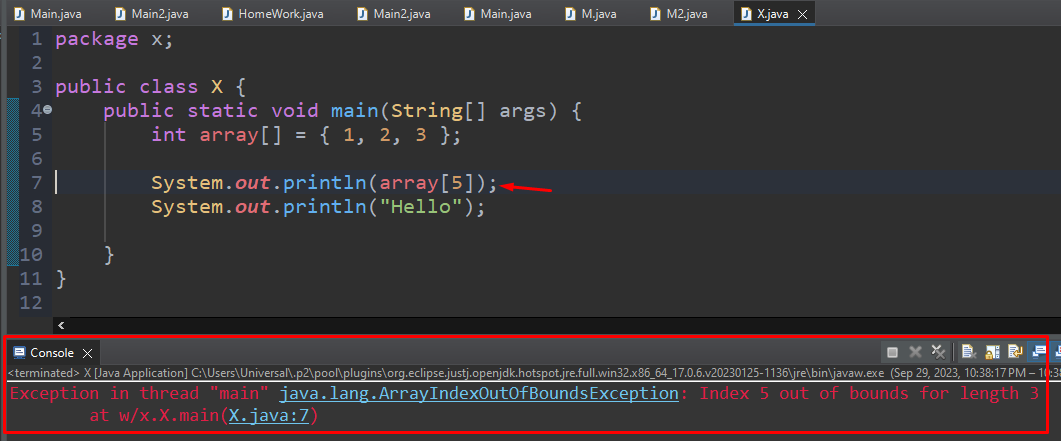
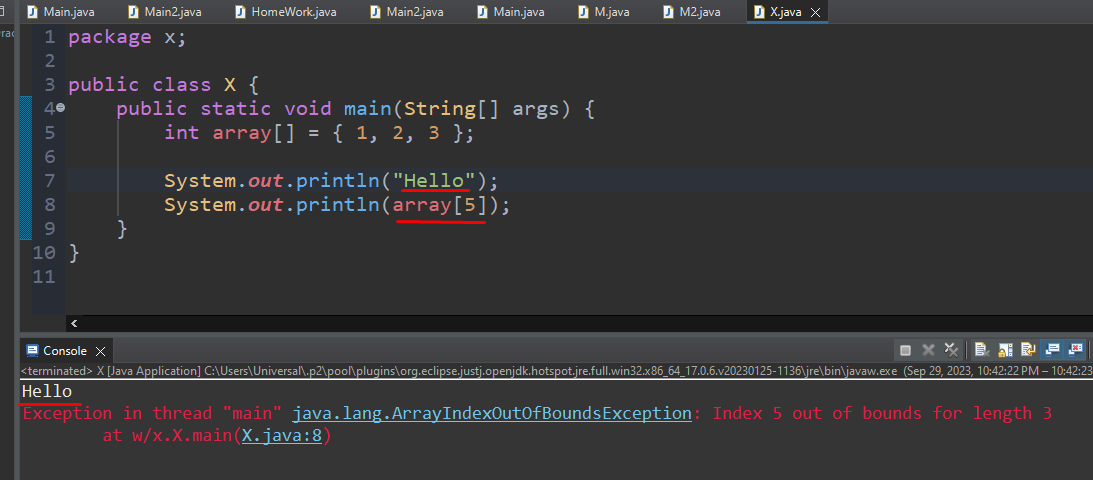
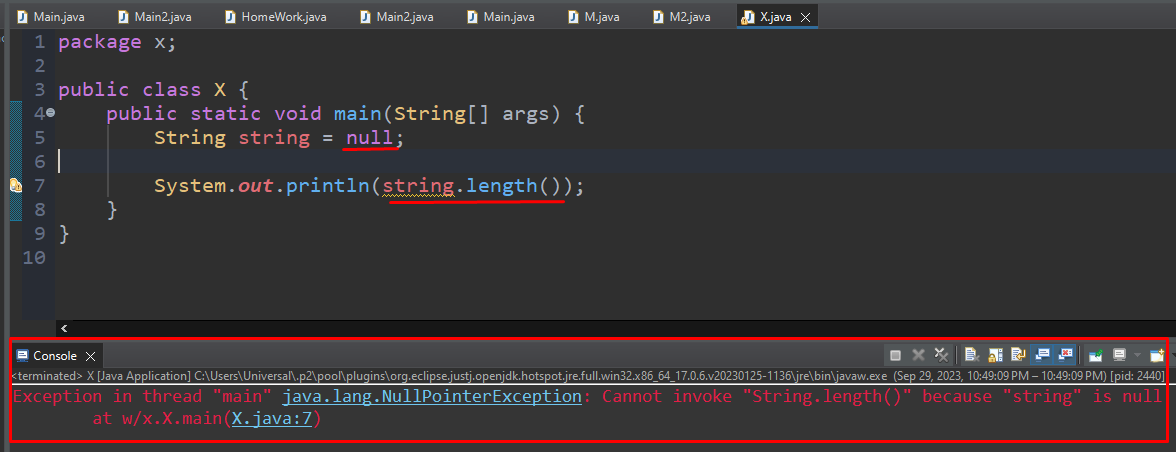
1. Javada agar kodimizni bajarish davomida qandaydir xatolik chiqsa, xatolik chiqqan joydan keying kodimiz ishlamaydi. Pastda **array** o’zgaruvchimizda 3 ta element bor bo’lib, **7-**qatorda **5-**index ni chiqar deyapmiz. Bu xatolikka olib keladi. Shuning uchun biz xatolik olyapmiz. Lekin **8-**qatordagi **sout** miz chiqmaydi. Error dan keyingi qism ishlamaydi:



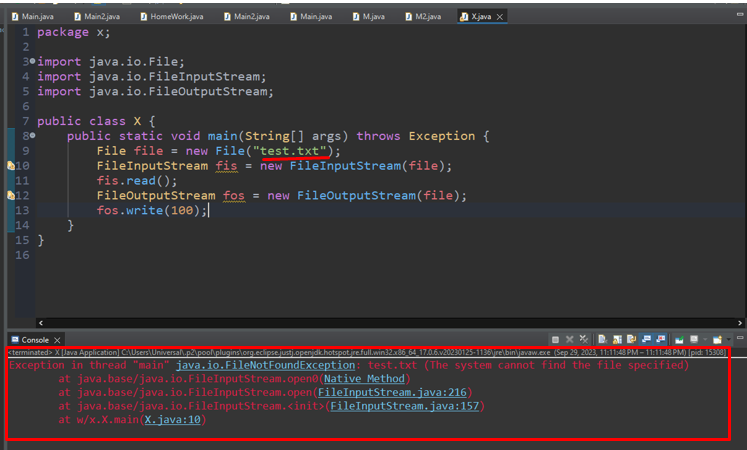
Lekin xatolikdan oldingi kodimiz ishlayveradi:



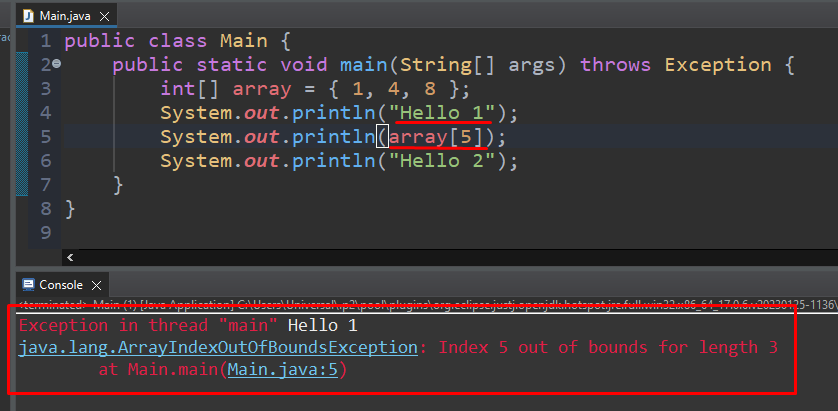
1. Javada qiymati **null** bo’lgan o’zgaruvchini **length()** ini ola olmaymiz, bunday holatda **NullPointerException** beradi:



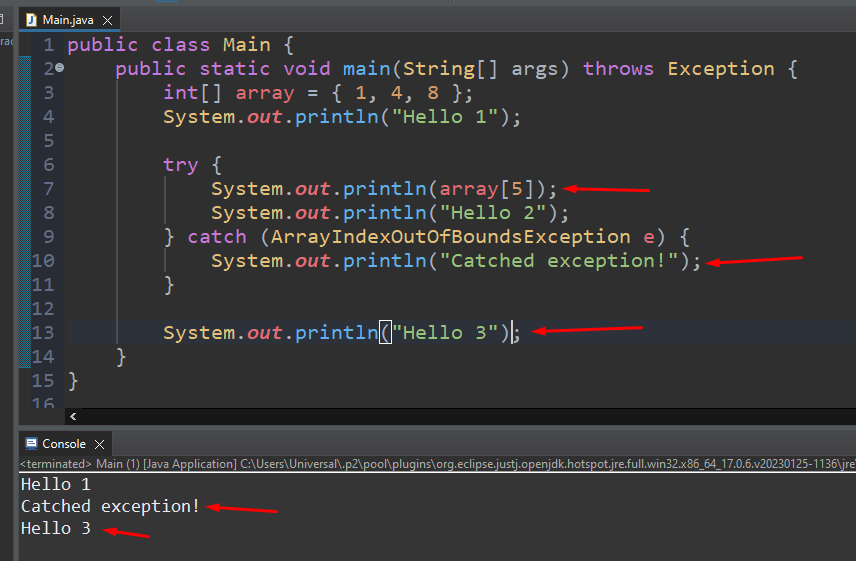
1. Agar file miz mavjud bo’lmasa, u holda **FileNotFoundException** xatoligi uzatiladi. Pastda ham **test.txt** filemiz hali mavjud emas, shuning uchun bizga shunday error berdi:

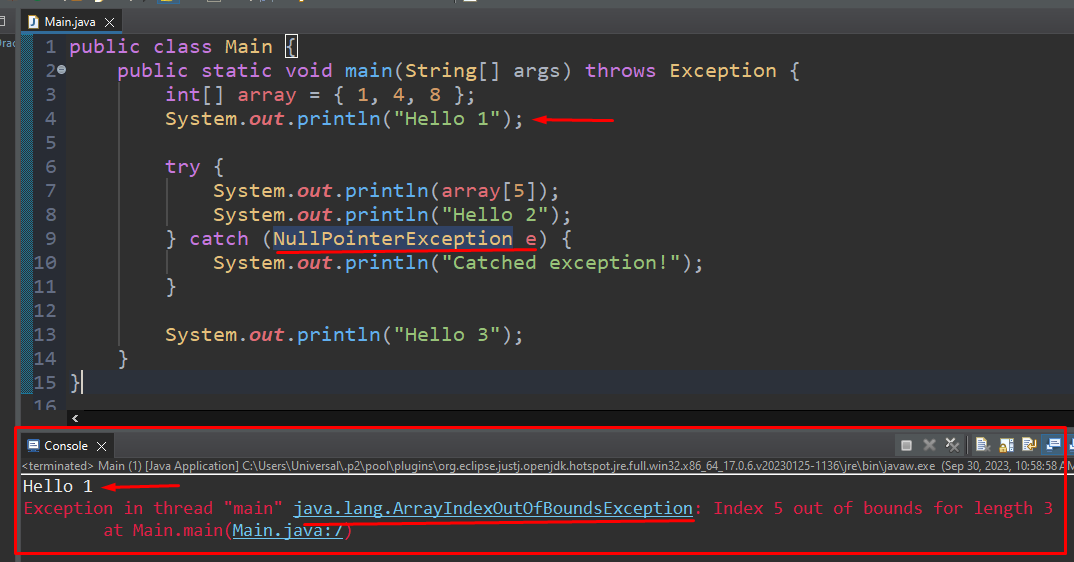


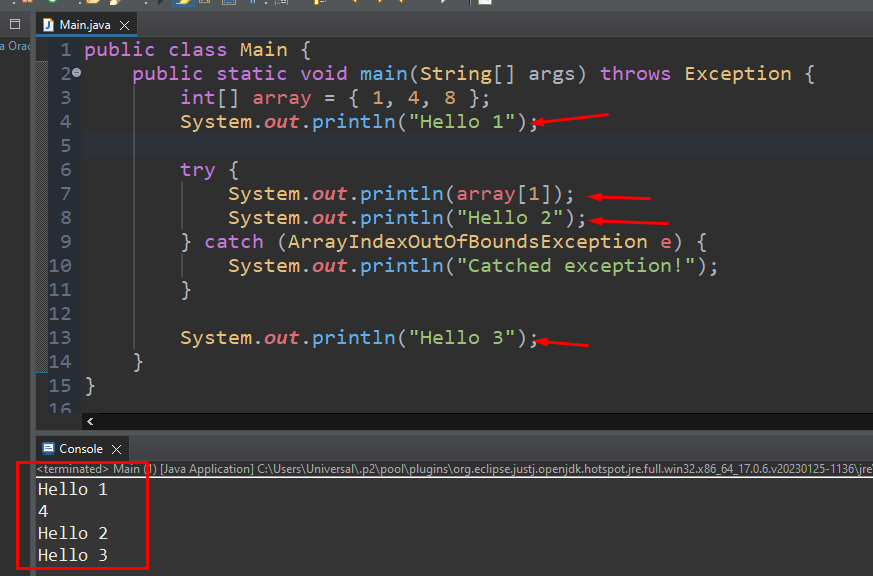
1. Try-catch ni ko’ramiz. Avval pastdagi misolni ko’raylik. Bizda array nomli massiv bor va u o’zida 3 element saqlaydi. Biz 5-qatorda shu array ni 5-indexini chiqargin deyapmiz. Lekin bunday index yoq’ligi uchun xatolik beradi. Lekin xatolikkacha bo’lgan qismi ishlaydi, ya’ni sout(“Hello 1”); qismi ekranga chiqadi, lekin xatolikdan keying qismi, ya’ni sout(“Hello 2”); qismi chiqmaydi. Bu degani xatolik bo’lsa, shu qismdan keying qismlar umuman ishlamaydi:



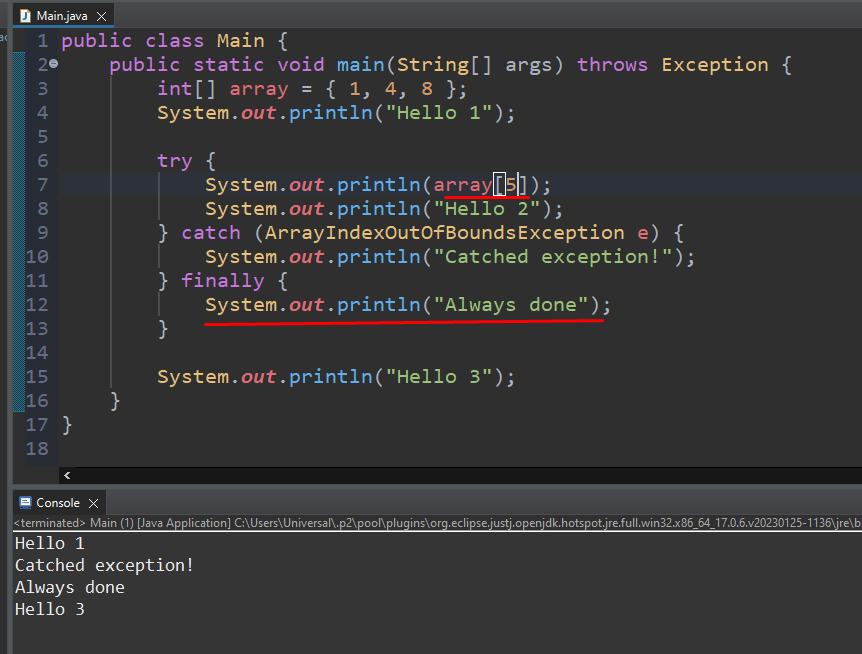
Keling endi yuqoridagi kodni biroz o’zgartiramiz, ya’ni faqat **try-catch** block qo’shamiz va **sout(“Hello 3”);** ni qo’shamiz xolos. Pastda kodimiz ishga tushushishni boshlaganda **4**-qatordagi **sout(“Hello 1”)** ni hiqaradi. Keyin **try-catch** block yaratdik va uni ichiga **array[5]** ni qatorni soldik. Bu qator bilamizki error beradi. Error chiqqan paytda shu error **try** blockda tutib olinadi va qayta ishlash uchun **catch** blockka uzatiladi. E’tiborli joyi, error chiqqandan keyin **sout(“Hello 2”);** qismi bajarilmaydi. **Catch** blockda biz oldindan **ArrayIndexOutOfBoundsException** nomli exception tashlashini bilganimiz uchun ham shu nomli exceptionni yozdik. Endi catch ni ichidagi **10**-qatordagi **sout()** bajariladi. E’tibor bergan bo’lsangiz, **13**-qatordagi **sout(“Hello 3”)** qismi ham bajarilyapti. Sababi **try-catch** da ushlangan xatoliklar qayta ishlanadi va chiqqan error kodni bloklab, keying qismini bajarilishiga to’sqinlik qilmaydi:



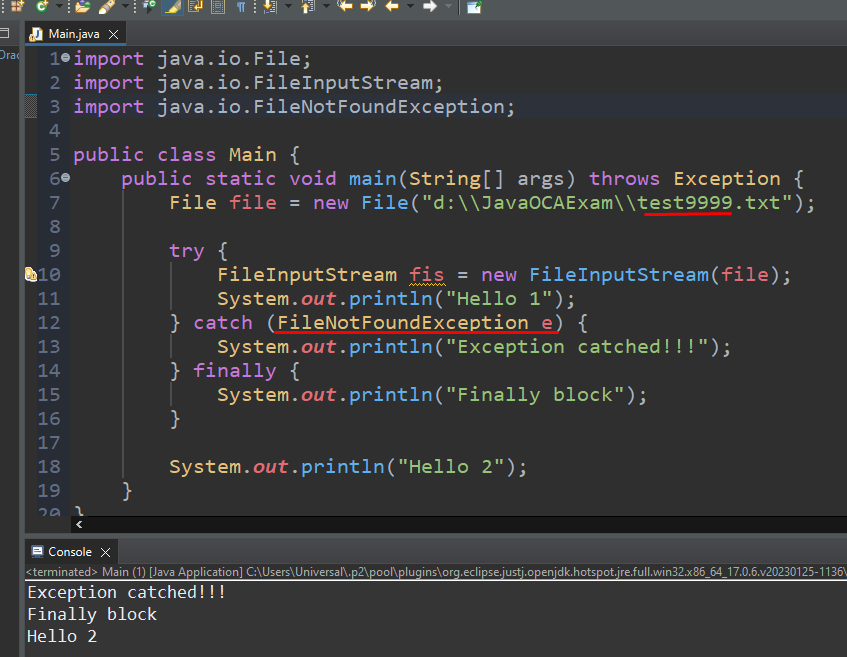
1. Keling endi catch blockda **ArrayIndexOutOfBoundsException** ni o’rniga **NullPointerException** yozib ko’raylik. Bunday holatda kodimiz bajarilib sout(“Hello 1”) ni chiqaradi, keyin esa try ga kiradi va array[5] da xatolik sodir bo’ladi. Shu yerda **ArrayIndexOutOfBoundsException** exceptioni hosil bo’lib, catch blockga yuboriladi. Lekin bizda faqat **NullPointerException** hosil bo’lgandagina tutib olinadigan catch block yozilgan **9**-qatorda. Shuning uchun **ArrayIndexOutOfBoundsException** exceptionni tutib oladigan **catch** block yo’qligi uchun, bu error qayta ishlanmaydi va kodimiz **7**-qatordayoq bajarilishdan to’xtaydi va qolgan qismlari bajarilmaydi, xattoki **9**-qatordagi **catch** ga ham kirmaydi chunki bu catch **NullPointerException** ni tutish uchun mo’ljallangan. Shuning uchun **13**-qatordagi **sout(“Hello 3”)** qismi bajarilmaydi. Demak, xatolik chiqsa ham o’ziga mos tushadigan **catch** block bo’lmasa, bu xatolik boshqa o’ziga mos tushmaydigan **catch** blockda tutib olinmaydi:
2. Agar kodimizda hech qanday xatolik bo’lmasa, u holda **catch** blockka kirib ham o’tirilmaydi. Pastdagi misolda **array[1]** index bor bizda. Shuning uchun xatolik bermaydi:

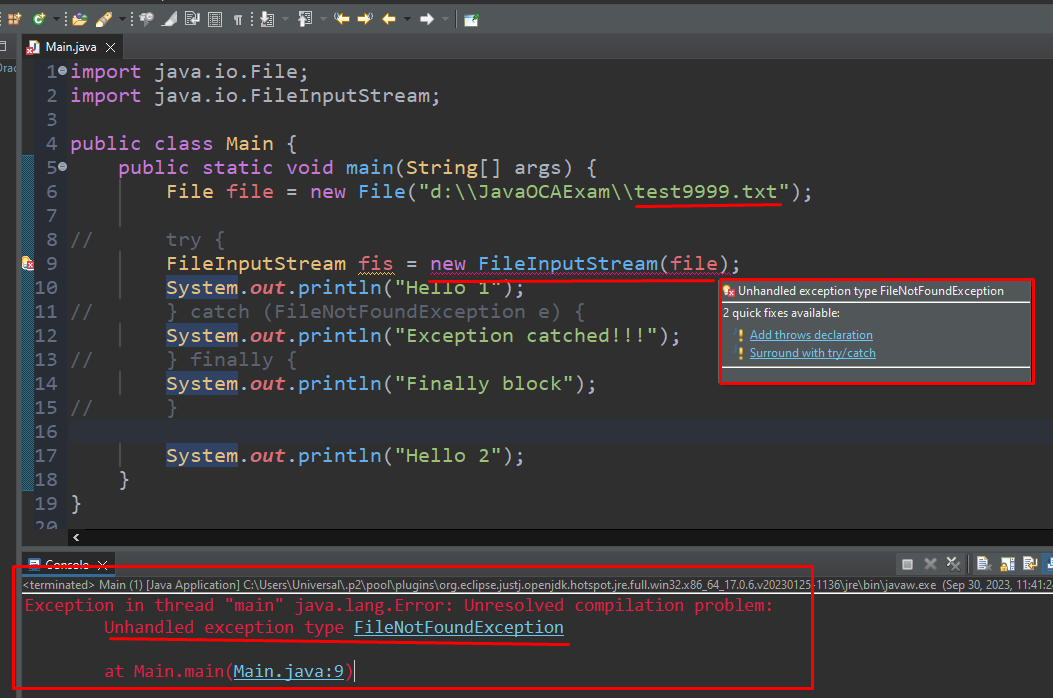


1. **Finally** block bizda kodimiz exception tashlaydimi yo’qmi farqi yo’q, doim bajariladi. Pastda xatolik tashlangan holat berilgan bo’lib, **finally** block bajarilganini ko’rishimiz mumkin:



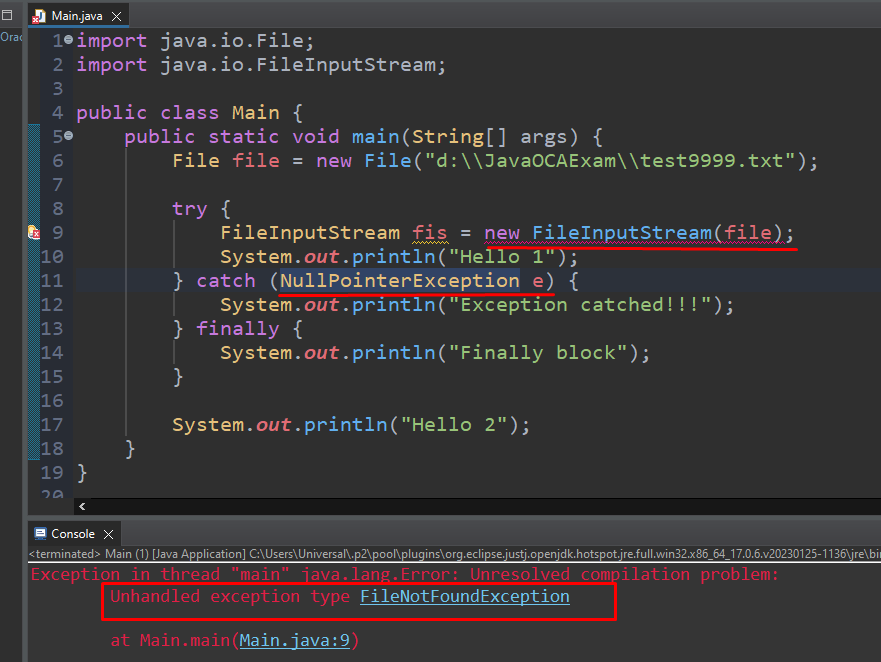
1. Asosan file lar bilan ishlashda xatoliklar kelib chiqishi mumkin. Sababi biz o’qimoqchi yoki yozmoqchi bo’lgan file miz yo’q yoki o’qish va yozishda file yopilib qolishi mumkin. Shunday holatlarda kodimizni keyingi qismini blocklamaslik uchun try-catch ga olishimiz zarur. Pastdagi misolda test9999.txt filemiz mavjud emas. Lekin biz bu fileni o’qimoqchi bo’lyapmiz. Albatta bizga xatolik berib, exception tashlaydi:



Xo’sh agar biz shunchaki try-catch blockni yozishni istamasak, u holda nima bo’ladi. Bunday holatda JAVA bizni try-catch block yozishga majburlaydi, aks holda xatolik beradi. Masalan pastda biz yuqoridagi misolni o’zini ko’ramiz, faqat try catch finally larni commentga olib qo’yamiz. E’tibor bergan bo’lsangiz compile time ni o’zidayoq xatolik beryapti eitorda 9 –qatorda tagiga chizib:

Nega try-catch block yozishimiz shart, sababi Java bilmaydi biz o’qimoqchi yoki yozmoqchi bo’lgan file miz borligini ham yo’qligini ham. Xattoki filemiz bo’lsa hamki, biz majburmiz try-catch blockka olishga. Shunday holatlarda biz try-catch ga olamiz kodimizni.

