1. Simplify the following Boolean algebra expressions as much as possible.

   (a) $(a \vee a')'$

   (b) $a \vee (a' \wedge b)$

   (c) $(a' \vee b) \wedge (a' \vee b')$

(5+5+5)

(a) $(a \vee a')' = 1' = 0$

↑ complement axiom

(b) $a \vee (a' \wedge b) = \underbrace{(a \vee a')}_{=1} \wedge (a \vee b)$ (distributivity)

(complement)

$= a \vee b$ (identity)

(c) $(a' \vee b) \wedge (a' \vee b') = a' \vee (\underbrace{b \wedge b'}_{=0})$ (distributivity)

(complement)

$= a'$ (identity)

2. Use a 7-bit floating-point representation where a floating point number has 1 sign bit followed by 3 exponent bits, and 3 bits for the significant. According to the IEEE standard, subnormal numbers have the exponent 000; the bias for a 3-bit exponent is $2^{3-1} - 1 = 3$. Answer the following questions:

   (a) Write out the bit pattern for the smallest positive normal number and determine its value in decimal.

   (b) Write out the bit pattern for the representation of -0.125. Note that the number may be subnormal!

   (c) Find three numbers a, b, and c where

   $$a \oplus (b \oplus c) \neq (a \oplus b) \oplus c$$

   when computed in this representation.

(5+5+5)

(a) Smallest significant is 000 which, as a normal number, means $(1.000)_2 = 1_{10}$

Smallest possible exponent bit pattern is 001, (because 000 indicates subnormals!)

the actual exponent, therefore, is $1 - \text{bias} = 1 - 3 = -2$

⇒ The bit pattern for the smallest positive normal number is 0 001 000

↑ sign, 001 exponent, 000 significant

Its value is $1 \cdot 2^{-2} = (0.25)_{10}$

(b) Since $0.125 < 0.25$, 0.125 is subnormal.

So its representation is $\text{significant} \cdot 2^{-\text{bias}} = \text{significant} \cdot 2^{-3}$

where the significant is interpreted as $x.yz$

Here, we need significant $= 1_{10}$, so bit pattern 100 works.

⇒ Full bit pattern is 1 000 100

(c) $1 \cdot 2^2 \oplus 1 \cdot 2^{-2} = 1 \cdot 2^2$

$\Rightarrow -1 \cdot 2^2 \oplus (1 \cdot 2^2 \oplus 1 \cdot 2^{-2}) = 0$ but $(-1 \cdot 2^2 \oplus 1 \cdot 2^2) \oplus 1 \cdot 2^{-2} = 1 \cdot 2^{-2}$

or $1\,101\,000 \oplus (0\,101\,000 \oplus 0\,001\,000) \neq (1\,101\,000 \oplus 0\,101\,000) \oplus 0\,001\,000$

3. Word processors can often do some spelling correction "on the fly". Construct a finite state transducer that does this in a very simple case: change a lower-case `i` to an upper-case `I` in sentences like

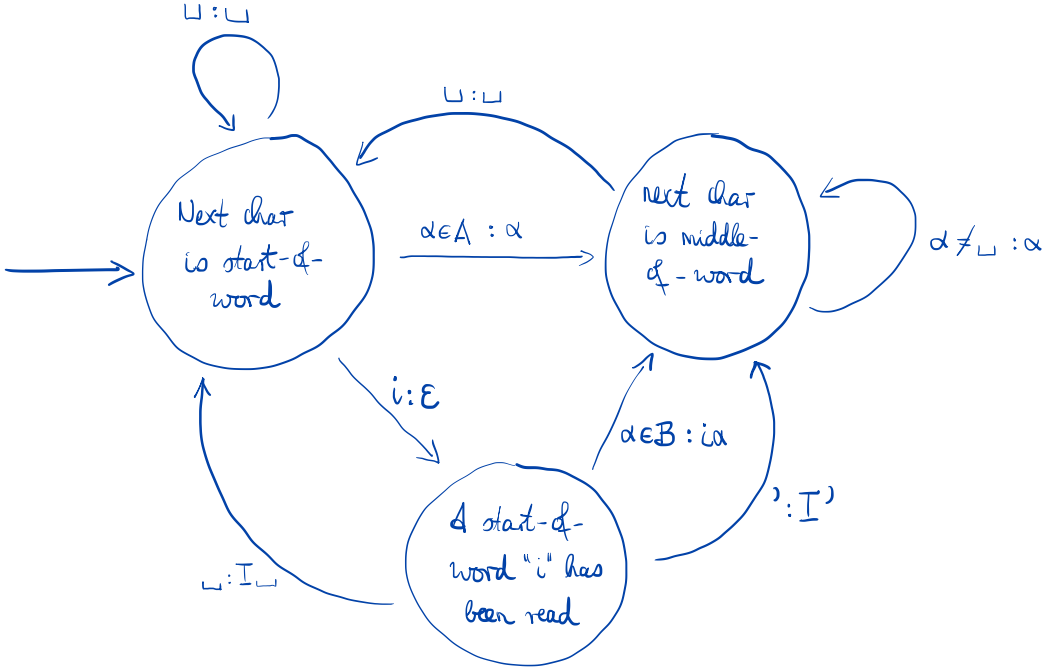
```
i am done
My dog and i go for a walk
The exam i've written
```

Specifically, change `i` to upper case if it is (i) the first letter in the string or preceded by ␣ (space) and (ii) followed by ␣ or ' (apostrophe).

