Solution of mathematical logic tutorial 1

Exercice 1 Unfolding the Turing machine on the given sequence will yield the following configurations.
We start by placing the state q_0 at the beginning of the sequence as follows :
$\#q_0s_0s_0s_2s_1s_2s_1s_2s_0s_2\#$
We apply the instruction $q_0s_0Dq_1$, we will have :
$\#s_0q_1s_0s_2s_1s_2s_1s_2s_0s_2\#$
Now, since the machine is in state q_1 and reads symbol s_0, we need to execute the instruction $q_1s_0Dq_1$, which is a loop on the symbols s_0. After the execution, we will have the following configuration :
$\#s_0s_0q_1s_2s_1s_2s_1s_2s_0s_2\#$
The new state is still q_1 , and it reads s_2 , so we need to execute the instruction $q_1s_2Dq_1$, and we will have :
$\#s_0s_0s_2q_1s_1s_2s_1s_2s_0s_2\#$
We continue in the same manner, and we will have :
$\#s_0s_0s_2s_1q_2s_2s_1s_2s_0s_2\#$
$\#s_0s_0s_2q_3s_1s_2s_1s_2s_0s_2\#$
$\#s_0s_0q_3s_2s_1s_2s_1s_2s_0s_2\#$
$\#s_0q_3s_0s_2s_1s_2s_1s_2s_0s_2\#$
$\#q_3s_0s_0s_2s_1s_2s_1s_2s_0s_2\#$
$q_3\#s_0s_0s_2s_1s_2s_1s_2s_0s_2\#$
We are in the configuration $q_3\#$ which is the condition, and we notice that there is no action, so the machine halts, and we say it is in a halting state.

Exercise 2 We will provide two solutions.
Solution 1 :
$q_0(a/b)Dq_1$
$q_1(a/b)Dq_1$ These two instructions are used to traverse the entire word (a loop until the end of the word.)
$q_1\#Gq_2$ The end of the word is found, so it is necessary to move back one position.
q_2aDq_3 If the last symbol is "a", move to the right with the action Dq_3.
$q_3\#Fq_4$ We will inevitably going to find the blank symbol, which is # in this case we replace it with the symbol "F" representing "False" so we write "F" in place of #.
q_4F Halt The new configuration of the TM has no action.
q_2bDq_5 If the last symbol is a "b", so with q_2 we read the final symbol which is a "b", move to the right and change the state.
$q_5\#Tq_6$ While moving to the right, we will encounter a #, so it is necessary to replace it with a "T" to indicate True.
q_6T Halt.
Solution 2 :
q_0aDq_1 If we read the symbol "a", we move to the state q_1 i.e., if the machine is at the state q_1 it indicates that it has just read an "a".
q_0bDq_2 If we read "b", we move to q_2.
q_1aDq_1 It also reads "a", so it stay at the state q_1.
q_1bDq_2 If it reads a "b", it moves to q_2.

q_2bDq_2 it reads a "b", So it retains the state q_2.
q_2aDq_1 It reads an "a", it goes to q_1.
$q_2\#Tq_2$ If it reads #This means that the end of the word is found, and state q_2 informs us that the last symbol read is a "b", so it writes a "T".
q_2T Halt.
$q_1\#Fq_1$ The last symbol is an "a", so it writes an "F".
q_1F Halt.

