Problem1 (a):

Let us consider a Turing machine M1 that checks the language on the input string say 'w'. Initially the head is at the leftmost input element i.e the most significant bit. Σ = (0,1) are the input and the tape symbols.

Step 1: Take the string 'w' as the input for the Turing machine M1. Scan the tape until we find a 0 on it. If we find a 0, then we have to mark (with a cross symbol/any symbol) that element on the tape. Then we move the head to the beginning of the string i.e to the leftmost element in the string. Then go to next step i.e Step 2. If there is no 0 or 1 on the tape then accept it, else we reject it.

Step 2: We have to scan the tape again. This time if a '1' is found in the string, we have to mark it (with cross symbol/any symbol). After marking the symbol on the tape go to step 3. If there is no '1' left in the string, then we have to reject it.

Step 3: We should continue scanning from the point where our head had stopped earlier in the previous step. This time we have to look for 1. If it is found on the tape, then we have to mark it (with a cross symbol/any symbol). Then move the head to the starting of the tape string i.e. to the beginning of the tape and then repeat all the steps again from step 1. If we fail to find a 1 in the string, then we reject the string.

Problem 1(b):

Let us consider a Turing machine M2 that checks the language on the input string say 'w'. Initially the head is at the leftmost input element i.e the most significant bit. Σ = (0,1) are the input and the tape symbols.

Step 1: Take the string 'w' as the input for the Turing machine M2. Scan the tape and look for a '1' on it. If we find a '1', then go to step 2 else go to step 5.

Step 2: We should continue scanning our tape from the point where our head is present in the previous step and look for an unmarked '1'. If we find a '1', then mark it. If we do not find a '1' then accept it. Else move the head to the beginning of the tape i.e to the left most part of the tape.

Step 3: Begin scanning the tape again and mark the first '0' that you encounter on the tape. If there's no unmarked '0' on the tape, then accept it.

Step 4: Now we have to move the head to the beginning of the tape i.e to the left most part of the tape and repeat again from step 1.

Step 5: Now we have to move the head to the beginning of the tape, and scan for unmarked 0's. If there are no unmarked 0's left on the tape, then reject it. Else accept it.

References:

1. Designing a Turing machine:
   Class lecture slides (http://infolab.stanford.edu/~ullman/ialc/spr10/slides/tm1.pdf)
2. Text book: Page 326; Chapter – 8; ISBN: 9780321455369; Introduction to automata theory, languages and computation; Author: Hopcroft, John E.,

Problem 2:

Design of a Turing machine that will compute the 2's complement of a binary number.

How to find 2's complement? (algorithm followed):

Method – Find the right most bit that is '1' i.e least significant '1' in the binary input and flip all the bits that are there to the left of first '1' encountered while moving from right to left.

Consider an input binary string '011010' and we try to find the 2's complement of the binary number by following the steps as follows.

[Note: We have put the string in the tape below & B = blank]
Z = (0,1)

Step 1: The input string is present on the tape below.

B	0	1	1	0	1	0	B

Step 2: q0 is the start state and is marked bold where the head is present (leftmost variable in string).

B	**0**	1	1	0	1	0	B

Step 3: We keep on moving to the right till we find a 'blank' on the tape. The head will be moving continuously to the right till it finds a blank space.
δ (q0, Z) = (q0, Z, R)

B	0	1	1	0	1	0	B

Step 4: Once we find a blank in the tape, we change the state from q0 to q1 and then we move to left.
δ (q0, B) = (q1, B, L).

B	0	1	1	0	1	0	**B**

Step 5: head will move to the left. If it encounters a 0, then leave it as it is and move to left.
δ (q1, 0) = (q1, 0, L)

B	0	1	1	0	1	**0**	B

Step 6: If we encounter 1, then leave it as it is but we change the state from q1 to q2 to identify that we encountered the first 1 while traversing from right to left. Now, again we move to left by 1 bit.
δ (q1, 1) = (q2, 1, L)

B	0	1	1	0	**1**	0	B

Step 7: After finding the first 1 in the tape, we flip each bit present to the left of the first one until we encounter a blank in the tape.

