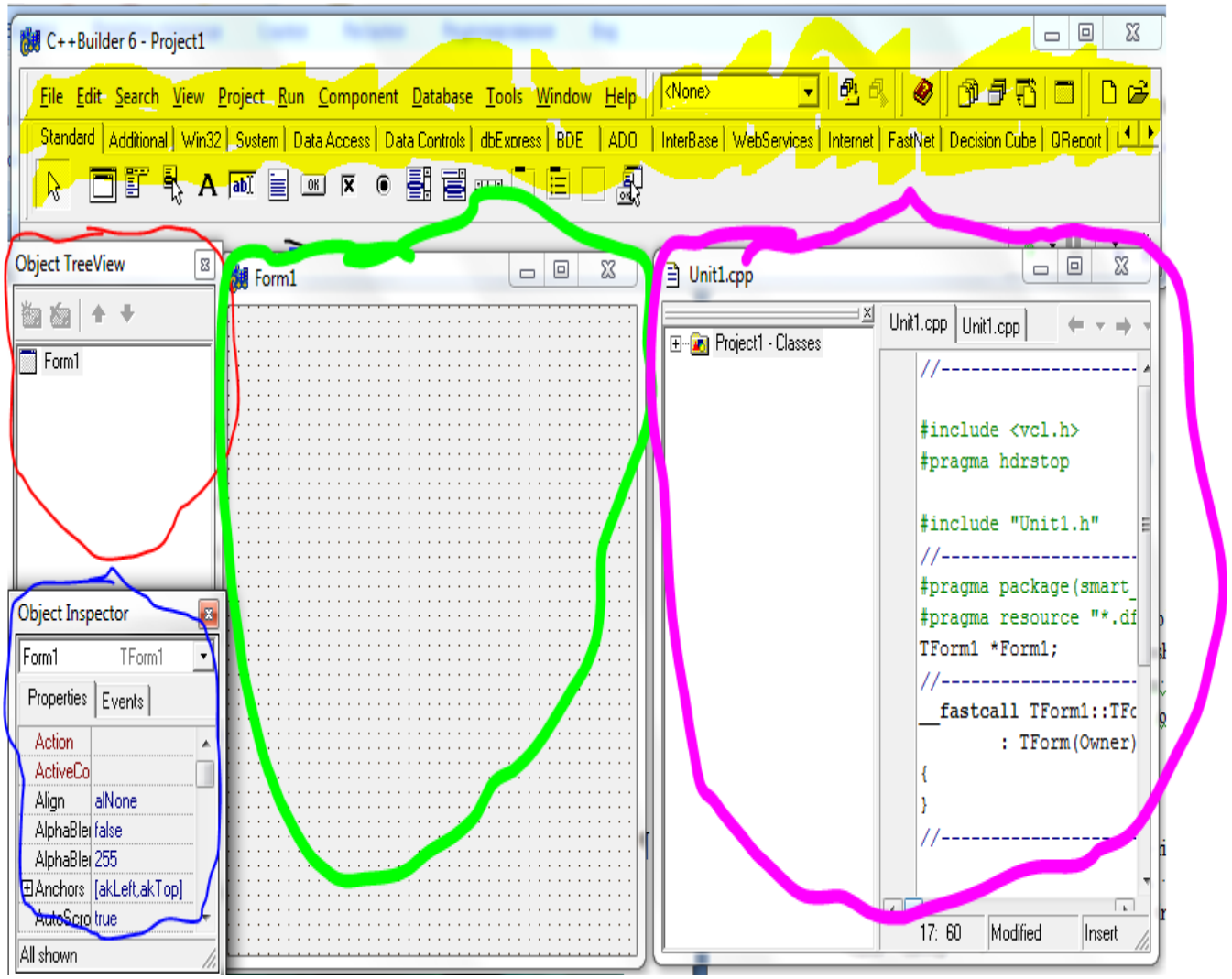
Borland C++ da matn muharriri

Dastur tasnifi

Vazifa: Matn fayllarini tashkil qiuvchi va o'zgartiruvchi dastur tuzilsin. Diskdagi fayllarni 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'limchalaridan iborat bo'ladi.



