Modul dasturlash asoslari

Alisher Shakirovich Ismoilov
Moxlaroyim Serobiddinova
Madina Mahmudova
Toshkent davlat iqtisodiyot universiteti

Annotatsiya: Modul dasturlash dasturiy ta’minotni loyihalash metodologiyasidir, bu metod dasturiy ta’minotni kichik, mustaqil modullarga bo‘lishni ta’kidlaydi, har biri dasturiy ta’minotning ma’lum bir qismiga javobgar. Ushbu yondashuv kodni tashkil etishni rivojlantiradi va qo‘shimcha o‘zgarishlar kiritishni osonlashtiradi, chunki dasturchilar alohida modullar bilan ishlashlari mumkin va bu butun tizimga ta’sir qilmaydi. Ushbu metodologiya murakkab dasturiy ta’minotni boshqarishni soddalashtiradi, chunki bir modulda o‘zgarishlar boshqa modullarda o‘zgarishlarni talab qilmaydi, bu esa samaraliroq rivojlanish jarayonlariga olib keladi.

Kalit so‘zlar: inkapsulyatsiya, SRP modul, qayta foydalanish, ajratish, sinov imkoniyati

Fundamentals of modular programming

Alisher Shakirovich Ismailov
Mokhlaroyim Serobiddinova
Madina Mahmudova
Tashkent State University of Economics

Abstract: Modular programming is a software design methodology that emphasizes dividing software into small, independent modules, each responsible for a specific part of the software. This approach improves code organization and makes it easier to make incremental changes because developers can work with individual modules without affecting the entire system. This methodology simplifies the management of complex software because changes in one module do not require changes in other modules, leading to more efficient development processes.

Keywords: encapsulation, SRP module, reusability, separation, testability

Kirish
Modul Dasturlashning Asosiy Tamoyillari:
Encapsulation

Inkapsulyatsiya - bu ma’lumotlarni va ularni boshqaradigan metodlarni bitta birlik yoki modulda to‘plab, tashqi aralashuvlardan himoya qilish texnikasidir.

- Foydasi:
  - Dastur kodining xavfsizligini oshiradi.
  - Dasturdagi o‘zgarishlarning boshqa qismlarga ta’sirini kamaytiradi.
  - Murakkablikni kamaytiradi va xatoliklarni oldini oladi.

Single Responsibility Principle (SRP)

SRP modul yoki sinf faqat bitta, va faqat bitta, o‘zgarish sababiga ega bo‘lishi kerakligini ta’kidlaydi, bu esa uning faqat bitta mas’uliyatni bajarishi kerakligini anglatadi.

- Foydasi**:**
  - Dasturiy ta’minotni yanada tushunarli qiladi.
  - Kodning o‘zgartirilishi va yangilanishini osonlashtiradi.
  - Testlashni osonlashtiradi.

Reusability

Qayta foydalanish mavjud modullar yoki komponentlarni yangi ilovalarda ishlatish qobiliyatini anglatadi, bu esa kodni takrorlash zaruratini kamaytiradi va samaradorlikni oshiradi.

- Foydasi**:**
  - Kod yozish va test qilish vaqtini kamaytiradi.
  - Xatoliklarni kamaytiradi, chunki qayta ishlatilgan kod allaqachon testdan o‘tkazilgan bo‘ladi.

Decoupling

Ajratish - modullar o‘rtasidagi bog’liqliklarni kamaytirish tamoyilidir, bu modullarning mustaqil ravishda ishlashiga imkon beradi va shu bilan texnik xizmat ko‘rsatishni va kengaytirishni soddalashtiradi.

- Foydasi**:**
  - Dasturiy ta’minotni boshqarish va texnik xizmat ko‘rsatishni osonlashtiradi.
  - Mustaqil testlash va o‘zgarishlarni amalga oshirishga imkon beradi.

Testability

Sinov Imkoniyati modullarni mustaqil ravishda sinab ko‘rish imkoniyatini ta’minlaydi, bu esa ularning funksionalligini tasdiqlashni osonlashtiradi va muammolarni aniqlash va tuzatishni osonlashtiradi.

- Foydasi**:**
  - Katta dasturiy ta’minotni sinovdan o‘tkazishni osonlashtiradi.
  - Xatolarni topish va tuzatishni tezlashtiradi.

