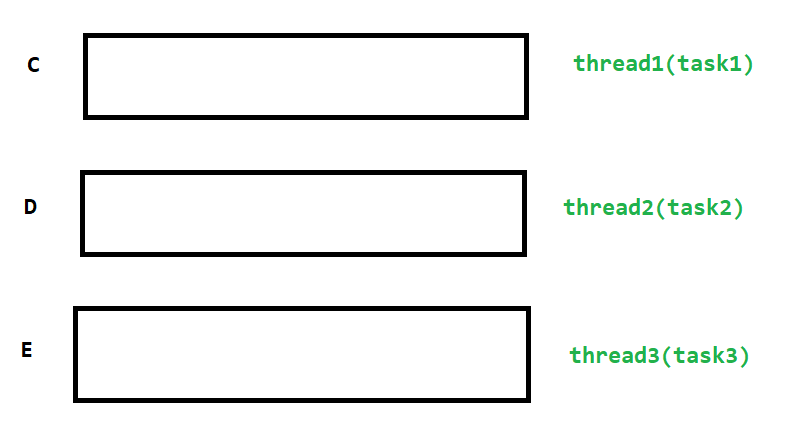
Multitasking(Ko’p vazifali) – bu bir vaqtning o’zida bir nechta ishni bajarishdir. Masalan biz bir vaqtning o’zida musiqa eshitib kitob o’qishimiz mumkin. Demak bir vaqtning o’zida 2 ta ish bajaryapmiz.

Multitasking ni o’zi 2 ga bo’linadi: 1)Process based multitasking, 2)Thread basic multitasking.

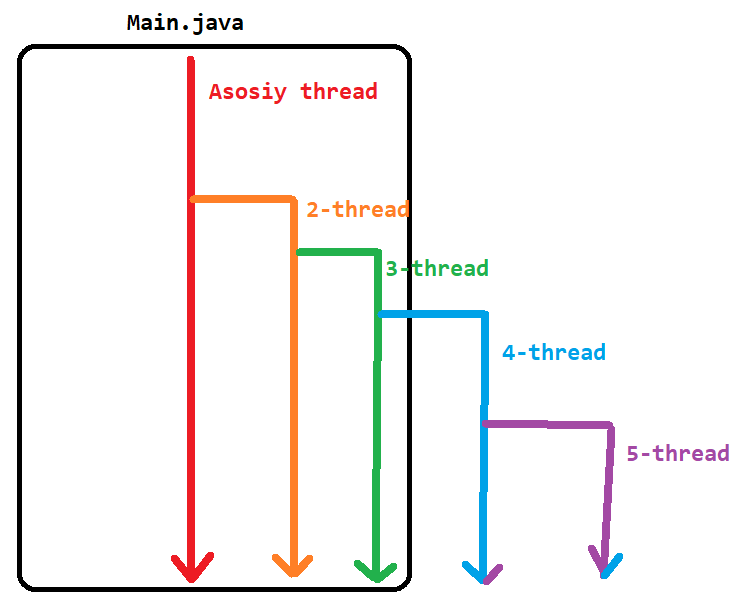
1. Process based multitasking bu – deylik bizni kompyuterimizda 3 ta dastur bir vaqtda ishlab turibdi. Masalan VSCode, Chrome Browser va Telegram. Bu 3 la dastur bir-biri bilan umuman aloqa qilmay mustaqil ravishda ishlayveradi. Bunda bittasi ishini to’xtatishi 2-ga umuman ta’sir qilmaydi.
2. Thread basic multitasking bu – deylik bizda 1 ta dastur bor bo’lib, uni ichida bir nechta tasklar bo’lsin. Bu tasklar bir-biridan mustaqil bo’lib, shu tasklarning bir vaqtda ishlashi Thread basic multitasking deyiladi. Masalan bitta dastur ishlab turibdi, bu dastur ishlashi uchun bir nechta tasklar bir vaqtda bir-biridan mustaqil ishlaydi. Lekin bu tasklarning hammasi bitta dasturni bo’lagi hisoblanadi. Multitaskingda esa bitta dastur emasda, balki bir nechta dasturni mustaqil ravishda ishlashidir.

