UNIVERSITATEA DIN BUCUREȘTI
FACULTATEA DE MATEMATICĂ ȘI INFORMATICĂ

RAPORT

asupra proiectului:

„XML”

Elaboratori: stud. Dorneanu Diana-Delia,
Miu Georgian-Fabian,
Preda Maria-Alexandra

Specializare: Informatică

Grupa: 141

Data: 09.01.2025

XML

I. Prezentarea cerinței proiectului

Scrieți un program (script) shell care să parseze fișiere XML. Acesta trebuie să poată citi și scrie date din, respectiv, în format XML.

- *Ce este limbajul XML?*

XML (*Extensible Markup Language*) este un limbaj de marcare similar cu HTML, care nu conține tag-uri predefinite, utilizatorul putând să își definească propriile tag-uri și structura documentului.

Este conceput pentru stocarea și pentru transmiterea datelor, fiind independent de software și hardware și utilizat pentru a crea limbaje de marcare precum XHTML, SVG, MathML.

- *Structura unui document XML*

Un document XML are o structură arborescentă și este format din:

- ✓ elemente: tag-uri <nume_tag> case sensitive
- ✓ date caracter: conținutul elementelor

Toate documentele XML trebuie să conțină un unic element rădăcină (eg. <root> … </root>).

- *Ce este un parser XML?*

Un parser XML este un software care poate procesa sau citi documente XML pentru a extrage date din ele.

Toate browserele moderne au un parser XML încorporat care poate converti textul într-un obiect XML DOM.

XML DOM (Document Object Model) definește proprietățile și metodele de accesare și de editare XML.

Cu toate acestea, înainte ca un document XML să poată fi accesat, acesta trebuie să fie încărcat într-un obiect XML DOM.

II. Descrierea programului

- Implementarea funcției de citire read_xml()

```
read_xml() {
  local file="$1"
  local element="$2"
  if grep -q "<$element>.*</$element>" "$file"; then
```

```
        echo "Valoarea elementului <$element> este:"
      grep -oP "(?<=<$element>).*?(?=</$element>)" "$file"
   else
      echo "Eroare: Elementul <$element> nu există în fișierul $file."
   fi
}
```

Mai sus am definit funcția de citire a elementelor de tip XML.

Această funcție primește numele fișierului XML în variabila *file* și elementul căutat în variabila *element*.

Comanda „grep" din Linux este un instrument pentru căutarea și pentru manipularea tiparelor de text din fișiere. Comanda „grep'' caută o expresie regulată și afișează liniile găsite în fișier ce ilustrează exact tiparul căutat.

If-ul verifică dacă tiparul căutat există în fișier sau nu. În cazul în care există, se va afișa valoarea dintre cele două tag-uri <element> și </element> , iar valoarea va fi găsită de următoarea linie de cod. În cazul în care nu există, vom afișa un mesaj negativ.

Opțiunea „-q'' din comanda „grep'' nu afișează în STDOUT daca tag-ul există în fișier.

Opțiunea „-oP'' din comanda „grep'' îmbină două funcționalități:

• „-o'' are rolul de a afișa doar partea dintr-o linie de cod care ilustrează tiparul căutat, restul fiind ignorat;

• „-P'' are rolul de a accepta expresii regulate mai complexe decât cele implicite ale comenzii „grep''.

Expresia regulată are rolul de a găsi tiparul „<element>valoare</element>'':

• (?<=<element>) - > asigură că există tag-ul <element> înaintea primei potriviri a textului cu tiparul căutat, dar <element> nu va face parte din rezultatul final;

• (?=</element>) -> asigură că există tag-ul </element> imediat după prima potrivire a textului cu tiparul căutat, dar </element> nu va face parte din rezultatul final;

• .* ? -> secțiunea care găsește textul dintre tag-uri:
  - „.'' -> reprezintă orice caracter, mai puțin cel de linie nouă;
  - „*'' -> identifică un număr nelimitat de caractere;
  - „?'' -> potrivește textul dintre primul <element> întâlnit și primul </element> întâlnit.

Din interiorul comenzii „grep'', „$file'' reprezintă fișierul în care se face căutarea.