Exercise 3 Solution :
q_0aDq_1 The first symbol is an "a".
q_0bDq_2 The first symbol is a "b".
$q_1(a/b)Dq_1$ Loop until the end of the word, knowing that the first symbol of the word is an "a".
$q_1\#Gq_3$ The end of the word is found, so it is necessary to move left by one position and change the state.
q_3aDq_4 The last symbol is an "a" considering that the first one is also an "a".
$q_4\#Tq_4$ Write "T" on the #.
q_4T Halt.
$q_2(a/b)Dq_2$ Loop until the end of the word, knowing that the first symbol of the word is an "b".
$q_2\#Gq_5$ The end of the word is found, so it is necessary to move left by one position and change the state.
q_5bDq_4 The last symbol is "b" considering that the first one is also "b", Move right and go to state q_4, , because if the machine is at q_4 and it reads # , it will directly replace it with "T" by executing the instruction $q_4\#Tq_4$.
q_3bDq_6 if the last symbol is a "b"considering that the first one is an"a", It moves to the right to state q_6 (a new stat).
$q_6\#Fq_6$ write an "F".
q_6F Halt.
q_5aDq_6 The last symbole is an "a" considering that the first one is a "b", Move to the right at q_6, because afterward, the machine's head will encounter a # so it replaces it with an "F" by executing the instruction $q_6\#Fq_6$.

Exercise 4 There are three solutions.
Solution 1 :
q_00Dq_1
q_11Dq_2 We will directly read the first two symbols "01" without modification as words always begin with "01".
q_210q_3 We reach the 3rd symbol, which is always "1", We replace it with "0".
$q_3(0/1)Dq_3$ go through the rest of the word until #.
$q_3\#$ Halt.The Turing machine halts without executing any action.
In this correct solution, we used 4 states"q_0, q_1, q_2, q_3" and 5 instructions.
Solution 2 :
q_00Dq_1
q_11Dq_2 We will directly read the first two symbols "01" without modification as words always begin with "01".

$q_2 10 q_3$ We reach the 3rd symbol, which is always "1", We replace it with "0".
$q_3 0$ Halt. As we are not going to change anything afterwards, we block the machine by not executing any action.
With the solution 2, We kept the same number of states but reduced the number of instructions to 4.

Solution 3 :

$q_0 0 D q_1$
$q_1 1 D q_2$ We will directly read the first two symbols "01" without modification as words always begin with "01".
$q_2 10 q_2$ We reach the 3rd symbol, which is always "1", We replace it with "0"But we keep the same state q_2 Because we know that we will never encounter a configuration $q_2 0$ since the 3rd symbol is always "1".
$q_2 0$ Halt.
With solution 3, we used only 3 states q_0, q_1, q_2 and 4 instructions. It is the most optimal solution in terms of execution time and memory space.

Exercise 5 Solution :

$q_0 0 D q_1$ We always start by reading the simplest word, in our case, the word 0001#, so we begin by reading the first "0".
$q_1 0 D q_2$ read the 2nd "0".
$q_2 0 D q_3$ read the 3rd "0".
$q_3 1 D q_4$ read "1".
$q_4 \# T q_5$ If after reading the "1", we find a #, this means that we have just read the sequence "0001" which represents the first word in the tape. So, we replace the # with the "T" to say that the word contains the sequence "0001".
$q_4 (0/1) D q_4$ If after reading the sequence "0001" we do not find the # directly, this means that the word is not finished but it contains the sequence. In this case, we loop with the state q_4 until the end of the word.
$q_5 T D q_5$ After writing the "T" we move to the right with the same state q_5.
$q_5 \#$ Halt. If we find a 2nd #, that means that it is the end of the tape which contains one word. So now we have treated the cases #0001#, #00010....#, #00011.....#, #0001##, #00010.....## and #00011.....##.
$q_5 0 D q_1$ The next word starts with "0", we move to q_1 because we just read a “0”.
$q_5 1 D q_6$ If the next word starts with "1", you have to change the state.
$q_6 1 D q_6$ Loop through the "1".
$q_6 \# F q_5$ We find a # with the state q_6 This means that we did not find the sequence, we must write an "F".
$q_5 F D q_5$ Test on the other word.
$q_6 0 D q_1$ We found the first "0" after a sequence of "1".
$q_1 \# F q_5$ We didn’t find the sequence and we found a #.
$q_2 \# F q_5$ We didn’t find the sequence and we found a #.
$q_3 \# F q_5$ We didn’t find the sequence and we found a #.
$q_1 1 D q_6$ After a "0", we found a "1", we go to the right and to the state q_6 to test on the "1s". If we find , we loop with the instruction $q_6 1 D q_6$, otherwise if we find a "0", we consider it to be the first "0" and we execute $q_6 0 D q_1$ otherwise if we come across the #, it means that it is the end of the word and we cannot find the sequence, we execute $q_6 \# F q_5$.