Keling thread basic multitaskingni ko’rib chiqaylik. Deylik bizda bitta dastur bor bo’lsin. Bu dastur kompyuterning disklaridan biz izlayotgan faylni topib beradi. Biz unga qidiruv bersak, avval C diskdan qidirib chiqadi, agar topilmasa D diskdan qidirib chiqadi, agar yana topilmasa E diskdan qidirib chiqadi, ya’ni dastur bitta diskni qidirib chiqmaguncha 2-diskka o’tmaydi. Masalan, deylik bitta diskni qidirish uchun 10minut vaqt ketdi. 3 ta diskni qidirish uchun esa, jami 30minut vaqt ketdi. Lekin bizga tezroq ishlashi kerak bo’layapti. Bunday vaziyatda har bir disk uchun alohida bittadan thread(task) yaratamiz va ular bir vaqtda ishni boshlashadi. Har bir thread 10 minutda qidirsa jami 10 minut ketadi 3la diskni qidirib chiqish uchun, chunki ular bir vaqtda qidiruvni boshlashgan. Demak vaqt 3 barobarga qisqaradi.

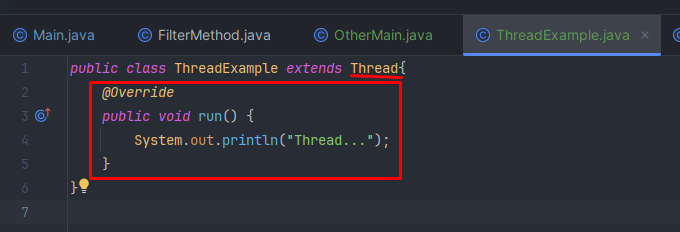


Lekin bitta yomon tomoni threadlar sonini ko’paytirsak, protsessordan ko’proq joy band qiladi. Chunki oldin bitta thread joy olgan bo’lsa, endi 3 ta thread joy olyapti.

Threadni ishlashini visual ko’ramiz. Deylik bizda Main.java class bor bo’lsin. Shu classimizning o’zini ham threadi bor bo’lib, qizil rangda berilgan “Asosiy thread” dir. Agar biz yana bitta “2-thread” ochmoqchi bo’lsak, u holda bu thread “Asosiy thread” ga parallel ishlashni boshlaydi. Istasak “2-thread” ga parallel ravishda “3-thread” ni ham ishga tushurishimiz mumkin va hokozo. Bu threadlar hammasi parallel ishlashni boshlaydi.



