מסדי נתונים – 89-281

תרגול 3,4 – XPath 1.0

אור כדראוי בשיתוף עם עמיעד רוזנברג

(מבוסס על מצגת של הטכניון)

- If at first you don't succeed… call it version 1.0 .
- Enter any 11-digit prime number to continue…

XPath

- שפת שאילתות המאפשרת שליפת נתונים בצורה קלה מתוך קבצי XML.
- מסתמכת על מבנה העץ של מסמך XML ופועלת לפיו.

- אינה שפת תכנות בפני עצמה אלא שפת שאילתות.
מאפשרת גישה לצמתי המסמך השונים (אלמנטים, תכונות ומידע) בצורה קלה ונוחה.
יכולת זו בשילוב עם שפת תכנות כלשהי. מאפשרת להציג כל מידע שנרצה בצורה נוחה פשוטה ומהירה.

XPath

- XPath קיבלה את שמה בעקבות הדימיון שלה למסלול במערכת הקבצים.
- ניקח כדוגמא מסמך XML שקובץ ה-DTD שלו נראה כך:

```
<!ELEMENT inventory (book)+>
<!ELEMENT book (title, author+, publisher+, price, chapter*)>
<!ELEMENT chapter (title, paragraph*, section*) >
<!ELEMENT section (title?, paragraph*) >
<!ELEMENT title (#PCDATA)>
<!ELEMENT author (#PCDATA)>
<!ELEMENT publisher (#PCDATA)>
<!ELEMENT price (#PCDATA)>
<!ATTLIST price currency CDATA #FIXED ”usd“>
<!ELEMENT paragraph (#PCDATA | emph | image)* >
<!ELEMENT emph (#PCDATA) >
<!ELEMENT image EMPTY >
<!ATTLIST image   file CDATA #REQUIRED
                  height CDATA #IMPLIED
                  width CDATA #IMPLIED >
```

XPath - בחירת צמתים

- ביטוי XPath פשוט – מבטא את המיקום של קדקוד (או לרוב – סט קדקודים) בעץ ה-XML של המסמך.
- כך למשל, **עבור מסמך עם הספר הראשון בלבד(!)**, אם נרצה את שם הספר נרשום את הביטוי הבא:
- `/inventory/book/title`
- ערך הביטוי הוא **הקדקוד**: <title>Snow Crash</title>,
  כלומר – את כל הקדקוד עצמו (ולא רק את הטקסט המופיע שם).
  קודקוד זה יכול להיות גם שורש של עץ בפני עצמו.
- בפועל – קדקוד זה הוא באמת שורש של עץ עם בן אחד מסוג PCDATA.
  (ערכו של קדקוד הבן הוא Snow Crash)

XPath - בחירת צמתים

- בכלליות: הסימון "/" אומר שיש לבצע חיפוש של הביטוי מימין אצל הקדקודים המתקבלים ע"י הביטוי השמאלי. (בהחלט יכול להיות יותר מקדקוד אחד שיתקבל ע"י הביטוי השמאלי!)
  הסימן "/" השמאלי ביותר אומר שיש להתחיל את החיפוש משורץ העץ.

- נסביר מה קורה בהרצת הביטוי `/inventory/book/title` :
  - הביטוי `/inventory` יחזיר את קדקוד השורש.
  - על קדקוד זה שחוזר מחפשים קדקודי `book`. חזרו 3 קדקודים.
  - על כל אחד מ-3 הקדקודים שחזרו מתבצע חיפוש לקדקודי `title`.
  - כל קדקודי ה- `title` שנמצאו הם אלה שחוזרים בתור התשובה לביטוי כולו.