- Implementarea funcției de scriere write_xml()

```
write_xml() {
   local file="$1"
   local element="$2"
   local value="$3"


   local root_tag
   root_tag=$(head -n 1 "$file" | sed 's/[<>]//g')
```

```
  if grep -q "</$root_tag>" "$file"; then
    if grep -q "<$element>.*</$element>" "$file"; then
      sed -i "s|<$element>.*</$element>|<$element>$value</$element>|" "$file"
    else
      sed -i "/<\/$root_tag>/i <$element>$value</$element>" "$file"
    fi
  else
    echo "Eroare: Fisierul XML nu contine un element radacina valid"
  fi
}
```

Funcția are 3 parametri:

1. *file*: fișierul XML în care se fac modificările;
2. *element*: numele elementului XML care urmează să fie adăugat;
3. *value*: valoarea care va fi inserată între tag-urile elementului ($2).

În primul rând, se extrage valoarea rădăcinii în variabila *root_tag*, folosind **head -n 1 "$file"**, ce preia prima linie din fișierul XML (unde se află de obicei tag-ul de rădăcină). **sed -n 's/<\([^>]\)>./\1/p'** folosește o expresie regulată pentru a extrage tot ce se află între simbolurile < și >, fără aceste simboluri, \([^>]*\) caută orice secvență de caractere care nu conține >, \1 este o referință la primul grup de caractere capturate (adică tag-ul rădăcină), p la final face ca rezultatul să fie afișat. Funcția sed (Stream Editor) procesează și manipulează textul din fișier. Opțiunea -n solicită să nu afișeze fiecare linie din fișier, ci doar cele care sunt explicit solicitate prin comenzi ulterioare.

În al doilea rând, se verifică corectitudinea fișierului XML, mai exact prezența elementul </$root_tag>, prin utilizarea funcției grep cu opțiunea -q (quiet), ce nu afișează în STDOUT dacă tag-ul există în fișier. În caz negativ, scriptul afișează un mesaj de eroare.

În caz afirmativ, se caută tag-ul citit in fișier ($file). Dacă acesta există deja, se actualizează cu noua sa valoare prin comanda de substituire „s|...|...|", unde primul câmp "<$element>.*</$element>" (.* reprezintă expresia regulată pentru orice text dintre tag-urile elementului) este înlocuit cu al doilea câmp "<$element>$value</$element>", mai exact cu noua valoare.

Altfel, elementul trebuie adăugat și se folosește funcția *sed* pentru a insera elementul nou înainte de tag-ul </$root_tag>, modificând în mod direct fișierul (opțiunea -i) prin introducerea textului "<$element>$value</$element>". "/<\/$root_tag>/" este un șablon de căutare, iar „i" reprezintă o comandă de inserare utilizată pentru a adăuga text înaintea liniei care corespunde expresiei.

- Implementarea meniului pentru utilizator și redirecționarea către opțiunea aleasă

```
echo "XML - Parser Script"
echo "1: Citire element"
echo "2: Scriere element"
```

```
read -p "Alegeti o optiune (1/2): " option

case $option in
  1)
    read -p "Introduceti numele fisierului XML: " file
    read -p "Introduceti numele elementului cautat: " element
    read_xml "$file" "$element"
    ;;
  2)
    read -p "Introduceti numele fisierului XML: " file
    read -p "Introduceti numele elementului ce trebuie scris: " element
    read -p "Introduceti valoarea elementului: " value
    write_xml "$file" "$element" "$value"
    echo "Elementul <$element> a fost adaugat/actualizat"
    ;;
  *)
    echo "Optiune nevalida"
    ;;
esac
```

Acest fragment de cod implementează un meniu interactiv pentru utilizator, care permite selectarea unei opțiuni între citirea și scrierea unui element XML.

- *echo*: Afișează titlul scriptului și opțiunile disponibile.

- *read -p*: Solicită utilizatorului să introducă o opțiune (1 sau 2). Textul dintre ghilimele este afișat ca prompt.

Avantajul utilizării flag-ului -p este că rezultatul inputului introdus de utilizator este direct salvat în variabila specificată, fără a fi nevoie să fie utilizată o comandă *echo* înainte de *read*. Totul se face într-o singură linie.