B	0	1	1	**0**	1	0	B

Step 8: As mentioned in the previous step we will flip the bits as seen below until the head reaches blank.
δ (q2, 0) = (q2, 1, L) or δ (q2, 1) = (q2, 0, L)
δ (q2, B) = (q2, B, F)

B	**1**	**0**	**0**	**1**	1	0	B

Step 9: The machine will come to halt – 'F'.

The output of the given input string will be i.e 2's complement of '**011010**' will be '**100110**'.

States = {q(start), f(final)}
Q = (q0, q1, q2)
Input symbols (Z) = (0,1)
Σ = (0,1) and the tape symbols are same

Transitions:

1. δ (q0, Z) = (q0, Z, R)
2. δ (q0, B) = (q1, B, L)
3. δ (q1, 0) = (q1, 0, L)
4. δ (q1, 1) = (q2, 1, L)
5. δ (q2, 0) = (q2, 1, L)
6. δ (q2, 1) = (q2, 0, L)
7. δ (q2, B) = (q2, B, f)

References:

1. https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=14&cad=rja&uact=8&ved=0ahUKEwitkrm4s9vQAhVEz1QKHeVXCOc4ChAWCC4wAw&url=http%3A%2F%2Facademic.regis.edu%2Fpsmallwo%2FSitePages%2FCS440%2FCD%2520files%2F1.5.4.1TwosComplementReview.doc&usg=AFQjCNGG0R4LewRQLqhBzIXQktQID0ETLg&sig2=Rqb3IP2KUNOu-h11l64-Aw&bvm=bv.140496471,d.cGw
2. Designing a Turing machine:
   Class lecture slides (http://infolab.stanford.edu/~ullman/ialc/spr10/slides/tm1.pdf)

Problem 3(a):

The given language below is non-Turing recognizable.

{<M>: M is a TM that accepts 3 or more different inputs}

As the number of inputs can be unaccountably infinite, we cannot come to a proper conclusion when the machine might come to halt or when it can reach a final state.

The complement of this language is solved in 3(b) which also has a property of undecidability by which we can say that, this given language is non-Turing recognizable.

Problem 3(b):

The given language below can be said as Turing recognizable. Because,

{<M>: M is a TM that accepts 3 or fewer different inputs}

- Turing machine will come to halt by accepting the given string 'w' which is in the language.
- Turing machine may come to halt in a rejected condition for the given input string 'w'.
- If the given string 'w' is not in the language, the Turing machine may keep on continue looping.

We have also seen an example in class for a machine M which is a Turing machine and that accepts fewer different inputs. Hence we can tell that this language is Turing recognizable.

We can consider a Turing machine example where there are equal number of 1's and 0's. Here's the algorithm for that below.

1. Change the first '0' encountered on the tape and change it to 'X'
2. Move the head right until we find the first '1' on the tape
3. If none, REJECT
4. Change the 1st '1' into 'Y'
5. Move head to left to the leftmost '0'
6. Go To line number 1 and repeat until we do not have any 0's
7. Check if any 1's remain on the tape
8. If YES then
9. REJECT
10. Else
11. ACCEPT

References:

1. http://www.ccs.neu.edu/home/rjw/csu390-f06/HowTo/How-to-Prove-Undecidability-or-Non-Recognizability.pdf
2. Textbook: Chapter 9 – Undecidability; ISBN: 9780321455369

Problem 4(a):

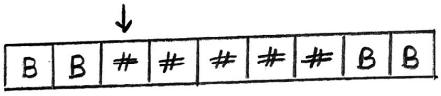
Problem 4:

(a) Given:

* M_1 → machine
* N → Number of occurances of "#"
* "#" → symbol which is provided as input to M_1.

Consider the example where the input to the Turing machine M_1 is ##### i.e N=5
for this the expected output is "101"

Step 1: Consider the tape below where the given input is used.