XPath - בחירת צמתים

- הסימון "//" מסמן שהחיפוש של הביטוי הימני צריך להתבצע בצאצאים של הביטוי המתואר משמאל (ולא רק בבנים) וגם בקדקוד עצמו.
- למשל, אם נרצה לחפש את כל קדקודי התמונה במסמך:
- //image
- כיצד נחפש את כל קדקודי הכותרות של הפרקים ותתי הפרקים הקיימים בקובץ?
- /inventory/book/chapter//title

בהנחה שיש לנו את הספר הראשון בלבד – הפלט יהיה:

<title>Snow Crash - Chapter A</title>

<title>Snow Crash - Chapter B</title>

<title>Chapter B - section 1</title>

<title>Chapter C</title>

XPath - בחירת צמתים שאינם אלמנטים

- text() – מאפשר לבחור בן מסוג טקסט
- /inventory/book/chapter//title/text()
  - פקודה זו מחזירה את הערך הטקסטואלי של צמתי ה-PCDATA.
- node() – מאפשר לבחור בן מכל סוג.
- /inventory/book/chapter//paragraph/node()
  - בשלב זה הפקודה לא תחזיר קודקודי תכונות. הסיבה לכך תובהר מאוחר יותר.
- /inventory/book/chapter/paragraph/image/node() = Ø
- * - כל **קדקודי האלמנטים** \ **תכונות** הממוקמים באותה הנקודה במסלול.
- @name – כל **קודקודי התכונות** בשם name.
- *@ - כל קודקודי התכונות בכלל.
- /inventory/book/chapter/paragraph/image/@*
  - שימו לב לכך שלא יחזרו קודקודי התמונות של פרק ב׳ משום שהם בנים של section ואין לנו את זה במסלול שלנו...

XPath - אופרטורים

- אופרטורים אריתמטים: +, -, *, div (לחילוק, במקום הסימן " / "), mod (שארית בחלוקה של שלמים).
- השוואות: >, <, =>, <=, =, =!.
- מחברים בוליאנים: and, or.
- איחוד בין שתי קבוצות צמתים מתבצע ע"י האופרטור " | ".
- /inventory/book | /inventory
  - מחזיר את שלושת קדקודי ה-book וכן את קדקוד ה-inventory.
  - יש לשים לב לכך שמדובר באיחוד. צומת שתופיע בשתי הקבוצות תופיע רק פעם אחת בתשובה.

- ניתן להשתמש **ב"ביטויי חיפוש"** (המתחילים ב-" / " או " // ") **וב"ביטויי בחירה"** (ביטויים בוליאניים) בתוך " [ ] ", כאופרטורים על קבוצת צמתים המתוארת בביטוי משמאל לסוגריים.

[] - XPath

- ניתן **לבחור** את הצומת לפי מספר ע"י הוספת []
- אם נרצה לבחור את הכותרת של הפרק הראשון בכל אחד מהספרים:
- /inventory/book/chapter[1]/title/text()

- Output:

Snow Crash - Chapter A
Burning Tower - Chapter A
Zodiac - Chapter A

- ניתן לשים בסוגריים גם ביטוי אריתמטי כלשהו.
- שימו לב: מספרי הקדקודים מתחילים מ-1!

- כיצד נקבל את הספר הראשון והספר השני?
- /inventory/book[1] | /inventory/book[2]

[] - XPath

- דגש חשוב מאוד!
המספור של הקדקוד ניתן ביחס למיקום בצומת ההקשר שלו (הצומת שממנו ביצענו את החיפוש שהביא לנו את אותו הקדקוד).
- מה יהיה הביטוי עבור שם הפרק השני בכל הקובץ שלנו?

- נבחן את הביטוי //chapter[2]/title/text()
  - התשובות שיתקבלו עבור ביטוי זה הן:
    - Snow Crash - Chapter B
    - Burning Tower - Chapter B
  - למה אנחנו מקבלים 2 צמתים ולא רק צומת אחת כפי שהיינו מצפים?
  - נסביר: מהביטוי //chapter מוחזרים כל קודקודי ה-chapter בקובץ. אך כאמור, כל אחד מהם מקבל מספור ביחס לצומת ההקשר שלו. שני קדקודים אלו הם היחידים שהם קדקודי chapter שממוקמים שניים ברשימת קדקודי ה-chapter שמתקבלת מצומת שממנה הם יצאו. לכן הם יהיו הקדקודים שנקבל בתשובה.

- הביטוי התקין עבור מה שחיפשנו יהיה:
- (//chapter)[2]/title/text()
  - הסוגריים "מאפסות" את צומת ההקשר ⇐ מתבצע מספור מחדש.