$q_2 1 D q_6$ The same with q_2.
$q_0 1 D q_6$ If the word starts with a "1", we test on the "1s" going to the state q_6.
$q_3 0 D q_3$ If we find more than "000", we continue to look for the "1" because we consider for example in the word "1000000010" there is the sequence 0001 which is in red "1000000010".

Now, if in the word "1000000010" we consider that the sequence "0001" does not exist, i.e., we only take 0001 into consideration, then we delete the instruction $q_3 0 D q_3$ and we replace it with these 4 instructions.
$q_3 0 D q_7$ We move to the right and we change the state.
$q_7 0 D q_7$ We loop over the "0"s with the new state.
$q_7 1 D q_6$ If we find a "1" after the "0"s, we go to state q_6.
$q_7 \# F q_5$ If we find a #, we write an "F" and move to q_5.

Exercise 6 Solution :
$q_0 a D q_1$
$q_1 a D q_2$
$q_2 b D q_3$
$q_3 (a/b) D q_3$
$q_3 \# T q_4$
$q_4 T D q_4$
$q_4 \#$ Halt.
$q_4 a D q_1$
$q_4 b D q_5$
$q_5 b D q_5$
$q_5 a D q_1$
$q_5 \# F q_5$
$q_5 F D q_4$
$q_0 b D q_5$
$q_1 b D q_5$
$q_1 \# F q_5$
$q_2 \# F q_5$
$q_2 a D q_6$
$q_6 a D q_6$
$q_6 b D q_5$
$q_6 \# F q_5$

Exercise 7 Solution :
$q_0 1 D q_1$ Read the first "1".
$q_1 1 D q_2$ Read the 2nd "1".
$q_2 01 q_3$ Replace the "0" that comes only after two "1s" with a "1".
$q_3 1 D q_4$
$q_4 \#$ Halt.
$q_4 0 D q_4$ Loop over the "0s" with state q_4.
$q_4 1 D q_1$ We have just read a first "1", so we must go to the right and go to the state q_1 .

$q_0 0 D q_4$ The word starts with a "0" you have to go through all the "0s".
$q_1 0 D q_4$ After a "1", we have a "0".
q_1# Halt.
$q_2 1 D q_5$ We have just read a 3rd "1", so we must loop over the "1s".
$q_5 1 D q_5$ Loop over the "1s".
$q_5 0 D q_4$ We find a "0" after three "1" or more.
q_5# Halt.
q_2# Halt.

Exercise 8 Solution :
The procedure to apply : you have to write a machine which goes through the "a" then the "b" and if it finds an "a" between the "bs", it transforms it into a "b" then it returns backwards and it transforms the first "b" in the list into "a".

$q_0 a D q_1$ We suppose that the word begins with an "a".
$q_1 a D q_1$ In the case where we have several "a" we must loop over all the "a".
$q_1 b D q_2$ The first "b" after one or many "a"s.
$q_2 b D q_2$ In the case where we have several "b" we must loop over all the "b". Now the word is sorted.
$q_2 a b q_3$ If we find an "a" after the "b", we replace it with a "b" and we change the state.
$q_3 b G q_3$ We go back through all the "b"s.
$q_3 a D q_4$ We find the first "a" which is before the "b"s.
$q_4 b a q_1$ We replace the first “b” with an “a”.
$q_0 b D q_2$ The word starts with a "b".
q_1# Halt.
q_2# Halt.
$q_3 \# D q_4$ The case where the word begins with "b" then we find an "a".