Muammolar

ifstream va ofstream sinflarining ob’ektlari fayllarni hosil qilish va ular bilan ishlashda uzatilayotgan fayl nomini belgilar massivi sifatida ishlatadi. Standart muloqatlar esa «satr» (AnsiString) turidagi qiymatlarni qaytaradi. Ya’ni standart muloqat oynalari qaytarayotgan qiymatlarni to’g’ridan-to’g’ri ifstream yoki ofstream ob’ektlarga uzatishning imkoni yo’q. Bu muammoni yechish uchun satrni belgilar massiviga almashtiruvchi protsedura tuzish tavsiya etiladi.

Mazkur dasturni yozish uchun dastur yaratish muhitining standart komponentalari, fayllarni izlashga mo’ljallangan muloqat oynalari bilan ishlashni bilish kerak. Undan tashqari matn holatdagi fayllarni o’qish va diskda saqlashni ham bilish zarur.

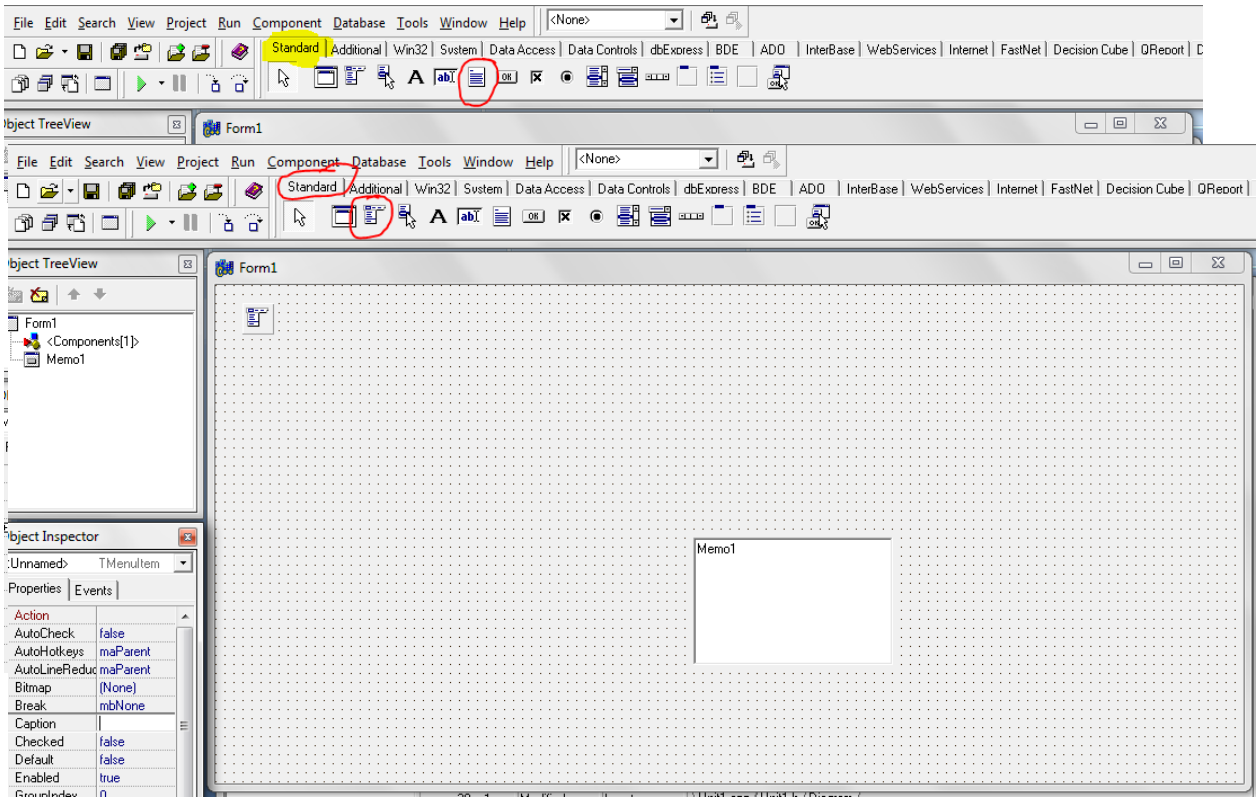
Yechish

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 bilan ishlashni 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.