II Asosiy qism:

Modul Dasturlashda Inkapsulyatsiya

Inkapsulyatsiyaning Ta’rifi, Maqsadi va Foydalari

- Inkapsulyatsiya dasturlash tamoyilidir, bu obyektning ba’zi komponentlariga to‘g’ridan-to‘g’ri kirishni cheklaydi, bu esa obyekt ichidagi metodlar va ma’lumotlardan nomaqsadli aralashuv va noto‘g’ri foydalanishni oldini olish usulidir.
- inKapsulyatsiyaning asosiy maqsadi ma’lumotlarni (atributlarni) va ma’lumotlar ustida ishlaydigan metodlarni (funksiyalarni) bitta birlik yoki sinfga to‘plab, modullikni oshirish va ma’lumotlarning yaxlitligini himoya qilishdir.
- Foydalari orasida yaxshilangan kodni boshqarish, interfeysni amalga oshirishdan aniqroq ajratish va obyektning ichki holatini tashqi manipulyatsiyalardan yashirish orqali xavfsizlikning oshishi mavjud.
- Masalan, C++ dasturlash tilida kapsulyatsiya private va public kirish spetsifikatorlari yordamida amalga oshiriladi. Bu sinf a’zolariga to‘g’ridan-to‘g’ri kirishni cheklaydi va umumiy metodlar orqali boshqarilgan o‘zaro aloqani ta’minlaydi.

Single Responsibility Principle (SRP)

SRP (Yagona Mas’uliyat Tamoyili) Ta’rifi va Muhimligi

- (SRP) sinf faqat bitta o‘zgarish sababiga ega bo‘lishi kerakligini, ya’ni u faqat bitta mas’uliyat yoki ishni bajarishi kerakligini ta’kidlaydi.
- SRP’ga rioya qilish orqali dasturchilar tushunarli va boshqarilishi oson kod yaratadilar, chunki har bir modul yoki sinf aniq maqsad va vazifaga ega bo‘ladi.
- SRP kodni qayta foydalanishni oshiradi, chunki maxsus vazifalar uchun loyihalangan sinflar o‘zgartirishlarsiz turli kontekstlarda osonlik bilan qayta ishlatilishi mumkin.
- Misol uchun, agar bir User sinfi foydalanuvchi ma’lumotlari va foydalanuvchi interfeysi ko‘rsatishni boshqaradigan bo‘lsa, bu SRP’ni buzadi. Buning o‘rniga, mas’uliyatlarni UserManager (foydalanuvchi ma’lumotlari uchun) va UserInterface (ko‘rsatish vazifalari uchun) kabi ajrating.

Modulardan Foydalanib Dasturlar Qurish

Analyzing Requirements

Dastur qanday funksiyalarni amalga oshirishi kerakligini aniqlang. Umumiy maqsadlarni mustaqil ravishda bajarilishi mumkin bo‘lgan aniq vazifalarga bo‘ling.

Module Partitioning

Aniqlangan vazifalarga asoslanib, dasturiy ta’minotni kichikroq, boshqariladigan modullarga bo‘ling. Har bir modul bitta mas’uliyat yoki funksionalni hal qilishi kerak.

Designing Interfaces

Har bir modul uchun qanday o‘zaro aloqalar o‘rnatilishini belgilovchi aniq interfeyslarni yarating. Bu bog’liqliklarni kamaytiradi va modullar o‘rtasidagi kommunikatsiyani yaxshilaydi.

Writing Modules

Har bir modulni alohida fayl yoki sinf sifatida amalga oshiring, uning belgilangan vazifasini bajarishini ta’minlang. Kodlash standartlariga rioya qiling, shunda kodning o‘qilishi va bir xilligi saqlanadi.

C++ Modulni Tashkil Etish

C++ da Modulni Tashkil Etishning Umumiy Ko‘rinishi:

- C++ modullarni sarlavha (.h) va amalga oshirish (.cpp) fayllari yordamida tashkil etadi.
- Sarlavha fayllari funktsiya prototiplari va sinf deklaratsiyalarini o‘z ichiga oladi.
- Amalga oshirish fayllari funktsiya va metodlarning aniqlanishlarini o‘z ichiga oladi.

C++ Modulni Tashkil Etish

Sarlavha va Amalga Oshirish Fayllari Misollari

- Sarlavha fayli misoli (math_utils.h):
- class MathUtils {public: // Static method to add two integers static int add(int a, int b); // Static method to subtract two integers static int subtract(int a, int b);};
- Amalga oshirish fayli misoli (math_utils.cpp):
- #include "math_utils.h"// Definition of the static method to add two integersint MathUtils::add(int a, int b) { return a + b;}// Definition of the static method to subtract two integersint MathUtils::subtract(int a, int b) { return a - b;}

Ajratish va Sinov imkoniyati

Decoupling

“Ajratish” dasturiy ta’minot modullari o‘rtasidagi bog’liqliklarni kamaytirishni anglatadi. Modullar orasidagi bog’liqliklarni minimallashtirish orqali, bir moduldagi o‘zgarishlar boshqa modullarga kamroq ta’sir qiladi, bu esa moslashuvchanlik va boshqarish osonligini oshiradi.