Exercise 9 Solution :

q_0 0 # q_1	q_4 (0/1) D q_4
q_0 1 # q_2	q_4 # G q_9
q_1 # D q_3	q_9 1 # q_6
q_2 # D q_4	q_5 # T q_5
q_3 (0/1) D q_3	$q_5 T$ Halt
q_3 # G q_5	q_5 1 F q_5
q_5 0 # q_6	q_5 F Halt
q_6 # G q_7	q_9 0 F q_5
q_7 (0/1) G q_7	q_9 # T q_5
q_7 # D q_8	q_8 1 # q_2
q_8 0 # q_1	q_8 # T q_5

This solution uses a minimal number of states, but it is not fast because each time we go back to the left (to the beginning) then we do the same processing again (to avoid this, we can do the processing on both sides). With the second solution, we will use a number of states which can go up to q_{17}.

Note : You can give this exercise (the second solution) as homework.

Exercicse 10 Solution :
$q_0(0/1)Dq_1$
$q_1(0/1)Dq_1$
$q_1\#Gq_2$
q_201q_3
q_210q_4
q_40Gq_2
$q_2\#1q_3$
$q_3(0/1)Dq_3$
$q_3\#Dq_5$
$q_5\#Dq_6$
$q_6\#$ Halt.
$q_6(0/1)Dq_1$

Exercise 11 Solution :
1- To show that the function plus = x + y is a primitive recursive function, we use the following recursion rule :

- plus (x,0) = x + 0 = x = $\pi_1^1(x)$.
- plus (x,y+1) = plus(x,y)+1 = succ (plus(x,y))= succ($\pi_2^3(x, plus(x,y), y)$)= succ$\circ\pi_2^3$(x,plus(x,y),y).

Conclusion :
The function h = $\pi_1^1(x)$ is a basic primitive recursive function because it is a projection, the function g = succ$\circ\pi_2^3$ is primitive recursive because it is constructed by composition of two basic primitive recursive functions succ and π. We have thus succeeded in constructing the "plus" function by the recursion rule, starting from the two primitive recursive functions h and g. The function "plus" is primitive recursive.

2- The function Sigma = $\Sigma_{i=0}^{x} i$
We will proceed to a construction using the recursion rule.

- Sigma(0) = 0 which is a constant.
- Sigma(x+1) = (0+.....+x)+ (x+1) = Sigma(x)+ (x+1) = plus(Sigma(x),x+1)= plus(Sigma(x),x)+1 = succ(plus(Sigma(x),x)) = plus ($\pi_1^2(Sigma(x), x)$,succ$\circ\pi_2^2$(Sigma(x),x))

Conclusion :
The constant function h=0 is a basic recursive primitive function (the zero() function), the plus function is a primitive recursive, the functions π_1^2, succ and π_2^2 are primitives recursive. So Sigma is primitive recursive.

3- The predecessor function (pred(x)).
pred(x)= $\{_{0 \quad si \quad x=0}^{x-1 \quad si \quad x\neq 0}$
Using the recursion rule we will have :

- pred(0) = 0, it is primitive recursive.

— pred(x+1) = x = g(x,pred(x)) = $\pi_1^2(x, pred(x))$
— or pred(x+1) = x = g(x,pred(x)) = $\sigma(\pi_2^2(x, pred(x)))$, it is primitive recursive.

It is primitive recursive because all the functions that compose it are primitive recursive. In this example, we used σ instead of succ (it's the same thing).

4-The subtraction function (Sub(x,y) or $\dot{-}(x, y)$ or x$\dot{-}$y) such that :
x$\dot{-}$y = $\begin{cases} x-y & si \quad x \geq y \\ 0 & si \quad x<y \end{cases}$.
By the recursion rule we will have :

— x$\dot{-}$0 = x = $\pi_1^1(x)$(identity function), it is primitive recursive because it is a basic function.
— x$\dot{-}$(y+1) = pred(x$\dot{-}$y) = pred($\pi_2^3(x, x \dot{-} y, y)$),it is primitive recursive.

