O’ZBEKISTON RESPUBLIKASI AXBOROT TEXNOLOGIYALARI VA KOMMUNIKATSIYALARINI RIVOJLANTIRISH VAZIRLIGI

TOSHKENT AXBOROT TEXNOLOGIYALARI UNIVERSITETI URGANCH FILIALI

KOMPYUTER INJINIRINGI FAKULTETI

C++ DA DASTURLASH FANIDAN

Kurs ishi

MAVZU: **Yorug’lik intensivlikni aniqlovchi dastur yaratish**

Bajardi:
912-15 guruh talabasi
Raximova Yulduz

Ilmiy rahbar:
Dasturiy injiniring kafedrasi assistenti
Aliyev Oybek

MUNDARIJA

KIRISH……...2

1.1.Masalaning qo`yilishi……...3

NAZARIY QISM……....4

2.1.Ob’ektli dasturlash asoslari. Ob’ektga yo’naltirilgan dasturlash...4

2.2 Dasturini yaratishda foydalanilgan Visual Studio Komponentalari…….....7

ASOSIY QISM……. 14

3.1. Dasturni Visual studio 2010 da yaratish……....14

3.2. Foydalanuvchi uchun yo’riqnoma……..... 15

XULOSA……. 23

FOYDALANILGAN ADABIYOTLAR…… 24

ILOVA……. 25

KIRISH

Bizning oldimizda algoritmlarni EHM tushunadigan tilda yozish masalasi turadi, buning uchun maxsus algoritmik tillar mavjud. EHM paydo bo'lganidan beri juda ko'plab algoritmik tillar yaratilgan. Ularni shartli ravishda uch tipga ajratish mumkin:

- Quyi darajadagi programmalash tillari (mashina tillari)
- O'rta darajadagi programmalash tillari
- Yuqori bosqichli programmalash tillari

Quyi darajadagi programmalash tillarida buyruqlar va amallar ma'lum kodlar (raqamlar) bilan ifodalangan bo'lib, ular EHM qurilmalari adreslari bilan bevosita ishlashga mo'ljallangan va mashina tili deb ham yuritiladi.

O'rta darajadagi programmalash tillarida amallar va buyruqlarni (komandalar)ni ifodalash uchun odam tushunishi uchun qulay bo'lgan har xil qisqartma so'zlardan foydalaniladi. Bunda ham programma tuzuvchi EHM qurilmalari adreslari bilan ishlashi lozim. Bu qiskartmalar-mnemokodlar deyiladi, bu turdagi programmalash tillari assemblerlar deb ataladi.

Yuqori darajadagi programmalash tillarida ko'rsatmalar inson tiliga yaqin bo'lgan so'zlar va birikmalardan iborat bo'lib, programma tuzish uchun juda qulay va biror maxsus amallardan tashqari hollarda adreslar va qurilmalar bilan bevosita bog'liq ko'rsatmalarni bilish zarur emas. Yuqori bosqich programmalash tillariga misol sifatida FORTRAN, Algol, RL-1, BASIC, PASCAL, C, C++ tillarini keltirish mumkin.

Masalani qo’yilishi.

Visual studio C++ dasturlashda Yorug’lik intensivlikni aniqlovchi dastur yaratish.Yorug’lik intensivligini aniqlashni yuzalarga bog’liq holda topish.

Kurs ishini bajarilishi uchun quyidagi talablar qo`yildi:

1. Yorug’lik intensivligini aniqlovchi dastur tuzish
2. Yorug’lik Intensivligni yuzalarga bog’liq holda topish.
3. Dastur oddiy foydalanuvchi uchun tushinarli bo`lishi lozim
4. Dasturni visual studio imkoniyatlaridan foydalanib yaratish

NAZARIY QISM

2.1 Ob’yektga yo’naltirilgan dasturlash tillari haqida umumiy tushuncha

Ob’ektga mo‘ljallangan yondoshuv (OMYO) bir kunda o‘ylab topilgan emas. Uning paydo bo‘lishi dasturiy ta’minotning tabiiy rivojidagi navbatdagi pog‘ona xolos. Vaqt o‘tishi bilan qaysi uslublar ishlash uchun qulay-u, kaysinisi noqulay ekanini aniqlash oson bo‘lib bordi. OMYO eng muvaffaqiyatli, vaqt sinovidan o‘tgan uslublarni o‘zida samarali mujassam etadi.

Dastlab dasturlash anchayin boshqotirma ixtiro bo‘lib, u dasturchilarga dasturlarni kommutatsiya bloki orqali kompyuterning asosiy xotirasiga to‘g‘ridan-to‘g‘ri kiritish imkonii berdi. Dasturlar mashina tillarida ikkilik tasavvurda yozilar edi. Dasturlarni mashina tilida yozishda tez-tez xatolarga yo‘l qo‘yilar edi, eng ustiga ularni tuzilmalashtirish imkoni bo‘lmagani tufayli, kodni kuzatib borish amalda deyarli mumkin bo‘lmagan hol edi. Bundan tashqari, mashina kodlaridagi dastur tushunish uchun g‘oyat murakkab edi.

Vaqt o‘tishi bilan kompyuterlar tobora kengroq qo‘llana boshladi hamda yuqoriroq darajadagi protsedura tillari paydo bo‘ldi. Bularning dastlabkisi FORTRAN tili edi. Biroq ob’ektga mo‘ljallangan yondoshuv rivojiga asosiy ta’sirni keyinroq paydo bo‘lgan, masalan, ALGOL kabi protsedura tillari ko‘rsatdi. Protsedura tillari dasturchiga axborotga ishlov berish dasturini pastroq darajadagi bir nechta protseduraga bo‘lib tashlash imkonini beradi. Pastroq darajadagi bunday protseduralar dasturning umumiy tuzilmasini belgilab beradi. Ushbu protseduralarga izchil murojaatlar protseduralardan tashkil topgan dasturlarning bajarilishini boshqaradi.

Dasturlashning bu yangi paradigmasi mashina tilida dasturlash paradigmasiga nisbatan ancha ilg‘or bo‘lib, unga tuzilmalashtirishning asosiy vositasi bo‘lgan protseduralar qo‘shilgan edi, Maydaroq funktsiyalarni nafaqat

tushunish, balki sozlash ham osonroq kechadi. Biroq, boshqa tomondan, protsedurali dasturlash koddan takroran foydalanish imkonini cheklab qo‘yyadi. Buning ustiga dasturchilar tez-tez «makaron» dasturlar ham yozib turishganki, bu dasturlarni bajarish likopdagi spagetti uyumini ajratishga o‘xshab ketar edi. Va, nihoyat, shu narsa aniq bo‘ldiki, protsedurali dasturlash usullari bilan dasturlarni ishlab chiqishda diqqatni ma’lumotlarga qaratishning o‘zi muammolarni keltirib chiqarar ekan. Chunki ma’lumotlar va protsedura ajralgan, ma’lumotlar inkapsullanmagan. Bu nimaga olib keladi? Shunga olib keladiki, har bir protsedura ma’lumotlarni nima qilish kerakligini va ular qaerda joylashganini bilmog‘i lozim bo‘ladi. Agar protsedura o‘zini yomon tutsa-yu, ma’lumotlar ustidan noto‘g‘ri amallarni bajarsa, u ma’lumotlarni buzib qo‘yishi mumkin. Har bir protsedura ma’lumotlarga kirish usullarini dasturlashi lozim bo‘lganligi tufayli, ma’lumotlar taqdimotining o‘zgarishi dasturning ushbu kirish amalga oshirilayotgan barcha o‘rinlarining o‘zgarishiga olib kelar edi. Shunday qilib, xatto eng kichik to‘g‘rilash ham butun dasturda qator o‘zgarishlar sodir bo‘lishiga olib kelar edi.