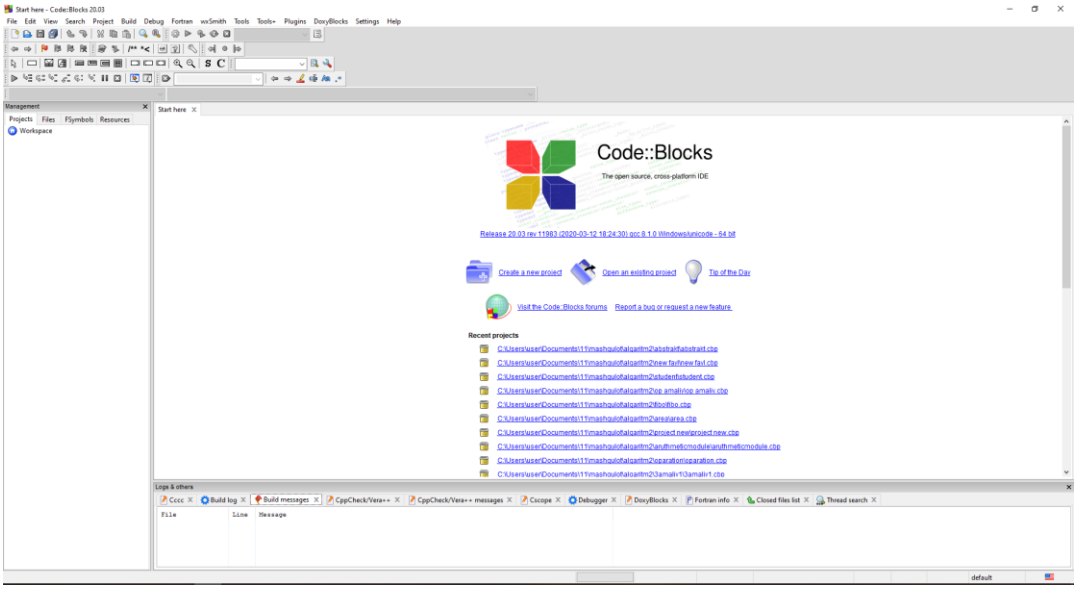
Importance of Decoupling

Ajratish dasturchilarga alohida modullarni o‘zgartirish, almashtirish yoki yangilash imkonini beradi, bu esa butun tizimga ta’sir qilmaydi. Bu, o‘z navbatida, xatolarni aniqlashni osonlashtiradi va dasturiy ta’minotning umumiy barqarorligini oshiradi.

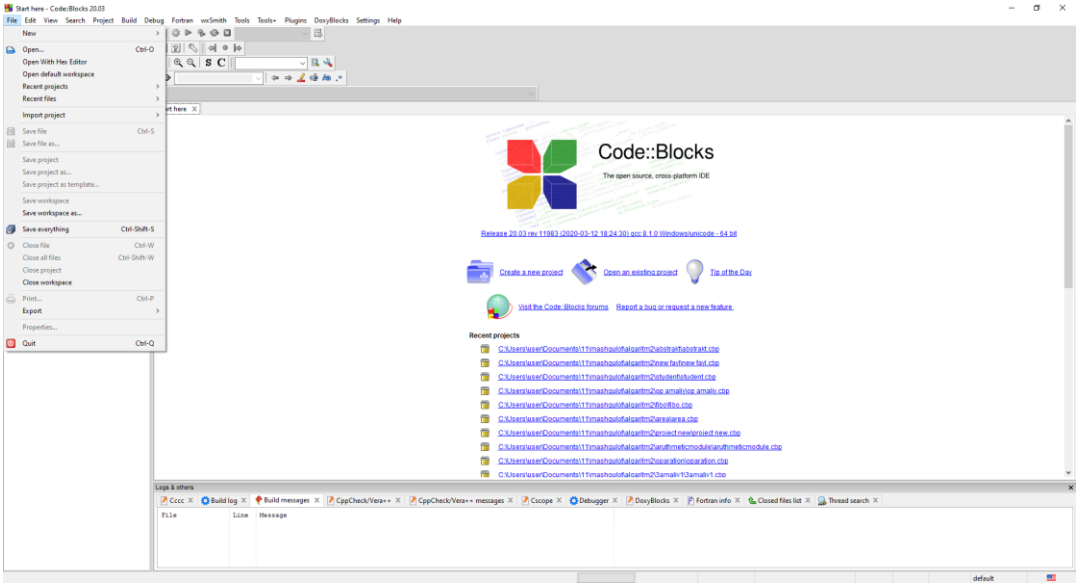
Testability

Sinov Imkoniyati modulning mustaqil ravishda sinab ko‘rilishi osonligini anglatadi. Yuqori sinov imkoniyati har bir modulning to‘g’riligi tekshirilishini ta’minlaydi, butun tizim bilan integratsiya qilish zarurati bo‘lmaydi, bu esa rivojlanish sikllarini tezlashtiradi.

