O’ZBEKISTON RESPUBLIKASI
OLIY VA O’RTA MAXSUS TA`LIM VAZIRLIGI

NAMANGAN MUHANDISLIK-PEDAGOGIKA
INSTITUTI

KASB TA’LIMI «INFORMATIKA VA AT»
kafedrasi

F. Irisqulov, E. Qosimov

«Ob’ektga yo’naltirilgan dasturlash tillari»
fanidan tajriba mashg’ulotlarini o’tkazish bo’yicha

Uslubiy ko’rsatma

(1-qism)

Namangan -2016

Ushbu uslubiy ko’rsatma «Informatika va axborotlar texnologiyasi» (Kasb ta’limi) bakalavr ta’lim yo’nalishi bo’yicha kunduzgi bo’limda ta’lim olayotgan talabalar uchun mo’ljallangan bo’lib, «Ob’ektga yo’naltirilgan dasturlash tillari» fanidan tajriba mashg’ulotlarini o’tkazish bo’yicha zarur yo’riqnomalarni va tajriba ishi variantlarini o’z ichiga olgan. Mazkur uslubiy ko’rsatmadan «Ob’ektga yo’naltirilgan dasturlash tillari» fanini o’rganayotgan barcha talabalar, C++ tilini mustaqil o’rganuvchi talabalar, magistrlar va o’qituvchilar foydalanishlari mumkin.

Tuzuvchilar: F. Irisqulov - NamMPI KT «Informatika va AT» kafedrasi assistenti
E. Qosimov - NamMPI KT «Informatika va AT» kafedrasi assistenti

Taqrizchi: P. Karimov - NamMPI KT «Informatika va AT» kafedrasi dotsenti

Uslubiy ko’rsatma NamMPI KT «Informatika va axborot tehnologiyalari» kafedrasining umumiy majlisida ko’rib chiqilgan va ma’qullangan.
(Bayonnoma №_____ «___» _____ 20__ yil)

Uslubiy ko’rsatma NamMPI ilmiy-uslubiy kengashida muxokama qilingan va chop etishga tavsiya berilgan.
(Bayonnoma №_____ «___» _____ 20__ yil

So’z boshi

«Jahon sivilizatsiyasiga dahldor bo’lgan eng zamonaviy ilmlarni egallamay turib, mamlakat taraqqiyotini ta’minlash qiyin», - degan edilar prezidentimiz I. Karimov. O’zbekistonning iqtisodiy va ijtimoiy sohalarida yuqori natijalarga erishishi jahon iqtisodiy tizimida to’laqonli natijalarga to’laqonli sheriklik o’rnini egallay borishi, inson faoliyatining barcha jabxalarida zamonaviy axborot texnologiyalaridan yuqori darajada foydalanish ko’lamlarining qanday bo’lishga hamda bu texnologiyalar ijtimoiy mehnat samaradorligining oshishida qanday rol o’ynashiga bog’liq. Xulosa qilib shuni aytish mumkinki, zamonaviy komp yuterlardan amalda, ya’ni o’zining kasbiy faoliyatida keng foydalana oladigan yetuk kadrlarni tayyorlash dolzarb masalalardan hisoblanadi. Xozirda turli ta’lim yo’nalishlarida ta’lim olayotgan talabalar «Informatika» fanini o’rganish natijasida hisoblash texnikasi vositalaridan foydalanishni mukammal o’zlashtiradilar. Shu bilan bir qatorda Beysik tilida dasturlash elementlari xaqida ham ma’lum bir tushunchalarga ega bo’ladilar. Lekin, dasturlash tillarini, ulardan foydalanib turli xil jarayonlarning dasturlarini yaratishni o’rganish uchun ajratilgan soat miqdorlarining kamligi sababli talabalar dasturlash asoslarini, uni yaratish texnologiyalarini talab darajasida o’zlashtira olmaydilar. Ana shu kabi kamchiliklarni yo’qotish, talabalarni dasturlash qoidalarini yaxshi o’zlashtirishlari uchun mazkur fan «Informatika va AT» ta’lim yo’nalishlarining ishchi o’quv rejasiga talabalar tanlovi asosida o’qitiladigan fanlar ro’yxatiga kiritilgan.

Har qanday fanni o’rganish fan bo’yicha nazariy ma’lumotlarni o’zlashtirishdan boshlanadi. So’ngra o’rganilgan nazariy ma’lumotlarni amalda ko’llay bilish malakasini xosil qilish uchun amaliy va tajriba mashg’ulotlari o’tkaziladi. Dasturlash asoslarini chuqur o’zlashtirishda ayniqsa tajriba mashg’ulotlarining salmog’i juda kattadir. Bu mashg’ulotlarda talaba mustaqil ravishda berilgan topshiriqlar bo’yicha qo’yilgan masalani yechishning mantiqiy ketma-ketligini ishlab chiqadi va yaratilgan dasturiy ta’minot bo’yicha natijalar olishga hamda ularni taxlil qilishga erishadi. Shuning uchun tajriba mashg’ulotlarini sifatli, yuqori saviyada o’tkazilishi talabalarning fan bo’yicha o’zlashtirish darajasiga ijobiy ta’sir o’tkazadi.

Mazkur uslubiy ko’rsatma «Ob’ektga yo’naltirilgan dasturlash tillari» fanidan tajriba mashg’ulotlarini o’tkazish bo’yicha zarur bo’lgan barcha yo’riqnomalarni o’z ichiga olgan va Davlat ta’lim standartlariga mos namunaviy dastur hamda unga mos ishchi dasturlar asosida tuzilgan.

Tajriba mashg’ulotlarini bajarish uchun talabalarga quyidagi talablar qo’yiladi:

1. Tajriba ishini bajarish uchun zarur bo’lgan nazariy ma’lumotlarni ko’rsatilgan adabiyotlardan to’plash;
2. Belgilangan variant bo’yicha shaxsiy topshiriqlarni o’z vaqtida olish;
3. Ish rejasida belgilangan masalalarning qo’yilishi, uning moxiyati va amaliy masalalarni yechishdagi axamiyati xaqida to’liq ma’lumotga ega bo’lish;
4. Masalani hal qilishning ishchi algoritmlarini ishlab chiqish va ularni blok - sxemalar ko’rinishida ifodalash;
5. Ishlab chiqarilgan algoritm bo’yicha C++ tilida dastur yaratish va uni ishga sozlash;

6. Tuzilgan dasturni tekshirib ko’rish va olinayotgan natijalarning to’g’riligiga ishonch bildirish;
7. Shaxsiy topshiriqning dasturidan olingan natijalarni zarur formada chop etishni tashkillash;
8. Bajarilgan ishlarni belgilangan reja asosida rasmiylashtirish va uni o’qituvchi huzurida ximoya qilish (tajriba darsi mobaynida);
9. Ximoya qilingan, o’qituvchi tomonidan ijobiy baholangan tajriba ishi hisobotini o’qituvchiga topshirish;

Eslatma:
1.Tajriba ishini o’z vaqtida bajarmagan talaba o’qituvchi belgilagan vaqtda (dars mashg’ulotlaridan bo’sh vaqtda) kafedraga kelib ishni bajaradi va belgilangan tartibda uni ximoya qiladi;
2. Agar talaba dars mobaynida va qo’shimcha belgilangan vaqtda ham tajriba ishini bajarmasa, shu mavzu uchun reyting ballarini olmagan va mavzuni o’zlashtirmagan hisoblanadi.