- *option*: Valoarea introdusă de utilizator este salvată în variabila *option*.

Blocul *case* evaluează valoarea introdusă de utilizator (*option*) și execută o secțiune corespunzătoare de cod, în funcție de alegerea făcută.

În cazul în care se alege opțiunea citirii unui element, mai întâi se afișează promptul pentru introducerea numelui fișierului XML, iar răspunsul este salvat în variabila *file*.

Apoi, se afișează promptul pentru introducerea numelui elementului XML de căutat, iar răspunsul este salvat în variabila *element*.

Ulterior, se apelează funcția read_xml(), definită anterior în script, cu argumentele *file* și *element*.

În cazul în care se alege opțiunea scrierii unui element, se procedează ca în cazul anterior pentru variabilele *file* și *element*, apoi se afișează un prompt pentru introducerea valorii elementului ce va fi scris, iar răspunsul este salvat în variabila *value*.

Se apelează funcția write_xml(), definită anterior în script, cu argumentele *file*, *element* și *value*.

Ulterior, se afișează un mesaj care confirmă utilizatorului că operația de scriere a fost realizată cu succes.

În cazul în care opțiunea aleasă este diferită de 1 sau de 2, se afișează un mesaj care notifică utilizatorul că a ales o opțiune nevalidă, iar programul se încheie.

III. Dificultăți întâmpinate

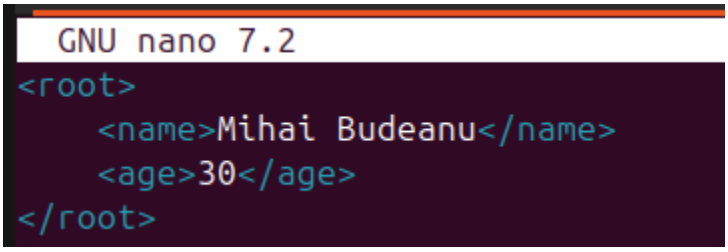
1. **Procesarea Regex în Shell**: Regex-ul folosit pentru extragerea valorii unui element XML poate deveni complicat pentru fișiere complexe.
- **Soluție**: Am utilizat grep -P pentru a suporta expresii Perl, care sunt mai flexibile.

2. **Unicitatea rădăcinii unui fișier XML:** Fiecare fișier XML are o rădăcină diferită (nu <root> … </root> neapărat), ce trebuie identificată pentru a ne raporta la ea în funcția write.
- **Soluție:** Declararea unei variabile locale care extrage rădăcina dintre <…> de pe prima linie.

3. **Scrierea unui element deja existent:** Un element care trebuie scris poate exista deja în XML. Astfel, acesta trebuie doar actualizat.
- **Soluție:** Verificarea existenței și utilizarea unor comenzi corespunzătoare pentru fiecare caz (actualizare / doar scriere).

IV. Rezultate experimentale

test1.xml



```
fabianmiu@Ubuntu:~/Desktop$ ./xml_parser.sh
XML - Parser Script
1: Citire element
2: Scriere element
Alegeti o optiune (1/2): 1
Introduceti numele fisierului XML: test1.xml
Introduceti numele elementului cautat: name
Valoarea elementului <name> este:
Mihai Budeanu
fabianmiu@Ubuntu:~/Desktop$ ./xml_parser.sh
XML - Parser Script
1: Citire element
2: Scriere element
Alegeti o optiune (1/2): 2
Introduceti numele fisierului XML: test1.xml
Introduceti numele elementului ce trebuie scris: email
Introduceti valoarea elementului: mihai.budeanu@gmail.com
Elementul <email> a fost adaugat/actualizat
fabianmiu@Ubuntu:~/Desktop$ cat < test1.xml
<root>
    <name>Mihai Budeanu</name>
    <age>30</age>
<email>mihai.budeanu@gmail.com</email>
</root>
```