Step 2: Initially the head is at the leftmost input element. Now we have to traverse the tape.

Step 3: Scan the tape until we find "#" and mark it. Then continue and do not mark the next one. So we have to mark every alternate # found in the tape like as shown below.

KIRANCHANDRASHEKHAR_SHETTAR@STUDENT.UML.EDU
UML ID - 01605800

Step 3 (cont'd) : If the last bit in the tape is marked, then GO TO step 4. Else go to step 5.

Step 4 : Now, we push '1' into the beginning of the tape. Our tape contents look as below. Now, repeat step 3. (Move head at the beginning initially)

Step 5 : Now, we push '0' into the beginning of the tape. Our tape contents look as below. Now, repeat step 3. (Move head at the beginning initially)

Final: From step 4 we will get the following output result.

Now, we can see that the output is 101 for ###### as expected.

Hence, the above designed algorithm for the Turing machine is proved.

KIRANCHANDRASHEKHAR_SHETTAR@STUDENT.UML.EDU
UML ID - 01605800

Problem 4(b):

Problem 4:

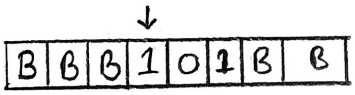
<b> Given:

* M_2 → Machine
* w_B → Number of occurances of "#"
* "#" → The output symbol by machine M.

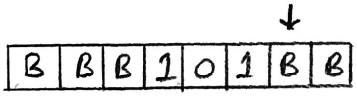
Consider the example where the input to the Turing-machine M_2 is 101.
for this the expected output is #####.

Step 1: Consider the tape below where the input is used

Step 2: Initially the head is at leftmost input element. We now traverse the tape until we encounter a blank on the tape.



→ part A

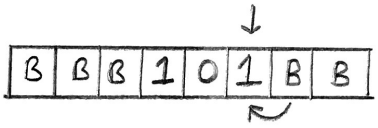


→ part B

KIRANCHANDRASHEKHAR_SHETTAR@STUDENT.UML.EDU
UML ID - 01605800

Step 3: We access each element in the tape by indexing every block to the power of base 2 where power = {0, 1, ... } i.e $2^0, 2^1, 2^2, \ldots$ so on. from right (least significant bit) to left (MSB).

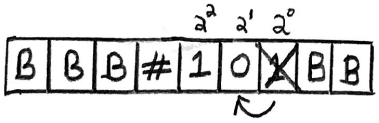
Step 4: As we encounter a blank on the tape, we keep moving to the left. If we encounter a '1' on the tape, go to step 5. If we encounter a '0' on the tape, just mark it and then move to the left. Uneless and until you reach the most significant bit (MSB) on the tape.



Step 5: Based on the head position and the corresponding index value, we calculate the number of #'s to be pushed into the tape.

For the current example we calculate as follows: Head is at ~~2nd~~ 2nd location. 2^{0th} location.

Now, $2^0 = 1$. Hence we push 1 # into the stack as follows.

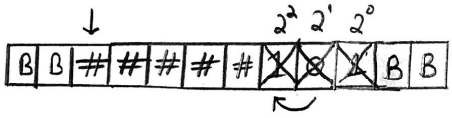


KIRANCHANDRASHEKHAR_SHETTAR@STUDENT.UML.EDU
UML ID - 01605800

Step 5: (cont'd): As we traverse to the left repeat step 4.

Final: The result will be as follows

At the end head will be at $2^{2^{th}}$ location. i.e $2^2 = 4$. Hence we push 4 #'s into the stack or tape as follows.



The tape will be modified at the end as seen in above image.

Now, we can see that the output is ##### for the input 101 as expected

Hence, the above designed algorithm for the Turing machine is proved.

References:
1. https://www.youtube.com/watch?v=9D47vcFhe7Y
2. Text book: Page 326; Chapter – 8; ISBN: 9780321455369; Introduction to automata theory, languages and computation; Author: Hopcroft, John E.,
3. Binary to unary & unary to binary conversion.