Hints: Three states will suffice to perform this task. Moreover, the following sets will be helpful in specifying the transition rules:

$$A = \{\text{any character that is not } \texttt{i} \text{ or } \sqcup\}$$
$$B = \{\text{any character that is not } \sqcup \text{ or } '\}$$
$$C = \{\sqcup, '\}$$

(10)



Note: I am using shorthand notation $\alpha \in C : \mathrm{I}\alpha$ to say:

"When reading any character α that is contained in C, i.e. is either ␣ or ', output the letter I followed by character α."

4. Consider a file system that may contain hard links (i.e., contain inodes with a reference count larger than one). Describe an algorithm that copies the contents of the entire file system onto another file system, preserving the hard links without creating duplicate data on the target file system.

   Describe your algorithm in precise language or use Python-like pseudo-code. (10)

The key is to maintain a mapping from source inode to destination inode.

Simple algorithm:

Traverse the source file tree

- when encountering a directory, create a directory under the same name in the destination tree
- when encountering a file:

  if source inode is a contained in the inode mapping:

    create a hard link on the destination file system to the target inode under the current name

  else:

    copy source file into the corresponding destination directory

    create a new corresponding entry in the inode mapping

To reduce the overhead of lookup and maintenance for the mapping:

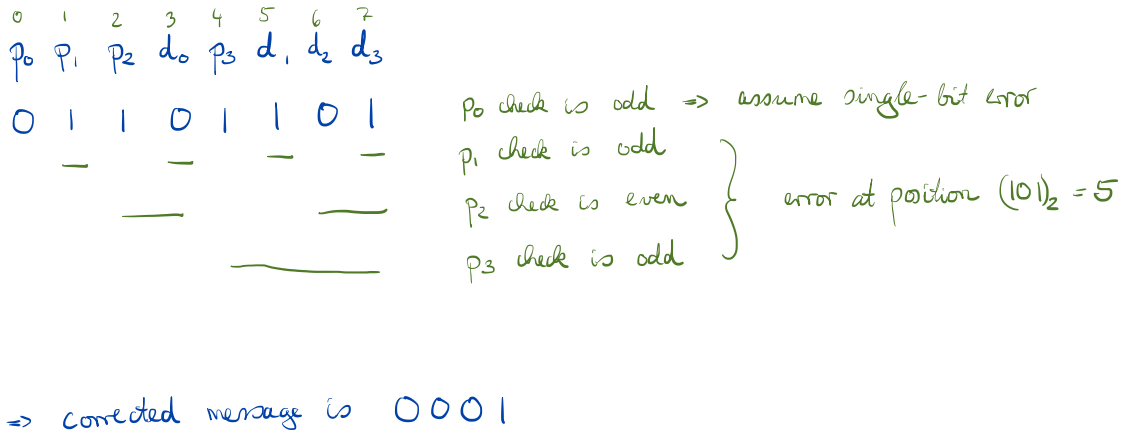
- Do lookup and creation of the mapping only if source inode has refcount > 1
- Keep a lookup counter and delete mapping entry once the mapping has been looked up refcount − 1 times.

Notes: The details on efficient tree traversals and data structures for maps will be discussed next semester in "Algorithms and Data Structures".

5. You receive the following Hamming-(8,4)-encoded message.

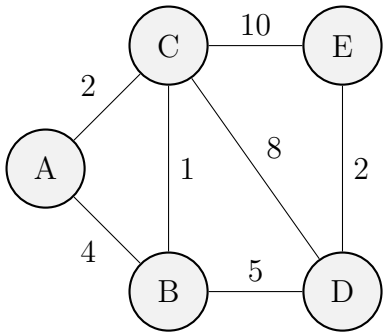
01101101

Extract the message, if possible, correcting single-bit errors as appropriate. The bit-order convention is the one used in class. Show all steps in your work. (5)



=> corrected message is 0001

6. Consider the following router network:



(a) Use Dijkstra's algorithm to compute the shortest path from router A to every other router in the network.

(b) Suppose the link between C and E goes down. How does the network adjust to this damage and how does it affect the shortest path from A to E?

(c) In general, which routing algorithm is faster to adjust to network damage? Link-state or distance-vector routing? Explain.

(10+5+5)

(a)

Step	M	$d_A(A)$	$d_A(B)$	$d_A(C)$	$d_A(D)$	$d_A(E)$
0	~~A~~,B,C,D,E	0 (A)	∞	∞	∞	∞
1	B,~~C~~,D,E	0	4 (B)	2 (C)	∞	∞
2	~~B~~,D,E	0	3 (C)	2	10 (C)	12 (C)
3	~~D~~,E	0	3	2	8 (C)	12 (C)
4	E	0	3	2	8	10 (C)
Routing Table:		A	C	C	C	C

(b) C-E is not part of any shortest path, so if it goes down, all routers can keep their routing tables. (But the link-state packets sent by C and E will change, so the computation of the routing table will be different, only the result remains the same.)

(c)
- With link state routing, only one of the routers attached to the disconnected link needs to flood the network with its LSP, which is small, so the process is fast. Then all routers can recompute their shortest-path table, which is typically also fast.
- With naive distance vector routing, spread of bad news can be very slow ("count to infinity problem"). Even with route poisoning, the route needs to be taken down and an alternative re-established, which takes several exchanges of distance vectors.

=> Link state routing reacts faster.