[] - XPath

- ניתן לשים ביטוי בוליאני בתוך הסוגריים ע״מ לבחור את הצמתים המקיימים את התנאי הבוליאני.
- /inventory/book[author="Neal Stephenson"]/title/text()
- Output:

Snow Crash

Zodiac

- ניתן לרשום בסוגריים שם של קודקוד.

הקודקודים שנקבל יהיו קדקודי האב (אלה שמחוץ לסוגריים) אשר קיימים עבורם הקדקודים שאותם אנחנו מחפשים בתוך הסוגריים.

בפועל, מתבצע חיפוש של הביטוי שבתוך הסוגריים בקדקודים שמחוץ לסוגריים. אם נמצאו קדקודים המתאימים לביטוי שבסוגריים – אזי הביטוי שבסוגריים יחזיר ״אמת״.

- /inventory/book/chapter[section]/title/text()
- Output:

Snow Crash - Chapter B

- /inventory/book/chapter[paragraph/image]/title/text()
- Output:

Snow Crash - Chapter A

Burning Tower - Chapter B

[] - XPath

- ניתן להשתמש בפונקציה ()not ע״מ לקבל את הקודקודים שבהם התנאי אינו מתקיים.
- /inventory/book/chapter[not(paragraph)]/title/text()
- Output:

Snow Crash - Chapter B

Burning Tower - Chapter A

Zodiac - Chapter A

- שימו לב שהתנאי של קיום paragraph נבדק עבור רמת הבנים של chapter. לכן, חוזרים גם קודקודים שבהם יש צאצא בשם paragraph אבל לא בן בשם זה.

Xpath - [][]

- כאמור, [] מחזירים קבוצת קודקודים העונים על התנאי.
  אם נרצה לבדוק על קבי קודקודים זו תנאי נוסף – נוכל שוב לבדוק קיום תנאי בעזרת [] נוסף.
- /inventory/book[author="Neal Stephenson"][price<10]/title/text()
- Output:

Zodiac

- השאילתה הזו זהה לשאילתה הבאה:
- /inventory/book[author="Neal Stephenson“ and price<10]/title/text()

- עם זאת, לא תמיד אפשר להחליף [][] בפקודת and.

[][] - Xpath

- לדוגמא:
- מה ההבדל בין המקרים הבאים:

- //author[2][1]/text()
- //author[1][2]/text()
- (//author)[2][1]/text()

- Jerry Pournelle
- Null – קבוצה ריקה
- Larry Niven

- נסביר את הסיבה להבדלים:
  - בביטוי הראשון //author מחזיר את כל הסופרים. כל אחד מהסופרים מקבל מספור המייצג אותו ביחס לצומת ההקשר שלו. התוספת [2] מחזירה את כל הסופרים שהמספור שהם קיבלו הוא 2. היחיד שעונה על הגדרה זו הוא Jerry Pournelle. נשים לב לכך שכעת הוא הסופר היחיד שחזר מצומת ההקשר שלו ולכן הוא גם הראשון – המספור שהוא מקבל הוא 1, לכן התוספת [1] מחזירה אותו.
  - בביטוי השני התוספת [1] מחזירה את כל הסופרים הראשונים ביחס לצומת ההקשר שלהם. כל שלושת הסופרים שחוזרים הם הסופרים היחידים עבור צומת ההקשר שלהם ולכן הם ממוספרים כ-1. לכן לא קיים אף קדקוד שהוא השני והתשובה שאנחנו מקבלים היא קבוצה ריקה.
  - בביטוי השלישי – הביטוי //author מחזיר 4 קדקודים. כפי שאמרנו – הסוגריים מאפסות לנו את צומת ההקשר ואנחנו מקבלים מספור חדש מ-1 עד 4 עבור כל הקדקודים שחזרו מהביטוי. כעת נלקח הקדקוד השני (בגלל [2]) וכעת זהו הקדקוד הראשון היחס לצומת ההקשר שלו ולכן הוא הקדקוד שחוזר.

Xpath - פונקציות

- position() - מחזירה את המיקום היחסי של צומת ההקשר בתוך קבוצת הייחוס שלו.
- /inventory/book[position()=2] = /inventory/book[2]

- מה מחזיר הביטוי הבא?
- //author[position()>=2][2]

- איזו שורה (אחת בלבד) ניתן להוסיף לקובץ (ואיפה) ע"מ שביטוי זה יחזיר תשובה?