KIRISH

Ma’lumki, EHMlar berilgan algoritmlarni formal bajaruvchi avtomat hisoblanadi. Shuning uchun biror masalani EHMda yechishda avval shu masalaga mos algoritmni aniqlash zarur. Algoritmni EHMga uzatishda esa uni maxsus «mashina tili»ga o’girib, mashina kodida yozilgan programmaga aylantiriladi. Shu bilan bir qatorda, EHMning turli xil tiplari turlicha tillarga ega bo’ladi ya’ni, biror EHM uchun yozilgan programma boshqa EHM uchun tushunarsiz bo’lishi mumkin. Chunki, har bir EHM faqat o’zining «mashina tili»da yozilgan programmalarnigina tushunishi va bajarishi mumkin.

Mashina kodida yozilgan programmalarning ko’rinish sifati juda kambag’aldir, chunki bu programmalar faqat 0 va 1 larning maxsus ketma-ketligidan tashkil topadi. Bu esa mutaxassis bo’lmagan odamga tushunarsiz bo’lib, programma tuzishga noqulaydir.

Aytib o’tilganlardan shuni xulosa qilish mumkinki, mashina tilidan foydalanish odam uchun uni qiziqtirgan ya’ni, yechishi lozim bo’lgan masalaning algoritmini ishlab chiqishda va yozishda juda katta qiyinchiliklar va muammolar tug’diradi.

Bu qiyinchiliklarni bartaraf qilish, dasturchining ishini osonlashtirish va yaratilgan programmalarning ishonchlilik darajasini oshirish maqsadida yuqori darajadagi programmalash tillari yoki algoritmik tillar yaratilgan.

Algoritmik tillarning mashina tillaridan asosiy farqlari sifatida quyidagilarni ko’rsatish mumkin:

- mashina tili alfavitidan algoritmik til alfavitining o’ta kengligi tuzilgan programma matnining ko’rinish sifatini keskin oshiradi;
- ishlatilishi mumkin bo’lgan amallar majmui mashina amallari majmuiga bog’liq emas;
- bajariladigan amallar odam uchun qulay ko’rinishda, ya’ni amalda qabul qilingan matematik belgilashlar yordamida beriladi;
- amallar operandlari uchun dasturchi tomonidan beriladigan shaxsiy ismlar qo’yish mumkinligi;
- mashina uchun ko’zda tutilgan ma’lumot tiplaridan tashqari yangi tiplar kiritish imkoniyati yaratilganligi;
- algoritmik tildagi programma matnini o’qish va tushunishga qulayligi;

Shunday qilib, qaysidir ma’noda aytish mumkinki, algoritmik tillar mashina tiliga bog’liq emas.

Yuqorida aytilganlardan kelib chiqqan holda ma’lum bo’ldiki, algoritmik tilda yozilgan masala yechimining algoritmi to’g’ridan-to’g’ri EHMda bajarilishi mumkin emas ekan. Buning uchun esa algoritm, ishlatilayotgan EHMning mashina tiliga translyator (kompilyator yoki interpretator) yordamida o’girilishi lozim. Translyator - mashina tilida yozilgan maxsus programma bo’lib, uning asosiy maqsadi algoritmik tillarda yozilgan programma matnini EHM tiliga tarjima qilishdan iboratdir.

Amalda programmalash uchun foydalanilayotgan algoritmik tillar o’z ma’nosiga ko’ra algoritmni so’zli-formulali yozish uslubiga o’xshab ketadi, ya’ni ma’lum bir qism ko’rsatmalar oddiy matematik formulalar, boshqa qismlar esa so’zlar yordamida ifodalanishi mumkin. Misol sifatida n va m natural sonlarning eng katta umumiy bo’luvchisi(EKUB)ni topish algoritmini ko’rib chiqaylik:

1. A=n, B=m deylik
2. Agar A=V bo’lsa 5-punktga, aks holda 3-punktga o’t.
3. Agar A>B bo’lsa A ning yangi qiymati deb A-V ni qabul qil, V ni qiymatini o’zgartirma; aks holda V ning yangi qiymati deb V-A ni qabul qil, A ning qiymatini o’zgartirma.
4. 2-punktga o’t.
5. EKUB=A va hisobni to’xtat.

Ushbu algoritmni qisqaroq ko’rinishda quyidagicha ifodalashimiz ham mumkin:

1. A=n, B=m deylik;
2. Agar A>B bo’lsa A=A-V aks holda V=V-A, A=V bo’lguncha 2-punktni takrorla.
3. EKUB=A va hisobni to’xtat.

Ushbu misoldan ko’rinib turibdiki algoritmlarning bunday yozish uslubi odam uchun ham qulay, ham tushunarli hisoblanadi. Lekin bu uslubda ham ma’lum kamchiliklar ko’zga tashlanadi:

- algoritmni ortiqcha ko’p so’zli va uzun deyish mumkin;
- bir xil ma’nodagi ko’rsatmani turli xil uslublarda berish mumkinligi;
- bunday ixtiyoriy ko’rinishda ifodalangan algoritmni EHM tiliga o’tkazish imkoniyatining kamligi.

Yuqoridagi kabi kamchiliklarni bartaraf qilish uchun formallashgan, qat’iy aniqlangan algoritmik tillar ishlab chiqilgan. Algoritmik tillar uchta o’zakdan tashkil topadi: til alfaviti, sintaksisi va semantikasi.

Til alfaviti shu tilgagina tegishli bo’lgan chekli sondagi belgilardan tashkil topadi. Programma matnini yozishda faqat shu belgilardangina foydalanish mumkin, boshqa belgilarni esa til tanimaydi.

Til sintaksisi alfavit harflaridan tashkil topib, mumkin bo’lgan konstruktsiyalarni aniqlovchi qoidalar tizimidir. Mazkur tilda ifoda etilgan to’la algoritm va uning alohida hadlari shu konstruktsiyalar orqali ifoda qilinadi. Shunday qilib, belgilarning har qanday ketma-ketligini, mazkur tilning matni to’g’riligi yoki noto’g’riligini til sintaksisi orqali bilib olamiz.

Til semantikasi uning ayrim konstruktsiyalari uchun qoidalar sistemasini tushuntirishga hizmat qiladi.

Endi algoritmik tillarning qaysilari amalda ko’proq ishlatilishi haqida fikr yuritsak. Ma’lumki 70-yillarda bir guruh muammoga yo’naltirilgan algoritmik tillar yaratilgan bo’lib, bu programmalash tillaridan foydalanib juda ko’p sohadagi muammoli vazifalar hal qilingan. Hisoblash jarayonlarining algoritmlarini ifodalash uchun Algol-60 va Fortran tillari, iqtisodiy masalalar algoritmlari uchun Kobol va Algek tillari, matnli axborotlarni taxrirlash uchun esa Snobol tillari ishlatilgan. Sanab o’tilgan bu algoritmik tillar asosan katta xajmli, ko’pchilikning foydalanishiga mo’ljallangan EHMlari uchun tuzilgan edi.

Hozirda insoniyat faoliyatining barcha jabhalariga shaxsiy elektron hisoblash mashinalari (ShEHM) shaxdam qadamlar bilan kirib bormoqda. Asosan ShEHMlarga mo’ljallangan, hamda murakkab jarayonlarning hisob ishlarini bajarish va juda katta ma’lumotlar tizimi bilan ishlashni tashkil etuvchi yangi algoritmik tillar sinfi borgan sari kengayib bormoqda. Bu tillar jumlasiga quyidagi ko’proq ishlatilayotgan tillarni

kiritish mumkin:

- Beysik tili;
- Paskal tili;
- Si tili va hakozo.

Programma tuzishni o’rganishni boshlovchilarga mo’ljallangan, savol-javob sistemasida ishlaydigan, turli-tuman jarayonlar algoritmini yozishga qulay bo’lgan tillardan biri BEYSIK(BASIC) tilidir. Beysik tilining nomi ingliz so’zi Beginner's All-purpose Symbolic Instruction Code ning o’qilishiga mos kelib, boshlovchilar uchun belgili ko’rsatmalar kodi(tili) degan ma’noni anglatadi. Beysik tilini yaratish ustidagi ishlar 1963 yilning yozidan boshlangan. Tilning ijodkorlari taniqli olimlar T.Kurts va J.Kemeni hisoblanadi. Hozirga kelib Beysik tilining turli xil yangi ko’rinishlari ishlab chiqilmoqda va ulardan foydalanib millionlab dasturchilar ajoyib programmalar yaratishmoqda.