Modulli dasturlashda, masalan, Modula2 kabi tilda protsedurali dasturlashda topilgan ayrim kamchiliklarni bartaraf etishga urinib ko‘rildi. Modulli dasturlash dasturni bir necha tarkibiy bo‘laklarga, yoki, boshqacha qilib aytganda, modullarga bo‘lib tashdlaydi. Agar protsedurali dasturlash ma’lumotlar va protsedtsralarni bo‘lib tashlasa, modulli dasturlash, undan farqli o‘laroq, ularni birlashtiradi. Modul ma’lumotlarning o‘zidan hamda ma’lumotlarga ishlov beradigan protseduralardan iborat. Dasturning boshqa qismlariga moduldan foydalanish kerak bo‘lib qolsa, ular modul interfeysiga murojaat etib qo‘yaqoladi. Modullar barcha ichki axborotni dasturning boshqa qismlarida yashiradi.

Biroq modulli dasturlash ham kamchiliklardan holi emas. Modullar

kengaymas bo‘ladi, bu degani kodga bevosita kirishsiz hamda uni to‘g‘ridan-to‘g‘ri o‘zgartirmay turib modulni qadamma-qadam uzgartirish mumkin emas. Bundan tashqari, bitta modulni ishlab chiqishda, uning funktsiyalarini boshqasiga o‘tkazmay (delegat qilmay) turib boshqasidan foydalanib bo‘lmaydi. Yana garchi modulda turni belgilab bo‘lsa-da, bir modul boshqasida belgilangan turdan foydalana olmaydi.

Modulli va protsedurali dasturlash tillarida tuzilmalashtirilgan va tuzilmalashtirilmagan ma’lumotlar o‘z «tur»iga ega. Biroq turni kengaytirish usuli, agar «agregatlash» deb ataluvchi usul yordamida boshqa turlarni yaratishni hisobga olmaganda, mavjud emas.

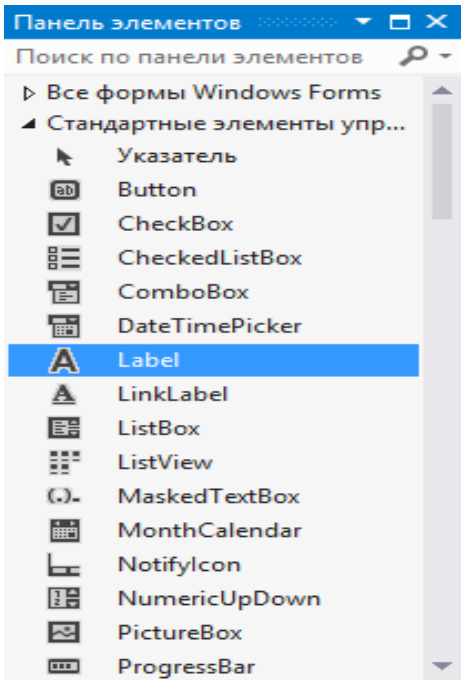
Va, nihoyat, modulli dasturlash - bu yana protseduraga mo‘ljallangan gibridli sxema bo‘lib, unga amal qilishda dastur bir necha protseduralarga bo‘linadi. Biroq endilikda protseduralar ishlov berilmagan ma’lumotlar ustida amallarni bajarmaydi, balki modullarni boshqaradi.

Ob’ektga mo‘ljallangan dasturlash (OMD) modulli dasturlashdan keyingi mantiqiy pog‘onani egallaydi, u modulga nasldan-naslga o‘tishni va polimorfizmni qo‘shadi. OMD dan foydalanr ekan, dasturchi dasturni bir qator oliy darajali ob’ektlarga bo‘lish yo‘li bilan tizimlashtiradi. Har bir ob’ekt hal qilinayotgan muammoning ma’lum bir tomonini modellashtiradi. OMD endilikda dasturni bajarish jarayonini boshqarish uchun dasturchi diqqatini protseduralarni ketma-ketlikda chaqirib olish ro‘yxatini tuzib o‘tirishga qaratmaydi. Buning o‘rniga ob’ektlar o‘zaro aloqada bo‘ladi. OMYO yordamida ishlab chiqilgan dastur hal qilinayotgan muammoning amaldagi modeli bo‘lib xizmat qiladi.

2.2 Dasturini yaratishda foydalanilgan Visual Studio komponentalari

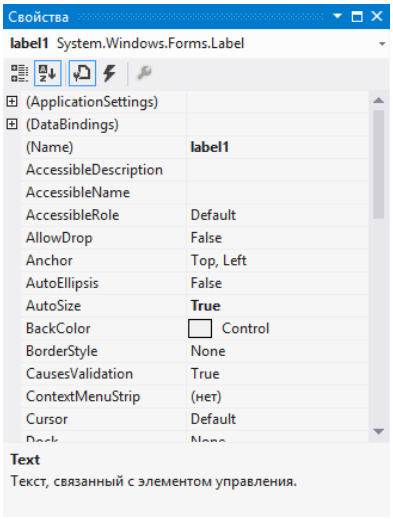
Label Yozuv (label komponenti) yorliqni ifodalaydi va ko`pincha o`zining Caption xususiyatiga ega bo`lmagan boshqa boshqaruv elementlarining sarlavasi sifatida ishlatiladi. Ko`pincha yozuvlarni tasvirlash uchun nishon deb nomlanadigan Label kompanenti ishlatiladi. U dastur bajarilish vaqti foydalanuvchi tahrirlashi mumkin bo`lgan oddiy matnni tasvirlash uchun ishlatiladi.

Visual Studio dasturida Label kompanentasini ishlatish uchun “Панель элементов” oynasida Label nomli kompanenta tanlanadi.



2.1-rasm. Label komponentasini ochish

Label komponentining xususiyatlari(Properties) (2.2-rasm)da ko`rsatilgan.



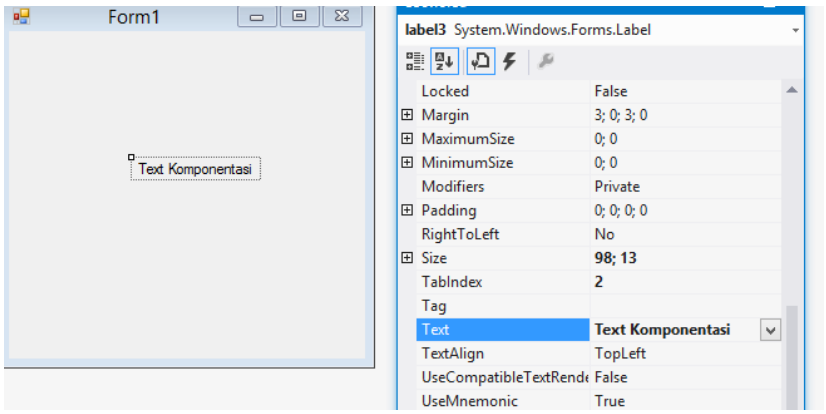
2.2-rasm. Label komponentining xususiyatlari

Bu xususiyatlardan bir nechtasini ko`rib chiqamiz.

BorderStyle — bu xususiyat orqali Label komponentasini chegara sohasini o`zgartirish mumkin.