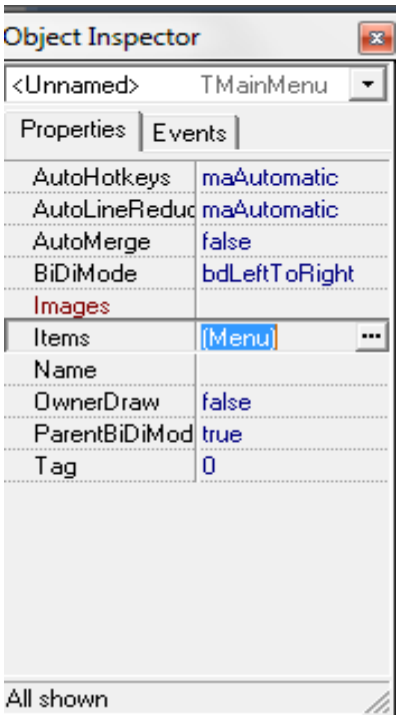
Dasturni tuzish jarayoni

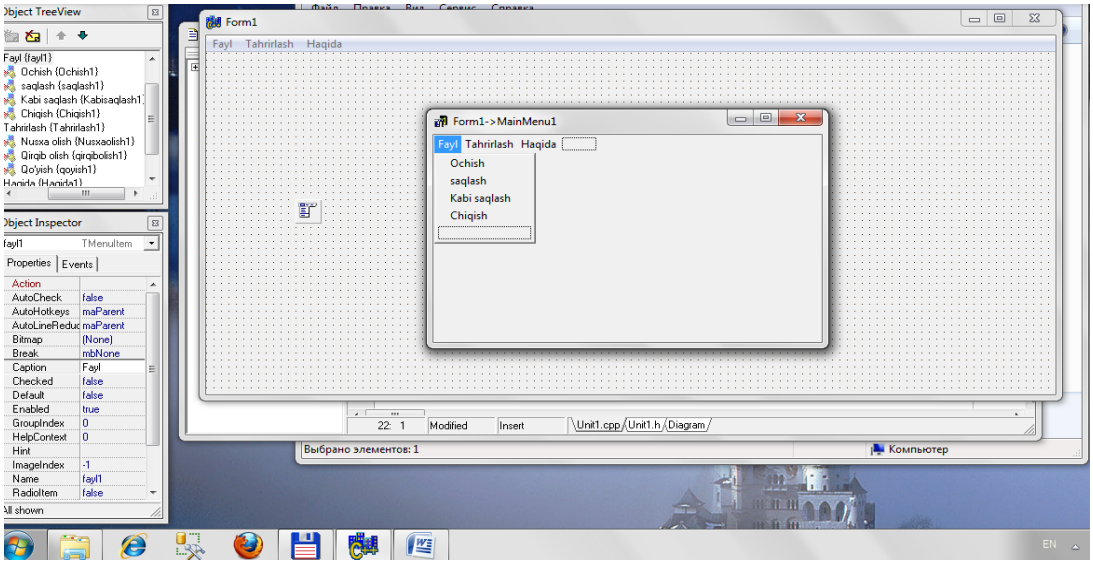
Dastlab C++ Builder dasturlash tilini ishga tayyorlab olamiz. Forma ochib formaga Standart funksiyalardan Memo komponentasini qo'yamiz.