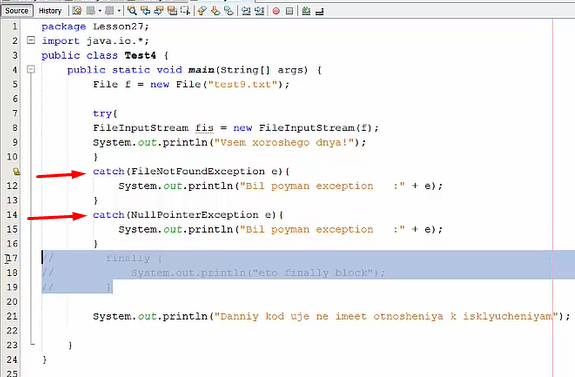
Odatda FileInputStream classi FileNotFoundException otadi. Lekin biz shu exceptionni o’rniga NullPointerEcxeption ni otadigan bo’lsak, uje compile time ni o’zidayoq xatolik beradi 9-qatorda. Chunki FileInputStream classi FileNotFoundException ni catch qilishi kerak, NullPointerEcxeption ni emas. Shuning uchun biz majburmizki FileNotFoundException exceptionni catch qilishga:



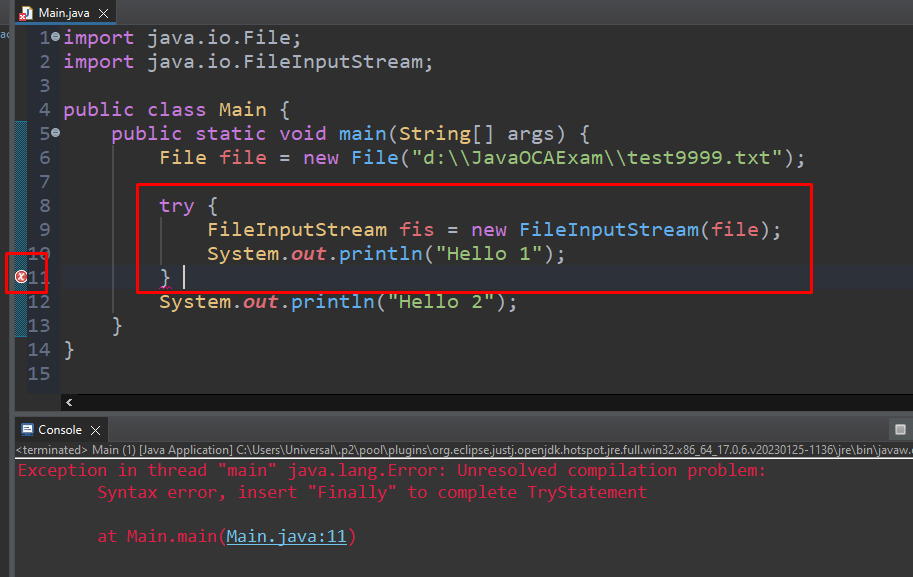
1. Try-catch-finally blockda try va finally blocklar doim bitta bo’ladi, catch block esa bir nechta bo’lishi mumkin. Bundan tashqari finally blockni yozish ixtiyoriydir, istasak yozmasligimiz mumkin:



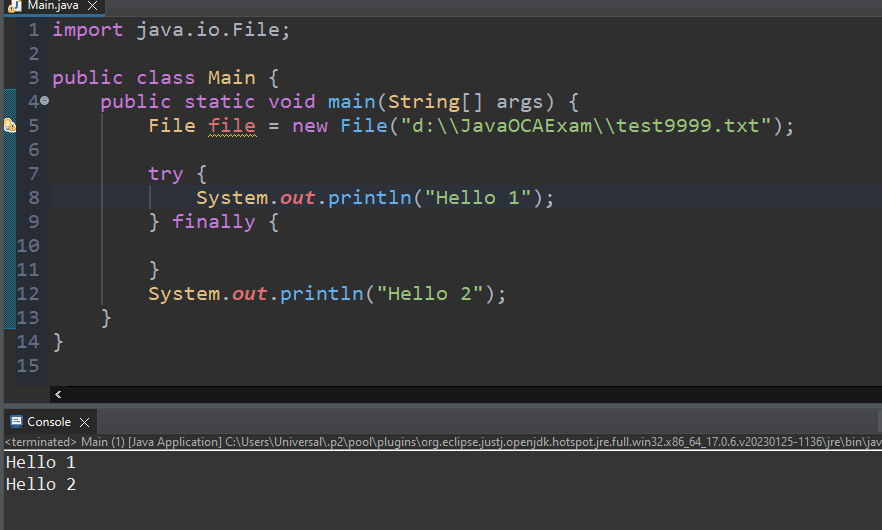
Yoki



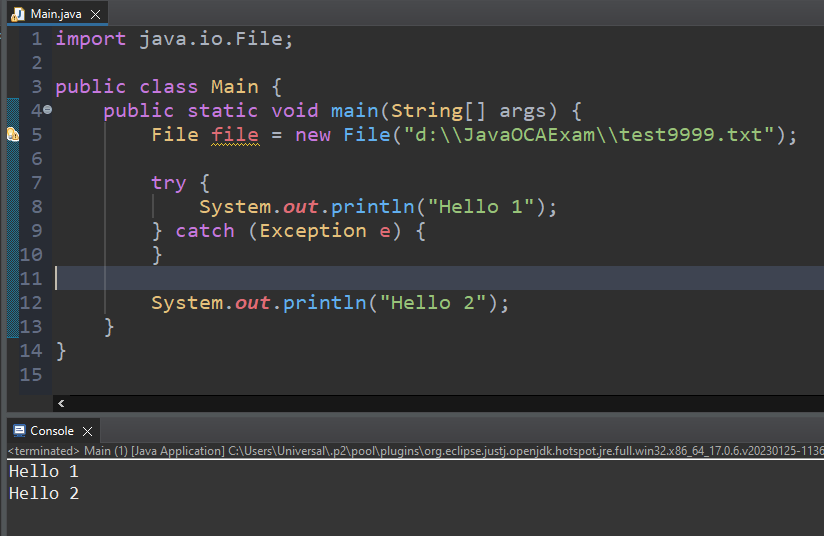
Agar biz faqatgina try blockni o’zini yozsak, yo catch ni yo finally ni tashlab ketib, u holda xatolik beradi:

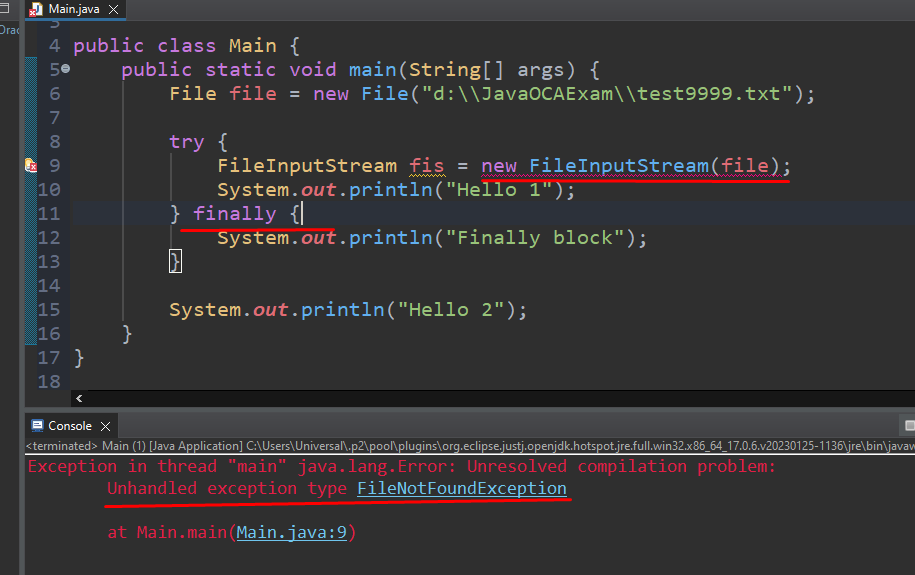


Agar shunchaki finally ni qo’shsak ham yuqoridagi xatolik yo’qoladi:



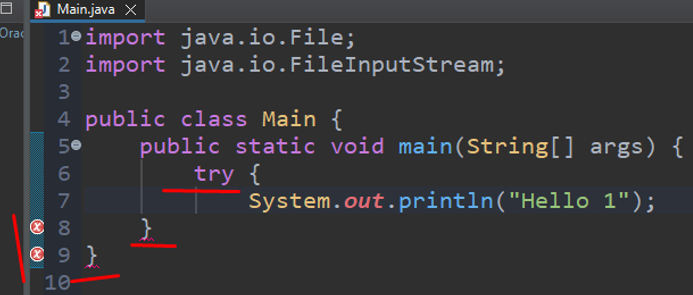
Yoki chunchaki catchni qo’shsak ham xatolik yo’qoladi:



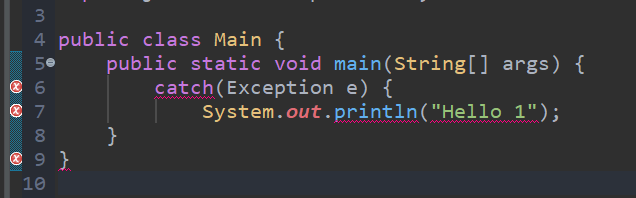
1. Agar kodimizda catch blockni tashlab ketsak, u holda bizni xatolik chiqqandagi exceptionlarimiz finally block tutib qolaolmaydi. Shuning uchun bizga xatolik beradi. Demak biz majburmiz har doim exception tashlab qolish ehtimoli yuqori bo’lgan joylarda, masalan file lar bilan ishlashda doim catch blockni yozishimiz kerak. Chunki finally block exception ni ushlamaydi va qayta ishlamaydi. Yana finally blockni yozish ixtiyoriydir:

Demak try blockni har doim ham finally block bilan ishlatib bo’lmas ekan. Exception tashlab qolishi aniq bo’lganda, masalan FileNotFoundException, biz majburmiz catch blokcni yozishga, aks holda xatolik beradi.