- last() – מחזירה את הקודקוד האחרון מבין הקודקודים העונים על התנאי.
- /inventory/book[last()] – מחזיר את הספר האחרון בקובץ

- count(arg) - מחזיר את מספר הצמתים בקבוצה המתוארת ע"י ביטוי החיפוש arg.
- לדוגמא, אם נרצה להחזיר את שמות הספרים בעלי לפחות 2 פרקים:
- inventory/book[count(chapter)>=2]/title/text()
- Output:

Snow Crash
Burning Tower

Xpath - השוואות

- ראינו שכאשר אנחנו משווים בין סוגים שונים של משתנים – מתבצעים המרות לצורך ביצוע ההשוואה. הקריטריונים הם:
  - ההשוואות =<, <, =>, > : הערכים המושווים יתורגמו במידת האפשר למספרים לצורך ההשוואה.
    שימו לב שמחרוזות שאינן מייצגות מספרים יגרמו כאן לתוצאה לא צפויה!
  - ההשוואות =, =! : התרגום יתבצע לפי סדר העדיפויות הבא:
    - אם לפחות אחד הערכים הוא בוליאני (למשל תוצאה של השוואה אחרת, או של אחת הפונקציות false(), true()), אז גם הערך השני יתורגם לערך בוליאני.
    - אם אחד הערכים הוא מספר והשני הוא מחרוזת, אז המחרוזת תתורגם גם היא למספר.
    - אם שני הערכים מחרוזות, תתבצע ביניהם השוואה.

Xpath - השוואות של קבוצות צמתים

- השוואה בין ערך לקבוצה של צמתים (הנתונה ע"י ביטוי חיפוש), תחזיר אמת "true()" אם ורק אם קיים צומת כל שהוא בקבוצה, שלאחר תרגומו לערך מהסוג המתאים ההשוואה המתאימה תתקיים.
  - לדוגמא: הביטוי //image/@width>50 יחזיר "אמת" אם ישנה תמונה שהרוחב שלה (המוצהר!) גדול מ-50.
- השוואה בין שתי קבוצות צמתים, תחזיר "true()" אם ורק אם קיים זוג צמתים, אחד מכל קבוצה, כך שבתרגומם לערכים ההשוואה המתאימה תתקיים.
  - לדוגמא: הביטוי

/inventory/book/chapter[1]//image/@file=/inventory/book/chapter[2]//image/@file

יחזיר ערך "true()" אם קיים לפחות קובץ תמונה אחד משותף בין שני הפרקים הראשונים.

- בדיקת שונות (=!) בין שתי קבוצות צמתים (כאשר ערך הוא קבוצת צמתים בעלת צומת אחד) תחזיר "true()" אם ורק אם קיים זוג צמתים אחד מכל קבוצה כך שבתרגומם לערכים ההשוואה המתאימה לא תתקיים.
  - שימו לב לכך שיכול להיווצר מצב שבו גם ביטוי וגם השלילה שלו יחזירו "אמת"!

Xpath - פונקציות

```
<!ELEMENT family (person)+>
<!ELEMENT person (name) >
<!ATTLIST person
    idnum ID #REQUIRED
    gender (male | female) #REQUIRED
    father IDREF #IMPLIED
    mother IDREF #IMPLIED
    children IDREFS #IMPLIED >
<!ELEMENT name (#PCDATA)>
```