Chegara sohasining uch xil ko`rinishi mavjud: none, (bir chiziqli) окантовка одной линией , окантовка под трехмерное пространство(uch o`lchovli);

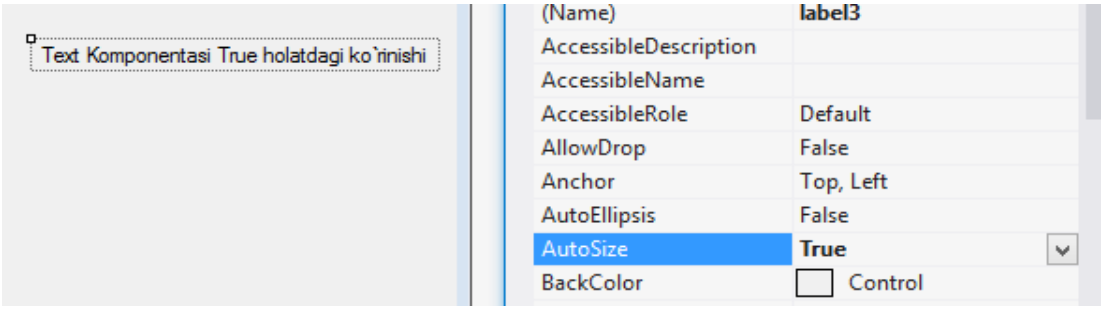
Text — xususiyatida Label komponentiga kiritiladigan matn yoziladi.(2.3-rasm)



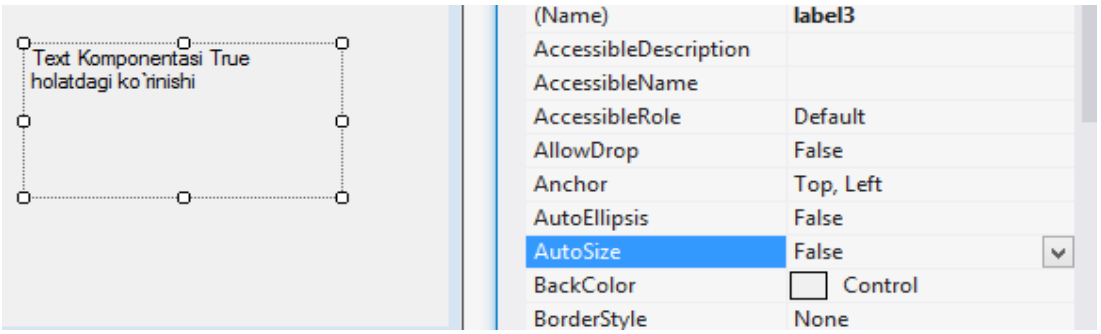
2.3-rasm. Text xususyati bilan ishlash

AutoSize — Nishon matniga bog`liq ravishda Label kompanenti o`lchamlarini avtomatik rostlashni boshqarish uchun Boolean turidagi AutoSize xususiyati xizmat qiladi. Agar xususiyat (odatdagidek) True qiymatga ega

bo`lsa, Label kompanenti Caption xususiyatidagi matnga ko`ra o`lchamlarini o`zgartiradi.(2.4,2.5-rasmlar)



2.4-rasm. AutoSize xususyati bilan ishlash



2.5-rasm. AutoSize xususyati bilan ishlash

TextAlign — yozilgan matnning Label chegarasi bo`ylab qanday joylashishini belgilash. Label kompanenti ichida matnni rostlash uchun quyidagi qiymatlarni qabul qiluvchi TAlignment turidagi Alignment xususiyati qo`llaniladi:

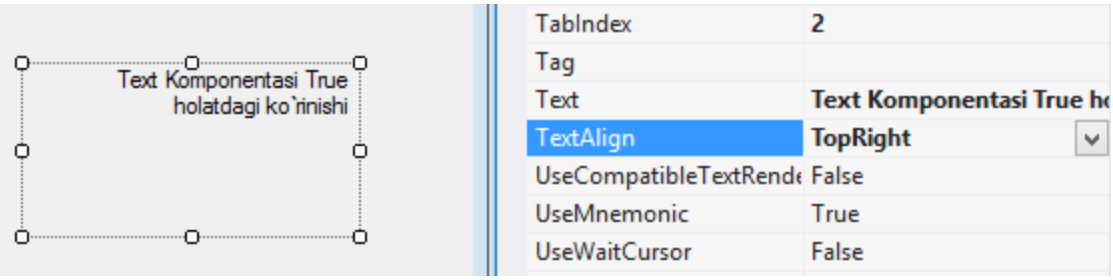
- TopLeft– chap yon bo`ylab rostlash;

- TopCenter – matnni markazlashtirish;

- TopRight – o`ng yon bo`ylab rostlash.

Nishonning shoffof yoki bo`yalganligi Boolean turidagi Transporent xususiyati belgilaydi. Bo`yoq rangi Color xususiyati yordamida o`rnatiladi.

Odatda, Transporent xususiyati False qiymatiga ega va nishon noshaffof bo`ladi. Shaffof Label kompanenti nishon rasm ustida joylashtirilganda va tasvirni yopmasligi zarur bo`lgan hollarda kerak bo`lishi mumkin. Masalan, geagrafik xaritada. (2.6-rasm)

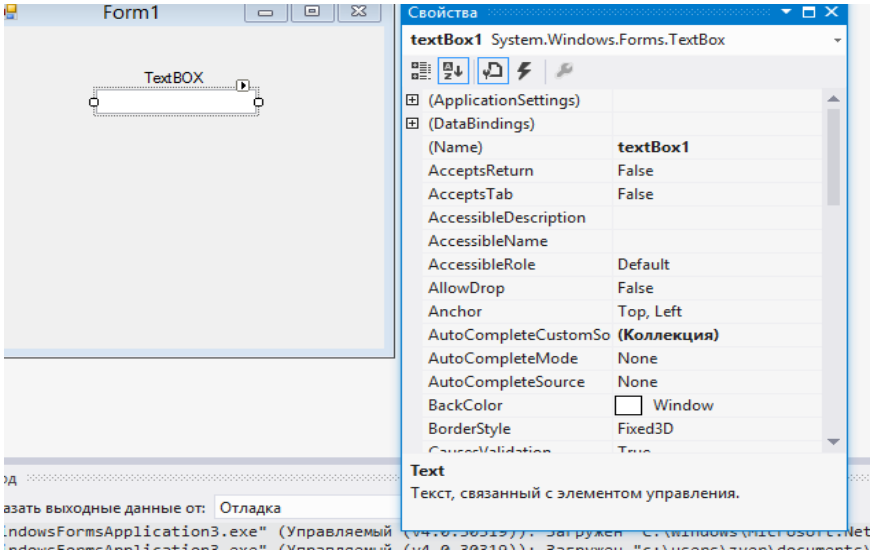


2.6-rasm TextAlign xususyati bilan ishlash

TextBox komponentasi

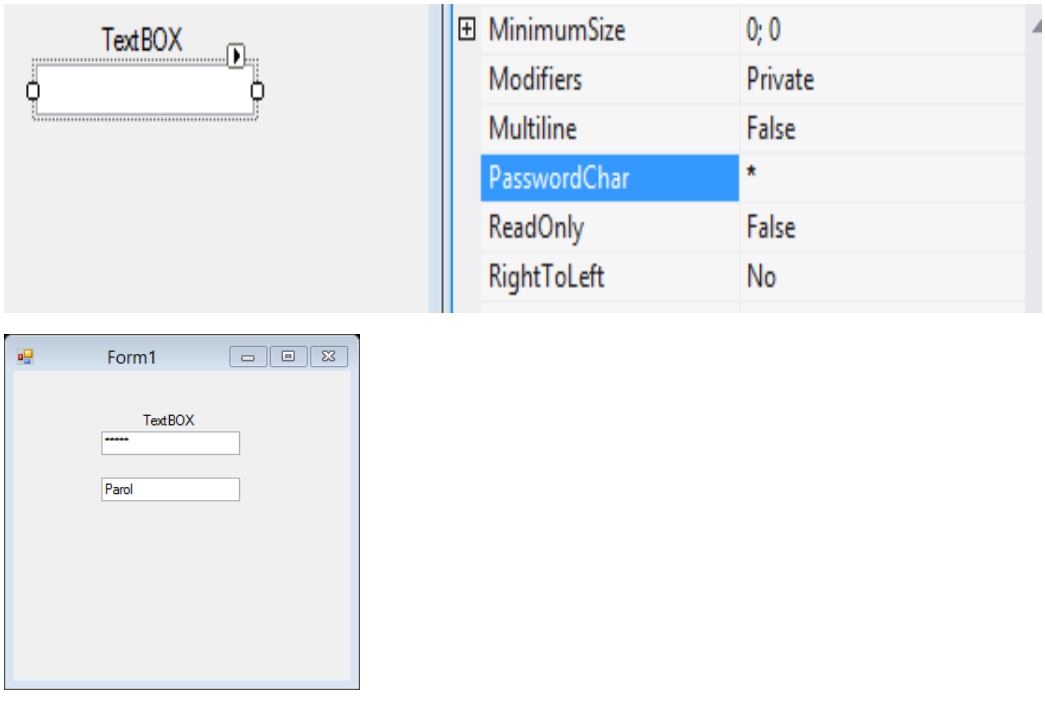
Bir satrli redaktor matn tasvirlanishi va o`zgartirilishi mumkin bo`lgan ma`lumot kiritish maydonini ifodalaydi. Visual Studioda bir qancha bir satrli redaktorlar mavjud bo`lib, ulardan TextBox kompanenti ko`p qo`llaniladi.