test2.xml

```
  GNU nano 7.2
<root>
    <titlu>Programare in XML</titlu>
    <autor>Maria Enache</autor>
    <an>2025</an>
    <editura>Paralela 45</editura>
<nr.pag.>256</nr.pag.>
</root>
```

```
fabianmiu@Ubuntu:~/Desktop$ ./xml_parser.sh
XML - Parser Script
1: Citire element
2: Scriere element
Alegeti o optiune (1/2): 1
Introduceti numele fisierului XML: test2.xml
Introduceti numele elementului cautat: titlu
Valoarea elementului <titlu> este:
Programare in XML
fabianmiu@Ubuntu:~/Desktop$ ./xml_parser.sh
XML - Parser Script
1: Citire element
2: Scriere element
Alegeti o optiune (1/2): 1
Introduceti numele fisierului XML: test2.xml
Introduceti numele elementului cautat: oras
Eroare: Elementul <oras> nu există în fișierul test2.xml.
fabianmiu@Ubuntu:~/Desktop$ ./xml_parser.sh
XML - Parser Script
1: Citire element
2: Scriere element
Alegeti o optiune (1/2): 2
Introduceti numele fisierului XML: test2.xml
Introduceti numele elementului ce trebuie scris: nr.pag.
Introduceti valoarea elementului: 256
Elementul <nr.pag.> a fost adaugat/actualizat
fabianmiu@Ubuntu:~/Desktop$ ./xml_parser.sh
XML - Parser Script
1: Citire element
2: Scriere element
Alegeti o optiune (1/2): 2
Introduceti numele fisierului XML: test2.xml
Introduceti numele elementului ce trebuie scris: an
Introduceti valoarea elementului: 2025
Elementul <an> a fost adaugat/actualizat
```

```
fabianmiu@Ubuntu:~/Desktop$ cat < test2.xml
<?xml version="1.0" encoding="UTF-8"?>
<root>
    <titlu>Programare in XML</titlu>
    <autor>Maria Enache</autor>
    <an>2025</an>
    <editura>Paralela 45</editura>
<nr.pag.>256</nr.pag.>
</root>
```

test3.xml

```
  GNU nano 7.2
<discografie>
        <trupa>3sudest</trupa>
        <album1>Visul meu</album1>
        <album2>Sentimental</album2>
        <album3>Top</album3>
        <album4>Best Of</album4>
</discografie>
```

```
fabianmiu@Ubuntu:~/Desktop$ ./xml_parser.sh
XML - Parser Script
1: Citire element
2: Scriere element
Alegeti o optiune (1/2): 1
Introduceti numele fisierului XML: test3.xml
Introduceti numele elementului cautat: album1
Valoarea elementului <album1> este:
Visul meu
fabianmiu@Ubuntu:~/Desktop$ ./xml_parser.sh
XML - Parser Script
1: Citire element
2: Scriere element
Alegeti o optiune (1/2): 1
Introduceti numele fisierului XML: test3.xml
Introduceti numele elementului cautat: album10
Eroare: Elementul <album10> nu exista in fisierul test3.xml.
fabianmiu@Ubuntu:~/Desktop$ ./xml_parser.sh
XML - Parser Script
1: Citire element
2: Scriere element
Alegeti o optiune (1/2): 2
Introduceti numele fisierului XML: test3.xml
Introduceti numele elementului ce trebuie scris: album5
Introduceti valoarea elementului: Ziua de maine
Elementul <album5> a fost adaugat/actualizat
fabianmiu@Ubuntu:~/Desktop$ ./xml_parser.sh
XML - Parser Script
1: Citire element
2: Scriere element
Alegeti o optiune (1/2): 2
Introduceti numele fisierului XML: test3.xml
Introduceti numele elementului ce trebuie scris: album4
Introduceti valoarea elementului: Ziua de ieri
Elementul <album4> a fost adaugat/actualizat
```

```
fabianmiu@Ubuntu:~/Desktop$ cat < test3.xml
<discografie>
        <trupa>3sudest</trupa>
        <album1>Visul meu</album1>
        <album2>Sentimental</album2>
        <album3>Top</album3>
        <album4>Ziua de ieri</album4>
<album5>Ziua de maine</album5>
</discografie>
```

Bibliografie:

https://cs.unibuc.ro/~cechirita/tw/c10/#/16
https://developer.mozilla.org/en-US/docs/Web/XML
https://aws.amazon.com/what-is/xml/
https://www.geeksforgeeks.org/sed-command-in-linux-unix-with-examples/
https://regex101.com/