It is primitive recursive car because it is composed of several primitive recursive functions.
Example :
2$\dot{-}$1 = 2 $\dot{-}$ (0+1) = pred (2 $\dot{-}$ 0) = pred (2) = 1.
1$\dot{-}$2 = 1 $\dot{-}$ (1+1) = pred (1 $\dot{-}$ 1) = pred (1 $\dot{-}$(0+1))= pred(pred(1 $\dot{-}$ 0)) = pred(pred(1))= pred(0)= 0.

5- The absolute difference function
$|x - y| = \begin{cases} x-y & si \quad x \geq y \\ y-x & si \quad six<y \end{cases}$
We can write $|x - y|$ = (x$\dot{-}$y)+(y$\dot{-}$x), so it is primitive recursive because it is composed of the function "plus" which is primitive recursive and the function $\dot{-}$ which is primitive recursive too.

6- The alpha function alpha such as $\alpha(x)= \begin{cases} 1 & si \quad x=0 \\ 0 & si \quad x \neq 0 \end{cases}$.
By the recursion rule, we have :

— $\alpha(0) = 1 = succ \circ zro()$, it is primitive recursive.
— $\alpha(x + 1) = 0$ = zero(), it is primitive recursive.

So the alpha function is a primitive recursive function because it is composed of several primitive recursive functions.
We can also write $\alpha(x)$ in the following form :
$\alpha(x) = 1 \dot{-} x$,i.e.,

— $1 \dot{-} 0 = 1$
— $\alpha(x + 1) = 1 \dot{-} x \dot{-} 1$ = pred (1 $\dot{-}$ x)= pred(π_2^2(x,1 $\dot{-}$ x)).

Example :
$\alpha(3) = 1 \dot{-} 3 = 1 \dot{-} (2+1)$= pred(1 $\dot{-}$ 2) = pred (1 $\dot{-}$ (1+1))= pred(pred(1 $\dot{-}$ 1))= pred(pred(1 $\dot{-}$(1 $\dot{-}$ 0)))= pred(pred(pred(1 $\dot{-}$ 0)))=pred(pred(1))=pred(0)=0.

7- The multiplication function mult = x * y.
By the recursion rule

— mult(x,0) = x * 0 = 0 = zero().
— mult(x,y+1) = mult(x,y)+x

If we follow the formalization given in class, we will have :
— mult(x,0) = x * 0 = 0 = zero()=h(x), it is primitive recursive.
— mult(x,y+1) = mult(x,y)+x = g(y,mult(x,y),x) =

π_2^3(y,mult(x,y),x) + π_3^3(y,mult(x,y),x)
We can also write g = succ∘π_3^3(mult(x,y),x-1,plus(mult(x,y),x-1)).
The mult function is primitive recursive Because it is composed of primitive recursive functions.

8- The factorial function Fact(x) = $x!$
Using the recursion rule, we will have :

- $0! = 1$ = succ∘zero() it is a composition of two primitive recursive functions.
- $(x + 1)!$ = g(x,$x!$)= (x+1)* $x!$ = succ(x) * $x!$ = succ($\pi_1^2(x, x!)$) * $\pi_2^2(x, x!)$ = mult(succ($\pi_1^2(x, x!)$),$\pi_2^2(x, x!)$), it ist primitive recursive (composition of two primitive recursive functions)

So the Fact(x) function is primitive recursive.
Example :
Fact(3) = fact(2+1) = succ(2)* Fact(2)= succ(2)* Fact(1+1) = succ(2)*succ(1)*Fact(1)= succ(2)*succ(1)*Fact(0+1) = succ(2)*succ(1)*succ(0)* Fact(0)= 3*2*1*1=6.