TextBox kompanenti klaviaturadan turli simvollarni kiritish va tahrirlashga imkon beradi. Bunda boshqaruv tugmalaridan foydalanib satr bo`ylab siljish, <BackSpace> va <Delete> tugmalari yordamida simvollarni o`cherish, matn (bo`lagini ) qismini belgilash va boshqa amallarni bajarish mumkin. Ta`kidlash joizki, bir satrli redaktorlar <Enter> va <Esc> boshqaruv tugmalariga javob bermaydi.(2.7-rasm)



2.7-rasm. TextBox kompanenti bilan ishlash

TextBox kompanentidan parolni kiritishda foydalanganda Char turidagi PasswordChar xususiyatidan foydalanish mimkin. U kiritish oynasiga tasvirlanadigan simvolni belgilaydi. Bu simvol matn kiritayotganda haqiqatda kiritilgan ma`lumot o`rnida paydo bo`ladi. Masalan, quyidagi(2.8-rasm)



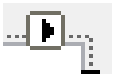
2.8-rasm. PasswordChar xususiyatidan foydalanish

Operatorlari bajarilganda so`ng (tahrirlash) redaktor satrida ***** satri

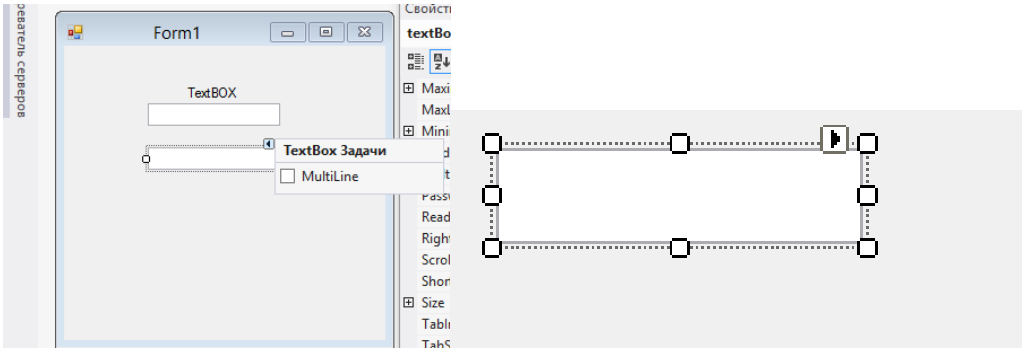
paydo bo`ladi. Aslida esa Text xususiyati Parol qiymatga ega bo`ladi.

Odatda PasswordChar xususiyati #0 qiymatga ega bo`ladi va redactor satrida real (haqiqiy) kiritilgan ma`lumot tasvirlanadi.

TextBox komponentasi xususiyatlari

 Textbox komponentida o`ng tomonda joylashgan uchurchak belgisiga bosilsa(2.3-rasm)da ko`rsatilgan kichik oyna chiqadi.

MultiLine – bu xususiyat True holatda bo`lsa TextBox komponentini maydoni rostlashni boshqarish mumkin bo`ladi.



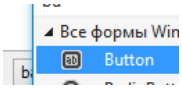
2.9-rasm. MultiLine xususiyati orqali maydoni rostlash

Button komponentasi

Tugmalar boshqaruv elementlari hisoblanadi va ma`lum funksional vazifalarni bajarishga buyruq berish uchun ishatiladi. Tugma yuzasida matn va yoki rasm tasvirlanishi mumkin.

Visual studio tizimi turli variantdagi tugmalarni tavsiya etuvchi bir qancha kompanentalarga ega. Quyidagi turdagi tugmalar mavjud :

- Button standart tugma;
- BitBtn rasmli tugma;
- SpeedButton tez murojaat tugmasi;



Visual studio standart tugma Button kompanenti yordamida berilgan. Tugma yuzasidagi bosilganda bajariladigan ish-harakatni tavsiflovchi

yozuvga ega bo`lishi mumkin.

Tugma uchun asosiy hodisa u bosilganda yuzaga keladigan OnClick hodisasi hisoblanadi. Bunda tugma bajarilayotgan harakatni Visual (ko`rinarli) tasvirlovchi mos ko`rinishga ega bo`ladi. OnClick hodisasini qayta ishlovchi qism dasturda joylashgan harakat tugma qo`yib yuborilishini bilanoq bajariladi.

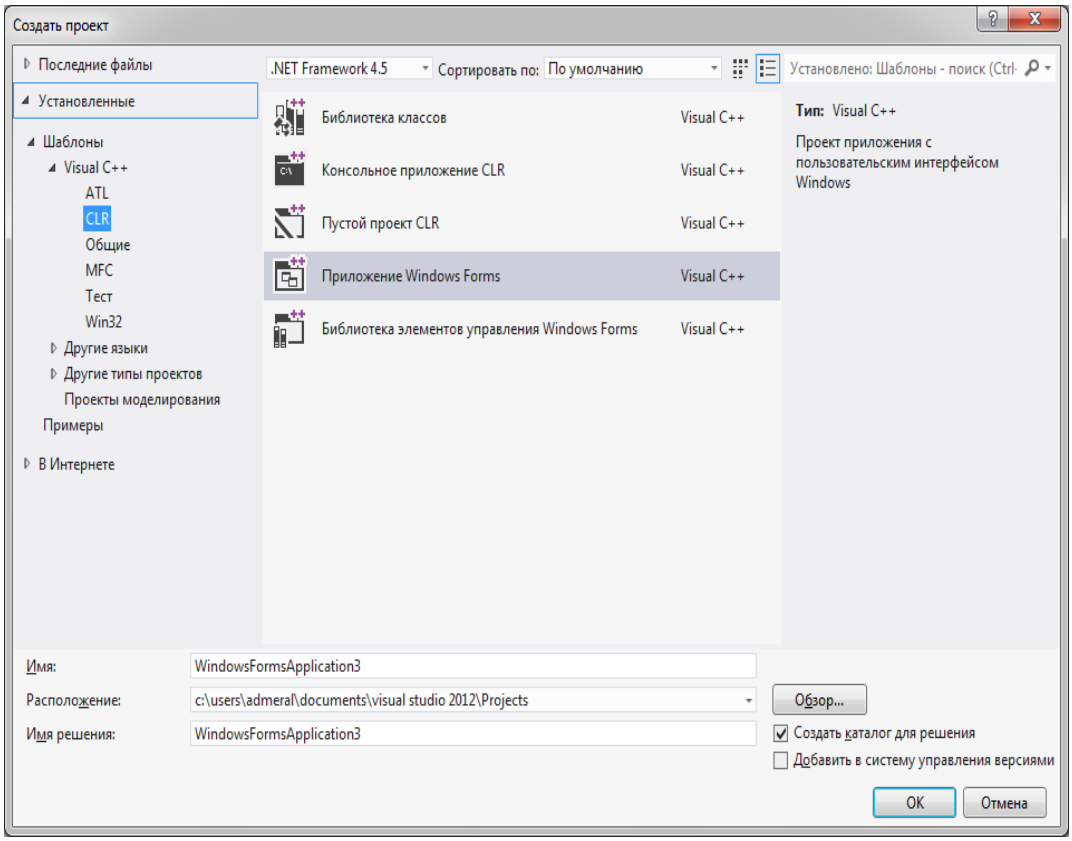
Tugmani quyidagi usullar bilan bosish mumkin:

- sichqonchani qirsillatish bilan ;
- Caption xususiyatida berilgan tugmalar majmuini tanlash bilan;
- <Enter> yoki probel tugmalarini bosish bilan;
- <Esc> tugmasini bosish bilan.