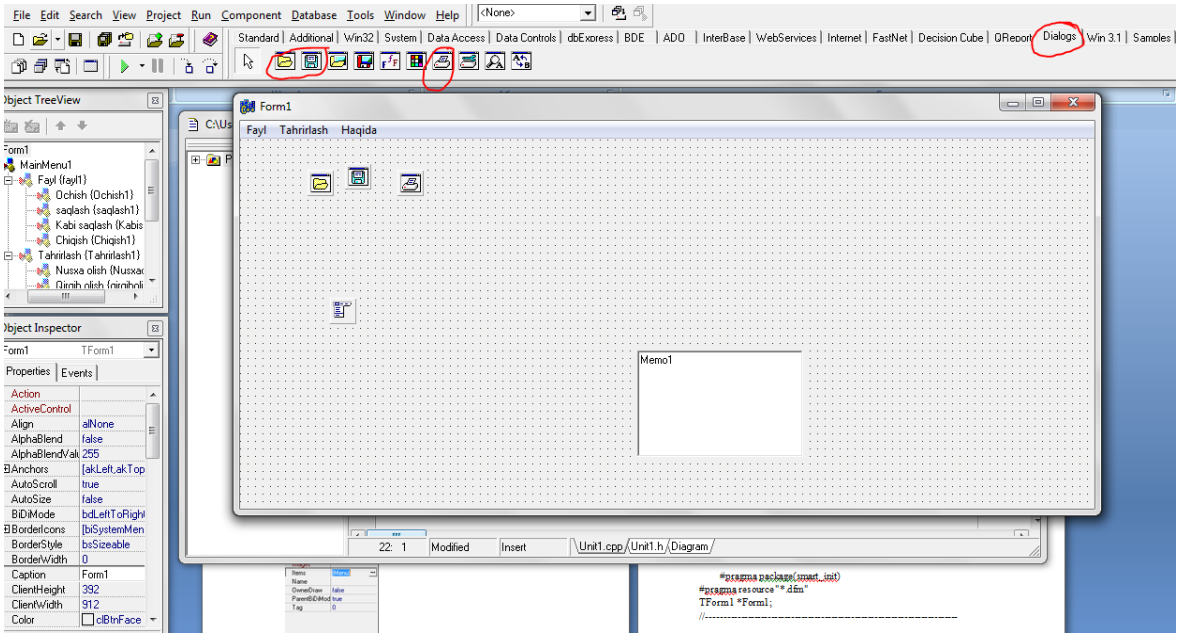
Undan keyin esa Mainmenu ni joylashtiramiz.

Mainmenu ning items xususiyati orqali kerakli menyuni yaratib olamiz.





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



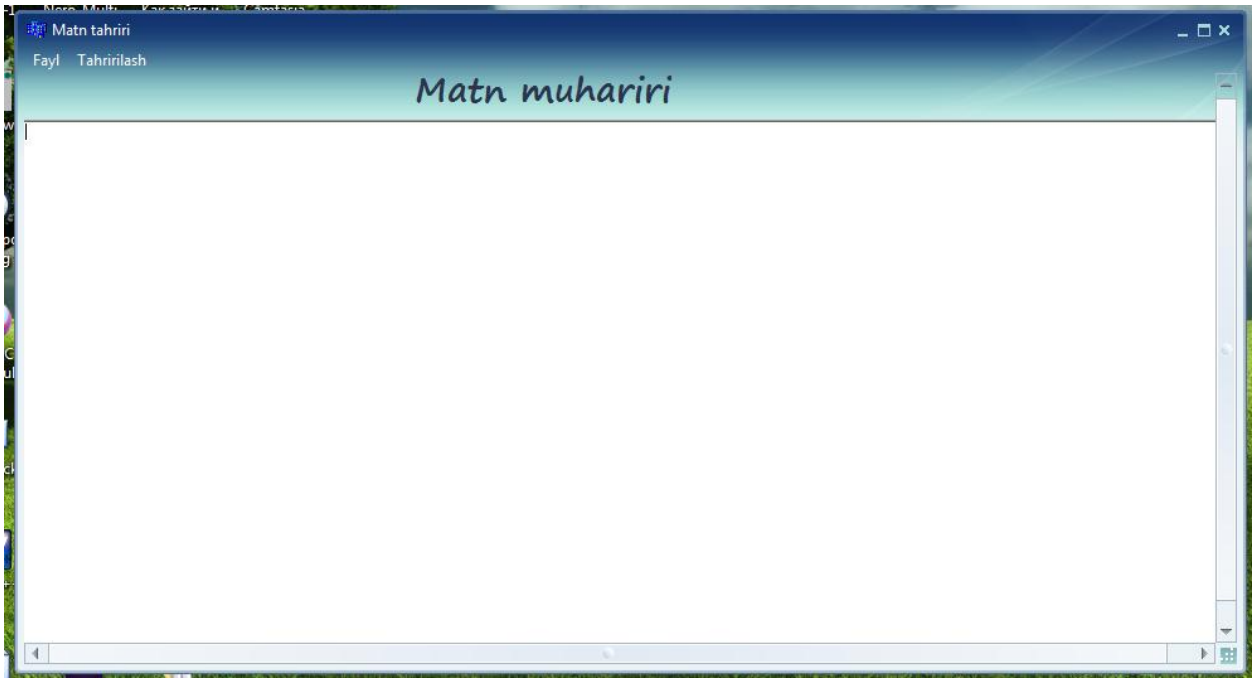
Shundan keyin Mainmenu ga sichqoncha bilan 2 marta bosib kod kiritamiz.

```
//---------------------------------------------------------------------

#include <vcl.h>
#pragma hdrstop

#include "Unit1.h"
//---------------------------------------------------------------------
#pragma package(smart_init)
#pragma link "sSkinManager"
#pragma link "sLabel"
#pragma resource "*.dfm"
TForm1 *Form1;
//---------------------------------------------------------------------
__fastcall TForm1::TForm1(TComponent* Owner)
        : TForm(Owner)
{
}
//---------------------------------------------------------------------

void __fastcall TForm1::Nusxaolish1Click(TObject *Sender)
{
 Memo1->CopyToClipboard();
}
//---------------------------------------------------------------------
```

```
void __fastcall TForm1::Qoyish1Click(TObject *Sender)
{
     Memo1->PasteFromClipboard();
}
//---------------------------------------------------------------------------

void __fastcall TForm1::Qirqibolish1Click(TObject *Sender)
{
       Memo1->CutToClipboard();
}
//---------------------------------------------------------------------------

void __fastcall TForm1::Tozalash1Click(TObject *Sender)
{
       Memo1->Clear();
}
//---------------------------------------------------------------------------




void __fastcall TForm1::Chiqish1Click(TObject *Sender)
{
   Close();
}
//---------------------------------------------------------------------------

void __fastcall TForm1::Saqlash1Click(TObject *Sender)
{
       if(OpenDialog1->Execute())
       Memo1->Lines->LoadFromFile(OpenDialog1->FileName);
}
//---------------------------------------------------------------------------

void __fastcall TForm1::Ochish1Click(TObject *Sender)
{
    if(SaveDialog1->Execute())
    Memo1->Lines->SaveToFile(SaveDialog1->FileName + ".txt");
```

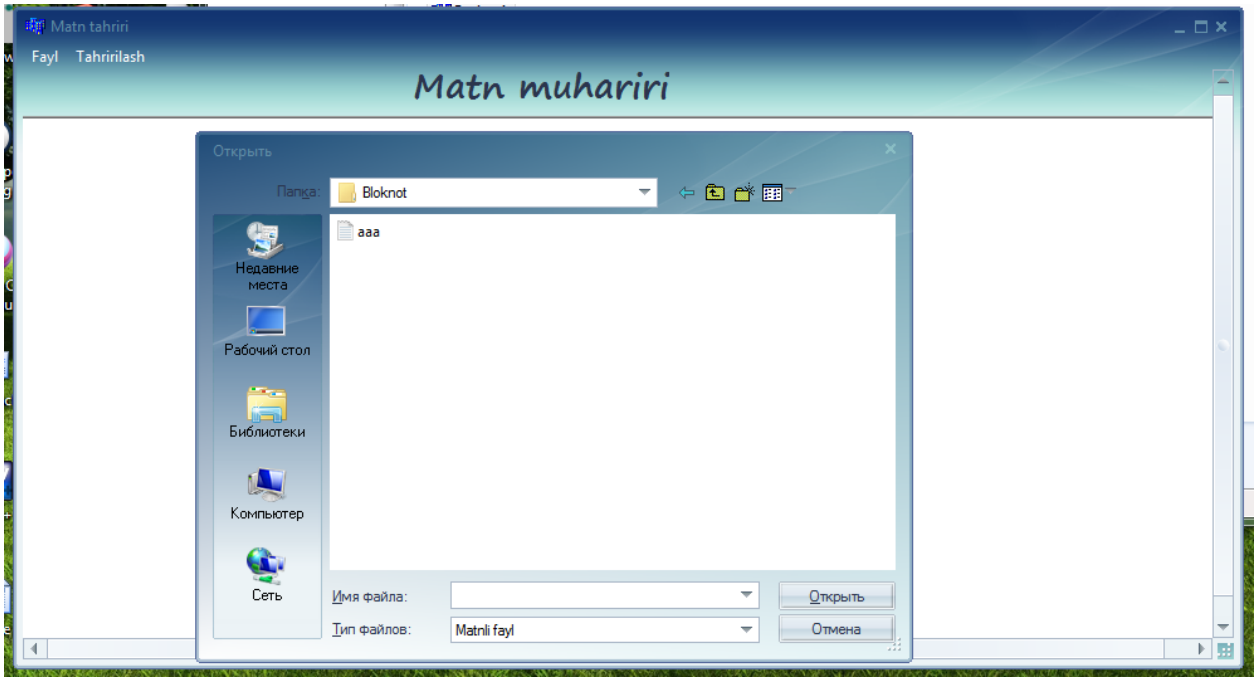
```
}
//---------------------------------------------------------------------------
```

Tuzgan dasturimizni ishlatganda quyidagi ko’rinishga keladi.

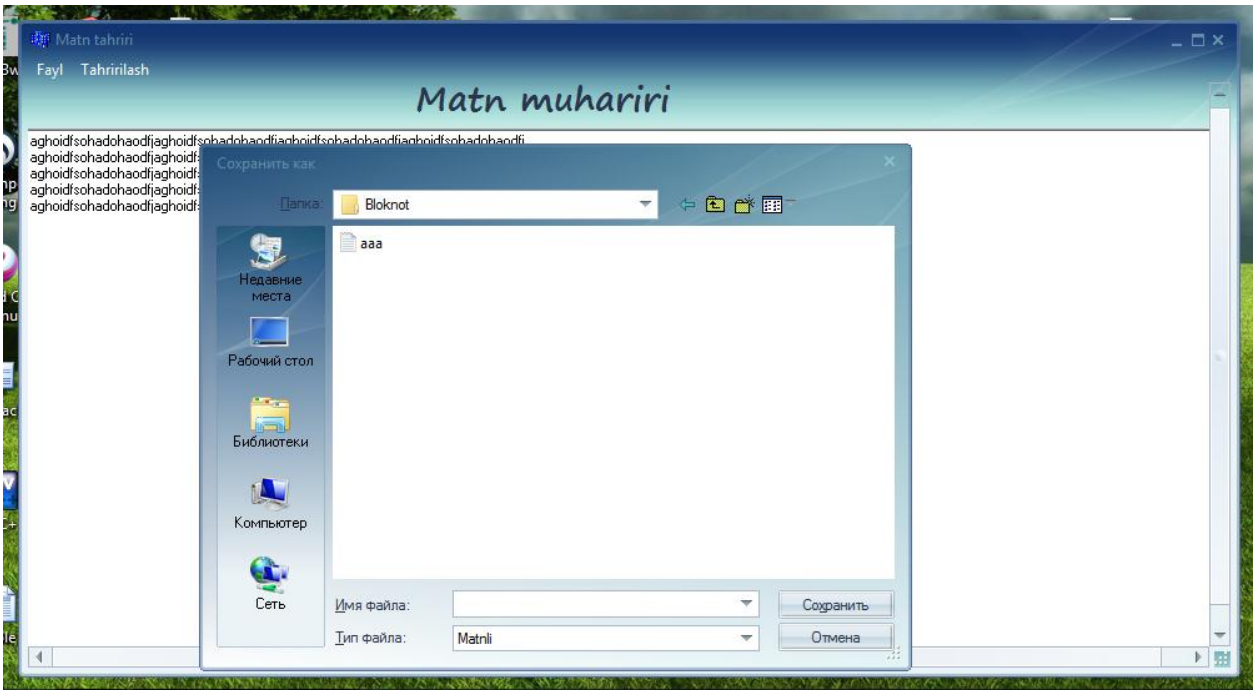


Hosil bo’lgan dasturimdan matn ustidagi quyidagicha amallarni bajarishimiz mumkin.

Tekstli fayllarni ochish.



O’zgartirilgan va yaratilgan fayllarni saqlash



Xulosa

Xullas, 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 chiqishimiz kerak. Dasturni kompilyatsiya qilishdan avval unga tegishli barcha fayllarni saqlab qo’yishimiz zarur. 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 muvofiq. Chunki dasturni boshqa joyga ko’chirishga to’g’ri kelib qolgan vaqtda dasturlarning o’xshash fayllarini adashtirib yuborish mumkin. Bunday holatda dasturda xatolik aniqlanib dastur ishlamay qoladi. Dasturni to’liq saqlash uchun faqat *save* ni emas balki *save all* ni bosib hamma fayllarni saqlab qo’yishimiz kerak.

Foydalanilgan adabiyotlar:

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.akmx.uz
7. www.ziyonet.uz