Bucșa Ecaterina, gr. 231

**Laborator 5**

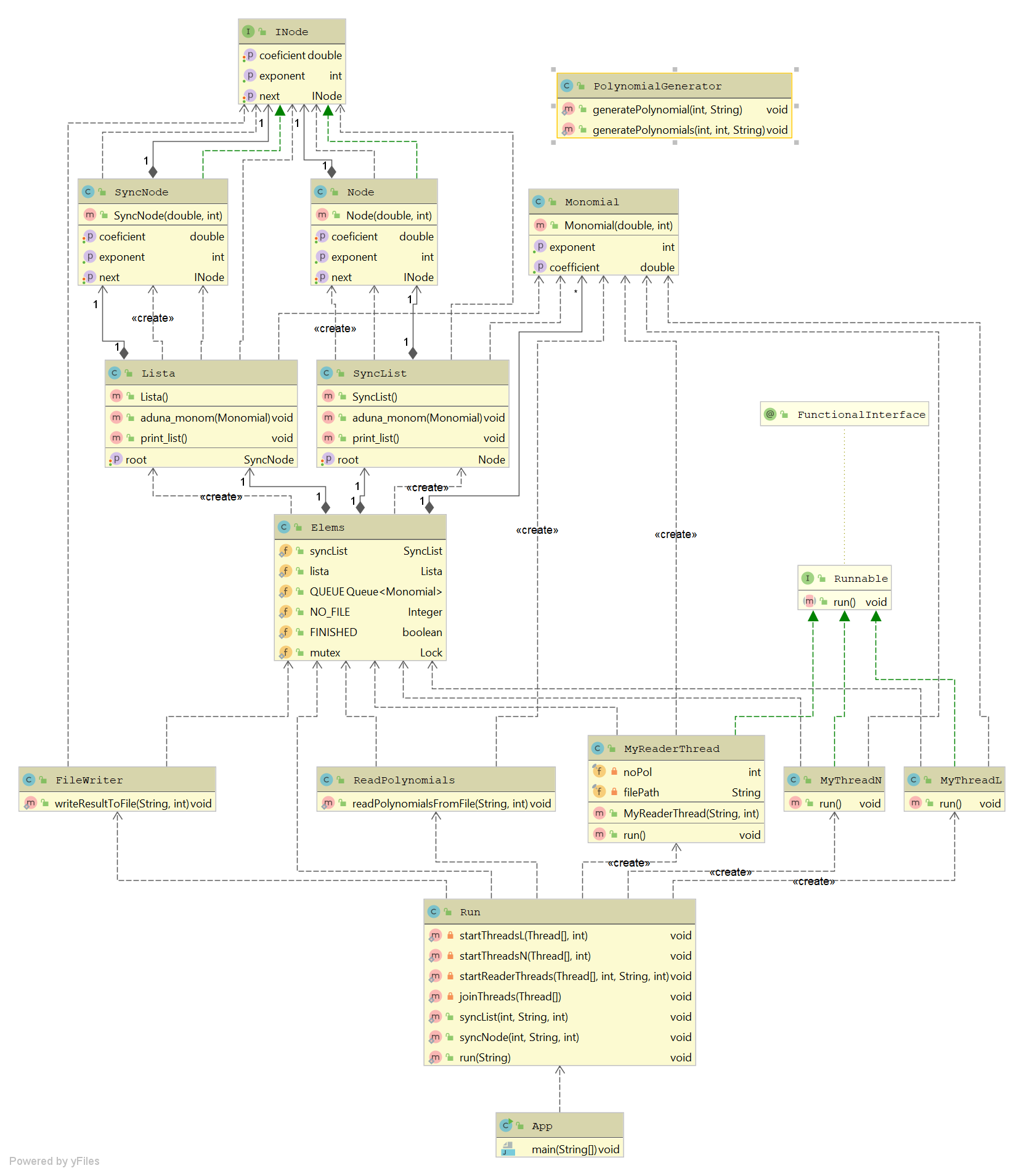
**-documentație-**

Laborator3 (Varianta 2)

