**MINISTERUL EDUCAȚIEI ȘI CERCETĂRII AL REPUBLICII MOLDOVA**

**Universitatea Tehnică a Moldovei**

**Facultatea Calculatoare, Informatică și Microelectronică**

**Departamentul Informatică și Ingineria Sistemelor**

**RAPORT**

Lucrare practică nr.1

la cursul „***Sisteme de operare***”

**Tema: Familiarizarea cu Git**

A efectuat: **St. gr. IA-221, Silistraru Adrian**

A verificat: **Asis.univ. Seniusin Anton**

**Chișinău 2024**

Cuprins

[REALIZAREA PRACTICĂ A SARCINII DE LUCRU 2](#_Toc178847390)

[Partea teoretică 2](#_Toc178847391)

[Partea practică 4](#_Toc178847392)

[CONCLUZII 7](#_Toc178847393)

[BIBLIOGRAFIE 8](#_Toc178847394)

# REALIZAREA PRACTICĂ A SARCINII DE LUCRU

**Scopul: Pregătirea inițială a mediului Git. Crearea unui depozit Git și local repertoriu.**

**Sarcina:**

* Instalați mediul Git pe dispozitivul dvs., pregătiți un folder pentru aplicație/proiect și inițializați depozitul local din acest folder (comanda git init).
* Implementați o aplicație simplă (din variantele propuse mai jos). Într-o echipă, atribuiți cine ce părțe a aplicației va implementa care va fi această parte și ce funcție (funcții) va îndeplini.
* Efectuați commit pentru modificări în codul aplicației/proiectului (comanda git commit). Creați un cont pe GitHub (dacă nu aveți unul) și trimiteți numele de utilizator la profesor sau linkul de profil. După ce ați primit mai departe instrucțiuni de la profesor, introduceți modificările în ramura dvs. separată depozite de pe GitHub.

**Varianta 2:**

O aplicație care extrage anumite informații din text introdus. Fiecare membru al echipei implementează o clasă care recupereaz informații solicitate în mod specific. În algoritmul principal (clasa) implementează o relație între cel puțin două clase. (nota: încercați să folosiți expresii regulate).

## Partea teoretică

Git este un sistem de control al versiunilor distribuit gratuit, open-source cod sursă, conceput pentru a lucra rapid și eficient cu orice proiecte - de la mici la foarte mari. Git este ușor de învățat și ocupă puțin spațiu și are performanțe fulgerătoare. Este superior unor astfel de SCMuri (Supply Managementul lanțului – trad. Supply Chain Management - instrumente precum

Subversion, CVS, Perforce și ClearCase, cu caracteristici precum local low-cost ramificații, zone de depozitare convenabile și fluxuri de lucru multiple.

**Ramificare și contopire**

Caracteristica Git care îl face cu adevărat să iasă în evidență de aproape toți ceilalți alte SCM-uri, este modelul de ramificare. Git permite și încurajează creația mai multe ramuri locale, care pot fi complet independente unele de altele prieten. Crearea, îmbinarea și ștergerea acestor ramuri durează doar câteva secunde.

Aceasta înseamnă că puteți face lucruri precum:

* Comutare de context fără contact. Creați o ramură pe care să o încercați orice idee, fă câteva comite, întoarce-te la locul de unde ramificație a fost făcută, aplicați remedierea, întoarceți-vă unde au fost efectuate experimente și combinate.
* Linii de cod pentru jocuri de rol. Ai o ramură care conține întotdeauna doar ce trimis la producție, altul, în care munca este fuzionată pentru testare și câteva ramuri mici pentru munca de zi cu zi.
* Flux de lucru bazat pe funcții. Creați filiale noi pentru fiecare funcție nouă la care lucrați, astfel încât să puteți face fără probleme comuta între ele și apoi șterge fiecare ramură când funcția vor fi îmbinate cu linia principală.
* Experimente unice. Creați un fir pentru experimentare, înțelegeți asta nu funcționează și ștergel, părăsește-ți jobul și nimeni nu o va mai vedea ea (chiar dacă în acest timp ai promovat alte ramuri).