- id(arg) - מחזיר את הצומת בעל תכונת ה-ID הזהה לביטוי arg.
  - אם arg הוא מחרוזת עם רווחים, יבוצע חיפוש עבור כל "מילה" במחרוזת.
  - אם arg הוא קבוצת צמתים, הם יתורגמו למחרוזות שיפורקו ל-"מילים", ועבור כל אלו יבוצע חיפוש.
- נחזור לדוגמא של העץ המשפחתי מהתרגול הקודם:
- אם נרצה את האנשים שיש להם בנות:
- //person[id(@children)/@gender="female"]
  - כיצד זה יעבוד?
    - הביטוי //person מחזיר לנו את כל האנשים בקובץ. על הקדקודים שחוזרים נבדק התנאי שבסוגריים. נזכור שבמקרה שהביטוי שבסוגריים הוא תנאי – הקדקודים החוזרים הם אלה שעבורם התנאי הנבדק מחזיר ערך "אמת".
    - במקרה זה – התנאי מתחיל ב-id(@children) ולכן חוזרים קדקודי ה-person במסמך אשר תכונת ה-id שלהם מתאימה לערך התכונה children עבור אותו קדקוד, כלומר - לכל אדם חוזרים קדקודי ה-person של הבנים שלו.
    - ההמשך בודק האם קדקודי ה-person שחזרו מכילים את תכונת ה-gender וכן אם ערך התכונה הוא female. במידה וכן – אזי ערך התנאי הוא "אמת" ולכן נחזיר את הקדקודים האלה.
    - לכן, בסה"כ – חוזרים קדקודי ה-person של אלה שיש להם בנות – בדיוק כפי שדרשנו.
- ישנן פונקציות נוספות ל-Xpath 1.0 שיכולות לסייע לכם לאתר קודקודים ובנים למיניהם. שימו לב לכך שאתם משתמשים בפונקציות בצורה נכונה!

Xpath - פונקציות

```
<!ELEMENT family (person)+>
<!ELEMENT person (name) >
<!ATTLIST person
    idnum ID #REQUIRED
    gender (male | female) #REQUIRED
    father IDREF #IMPLIED
    mother IDREF #IMPLIED
    children IDREFS #IMPLIED >
<!ELEMENT name (#PCDATA)>
```

- id(arg) - מחזיר את הצומת בעל תכונת ה-ID הזהה לביטוי arg.
  - אם arg הוא מחרוזת עם רווחים, יבוצע חיפוש עבור כל "מילה" במחרוזת.
  - אם arg הוא קבוצת צמתים, הם יתורגמו למחרוזות שיפורקו ל-"מילים", ועבור כל אלו יבוצע חיפוש.
- נחזור לדוגמא של העץ המשפחתי מהתרגול הקודם:
- אם נרצה את האנשים שיש להם בנות:
- //person[id(@children)/@gender="female"]
  - ננסה להבין יותר לעומק:
    - נבחן את קדקוד ה-person הראשון שחוזר מ-//: עבורו – הביטוי @children מחזיר: "T13 T14 T15".
    - הביטוי id("T13 T14 T15") מחזיר לנו את הקדקודים שערך השדה idnum שלהם הוא אחד מהשלושה האלה (כי בעצם מבוצע חיפוש עבור כל אחת משלושת המילים במחרוזת). כלומר – חוזרים לנו 3 קודקודי person.
    - עבור כל אחד משלושת הקדקודים הללו חוזר לנו בן מסוג @gender כלומר – קדקוד המייצג את התכונה עבור כל אחד מהקדקודים. בעצם – חוזרים לנו 3 קדקודים כאלה שהערך של 2 מהם הוא male והערך של אחר מהם הוא female.
    - כעת, תתבצע השוואה בין קבוצת הקדקודים שחזרה לבין המחרוזת female. כזכור, מכיוון שהקדקודים הם קדקודי תכונה ואנחנו משווים אותם למחרוזת – תתבצע המרה של אותם קדקודים למחרוזת.
      בנוסף, מכיוון שאנחנו משווים קבוצה על ערכים לערך יחיד, מספיק שיהיה ערך אחד באותה הקבוצה ששווה לערך שאליו אנחנו משווים ע"מ שהשאילתה תחזיר ערך "אמת".
      במקרה שלנו, אכן יש לנו קדקוד עם הערך female בקב' הקדקודים שחזרה ולכן ערך השאילתה תהיה "אמת".
    - בעקבות זאת קדקוד ה-person הראשון אכן יחזור בתור תשובה לשאילתה.
    - תהליך בדיקה דומה יתבצע עבור אל אחד משאר הקדקודים שחוזרים מ-//person.