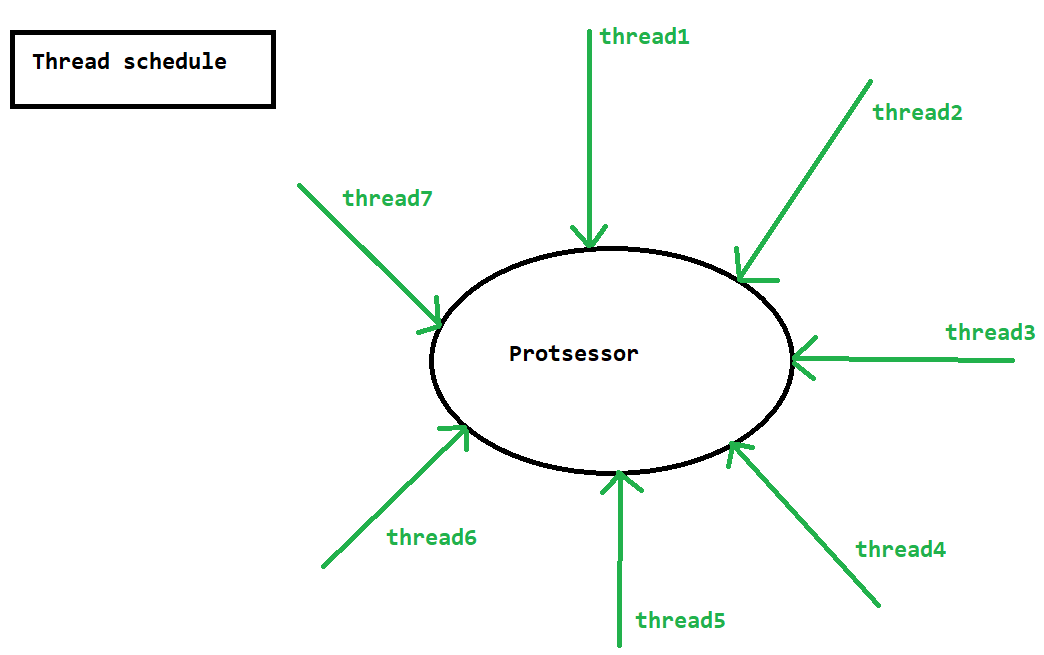
Thread yaratishni 2 ta usuli mavjud. 1-usulini ko’raylik avval. 1-usul bu Thread classdan meros olishdir. Bu classni asosiy methodlaridan biri bu run() methodi bo’lib, u asosiy ishni bajaradi.



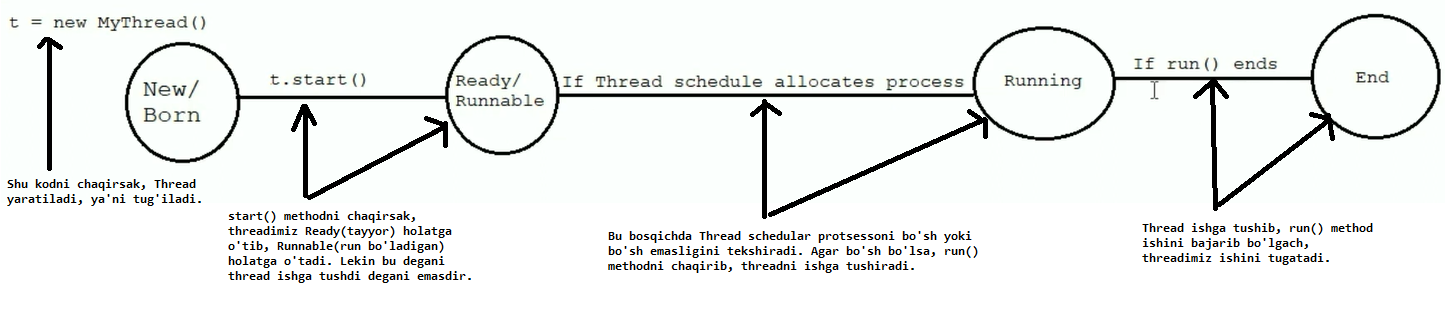
Endi Threadimizni yurgizish uchun esa start() methoddan foydalanamiz. start() methodi ham asosiy methodlardan biri bo’lib, threadimizni run qiladi. run() method threadni run qilmaydi, aksincha ish bajaradigan methoddir, balki start() method orqali threadimizni yurgizamiz va run() methodni chaqirib ishlatib beradi:



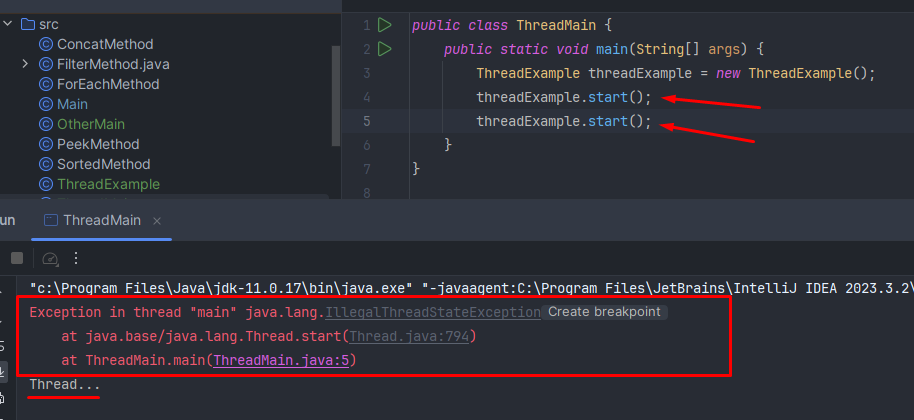
Threadda schedule degan tushuncha bo’lib, bunda threadlar protsessorga kelganda qaysi biri birinchi bajarishini bildiradi. Pastdagi rasmdan ko’rish mumkinki, protsessorga 7 ta thread bir vaqtda kelib qolyapti. Shnday paytda Thread schedule qaysi threadni birinchi bajarishni hal qiladi:



Thread Lifecycle – bunda thread 4 ta state dan tashkil topadi. 1) New/Born(tu’gulish), 2) Ready/Runnable 3) Running 4)End. Pastda ularga batafsilroq tushuntirish berilagan rasm bor:

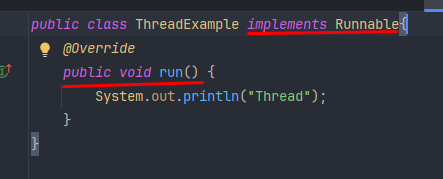


Yana bir narsani bilish muhimki, 2 marta bitta threadni start() methodini chaqirib bo’lmaydi. Sababi 1-marta start() method chaqirilganda New/Born statedan Ready/Runnable statega o’tadi. 2-marta chaqirganda Ready/Runnable statedan yana orqaga qaytib New/Born statega o’ta olmaydi. Bu degani 2 marta tug’ila olmaydi degani. Shuning uchun xatolik olamiz:

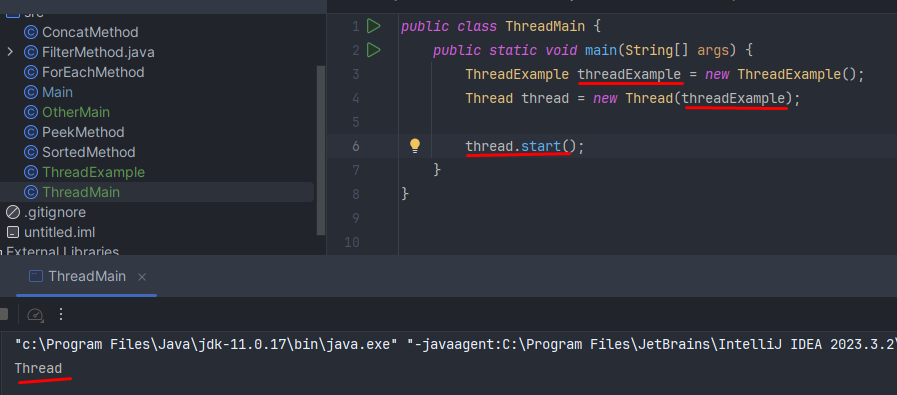


**Runnable interface**

Threadni yaratishni 2-usuli bu Runnable interfacedan implement olib yaratishdir. Bu interfaceni atigi bitta run() nomli abstract methodi mavjuddir va uni override qilishimiz shart:



Endi shu classdan object yaratib, theadni start() methodini chaqirib ishlatamiz:



Threadni nomini olish uchun esa **getName()** methodini ishlatamiz. Threadni nomini JVM o’zi avtomatik beradi. Pastda ko’rish mumkin. Istasak threadni nomini o’zimiz **setName()** methodi bilan ham berishimiz mumkin. **currentThread()** method bizga ayni paytda ishlab turgan threadni ko’rsatadi va pastdagi misolda current thread **“main”** ni ko’rsatyapti. Bu “main” o’zimizni main() methodidagi asosiy thread dir:

