Vom implementa un Automat Finit Nedeterminist (AFN) prin intermediul clasei Automaton, care conține următoarele elemente:

* mulțimea stărilor și cardinalul ei; stările vor fi reprezentate prin numere întregi, pentru simplitatea implementării
* alfabetul (mulțimea de simboluri) și cardinalul său
* tranzițiile și numărul lor; tranzițiile vor fi reprezentate sub forma unui graf ponderat (ponderile vor fi simbolurile)
* starea inițială
* mulțimea stărilor finale și cardinalul ei

Automaton::Automaon()

* constructor fără parametri, care se apelează în momentul declarării unui obiect și care inițializează acel obiect
* citește datele care definesc automatul din fișierul *dataNFA.txt*

Automaton::~Automaton()

* destructorul clasei Automaton
* eliberează memoria alocată pentru automat

void Automaton::isWordAccepted(int currentState, string word, int pos, bool &done)

* verifică dacă cuvântul word este acceptat de către automat
* implementează un DFS ușor modificat, care are condiții mai restrictive: dacă, până la pasul curent, cuvântul a fost acceptat, și dacă mai avem litere în cuvânt (condiție de oprire); deoarece ne interesează doar dacă există cel puțin o succesiune de stări pentru care cuvântul este acceptat, ne vom opri după ce găsim prima astfel de succesiune
* currentState – număr întreg ce reprezintă starea curentă
* word – string ce reprezintă cuvântul verificat
* pos – număr întreg ce reprezintă poziția literei din cuvânt care se verifică la pasul curent
* done – variabilă de tip bool ce ține evidența faptului că am găsit cel puțin o succesiune de stări pentru care cuvântul dat a fost acceptat

int Automaton::getInitialState()

* getter pentru starea inițială a AFN-ului