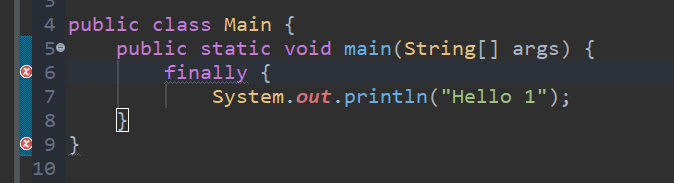
Bundan tashqari try-catch-finally blcoklarni shunchaki o’zini yakka-yakka holda alohida yozish mumkin emas, xatolik beradi. Masalan try blockni o’zini alohida ishlatsak:



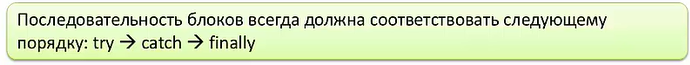
Yoki catchn o’zni alohida ishlatsak:



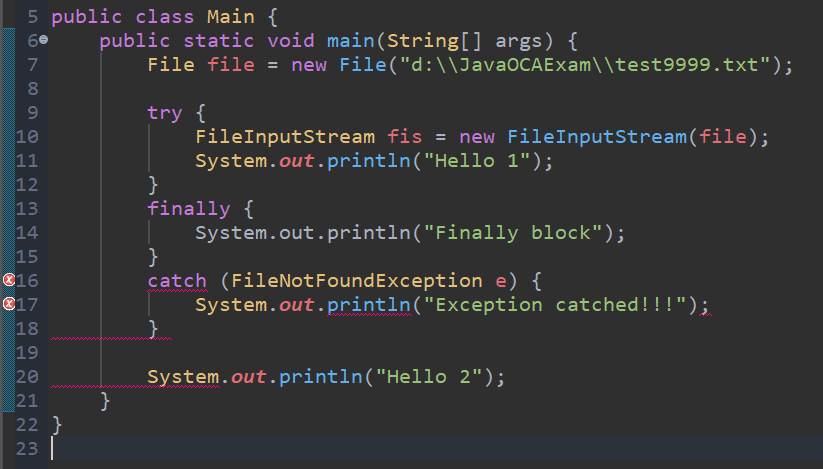
Yoki finally blockni o’zini alohida ishlatsak:



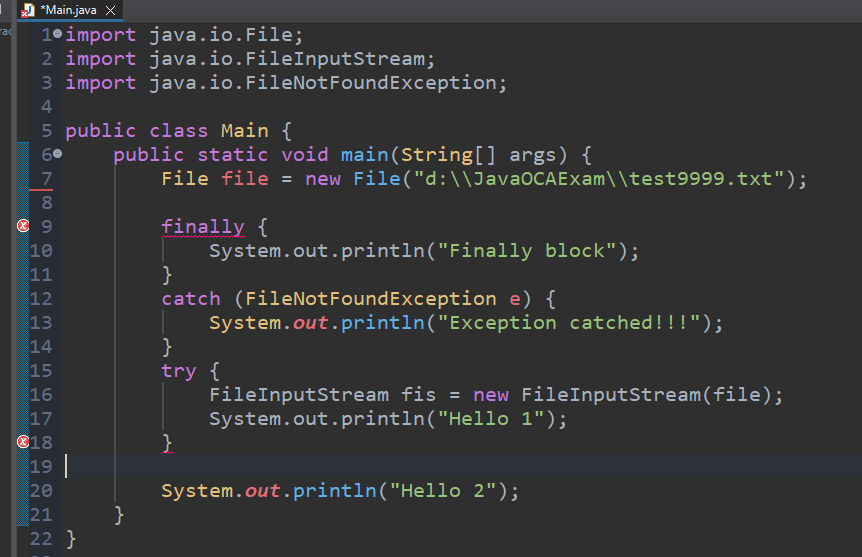
1. Try-catch-finally bilan ishlayotganda, ularni joylashgan tartibi juda muhim. Har doim avval try keyin catch keyin finally yozilishi shart. Xuddi pastdagi tartibda:



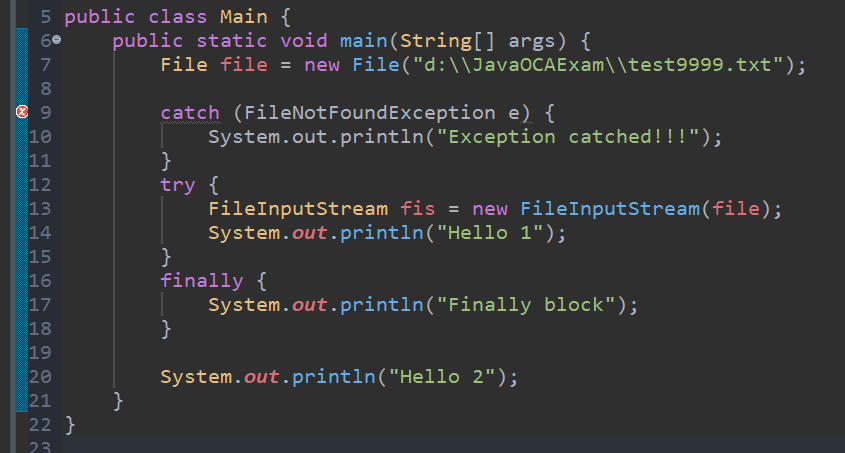
Masalan pastda finally blockni try va catch blockni orasiga yozganimizda, xatolik chiqdi:



Yoki bo’lmasam finally ni birinchi try ni esa oxiriga yozsak xatolik berdi:



Yoki catch ni birinchi yozsak ham xato berdi:



Demak ularni yozilishida tartib va ketma-ketlik juda muhim ekan.