ASOSIY QISM

3.1 Dasturni Visual studio 2010 da yaratish

Visual Studio 2010 dasturida form oynalar yaratish uchun quidagi ketma-ketlik bajariladi.



1-rasm

Dastur yaratilishida oldi bizga qanday komponentalar kerak ekanligini bilishimiz lozim va bu komponentalarni form oynaga joylashtirishimiz lozim.

Masalan, dastur menyu qismi uchun menustript komponentasi tanlanadi:

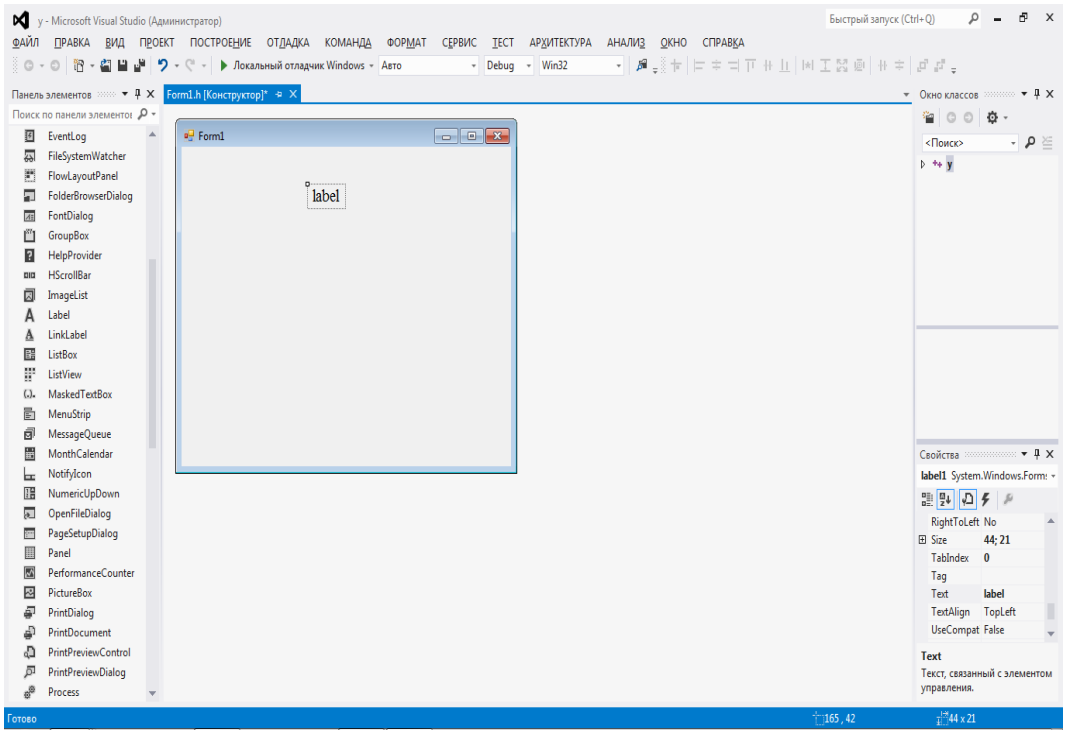
Bu dasturimizning asosiy maqsadi Visual Studio da yorug’lik intensivligini aniqlovchi dastur yaratish.Buning uchun yorug’lik o’zi nima degan savolga to’xtalib o’tamiz.Yorug’lik to’g’risidagi birinchi tassavurlar

qadimgi greklar va misrliklarda paydo bo’lgan 17 asr oxiriga kelib yorug’likning ikkita nazaryasi I.Nyuton tomonidan korpuskulyar nazariyasi

R.Guk va X.Gyuygens tomonidan to’lqin nazariyasi shakillana boshladi.

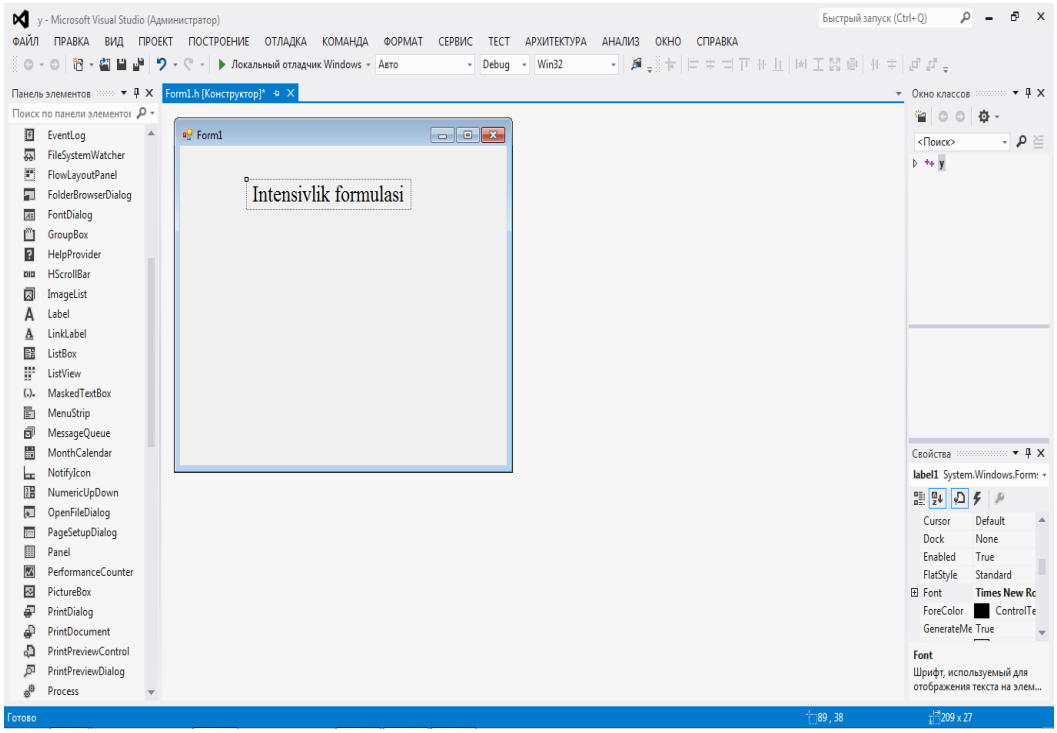
Yaniy Nyuton yorug’likni zaracha deb aytgan bo’lsa Gyuygens esa uni to’lqin deb tariflagan shundan kelib chiqan holda yorug’lik biron jisimga tushganda to’lqin bo’lib keladi va zara bo’lib yutiladi degan xulosaga kelishgan. Bundan kelib chiqib yorug’lik to’lqin ham zara ham ekan.

Yorug’lik intensivligi bu asosan yuzalarga bog’liq bo’lib biron vaqtda yuza birligiga tushgan energiya hisoblanadi.I=W/S*t formulaga asosan yorug’lik intensivlikini hisoblovchi dastur yaratamiz.



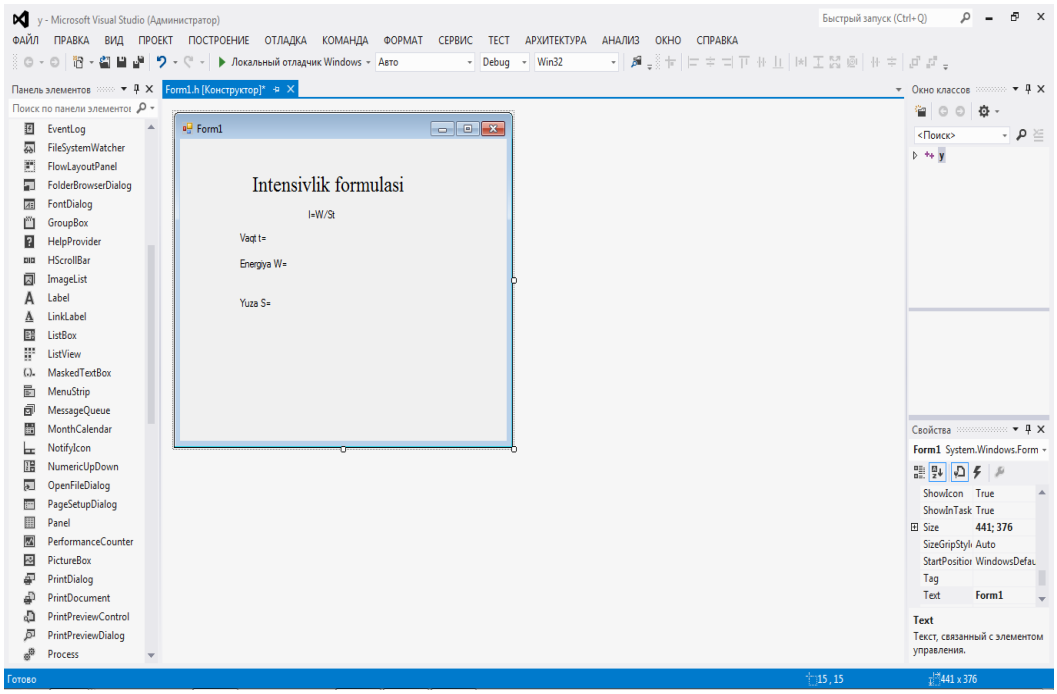
2-rasm

Shu ketma- ketlikda dasturni tuzilishini tayorlab olamiz:



3-rasm

Intensivlik formulasini kiritamiz.



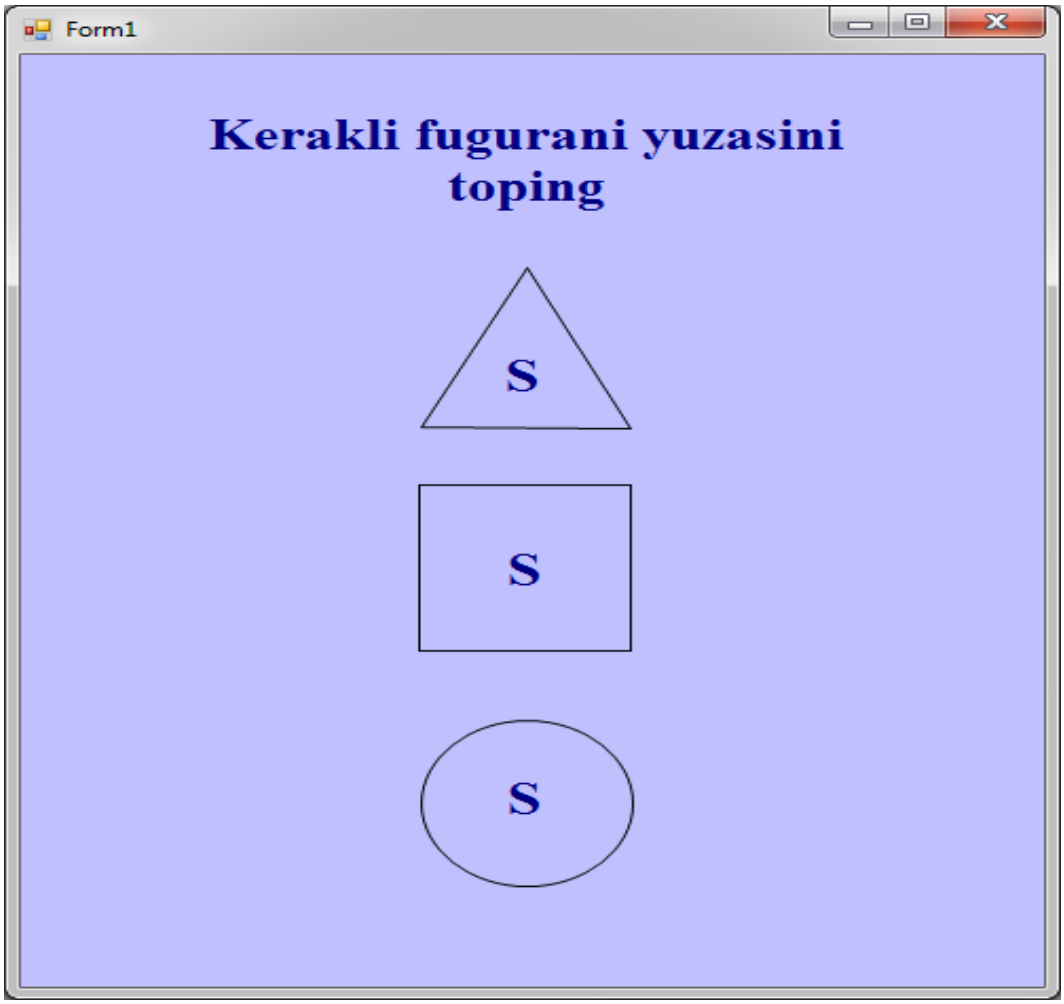
4-rasm.

Dastur umumiy ko`rinishi:

5-rasm

Biz dasturimizni ishga tushradigan bo’lsak birinchidan formulaga asosan ularga qiymat beramiz va turli xil yuzalarga tushayotgan intensivlikni aniqlaymiz.Asosan bu dasturimizda uchta yuzaga tushayotgan yorug’lik intensivligining yuzalari aniqlangan, birinchi yuzamiz uchburchak, ikkinchi yuzamiz kvadrat uchinchi yuzamiz doira hisoblanadi.

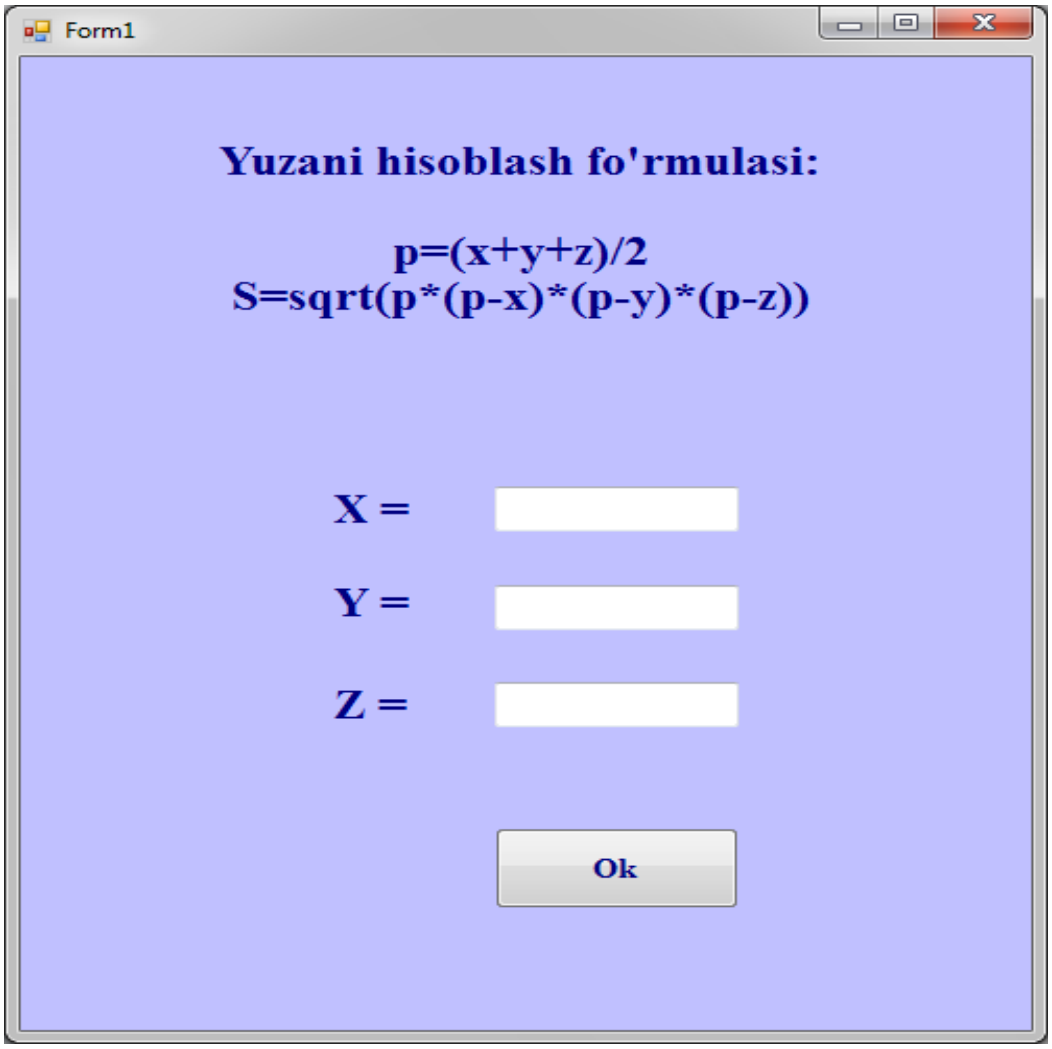
Vaqtga energiyaga hamda qiymat beradigan bo’lsak yuzaga kelganda ikkinchi oynamiz hosil bo’adi.



6-rasm.

Datur ishga tushirilgan holati

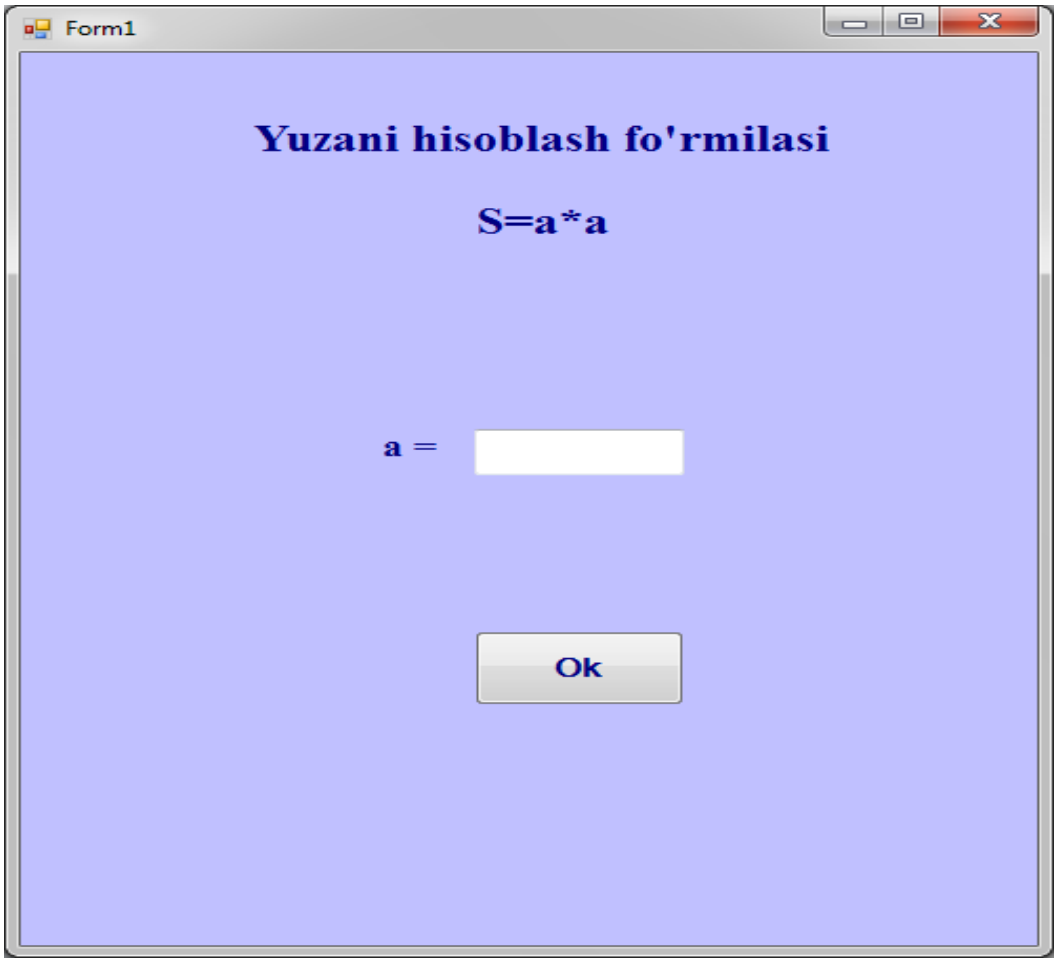
Birinchi uchburchak yuzi bu ixtiyoriy uchburchak yuzi hisoblanadi Shuning uchun yarim peremetrni hisoblab Geron formulasiga asosan yuzani hisoblaymiz.



7-rasm.

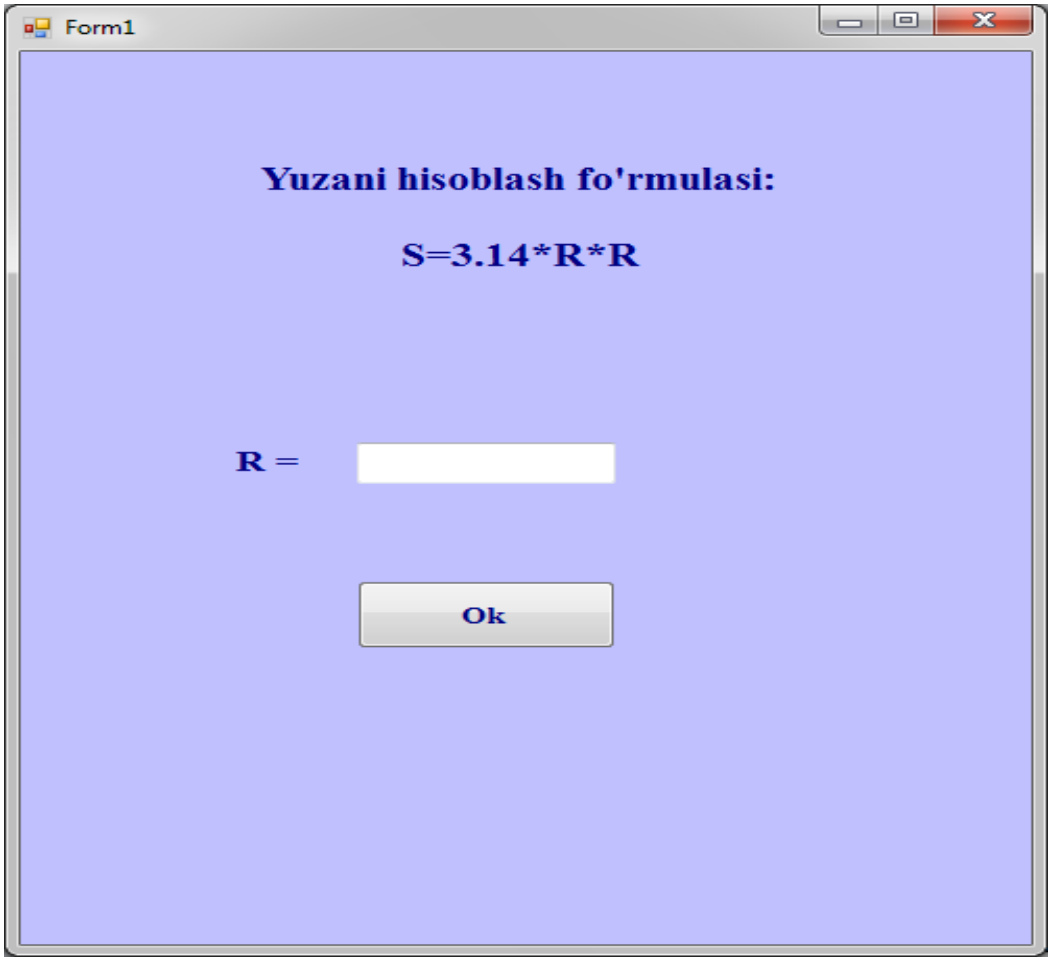
TextBox ga qiymat bersak formulaga asosan yuzani hisoblab natijani birinchi oynamizga label da ko’rsatadi.