**Distribuire**

Una dintre cele mai utile caracteristici ale oricărui SCM distribuit este inclusiv Git, este distribuția sa. Aceasta înseamnă că în loc de efectuați o „verificare” a vârfului codului sursă curent, efectuați o „clonă” întregul depozit.

**Copii de rezervă multiple**

Aceasta înseamnă că, chiar dacă utilizați un flux de lucru centralizat, fiecare utilizator are în esență o copie de rezervă completă a serverului principal.

Fiecare dintre aceste copii poate fi ridicată pentru a înlocui serverul principal în caz de defecțiune sau deteriorare. În esență, nu există un singur punct de eșec în Git decât dacă există o singură copie a depozitului.

**Flux de lucru**

Datorită naturii distribuite a lui Git și sistemului excelent de ramificare este posibil să se implementeze cu relativă uşurinţă un număr practic infinit procesele de lucru.

**Flux de lucru în stil subversiune (Subversion)**

Fluxul de lucru centralizat este foarte comun, mai ales în rândul utilizatorilor, care au trecerea dintr-un sistem centralizat. Git nu te va lăsa să realizați push dacă cineva a făcut un push de la ultima extracție, deci centralizat funcționează modelul în care toți dezvoltatorii realizează push pe același server pur si simplu minunat.

**Instalarea Git**

Descărcați programul de instalare, rulați programul de instalare, selectați opțiunile de care aveți nevoie (sau lăsați totul așa cum este, dacă nu prea mult înțelegeți), așteptați finalizarea procesului de instalare și ați terminat.

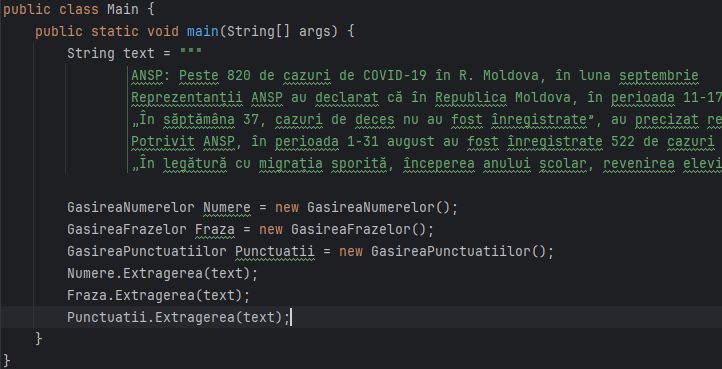
**Utilizare Git**

În folderul de lucru în care se află fișierele sursă ale aplicației/proiectului, prin terminal sau linia de comandă, utilizați comanda git init pentru a inițializa depozitul local, după ce Git va urmări modificările la fișierele sursă și le va remedia modificările - utilizați comanda git commit.

În viitor, dacă trebuie să sincronizați modificările în comun pentru toate depozitele - utilizați comanda git push.

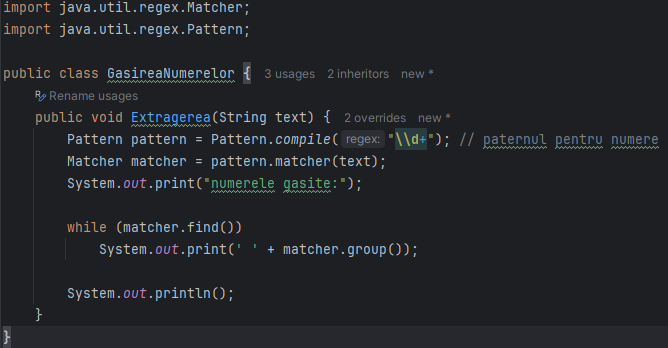
## Partea practică

Pentru sarcina oferită am elaborat 3 clase pentru extragerea informației dintr-un text, iar în clasa Main am utilizat clasele pentru recupererea informației solicitate în mod specific.