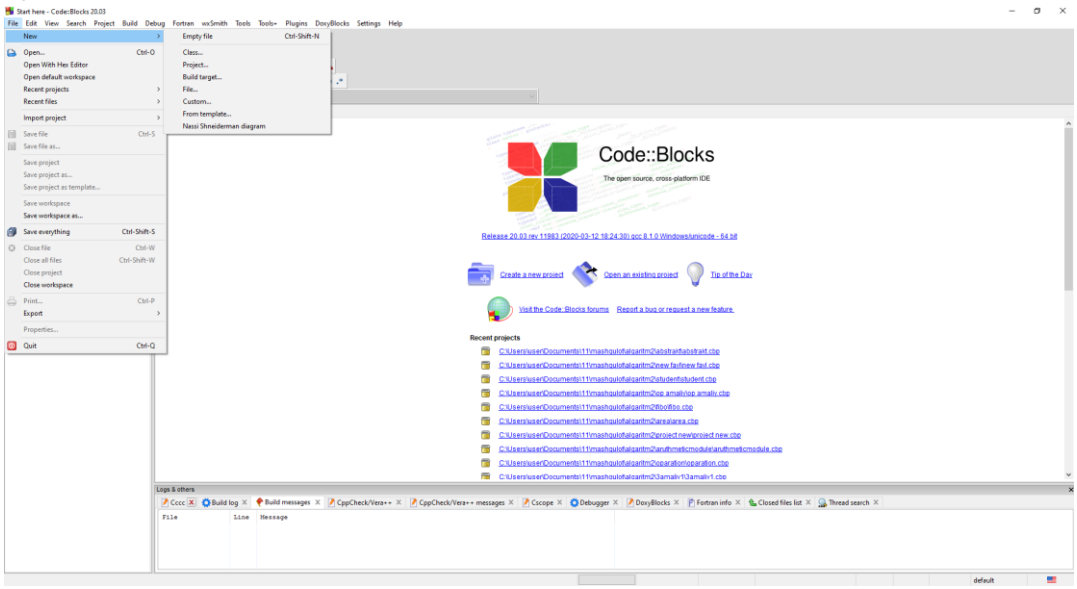
Code::block ga kiriladi:

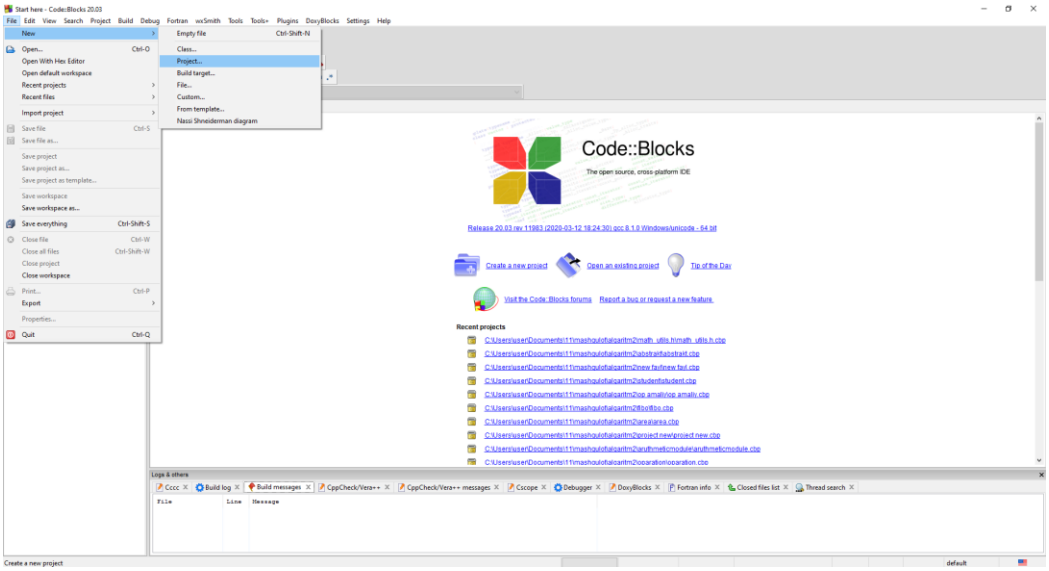


New:

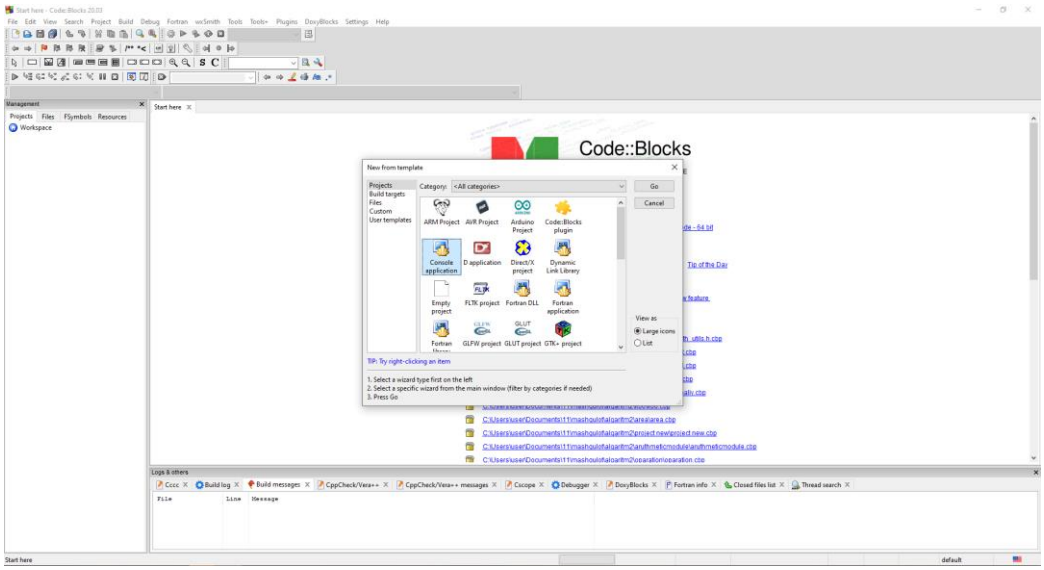


Project:

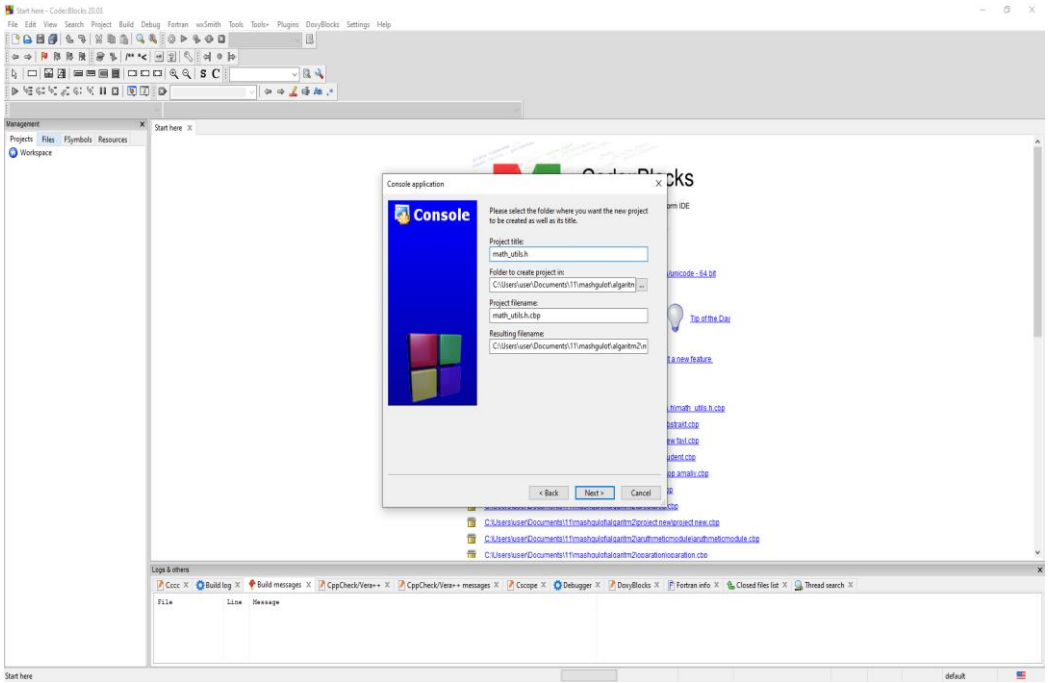


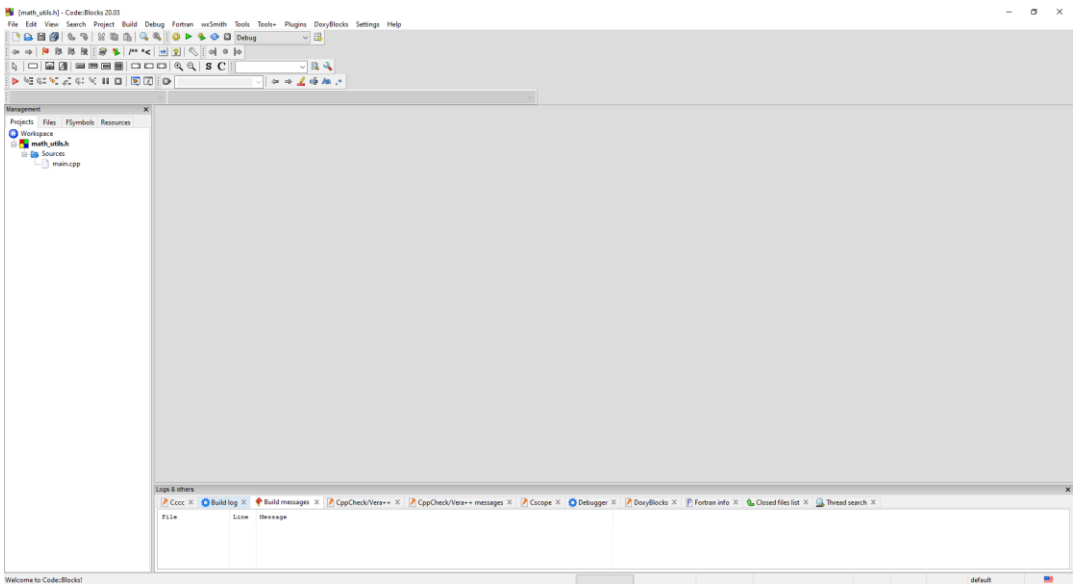


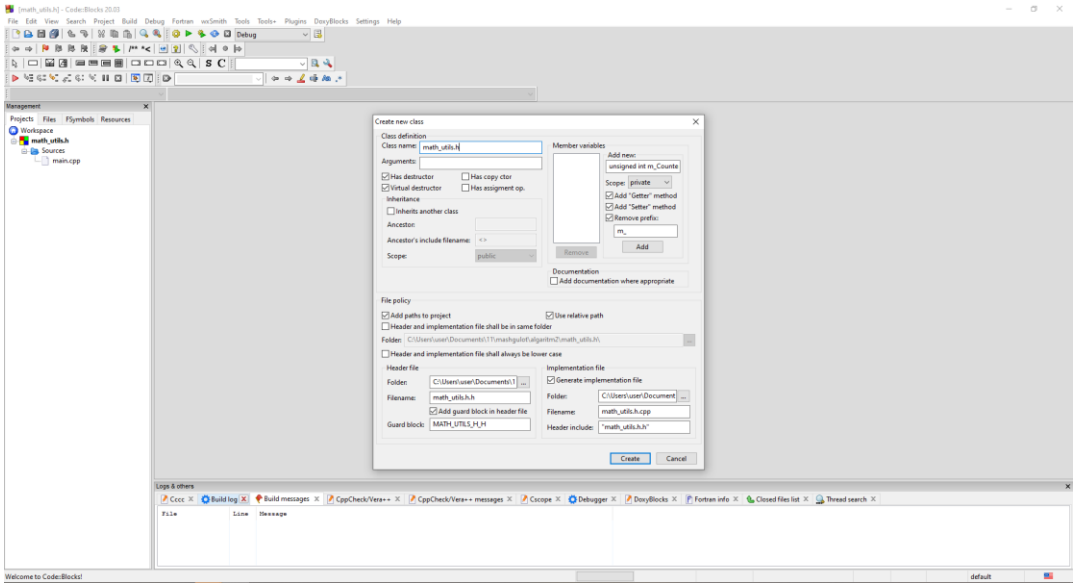
Console aplication

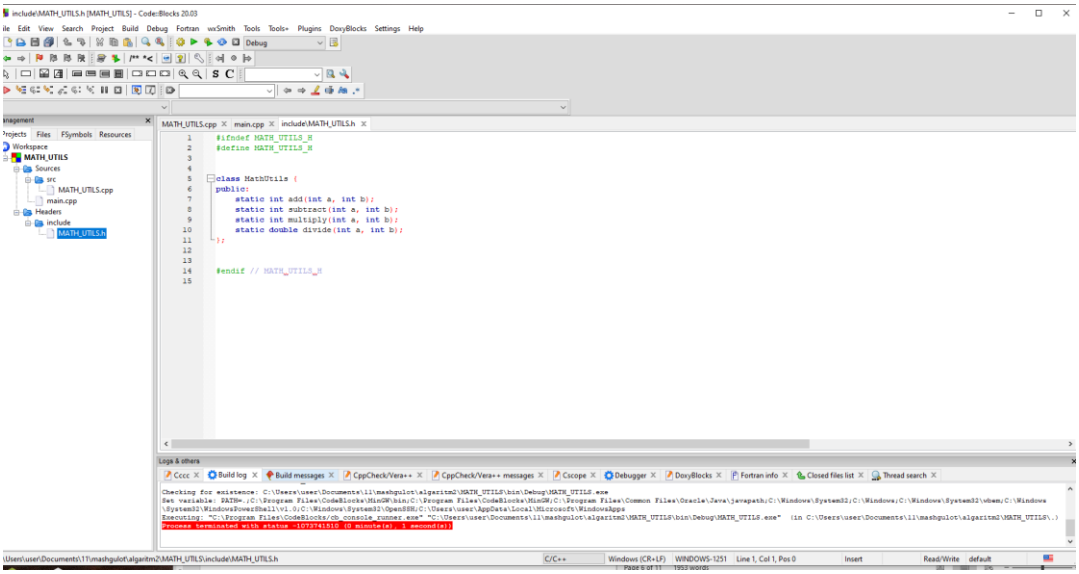


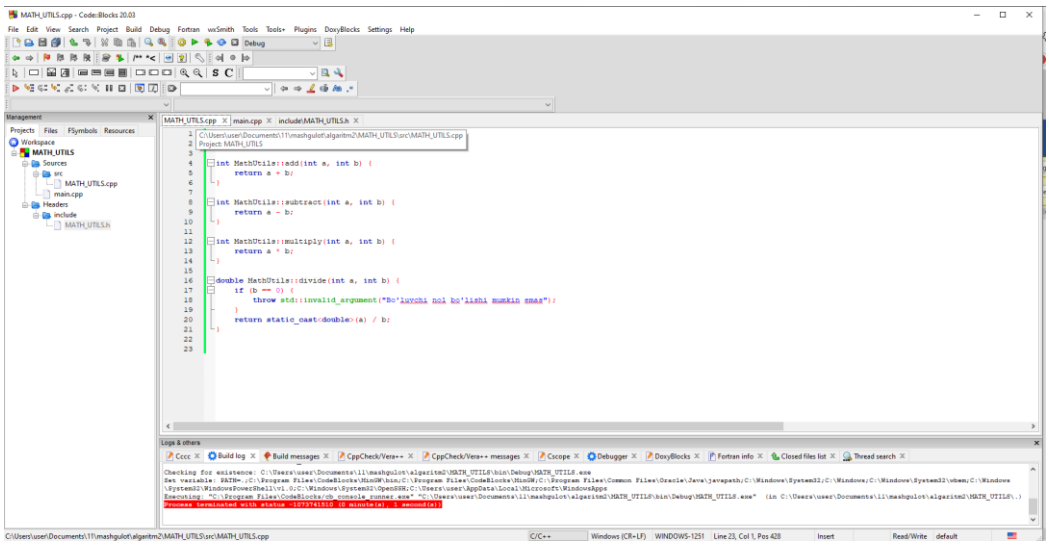
Fayl nomi yoziladi:

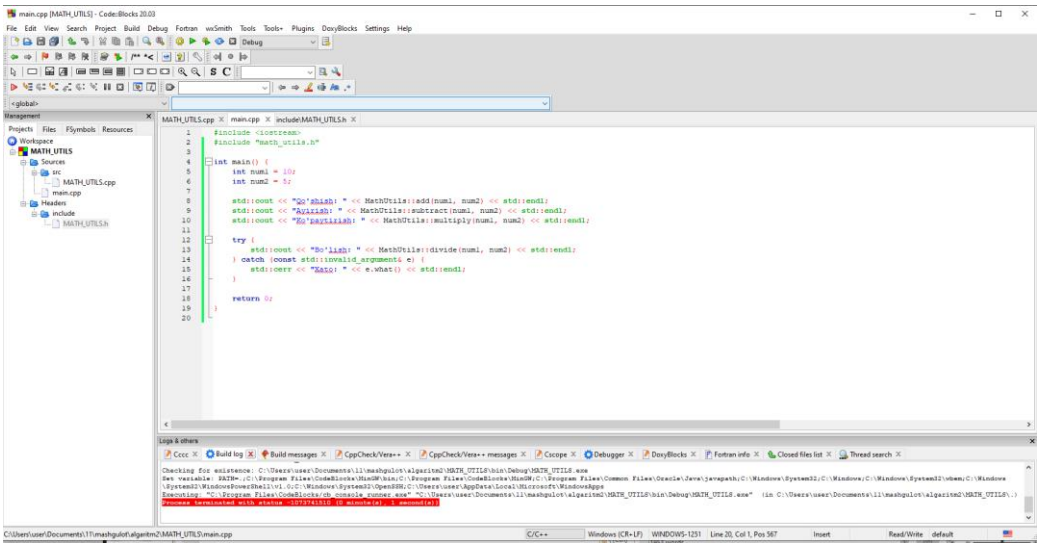




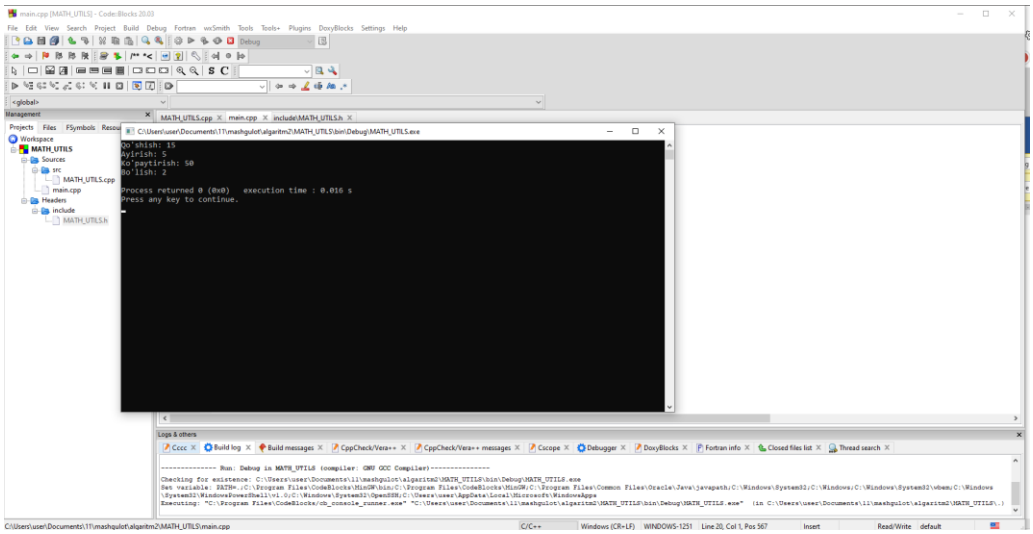








Natija:



Xulosa:

Modullar asosida dasturlarni yaratish dasturiy ta’minotni boshqarishni, sinashni va kengaytirishni osonlashtiradi. Dastur tarkibini modullarga bo‘lish va ularni interfeyslar orqali bog’lash modullararo bog’lanishni kamaytiradi, dasturiy ta’minotni

qayta ishlatish imkoniyatini yaratadi va katta tizimlarni boshqarishni soddalashtiradi. C++ tilida modullarni tashkil qilish header va implementatsiya fayllari orqali amalga oshiriladi, bu esa modullarni bir-biridan ajratish va kodni tartibli saqlashni ta’minlaydi.

Foydalanilgan adabiyotlar

[1] Ismailov, A., Jalil, M. A., Abdullah, Z., & Abd Rahim, N. H. (2016, August). A comparative study of stemming algorithms for use with the Uzbek language. In 2016 3rd International conference on computer and information sciences (ICCOINS) (pp. 7-12). IEEE.

[2] Informatika va programmalash.O‘quv qo‘llanma. Mualliflar: A.A.Xaldjigitov, Sh.F.Madraximov, U.E.Adamboev, O‘zMU, 2005 yil

[3] O.Shukurov, F.Qorayev, E.Eshboyev, B.Shovaliyev “Programmalashdan masalalar to‘plami”. Toshkent 2008,160 bet

[4] "Grokking algaritmlash" - Aditya Bhargava

[5] "INtroduction to Algorithms" - Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rives

[6] Y. AI-Nashashibi, D. D. Neagu and Y. Ali, “Stemming Techniques for Arabic Words: A Comparative Study,” 2nd International Conference on Computer Technology and Development (ICCTD), 2010, pp. 270-276.

[7] H. Mohammad, B. Zuhair, C. Keely and M. David, “An Arabic Stemming Approach Using Machine Learning with Arabic Dialogue System,” ICGST AIML-11 Conference, Dubai, April 2011, pp. 9-16.

[8] L. S. Leah, B. Lisa and C. E. Margaret, “Improving Stem- ming for Arabic Information Retrieval: Light Stemming and Co-occurance Analysis,” SIGIR, ACM, 11-15 August 2002.

[9] L. S. Leah, B. Lisa and C. E. Margaret, “Conservatice Stemming for Search and Indexing,” ACM, August 2005, pp. 15-19.

[10] S. Jikitsha and P. C. Bankim, “Stemming Techniques and Naïve Approach for Gujarati Stemmer,” International Conference in Recent Trends in Information Technology and Computer Science, IJCA, 2012, pp. 9-11.