Se dau n polinoame fiecare reprezentat într-un fișier prin monoamele sale. Un fișier conține informații de tip (coeficient, exponent) pentru fiecare monom al polinomului. Metoda: - Se creează lista înlănțuită - L corespunzătoare unui polinom nul. - Primul thread citește câte un monom și îl adaugă într-o structură de date de tip coadă. - Celelalte threaduri preiau câte un monom din coadă și îl adună la polinomul reprezentat în lista L. - Se continuă până când toate monoamele, din toate fișierele, sunt adunate la lista L. - Rezultatul obținut în lista L se scrie într-un fișier rezultat. Ce implicații ar avea considerarea mai multor threaduri „cititoare”?

**Ideea de rezolvare:** Thread-ul principal generează thread-urile, pornește thread-urile care vor citi din coadă, thread-uri care vor intra în starea de *wait,* iar apoi începe să citească monom cu monom polinoamele generate și să le adauge în coadă. La fiecare adaugare de monom în coadă, threadă-ul principal face *notify* iar unul dintre thread-urile care așteptau după monitorul cozii va ieși din starea de *wait* și va scoate din coadă ultimul element inserat, adunându-l la lista rezultat. După citirea tuturor polinoamelor, thread-ul principal notifică thread-urile care sunt în *wait* că a terminat de citit, setând o variabilă la valoarea *true.* Thread-urile consumator vor ieși din starea de *wait* și vor verifica din nou dacă coada este vidă și thread-ul principal a terminat citirea( variabila e setată cu valoarea *true*), iar în caz afirmativ își vor termina execuția.

Diagrama de clase Java:



Cazuri de testare:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Nr. polinoame | Grad polinoame | Limbaj | Nr. threaduri | Varianta a | Varianta b |
| 4 | 4 | Java | 2 | 0.011 | 0.002 |
| 10 | 4 | Java | 2 | 0.016 | 0.005 |
| 10 | 10 | Java | 2 | 0.018 | 0.008 |
| 10 | 10 | Java | 4 | 0.017 | 0.005 |
| 100 | 200 | Java | 2 | 0.162 | 0.064 |
| 100 | 400 | Java | 2 | 0.277 | 0.141 |
| 100 | 1000 | Java | 2 | 0.486 | 0.103 |
| 100 | 1000 | Java | 4 | 0.459 | 0.11 |
| 100 | 2500 | Java | 2 | 0.978 | 0.23 |
| 100 | 2500 | Java | 4 | 0.957 | 0.237 |
| 100 | 10000 | Java | 2 | 12.684 | 12.131 |
| 100 | 10000 | Java | 4 | 12.63 | 12.936 |
| 150 | 1000 | Java | 2 | 0.51 | 0.309 |
| 150 | 1000 | Java | 4 | 0.453 | 0.238 |
| 150 | 5000 | Java | 2 | 4.825 | 4.064 |
| 150 | 5000 | Java | 4 | 4.879 | 4.578 |
| 500 | 300 | Java | 2 | 0.527 | 0.232 |
| 500 | 300 | Java | 4 | 0.49 | 0.173 |
| 500 | 700 | Java | 2 | 0.556 | 0.372 |
| 500 | 700 | Java | 4 | 0.742 | 0.411 |
| 500 | 2000 | Java | 2 | 2.623 | 2.219 |
| 500 | 2000 | Java | 4 | 2.766 | 2.692 |
| 500 | 5000 | Java | 2 | 13.923 | 15.065 |
| 500 | 5000 | Java | 4 | 14.969 | 17.401 |
| 1000 | 1000 | Java | 2 | 1.683 | 1.264 |
| 1000 | 1000 | Java | 4 | 1.733 | 1.485 |
| 1000 | 2000 | Java | 2 | 7.058 | 5.665 |
| 1000 | 2000 | Java | 4 | 5.894 | 6.304 |
| 1000 | 5000 | Java | 2 | 34.641 | 34.88 |
| 1000 | 5000 | Java | 4 | 33.282 | 40.313 |
| 1000 | 10000 | Java | 2 | 117.526 | 126.712 |
| 1000 | 10000 | Java | 4 | 122.716 | 149.436 |