Endi nisbatan mukammalroq bo’lgan Paskal va Si algoritmik tillari haqida qisqacha fikr yuritsak.

Paskal tili 1969 yili N.Virt tomonidan yaratilib mashhur olim Blez Paskal nomi bilan ataldi. Bu til N.Virtning o’ylashi bo’yicha programmalashning zamonaviy texnologiyasiga va uslubiga, strukturali programmalash nazariyasiga asoslangan va boshqa programmalash tillaridan muayyan yutuqqa ega til bo’lishi lozim edi. Mazkur til:

1. Programmalashtirish kontseptsiyasini va strukturasini sistemali va aniq ifodalaydi;
2. Programma tuzishni sistemali olib borish imkonini beradi;
3. Programma tuzish uchun boy termin va struktura sxemalariga ega;
4. Yo’l qo’yilgan xatoliklarni tahlil qilishning yuqori darajadagi sistemasiga ega.

1981 yili Paskal tilining halqaro standarti taklif etildi va IBM PC tipidagi shaxsiy komp yuterlarda Paskal tilining Borland firmasi tomonidan ishlab chiqilgan Turbo-Paskal oiladosh tili keng qo’llanila boshlandi.

C++ dasturlash tili kompilatsiyalanadigan, stataik tiplashtirilgan umumiy qo’llanishga mo’ljallangan dasturlash tili hisoblanadi. C dastlab UNIX operatsion tizimi va C dasturlash tili yaratilgan va C tili asosida **C++** tili yaratildi. C esa o‘z navbatida B va BCPL tillaridan kelib chiqgan.

C++ tilini 80 yillarda AT&T Bell Labs korxonasi ishchisi Byarnom Straustrup tomonidan yaratilgan. **C++** dasturlash tilining asosi C hisoblanadi va shu tilni (C) misolida C++ tili yaratilgan. C dasturlash tilini mukammalashtirgan eng asosiy narsa bu – ob’yektga mo‘ljallangan dasturlashni olib kirgani hisoblanadi.

ANSI-ISO (ANSI X3J16; ISO WG21/N0836) birlashmasi 1989 yilda, birlashga holda ish boshladi. Bu korxonaning dastlabki ishi **C++** dasturlash tiliga va uning kutubxonasiga standart ishlab chiqishdan boshlandi. Buning uchun 1990 yildagi **C++** tili asos qilib olindi.

1990 yilda **C++** standarti ishlab chiqildi va bu standart hozir ANSI C nomi bilan ataldi. **C++** funksiya va ob’yektlarning boy kutubxonasiga ega.

C++ dasturlash tili kompilyatsiya qilinadigan til hisoblanadi bu degani yozilgan kod oldin mashina tiliga o‘giriladi va keyin ishga tushiriladi.

C++ dan tashqari boshqa ko'p ob'ektli dasturlshga yo'naltirilgan tillar paydo

bo'ldi. Shulardan eng ko'zga tashlanadigani Xerox ning Palo Altoda joylashgan ilmiy-qidiruv markazida (PARC) tuzilgan Smalltalk dasturlash tilidir. Smalltalk da hamma narsa ob'ektlarga asoslangan. C++ esa gibrid tildir. Unda C ga o'hshab strukturali dasturlash yoki yangicha, ob'ektlar bilan dasturlash mumkin. Yangicha deyishimiz ham nisbiydir. Ob'ektli dasturlash falsafasi paydo bo'lganiga ham yigirma yildan oshayapti.

Ma’lumki, biror-bir dasturlash tilini mukammal o’rgangan foydalanuvchi uchun boshqa dasturlash tillarini o’rganish juda oson kechadi. Shuning uchun, talaba yuqorida sanab o’tilgan tillarning birortasini yaxshi o’zlashtirib olishi lozim. Keyinchalik esa yechilayotgan masalaning moxiyatidan, zamon va mezon talabidan kelib chiqib yuqori darajali dasturlash tillarini ketma – ket o’rganishni davom ettirish mumkin.

Mazkur uslubiy ko’rsatma dasturlash strukturasi va texnologiyasini o’rganishga asosdangan **C++** tilini mukammal o’rganishga mo’ljallangan bo’lib, turli xil jarayonlarni hisoblash algoritmlarini ishlab chiqish va dasturlarini yaratish bo’yicha tajriba ishlarini o’z ichiga olgan.

1-Tajriba ishi

Mavzu: Dev C++ muhitida ishlash.

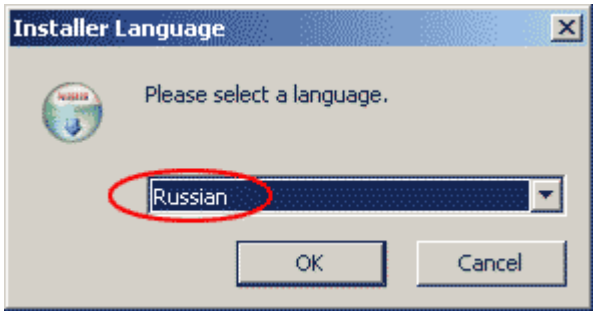
Ishdan maqsad: Dev C++muhitini o'rnatish, Dev C++ muhitida sozlash va ishlash, C++ tilining asosiy tushunchalariga ega bo'lish, C++ tilida matematik amallarni ifodalay olish, o'zgaruvchi tiplari haqida to'liq ma'lumotga ega bo'lish ko'nikmalarini xosil qilish.

Ish rejasi:

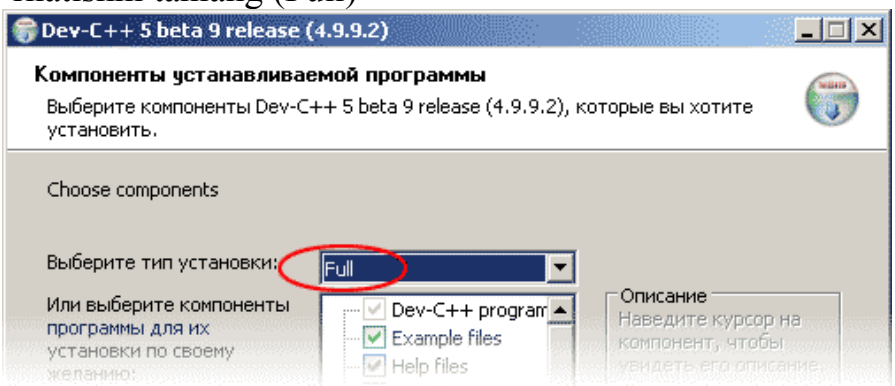
1. Dev C++ dasturini o'rnatish
2. Dev C++ muhitini o'rnatish, muhitini sozlash va ishlash ko'nikmasini hosil qilish.
3. C++ algoritmik tilining asosiy tushunchalari
4. Matematik amallarni C++ dasturlash tilida ko'rinishi
5. C++ da tiplar
6. C++ da kutubhonalar

1. Dev C++ dasturini o'rnatish.

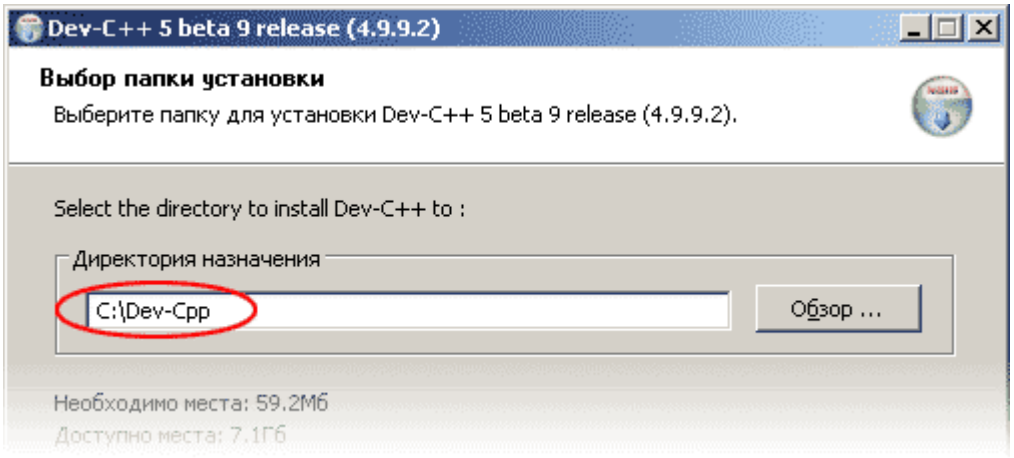
1. devcpp-4.9.9.2_setup.exe instalyator dasturini ishga tushiring.
2. Rus tilida o'rnatish uchun (Russian) bo'limini tanlang



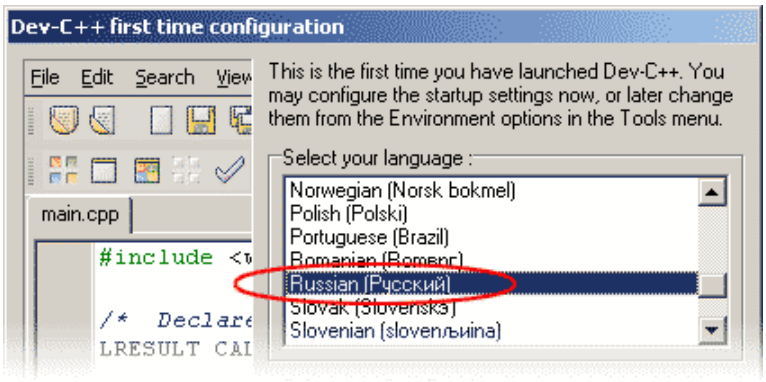
3. Lisenziya talablarini qabul qiling
4. To'liq o'rnatishni tanlang (Full)



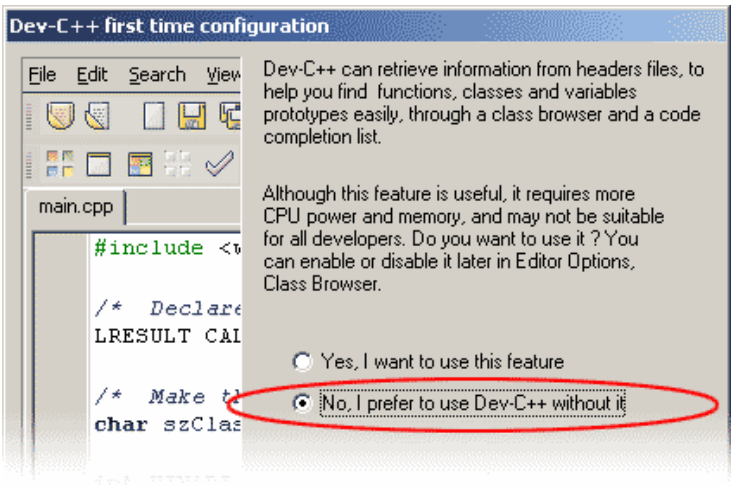
5. Dasturni tavsiya etgan yo’nalishga o’rnating (C:\Dev-Cpp)



6. Dastur o’rnatilgandan so’ng dastur avtomatik yuklanadi.
7. Menyu uchun rus tilini tanlang

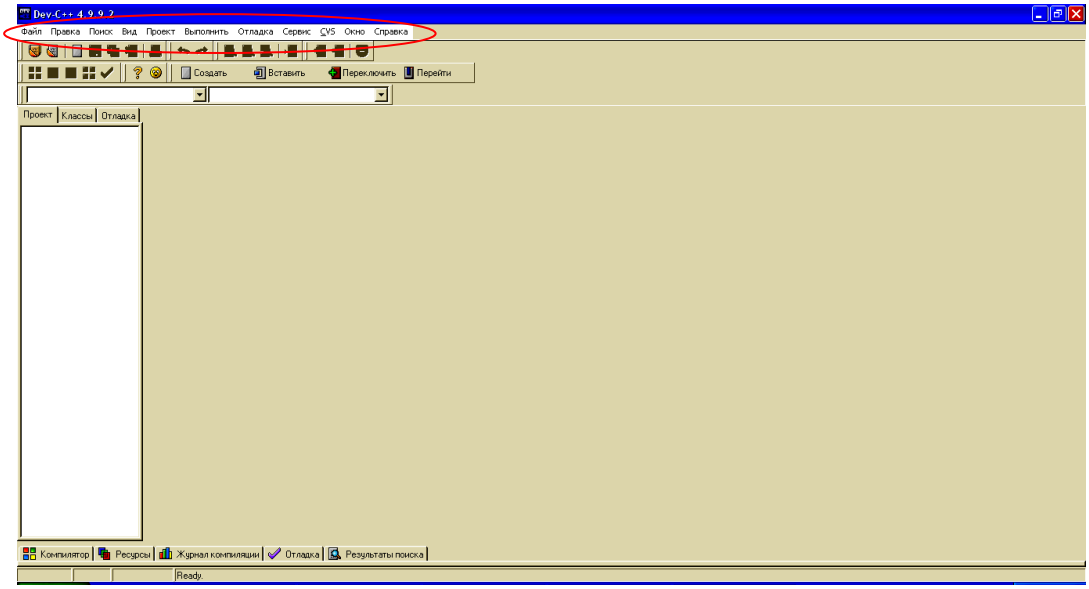


8. Keyingi oynada rus tili parametrlarini qo’llash kerak bo’lsa “Yes” yoki “No” tugachasi bosiladi.



2. Dev C++ dasturini muhitini o’rganish

Dasturni ishga tushirganimizda bizda quyidagi oyna hosil bo’ladi



Файл Правка Поиск Вид Проект Выполнить Отладка Сервис CVS Окно Справка

Menyular ko’rinishi

Menyular vazifalari:

“Файл” menyusi – Loyiha yoki C faylini tashkil etish, ularni saqlash, qayta nomlash, ochish, fayllarni eksport va import ishlari va dasturdan chiqish amallari mavjud.

“Правка” menyusi – Loyiha yoki C faylini tashkil etilgan fayllarni to’g’rilash, matnlardan nusxa olish, olingan nusxani qo’yish va matn ustida bajarish mumkin bo’lgan taxrirlash ishlarini amalga oshirish

“Поиск” menyusi – Loyiha yoki C faylini tashkil etilgan fayllar ichidan kerakli bo’lgan matnlarni qidirish, matn o’rniga matn qo’yish ishlarini amalga oshirish ishlari mavjud

“Вид” menyusi – Dev C++ muhiti ko’rinishlarini sozlash bo’limi

“Проект” menyusi – bo’sh C++ faylini hosil qilish, Lohihalarga faylllar qo’shish amallari

“Выполнить” menyusi – Kompilatsiya qilish, joriy faylni kompilatsiya qilish, bajarish, kompilatsiyalash va bajarish, kompilatsiya parametrlari va boshqa vazifalarni

bajaradi

“Отладка” menyusi – dastur ishlash jarayonidagi xatoliklarni aniqlash, bajarish ishini tugatish, otladka parametrlari, qadamva qadam dasturni bajartirish va boshqa amallar kiradi.

