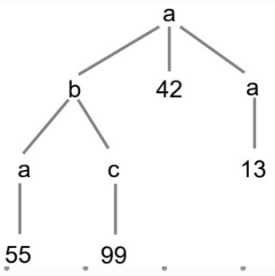
Exercise 3.1

Represent the following tree as an XML document:




```
<a>
    42
    <b>
        <a> 55 </a>
        <c> 99 </c>
    </b>
    <a> 13 </a>
</a>
```

Exercise 3.2 – obligatory (4 points)

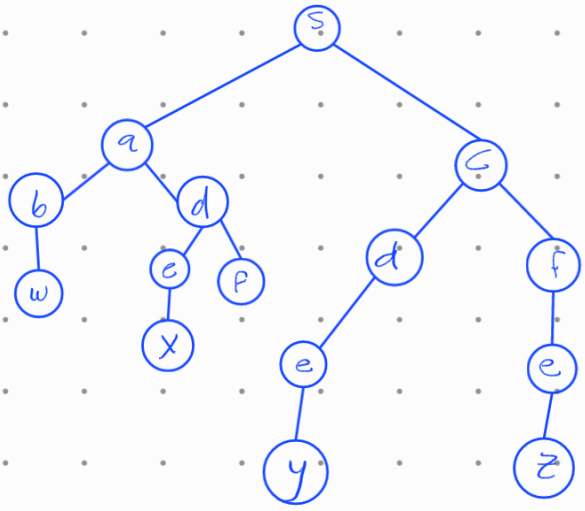
a) Are the following XML documents well-formed? If not, indicate all errors.
b) Represent the structure of the well-formed XML documents as trees.

(1)

a) No, has two XML elements, missing end tags,

```
<a>
    <b> w </b>
    <d>
       <e> x <e> </e>.  => e has 2 opening tags
       <f> </f>         => f has no closing tag
    </d>
</a>
<c>   <-- There can only be one root element <a>
   <d>
       <e> y </e>
   <f>
   </d>   <-- d ends before f
      <e> z </e>
   </f>
</c>
```

```
<s>
    <a>
    <b> w </b>
        <d>
            <e> x </e>
            <f></f>
        </d>
    </a>
    <c>
        <d>
            <e> y </e>
        </d>
        <f>
            <e> z </e>
        </f>
```



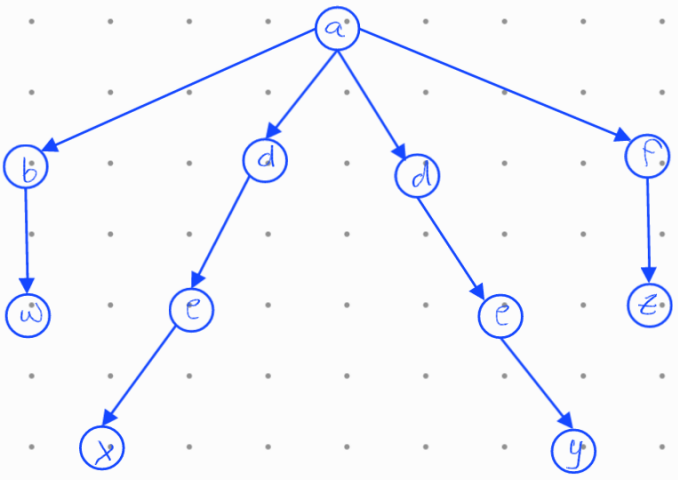
```
    </c>
</s>
```

(2)

a) Yes, the XML document is well-formed on a single horizontal line.

```
<a><b>w</b><d><e>x</e></d><d><e>y</e></d><f>z</f></a>
```

```
<a>
    <b>w</b>
    <d>
        <e>x</e>
    </d>
    <d>
        <e>y</e>
    </d>
    <f>z</f>
</a>
```

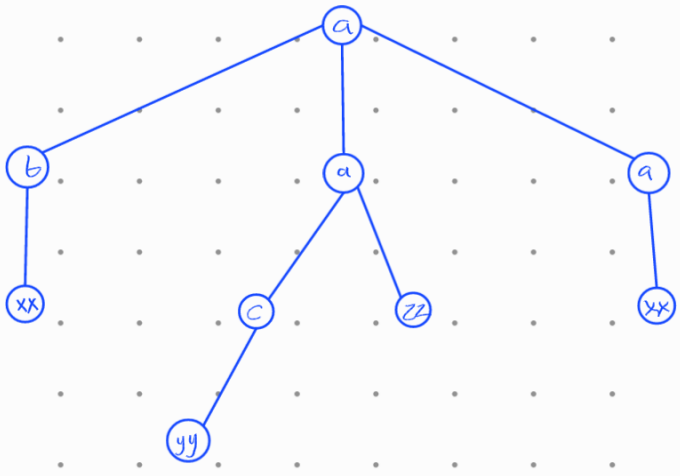


(3)

a) Yes, no missing nested XML elements

```
<a>
   <b>xx</b>
   <a>
      <c>yy</c>
      zz
   </a>
   <a>
      xx
   </a>
</a>
```

```
<a>
    <b>xx</b>
    <a>
        <c>yy</c>
        <d>zz</d>
    </a>
    <a>
        <e>xx</e>
    </a>
</a>
```