Xpath - ניווט מתקדם בעזרת צירים (axes)

- עד עכשיו ראינו איך אנחנו יורדים בעץ ואיך אנחנו מגיעים לצאצאים.
- אבל לפעמים אנחנו רוצים לבדוק גם דברים בכיוונים אחרים...
  הדבר נעשה בעזרת "הכוונה" של מסלול החיפוש לכיוונים אחרים.
- child:: - ירידה בכיוון הבן – ברירת המחדל.
  - זו בעצם הירידה שאנחנו מכירים
- /inventory/book = /child::inventory/child::book
- attribute:: - ירידה לכיוון התכונות.
  - גם ירידה כזו אנחנו כבר מכירים:
- //image/@file = //image/attribute::file

- שימו לב לכך שה-"/" הרגיל מהווה תחליף ל-child::/ והוא מהווה ציר שונה מהציר attribute::/
  זה בדיוק מה שגורם לכך שהביטוי node()/ אינו מחזיר קדקודי תכונות.
  עם זאת – הביטוי attribute::node()/ או @node()/ אכן יחזיר קדקודי תכונות.

Xpath - ניווט מתקדם בעזרת צירים (axes)

- descendant:: - כל צאצא אפשרי של הצומת הנוכחי.
- descendant-or-self:: - כל צאצא אפשרי כולל הצומת עצמו.
  - ראינו קודם קיצור **דומה:**
- // = /descendant-or-self::node()/
- כאמור, הקיצור הוא דומה אך לא זהה!
  שימו לב להבדלים הנובעים מהמשמעויות השונות:
- /inventory/book/descendant-or-self::image[1]
  - יחזיר את התמונה הראשונה בכל ספר.
- /inventory/book//image[1]
  - ייתן את התמונה הראשונה בכל פסקה בספר המכילה תמונה. ה-"[1]" יהיה ביחס לצומת ההקשר של צומת ה-image, כי למעשה הביטוי שקול לביטוי:
- /inventory/book/descendant-or-self::node()/child::image[1]

Xpath - ניווט מתקדם בעזרת צירים (axes)

- self:: - הצומת הנוכחי.
  - לדוגמא, אם נרצה לבחור את כל צמתי הכותרת שאינם הקדמה:
- /inventory/book/chapter/title[self::title!="Introduction"]
- parent:: - האבא של הצומת הנוכחי.
  - לדוגמא, מה יחזיר לנו הביטוי הבא:
- /inventory/book/chapter/parent::book
  - את צמתי הספר <u>שיש בהם פרקים</u>
    - איזה ביטוי אחר (שאנחנו מכירים) יביא לאותה התוצאה?
- קיצורים:
  - "." הוא סימון מקוצר ל- "self::node()"
  - ".." הוא סימון מקוצר ל- "parent::node()"

Xpath - ניווט מתקדם בעזרת צירים (axes)

- self:: - שימוש חשוב!
- נזכור שראינו שיש לנו בעיה כשאנחנו רוצים לבדוק שוויון בין שני קבוצות קדקודים.
  - מספיק שיהיה קדקוד זהה בשתי הקבוצות כדי לקבל "אמת" מבדיקת השוואה (למשל, בדיקת השוואה בין קבוצה המכילה את הקדקודים 1 ו-2 לקבוצה המכילה את הקדקודים 1 ו-3 תחזיר "אמת").
  - מספיק שיהיו שני קדקודים שונים כדי לקבל "שקר" מבדיקת שלילת השוני (למשל, אם נרצה להשוות 2 קבוצות המכילות את הקדקודים 1 ו-2, אזי בדיקת השונות תחזיר "אמת" משום שהקדקוד 1 בקבוצה הראשונה שונה מהקדקוד 2 בקבוצה השנייה, ושלילה של זה תחזיר "שקר").
- נניח שנרצה להשוות בין 2 קב' קדקודים:
  - a = /inventory/book/chapter[1]//image/@file
  - b = /inventory/book/chapter[2]//image/@file
- נבצע את ההשוואה כך:
  - count(a[self::node()=b])=count(a) and count(b[.=a])=count(b)
- נסביר:
  - בעצם, אנחנו בודקים הכלה דו כיוונית של כל קבוצת קדקודים אחת בשנייה.
  - מהביטוי a חוזר סט קדקודים כלשהו. על כל אחד מסט הקדקודים הזה נבדוק את התנאי שבסוגריים.
  - נשים לב שהתנאי שבסוגריים בעצם שואל האם הקדקוד שנבדק כרגע נמצא בתוך הקבוצה b. אם כן – ערך הביטוי שבסוגריים יהיה "אמת" והקדקוד יחזור מהשאילתה.
  - במידה וכל קדקודי a נמצאים ב-b אזי כמות הקדקודים שתחזור מהביטוי תהיה זהה לכמות הקדקודים שנמצאת מלכתחילה בביטוי a עצמו.
  - בצורה דומה נבדוק האם כל קדקודי b נמצאים בתוך a וקיבלנו יכולת לבצע בדיקת השוואה.

Xpath - ניווט מתקדם בעזרת צירים (axes)

- following-sibling:: - האחים של הצומת הנמצאים אחריו במסמך. (צמתי תכונות נחשבים כחסרי אחים)
  - לדוגמא:
- /inventory/book/chapter[title="Introduction"]/following-sibling::chapter
  - הביטוי יחזיר את כל צמתי הפרקים לאחר פרק ההקדמה.

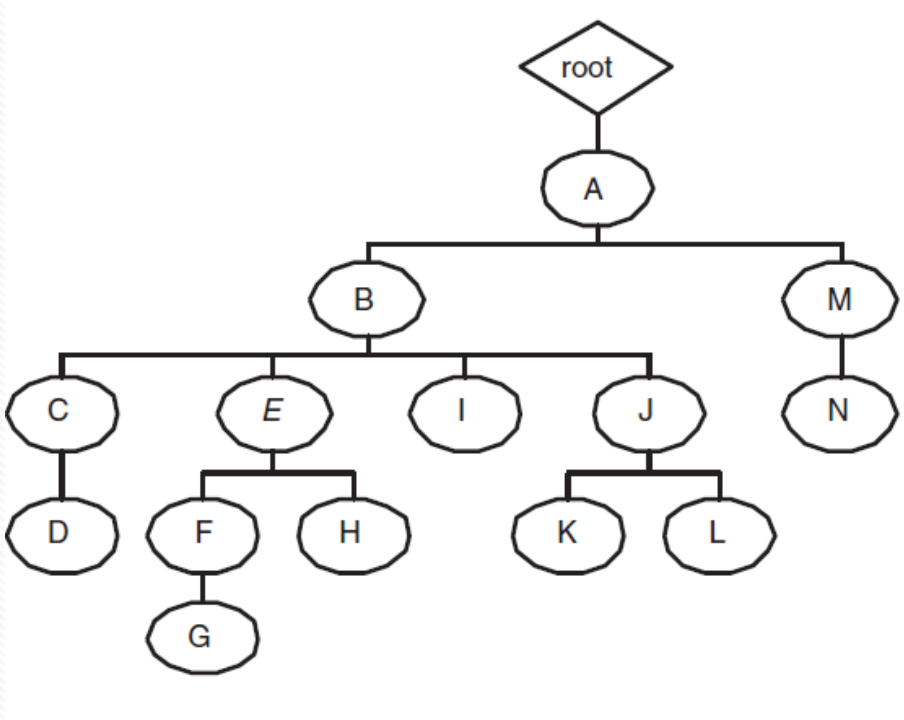
- preceding-sibling:: - האחים של הצומת הנמצאים לפניו במסמך.
  - בניגוד לצירים האחרים, ציר זה הוא ציר <u>הפוך</u>, כלומר – סדר הצמתים יהיה הפוך מסדר ההופעה במסמך.

Xpath - ניווט מתקדם בעזרת צירים (axes)

- צירים נוספים:
  - מציאת האבות הקדמונים (צירים הפוכים):
    - ancestor-or-self::
    - ancestor::
  - למציאת כל הצמתים במסמך לאחר הצומת הנוכחי (על עץ ה-XML), פרט לצאצאים ותכונות:
    - following::
  - למציאת כל הצמתים לפני הצומת הנוכחי (על עץ ה-XML), פרט לאבות הקדמונים ותכונות (ציר הפוך):
    - preceding::

Xpath - ניווט מתקדם בעזרת צירים (axes)

- דוגמא:



Axis	Nodes (relative to E)
self	E
parent	B
child	F,H
descendant	F,G,H
descendant-or-self	E,F,G,H
ancestor	B,A,root
ancestor-or-self	E,B,A,root
preceding	D,C
preceding-sibling	C
following	I,J,K,L,M,N
following-sibling	I,J