“Сервис” menyusi –Kompilyator parametrlari, Muhit parametrlari, muharrir parametrlari, klavishlar qo’shilmasi sozlamasi, uskunalar sozlamalari va boshqa sozlamalar bo’limi

“Окно” – menyusi – Oynalar ustida amallar

3. C++ algoritmik tilining asosiy tushunchalari

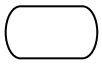
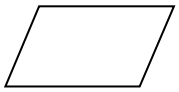
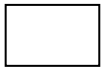
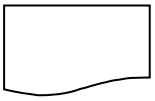
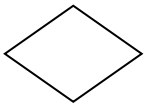
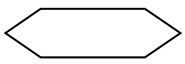
Algoritm deganda, biror maqsadga erishishga qaratilgan ijrochi bajarishi uchun mo'ljallangan ko'rsatma (buyruq) larning aniq, tushunarli va chekli ketma-ketligi tushuniladi.

Algoritm quyidagi asosiy xossalarga ega: uzluklilik, aniqlik, natijaviylik va ommaviylik.

Algoritmni ishlab chiqishda uni bir necha xil usul bilan ifodalab bersa bo’ladi. Shulardan uchtasi keng tarqalgan. Bular:

1. Algoritmni oddiy tilda ifodalash;
2. Algoritmni tuzim ko’rinishida ifodalash;
3. Algoritmni maxsus (algoritmik) tilda yozish.

Quyida algoritmni tuzim ko’rinishida ifodalash uchun asosiy foydalaniladigan shakllar keltirilgan:

Shakl	Qaysi xolda ishlatiladi	Shakl	Qaysi xolda ishlatiladi
	Boshlanish va oxirida		Axborotni kiritish va chiqarish
	Xisoblashlar uchun		Natijani chop etish uchun
	Tarmoqlanish shartini tekshirishda		sikl boshlanishida

Alfavit. C++ alfavitiga quyidagi simvollar kiradi.

- Katta va kichik lotin alfaviti xarflari (A,B,..,Z,a,b,…,z)

- Raqamlar: 0,1,2,3,4,5,6,7,8,9
- Maxsus simvollar: “ , {} | [] () + - / % \ ; ‘ . : ? < = > _ ! & * # ~ ^

4. Matematik amallarni C++ dasturlash tilida ko’rinishi

Matemetik funksiyalardan programmada foydalanish uchun math.h faylini progarmmaga qo'shish kerak.

#include <math.h>

Matematik ifodalanishi	**C++ tilida ifodalanishi**	**Ma’nosi**
cos x	cos(x)	Kosinus x radianda
sin x	sin(x)	Sinus x radianda
tg x	tan(x)	tangenes x radianda
chx	cosh(x)	Giperbolik kosinus x radianda
shx	sinh(x)	Giperbolik sinus x radianda
thx	tanh(x)	Giperbolik tangenes x radianda
arccosx	acos(x)	Arkkosinus x
arcsinx	asin(x)	Arksinus x
arctg x	atan(x)	Arktangenes x
e^x	exp(x)	Exponenta x
x^y	pow(x,y)	x ning darajasi y
|x|	1. abs(x) - butun sonlar uchun 2. fabs(x) - haqiqiy sonlar uchun 3. labs(x) - uzun butun son uchun	x ning moduli
√x	sqrt(x)	Ildiz ostida x
ln(x)	log(x)	x ning natural logorifmi
$\log_{10}x$	Log10(x)	Asosi 10 asosga ko’ra

		lagorifmi
ceil(x)	haqiqiy toifadagi x o'zgaruvchisi qiymatini unga eng yaqin katta butun songa aylantiradi.	
floor(x)	haqiqiy toifadagi x o'zgaruvchisi qiymatini unga eng yaqin kichik butun songa aylantiradi	

5. C++ da tiplar

C++ tilining asosiy tushunchalaridan biri nomlangan hotira qismi – ob’ekt tushunchasidir. Ob’ektning xususiy holi bu o’zgaruvchidir. O’zgaruvchiga qiymat berilganda unga ajratilgan hotira qismiga shu qiymat kodi yoziladi. O’zgaruvchi qiymatiga nomi orqali murojaat qilish mumkin, hotira qismiga esa faqat adresi orqali murojaat qilinadi. O’zgaruvchi nomi bu erkin kiritiladigan identifikatordir. O’zgaruvchi nomi sifatida xizmatchi so’zlarni ishlatish mumkin emas.

O'zgaruvchi. Xotiraning nomlangan qismi bolib, o'zida ma'lum bir toifadagi qiymatlarni saqlaydi. O'zgaruvchining nomi va qiymati bo'ladi. O'zgaruvchining nomi orqali qiymat saqlanayotgan xotira qismiga murojaat qilinadi. Programma ishlashi jarayonida o'zgaruvchining qiymatini o'zgartirish mumkin. Har qanday o'zgaruvchini ishlatishdan oldin, uni e'lon qilish lozim.

O’zgaruvchilar quyidagi tiplarga bo’linadi:

Butun sonlar

Toifa ko'rinishi	Qabul qiladigan qiymatlar oralig'i	Kompyuter xotirasida
unsigned short int	0..65535	2 bayt
short int	-32768..32767	2 bayt
unsigned long int	0..42949667295	4 bayt
long int	-2147483648..2147483647	4 bayt

int (16 razryadli)	-32768..32767	2 bayt
int (32 razryadli)	-2147483648..2147483647	4 bayt
unsigned int (16 razryadli)	0..65535	2 bayt
unsigned int (32 razryadli)	0..42949667295	4 bayt

Haqiqiy sonlar

Toifa ko'rinishi	Qabul qiladigan qiymatlar oralig'i	Kompyuter xotirasida egallagan hajmi
float	1.2E-38..3.4E38	4 bayt
double	2.2E-308..1.8E308	8 bayt
long double (32 razryadli)	3.4e-4932..-3.4e4932	10 bayt

Boshqa toifalar

Toifa ko'rinishi	Qabul qiladigan qiymatlar oralig'i	Kompyuter xotirasida
bool	true yoki false	1 bayt
char	0..255	1 bayt
void	2 yoki 4	

O'zgaruvchilarni e'lon qilish. Programmada ishlatilgan barcha o'zgaruvchilarni qaysi toifaga tegishli ekanligini e'lon qilish kerak. Ma'lulotlarni e'lon qilishning umumiy ko'rinishi quyidagicha:

toifa_nomi o'zgaruvchi;

Agar bir nechta o'zgaruvchi bir toifaga mansub bo'lsa, ularni vergul bilan ajratib berish mumkin.

Butun sonlarni ifodalash uchun **int** va haqiqiy sonlarni ifodalash uchun **float** xizmatchi kabi so'zlaridan foydalaniladi.

int x,y; // butun toifadagi o'zgaruvchilarni e'lon qilish

float a,b,c; // haqiqiy toifadagi o'zgaruvchilar e'lon qilish.

Kiritish chiqarish operatorlari.

Programmada klaviatura orqali ma'lumot kiritish va ekranga chiqarish uchun preprotsessor direktivasini, ya'ni **#include <iostream>** ni programmaga qo'shish shart.

Ma'lumotlarni kiritish **std::cin** >>, ma'lumotlarni chiqarish **std::cout** << operatori orqali amalga oshiriladi.

Misol:

```
std::cin >> a;
std::cout<<a;
```

Dasturda standart nomlar fazosidan foydalanib e'lon qilinganda (**using namespace std**) shunchaki **cin** va **cout** deb yozishimiz mumkin.