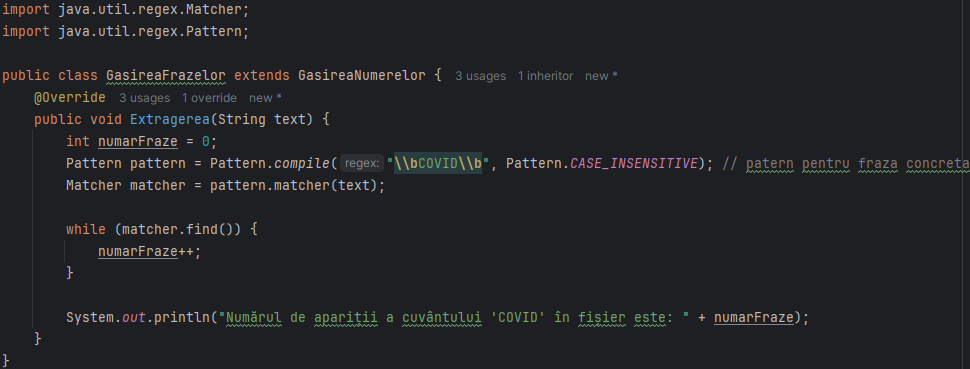
**Figura 1.** Codul clasei Main

În figura 1 este reprezentat textul, utilizarea claselor și apelarea funcției Extragerea fiecărei clase.



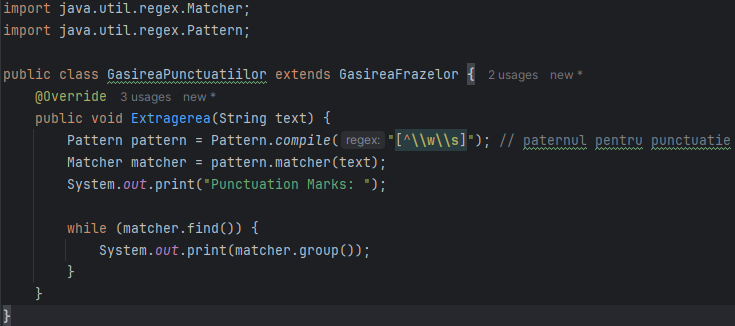
**Figura 2.** Codul clasei GasireaNumerelor

În figura 2 este reprezentat algoritmul de extragerea numerelor din textul propus. Cand găsește un număr, atunci îl printează instant.



**Figura 3.** Codul clasei GasireaFrazelor

În figura 3 este reprezentat algoritmul de găsirea numărului de apariție a frazei indicate în patern. Variabila numarFraze se incrementează de fiecare dată cand paternul coincide cu un subșir din șirul indicat.



**Figura 4.** Codul clasei GasireaPunctuatiilor

În figura 4 este reprezentat algoritmul de extragerea caracterelor care nu sunt literă, cifră, underscore sau spațiu. Printează simbolul cănd coincide cu paternul indicat.

# CONCLUZII

În cadrul acestei lucrări de laborator am facut cunoștință cu platforma Git. Astfel sarcina a oferit o experiență valoroasă în gestionarea unui proiect de dezvoltare software într-un mediu colaborativ. Folosirea Git și GitHub facilitează colaborarea și gestionarea versiunilor.

Implementarea claselor pentru extragerea informațiilor din text și utilizarea expresiilor regulate arată o abordare tehnică bine gândită și eficientă. Această sarcină abordează să se utilizeze instrumente de dezvoltare esențiale și să dezvolte abilități practice în programare.

Am însușit modalitatea de lucru cu Git, astfel am obținut experiență importantă de lucru cu proiecte și modul de interacțiune prin platforma Git.

# BIBLIOGRAFIE

* [git-scm](https://git-scm.com/docs) [sursă web] – documentație oficială în utilizarea instrumentului Git.