Ikkinchi yuzamiz kvadrat yuzi hisoblanadi .



7-rasm.

Uchinchi yuzamiz doira yuzi.



8-rasm.

Dasturni ishga tushiradigan bo’lsak.

9-rasm

Agar ixtiyoriy yuzani tanlab unga qiymat berganimizda formulaga asosan yuzani hisoblab natijani birinchi oynamizga yuboradi va natijani yorug’lik intensivligiga asosan hisoblab label da ko’rsatadi.

Xulosa.

Dastur yaratish jarayonida ko’plab malumotlarga ega bo’lindi. Yorug’lik intensivligini aniqlab beruvchi dasturimiz ko’plab imkoniyatlarni beradi asosan yorug’lik intensivligini asosoiy formulasi uni kelib chiqishi haqida bir qancha tushunchaga ega bo’ldindi. Dastur yaratilish jarayonida visual studio imkoniyatlaridan foydalanib bu imkoniyatlar haqida ma`lumotga va malakaga hosil qilindi. Yorug’lik turli xil yuzalarga tushganida undagi intensivlik ham o’zgaradi shuning uchun ham yuzallarni turli xil qilib tanladi.

1.Birinchi ixtiyoriy uchburchakni yuzi hisoblash formulasi kirtildi

2.Ikkinchi kvadratni yuzini hisoblovchi formula kiritildi.

3.Uchinchi doira yuzini hisoblovchi formula kiritildi.

Masalaning qoyilgan talablari quyidagicha bajarildi:

1 Yorug’lik intensivlikni hisoblovchi dastur tuzish.

2 Yorug’lik intensivligini yuzalarga bog’liq holda hisoblash .

3 Dastur oddiy foydalanuvchi uchun tushinarli hosil qilindi.

4 Dasturni visual studio imkoniyatlaridan foydalanib yaratildi.

Foydalanilgan adabiyotlar:

I.Asosiy adabiyotlar:

1. Борис Пахомов – C,C++ и MSVisualC++ 2012 для начинающих Санкт-Петербург «БХВ-Петербург» 2013
2. Зиборов В. – MS Visual C ++2010 в среде NET Питер, 2012
3. Khorton_Visual_C__2010_Polny_kurs-2011g Санкт-Петербург 2011
4. Horton.I-Beginning Visual C++ 2012.

II.Qo’shimcha adabiyotlar:

1.И.В.Савельев-Umumiy fizika kursi 3-tom.

2.Abduraxmanov Q.Egamov-Fizika darslik.

ILOVA

```
private: System::Void button1_Click(System::Object^  sender,
                                    System::EventArgs^  e) {

if (textBox1->Text != "" && textBox2 ->Text != "" && label_s->Text != "" )
                                    }
          t = Convert::ToInt32(textBox1->Text);
          w = Convert::ToInt32(textBox2->Text);
          String ^ p = "" , ^ p1 = label_s->Text;
          a1=p1->Length;
          n=1;
          s=0;
          for ( i=0 ;i<a1 ;i++){
          p=p1[i]+p;}
          for ( i=0 ;i<a1 ;i++){
          if (p[i] != ',' && p[i] != '.' ){
          s=s+((char)p[i]-48)*n;
          n*=10;}
          else{
          s=s/n;
          n=1;}}
          i=w/(s*t);
          label_i->Text = Convert::ToString(i);}
            if ( textBox1->Text == "" && textBox2 ->Text == "" && label_s-
      >Text == "" ){
       MessageBox::Show(" t, W, S - ni qiymatini kiriting "," Ошбика");}


             else {
```

```
 if ( textBox1->Text == "" && textBox2 ->Text == ""  ){
 MessageBox::Show(" t, W - ni qiymatini kiriting "," Ошбика");}
 else {
 if (  textBox2 ->Text == "" && label_s->Text == "" ){
 MessageBox::Show(" W, S - ni qiymatini kiriting "," Ошбика");}
 else {
 if ( textBox1->Text == ""   ){
 MessageBox::Show(" T - ni qiymatini kiriting "," Ошбика");}
 else {
 if (  textBox2 ->Text == ""  ){
 MessageBox::Show(" W - ni qiymatini kiriting "," Ошбика");}
 else if (    label_s->Text == "" ){

 MessageBox::Show(" S - ni qiymatini kiriting "," Ошбика");}}}}}
  private:   System::Void   button3_Click(System::Object^      sender,
System::EventArgs^   e) { if (textBox4->Text != "" && textBox5 ->Text != ""
&& textBox6->Text != "" ){
 x = Convert::ToInt32(textBox4->Text);
 y = Convert::ToInt32(textBox5->Text);
 z = Convert::ToInt32(textBox6->Text);
 if (x+y>z && x+z>y && y+z>x){
 a1=0.1;
 p=(x+y+z)/2;
 s=(p*(p-x)*(p-y)*(p-z));
 p=0;
 if (s > 1000){
  a=s;}
```

```
else{
a=1000;}
for (i=1;i<a;i++){
if (i == s/i){
label_s ->Text = Convert::ToString(i);
panel -> Controls-> Clear();
panel -> Controls-> Add(this->panel1);
return;}
if (i*i < s ){
x1 = i;}
else { y1=i; i = a; }{
for ( i=1;i<1000;i++){
if ( (x1+a1)*(x1+a1) < s ){
x1=x1+a1;}
else{ if ( (x1+a1)*(x1+a1) > s ){
y1=x1+a1;}}
if ( (y1-a1)*(y1-a1) < s ){
x1=y1-a1;}
else{ if ( (y1-a1)*(y1-a1) > s ){
y1=y1-a1;}}
if ( x1+a1 == y1 ){
a1=a1/10;}}
label_s ->Text = Convert::ToString(x1);
panel -> Controls-> Clear();
panel -> Controls-> Add(this->panel1);}

else {
```

```
        MessageBox::Show(" Bu uzunliklardan uch burchak yasap bo'lmaydi "," Ошбика");} }
        if ( textBox4->Text == "" && textBox5 ->Text == "" && textBox6->Text == "" ){
                MessageBox::Show(" X, Y, Z - ni qiymatini kiriting "," Ошбика");}
               else {
                if ( textBox4->Text == "" && textBox5 ->Text == ""  ){
                MessageBox::Show(" X, Y - ni qiymatini kiriting "," Ошбика");}
                else {
               if (  textBox5 ->Text == "" && textBox6->Text == "" ){
               MessageBox::Show("  Y,  Z  -  ni  qiymatini  kiriting  "," Ошбика");}
               else {
               if ( textBox4->Text == ""   ){
                MessageBox::Show(" X - ni qiymatini kiriting "," Ошбика");}
                else {
                if (  textBox5 ->Text == ""  ){
                MessageBox::Show(" Y - ni qiymatini kiriting "," Ошбика");}
               else if (    textBox6->Text == "" ){
                MessageBox::Show(" Z - ni qiymatini kiriting "," Ошбика");}}}}}
      private:      System::Void      label_s_Click(System::Object^          sender, System::EventArgs^  e) {
               this->panel->Controls->Clear();
                this->panel->Controls->Add(this->panel2);]
      private:      System::Void      button4_Click(System::Object^          sender, System::EventArgs^  e) {
                if (textBox7->Text != ""){
                r = Convert::ToInt32(textBox7->Text);
```

```
            if (r>0){
            s=3.14*r*r;
            label_s ->Text = Convert::ToString(s);
            panel -> Controls-> Clear();
            panel -> Controls-> Add(this->panel1);}
            else{
            MessageBox::Show(" r - ni qiymati 0 dan katta bo'lishi kerak "," Ошбика");}}
             if (textBox7->Text == ""){
             MessageBox::Show(" r - ni qiymatini kiriting "," Ошбика");}
```