Misol:

```
#include <iostream>
// standart nomlar fazosidan foydalanishni e'lon qilish
using namespace std;
int main()
{
    int a, b, c;
    cout << "a="; cin >> a;
    cout << "b="; cin >> b;
    c = a + b;
    cout << c << endl;
  return 0;
}
```

cin va **cout** operatorlari parametrsiz kiritish va chop etish operatorlari hisoblanib, kiritish va chiqarish jarayonini oqim tarzida hosil qilishimiz mumkin.

Misol:

```
cout<< “X ning qiymati \t”<<x<<“y ning qiymati \t”<<y<<endl;
```

endl-yagi satrga o’tish buyrug’i.

C++ sistemasi asosan quyidagi qismlardan iborat. Bular:

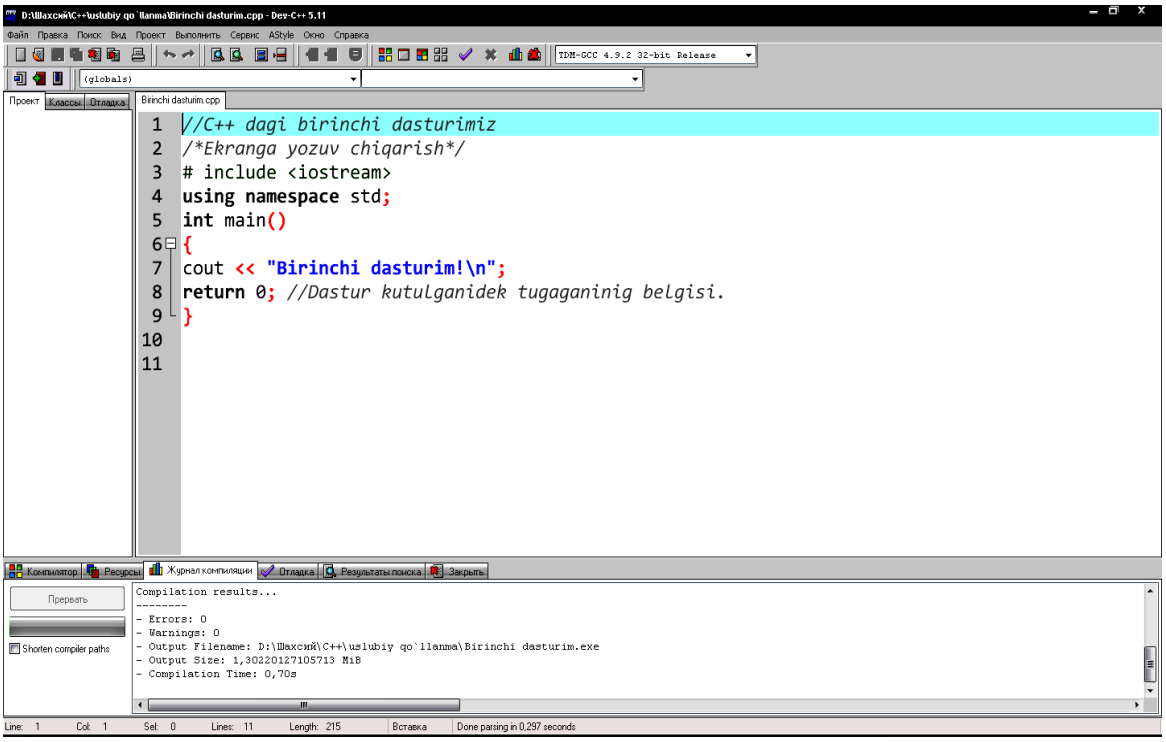
Dasturni yozish redaktori, C++ tili va standart kutubhonalardir. C++ dasturi ma'lum bir fazalardan o'tadi. Birinchisi dasturni yozish va tahrirlash, ikkinchisi preprosessor amallarini bajarish, kompilyatsiya, kutubhonalardagi ob'ekt va funksiyalarni dastur

bilan bog'lash (link), hotiraga yuklash (load) va bajarish (execute).

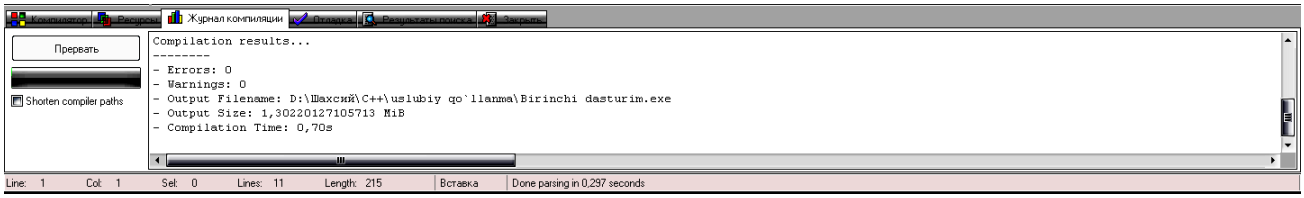
C++ da oddiy misolni ko'rib chiqamiz.

```
//C++ dagi oddiy dastur
/*Ekranga yozuv chiqarish*/
# include <iostream>
using namespace std;
int main()
{
cout << "Birinchi dasturim!\n";
return 0; //Dastur kutulganidek tugaganinig belgisi.
}
```

Dasturni ishga tushirishdan oldin uni kompilyatsiya qilib olamiz. Buning uchun standart funksional tugmalardan **F9** tugmasini bosamiz.



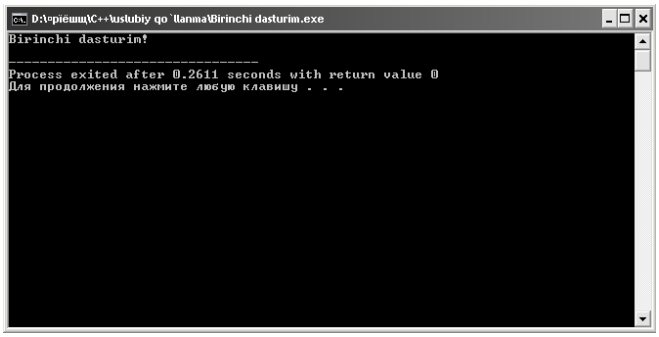
Bu erda quyidagi **Kompilyator(*To'plam*)** darchasiga duch kelamiz. Agar qastur kodi to'gri yozilsa kompilyator darchasida quyidagi yozuvlar aks etadi.



Compilation results… //(To’plamda natija)
- ----------
- Errors: 0 //(Xatolar)
- Warnings: 0 //(Ogohlantirishlar)
- Output Filename: D:\Birinchi dasturim.exe //(Chiqish fayl nomi)
- Output Size: 1,3002201247212 Mib //(Chiqish hajmi)
- Compilation Time: 0,70 s //(To’plam vaqti)

To’plamda quyidayi oyna namoyon bo’lsa demak siz to’gri kodlashtiribsiz. Endi natijani ekranga chiqarish uchun **F10** tugmasini bosamiz.

Ekranda: Quyidagi natija namoyon bo’ladi.



 # include <iostream.h> bu preprosessorga beriladigan buyruqdir. Preprosessor kompilyatsiyadan oldin fayllarni ko'rib chiqadi va kerakli amallarni bajaradi. Unga tegishli bo'lgan buyruqlar **(#)** belgisi bilan boshlanadi lekin buyruq ohiriga nuqta-vergul **(;)** qoyilmaydi. Bu yerda **include** (***kiritmoq, qamrab olmoq***) buyrug'i **iostream.h** faylini asosiy dasturimiz ichiga kiritadi. Bu fayl ichida biz ishlatayotgan **cout** oqim (stream) ob'ektining e'loni berilgan. C++ stilida ekran yoki klaviaturadan ***kirish/chiqishni*** bajarmoqchi bo'lgan barcha dasturlar ushbu boshliq (header) faylni yoki uning yangi ko'rinishini include bilan o'z ichiga olishi kerak. Bu kabi fayllarni biz bundan keyin e'lon fayllari deb ataymiz.