Exercise 3.3

Specify regular expressions for the following languages:

a) La = {w ∈ {0,1}* | w contains 01 as a substring}

(0 | 1)*(01)(0 | 1)*

b) Lb = {w ∈ {a,b,c}* | w either starts with a and ends with bc or cb, or starts with b and ends with ac }

(a)(a | b | c)*(bc | cb) | (b)(a | b | c)*(ac)

c) Lc = {w ∈ {a,b,c}* | |w|b is even} (0 is also considered as even)

(a|c|bb)*

Exercise 3.4 – obligatory (6 points)

Specify regular expressions for the following languages:
a) La = {w ∈ {a,b}* | |w| is odd}

(a|b)(aa|ab|bb|ba)*

b) Lb = {xwx | x ∈ S, w ∈ S* }, where S = {a, b, c},
i.e. all strings that start and end with the same symbol

(a)(a|b|c)*(a) | (b)(a|b|c)*(b) | (c)(a|b|c)*(c)

c) Lc = {w ∈ {a, b}* | w does contain neither aa nor bb as a sub-string}

(ab)* | (a) | (ba)* | (b)

Exercise 3.5

There exist many different number formats. One of them is as following, where groups of three digits are separated by apostrophes:
0
42
486
9’386
719’528
83’748’694’846
Indicate a regular expression that describes all possible numbers written this way.

((1|2|...|9)(0|1|...|9){0,2}')* ((1|2|...|9){0, 2}(0|1|...|9)+)

Exercise 3.6 – obligatory (5 points)

Define the format of telephone numbers using regular expressions. Telephone numbers are allowed in the following format variants:

Format Examples Remark
Local number 123450

-Does not start with 0

(1|2|...|9)(0|1|...|9)*

National number with area code
01234-123450
01234/123450
-Starts with a single zero.
-Characters - or / are used as separator between area code and local number

(0)(1|2|...|9)(0|1|...|9)*([-]|[/])(1|2|...|9)(0|1|...|9)*

International number with country code
(0049)1234-123450
(00500)800/9080
-Starts with 00 and one to three digits as country indication.
-National area code is without leading 0.
-Area code and local number may consist of one or more digits.

[(](00)(1|2|...|9)(0|1|...|9){0, 2}[)](1|2|...|9)+(0|1|...|9)*([-]|[/])(1|2|...|9)(0|1|...|9)*

Exercise 3.7
Simplify the following regular expressions.
a) aa | a(be | ea)
= aa | a(be | ea) =
= aa | a(b | a) = // 8
= aa | ab | aa = // left distributivity
= aa | ab // 5

b) a(a | b) | (aa | ab)* e
a(a | b) | (aa | ab)* e =
= a(a | b) | (aa | ab)* = // 8
= aa | ab | (aa | ab)* =
= (aa | ab)*

Exercise 3.8 – obligatory (5 points)
Simplify the following regular expressions. Show the intermediate steps of your transformations.
a) a((a | bb) a) | aaa
a((a | bb) a) | aaa =
= a(aa | bba) | aaa = //right distributivity
= (aaa | abba) | aaa = //left distributivity
= aaa | abba | aaa = //associativity
= aaa | abba

b) (e | a(bd|cd) ((ac|ab)d)*)*

(e | a(bd|cd) ((ac|ab)d)*)* = // = (a(bd|cd))*
= (a(bd|cd) ((ac|ab)d)*)* =
= (a(bd|cd) (ac|ab)d)* =
= ((abd | acd) (acd | abd))* =
= ((acd | abd) (acd | abd))*

Exercise 3.9

Are the following equivalences valid for all regular expression R and S? Give a short justification.

(1) R R* = R* R //valid, R has to occur at least once on both sides
(2) (RS)* = R* S* //invalid, the left side can be RRSSR while the right can't
(3) (R | e)* = R* //valid, empty string is always included in R*
(4) (R | S) * = R* | S* //invalid, right side can only contain empty string, R or S(but not both)
(5) (R* | S*)* = (R | S)* //valid, (rule 10) R* already includes all combinations, R** provides no additional results