Birinchi usulda e'lon fayli <> qavslari ichida yoziladi. Bunda C++ sistemasi ushbu faylni oldindan belgilangan kataloglar ichidan qidiradi. Bu usul bilan asosan standart kutubhona fayllari qo'llaniladi.

Ikkinchi usulda, fayl nomi qo'shtirnoqlarga olinganda, kiritilishi kerak bo'lgan fayl joriy katalogdan qidiriladi.

Biz bu o'zgarishlarga keyinroq qaytamiz, hozircha esa eski tipdagi e'lon fayllaridan foydalanib turamiz.

int main() har bir C++ dasturining qismidir. main dan keyingi () qavslar C++ ning funksiya deb ataluvchi blokining boshlangangini bildiradi. C++ dasturi bir yoki bir necha funksiyalardan iborat. Va shulardan aniq bitta funksiya main deb atalishi shart. Bunda main dastur ichida keladigan birinchi funksiya bo'lmasligi ham mumkin. Operatsion sistema dastur ijrosini main() funksiyasidan boshlaydi. **main()** dan oldin kelgan **int** esa **main** funksiyasidan qaytish qiymati tipini belgilaydi. Bunda **int integer**, yani butun son deganidir. main() ning qaytargan qiymati operatsion sistemaga boradi.

{ *qavs funksiya va boshqa bloklar tanasini boshlaydi. Blokni yopish uchun* }qavsi ishlatilinadi.

cout << "**Birinchi dasturim**!\n"; satri C++ da ifoda deb ataladi. C++ dagi har bir ifoda ; (***nuqta-vergul***) bilan tugatilishi shart. Ortiqcha; bo'sh ifoda deyiladi. Uni qo'yish dastur tezligiga ta'sir qilmaydi.

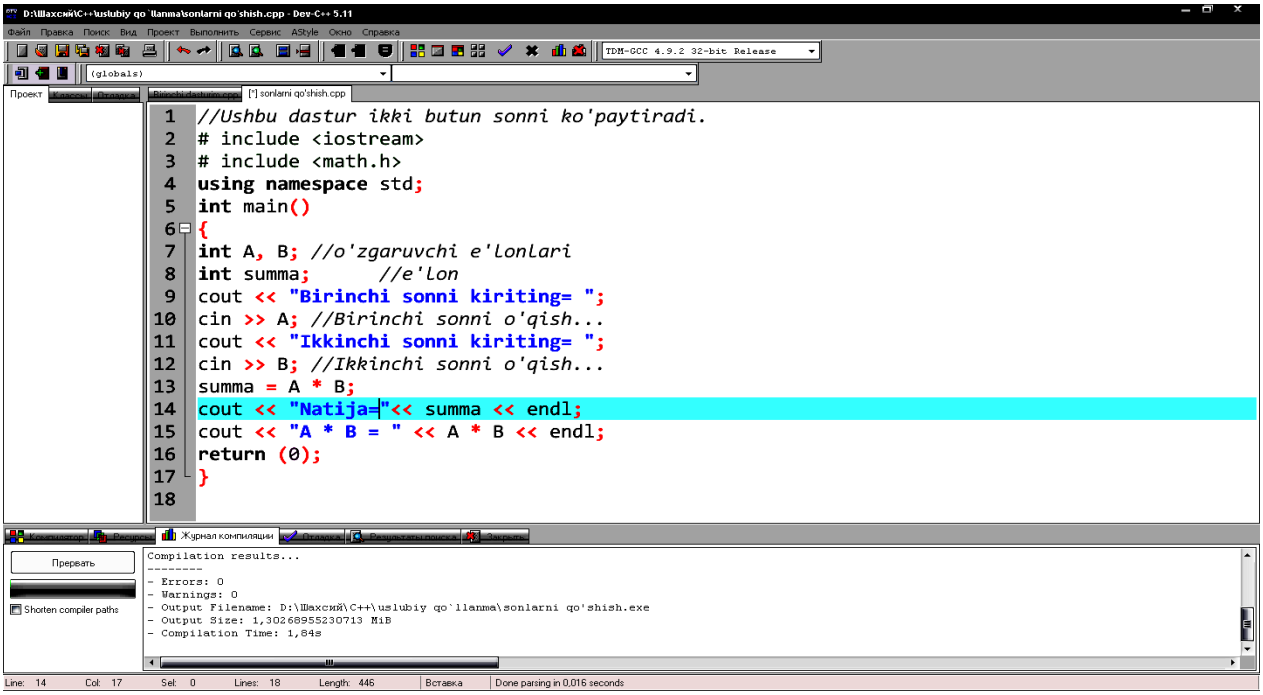
return 0; (***return - qaytmoq***) ifodasi *main()* funksiyasidan chiqishning asosiy yo'lidir. 0 (nol) qiymatining qaytarilishi operatsion tizimga ushbu dastur normal bajarilib tugaganini bildiradi. return orqali qaytadigan qiymat tipi funksiya e'lonidagi qaytish tipi bilan bir hil bo'lishi kerak. Bizda bu e'lon int main(){...} edi. Va 0 int tipiga mansubdir. Bundan keyin return orqali qaytarilayotgan ifodani qavs ichiga olamiz. Misol uchun return (6). Bu qavslar majburiy emas, lekin bizlar ularni programmani o'qishda qulaylik uchun kiritamiz

C++ da yana bir oddiy misolni ko’rib chiqamiz.

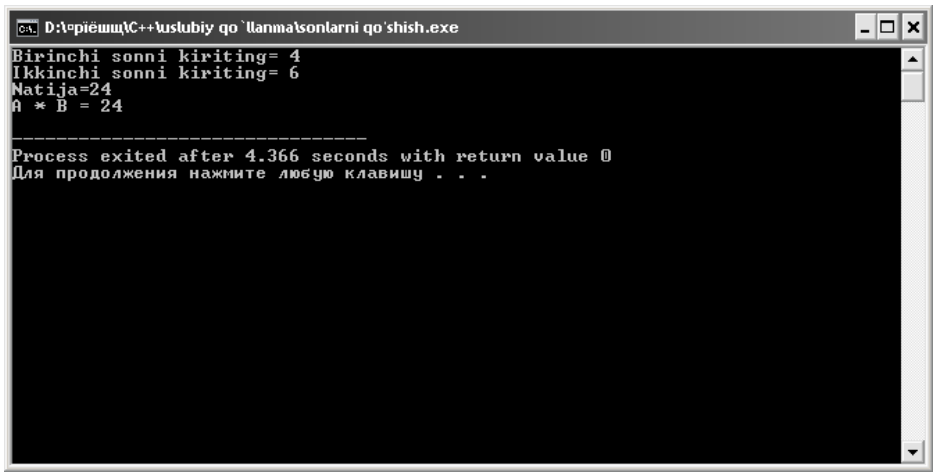
```
//Ushbu dastur ikki butun sonni ko'paytiradi.
# include <iostream>
# include <math.h>
using namespace std;
int main()
{
int A, B; //o'zgaruvchi e'lonlari
int summa; //e'lon
cout << "Birinchi sonni kiriting= ";
cin >> A; //Birinchi sonni o'qish...
```

```
cout << "Ikkinchi sonni kiriting= ";
cin >> B; //Ikkinchi sonni o'qish...
summa = A * B;
cout << "Natija="<< summa << endl;
cout << "A * B = " << A * B << endl;
return (0);
```

Dasturni Dev C++ da kiritamiz:



F9 tugamachasi orqali ham kompilatsiya, ham bajarishni tashkil etamiz:



6. C++ da kutubhonalar

Error.h – hatolarni tekshirish.

Float.h – haqiqiy sonlar bilan ishlash.

Math.h – matematik hisoblashlar.

Stdarg.h – o’zgaruvchi sonli parametrlarni qo’llash.

Conio.h – ekran parametrlarini yuklash.

Iostream.h – kiritish-chikarish vositalari.

Stdlib.h – hotira bilan ishlash.

String.h – simvolli katorlar bilan ishlash.

Time.h – sana va vaqtni aniqlash.

Graphic.h –grafik rejimni aniqlash.

Tajriba ishi yuzasidan topshiriqlar:

1. Matematik model
2. Turli xil jarayonlarni hisoblash algoritmlarini ifodalash usullari
3. Algoritmlarni blok – sxemalar haqida ma’lumot
4. Arifmetik o’zlashtirish operatori
5. Arifmetik ifoda
6. Arifmetik standart funksiyalar
7. Arifmetik ifodalarni yozish qoidalari
8. Ma’lumotlar tiplari
9. Dev C++ dasturini o’rnatish
10. Algoritm nima?
11. O'zgaruvchilarni e'lon qilish
12. Tiplarni dasturlashdagi axamiyati
13. Standart tiplarning qiymatlar qabul qilish oraliqlari
14. Dastur yaratishda tiplarni to`g`ri o`rnatish qoidalari
15. O’zgarmas qiymatlar
16. Butun sonlar toifalarini sanab bering
17. Haqiqiy sonlar toifalarini sanab bering
18. Identifikatorlar
19. Xizmatchi so’zlar
20. C++ da maxsus belgilar
21. Kiritish chiqarish operatorlari
22. Dastur yozish qoidalari
23. Dastur sxemasi (tuzilmasi)
24. C++ da kutubhonalar
25. C++ da asosiy funksiya va uning vazifasi
26. C++da standart nomlar fazosi
27. Kompilyatorlar
28. Komyilatsiya

29. C++ da muhitni ishga yuklovchi fayllar
30. C++ da muhiti
31. C++ da muhiti menyulari
32. C++ da fayl xosil qilish yo’llari va ularning farqi
33. C++ da muhitida uskunalar paneli vazifalari
34. C++ da muhitida quyi paneli vazifalari
35. C++ muhitini sozlash (Servis menyusi)
36. C++ da xosil bo’lgan fayllar, tiplari va vazifalari.
37. C++ da fayllar ustida amallar (Fayl menyusi)
38. C++ da muhitida hatoliklar

Foydalanilgan adabiyotlar:

1. Sh. A. Nazirov, R. V. Kabulov. C++ tili asoslari. KHK uchun o’quv qo’llanma. Toshkent. 2012
2. V. V. Pobelskiy. Yazik C++. Moskva. “Finansi i statistika”.
3. James O. Coplien. Advanced C++ programming styles and idioms. Piter. 2005
4. A.R. Azamatov. Algoritmlash va dasturlash asoslari. Kasb-hunar kollejlari uchun o'quv qo'llanma. Cho'lpon nomidagi nashriyot-matbaa ijodiy uyi Toshkent — 2010
5. Bjarne Stroustrup. The C++ Programming Language. Москва. Издательство БИНОМ -2011
6. http://dasturchi.uz sayti
7. C++ — Википедия sayti

_____ guruh talabasi_____ning "Ob'yektga yo'naltirilgan dasturlash tillari" fanidan joriy va oraliq baholashlarini qayd etuvchi ***reyting varaqasi***

Baho-lash turi	Topshiriq mazmuni	Maksi-mall ball	Bajarish muddati	Olin-gan ball	2-muddat (-1) ball	Natijavi y ball
1-JN	1-amaliy. Amaliy masalalarning matematik modellarini qurish. 2-amaliy C++ muhiti bilan ishlash 3-amaliy C++ tilining asosiy tiplari 4-amaliy Arifmetik o`zlashtirish operatori 1-tajriba. C++ muhiti bilan ishlash. Muhit menyulari ishini va menyu bo`limlari vazifalarini o`rganish. C++ tilining asosiy tiplari. Arifmetik o`zlashtirish operatori 5-amaliy. Chiziqli jarayonlarni hisoblashga doir masqlar bajarish 2-tajriba. Chiziqli jarayonlarning hisoblash algoritmlari va dasturlari. Jarayonda qatnashuvchi arifmetik ifodalarni C++ tiliga o`girish. Dasturni kompyuterga kiritish va sozlash.	2 8 1 7				
	Mustaqil ish topshiriqlarini bajarganligi uchun	1				
	Talabaning amaliy va tajriba mashg'ulotlardagi ishtiroki, kichik guruhdagi faolligi, ijodiy fikrlashi, mustaqil qarorlar qabul qila olishi, mantiqiy xulosalar chiqara olganligi uchun	1				
	Jami: 1-joriy	20 ball				
1-ON	Yozma ish:	10				
	Mustaqil ish topshiriqlarini bajarganligi uchun	3				
	Talabaning ma`ruza mashg'ulotlardagi ishtiroki, ijodiy fikrlashi, mantiqiy xulosalar chiqara olganligi, innovasion g'oya va takliflari uchun	2				
	Jami: 1-oraliq	15 ball				
2-JN	6-amaliy. Tarmoqlanuvchi jarayonlarni hisoblash dasturlari. 7-amaliy. Tarmoqli jarayonlarni hisoblashni tashkil etishda o`tish, shartli va variant tanlash operatorlaridan foydalanish 3-tajriba. Tarmoqlanuvchi jarayonlarni hisoblash algoritmlari va dasturlari. Jarayondagi arifmetik va mantiqiy ifodalarni farqlash va ularni C++ tiliga o`girish. Dasturni kompyuterga kiritish va sozlash. Chiziqli va tarmoqli jarayonlarni farqlash 8-amaliy. Takrorlanuvchi jarayonlarni hisoblash dasturlari. 4-tajriba. Takrorlanuvchi jarayonlarni hisoblash algoritmlari va dasturlari. Jarayonni hisoblash dasturini loyihalash. Chiziqli, tarmoqli va takrorlanuvchi jarayonlarni farqlash. 9-amaliy. Murakkab takrorlanuvchi jarayonlarni hisoblashda takrorlash operatorlarning rekursivligidan unumli foydalanish 5-tajriba. Chekli va cheksiz xadli qator (yig`indi)larni hisoblash algoritmlari va dasturlari. Chekli va cheksiz xadli qator yig`indilarini hisoblash algoritmlari. Qatorlar (chekli va cheksiz) yig`indisi hisobini bajaruvchi dasturni prosedura va funksiyalar yordamida loyihalash.	1 5 1 5 1 5				
	Mustaqil ish topshiriqlarini bajarganligi uchun	1				
	Talabaning amaliy va tajriba mashg'ulotlardagi ishtiroki, kichik guruhdagi faolligi, ijodiy fikrlashi, mustaqil qarorlar qabul qila olishi, mantiqiy xulosalar chiqara olganligi uchun	1				
	Jami: 2-joriy	20 ball				
2-ON	Test nazorati:	10				
	Mustaqil ish topshiriqlarini bajarganligi uchun	3				
	Talabaning ma`ruza mashg'ulotlardagi ishtiroki, ijodiy fikrlashi, mantiqiy xulosalar chiqara olganligi, innovasion g'oya va takliflari uchun	2				
	Jami: 2-oraliq	15 ball				
		Jami _____				

Talabaning reyting bali (JN+ON) :_____+ (YAN)_____= _____

Izoh*: Har bir olingan ball o'qituvchi imzosi bilan tasdiqlanadi. Ushbu varaqa talabaning qo'lida semestr yakunlanguncha turishi shart. Reyting varaqasi yo'qotilsa, uni qayta to'ldirilmaydi.*

O'qituvchi___ _____ _____ Talaba _____ _____ Muhr

(Imzo) (Imzo)