Lab Topic: Additional Notes

1. In Section Z (String Generation)

- The Z function **RandNumber(N)** `[AleaNombre in French]` generates a random number between 0 and N-1.
- The Z function **RandString(N)** `[AleaChaine in French]` generates a random string composed of N characters.

To thoroughly test your program, frequently generate strings starting with the following characters:

- X = 'Y'
- Y = 'Z'
- Z = 'a'

You can implement this as follows:

```
I := RandNumber(10);
If I = 0 then Word := 'Y' + RandString(RandNumber(5) + 3)
If I = 1 then Word := 'Z' + RandString(RandNumber(5) + 3)
If I = 2 then Word := 'a' + RandString(RandNumber(5) + 3)
Else Word := RandString(RandNumber(5) + 3)
```

Feel free to adjust the value `10` as needed to control the frequency of specific cases.

2. In Section Z (Using Files)

You can build and read a Z file as follows:

French Version:

```
SOIT
  F UN FICHIER DE (CHAINES, chaine) BUFFER S ;
  I : ENTIER ;

DEBUT
  OUVRIR ( F , 'F2.z' , 'N' ) ;
  POUR I := 1 , 101
    init_struct(S, [ALEACHAINE ( 5 ) , ALEACHAINE ( 5 )]);
    ECRIRESEQ ( F , S )
  FPOUR ;
  FERMER ( F ) ;
  OUVRIR ( F , 'F2.z' , 'A' ) ;
  I := 0 ;
  TQ NON FINFICH ( F )
    LIRESEQ ( F , S ) ;
```

```
      ECRIRE ( S ) ;
      I := I + 1
    FTQ ;
    FERMER ( F ) ;
    ECRIRE ( 'Compte = ' , I ) ;
  FIN
```

English Version:

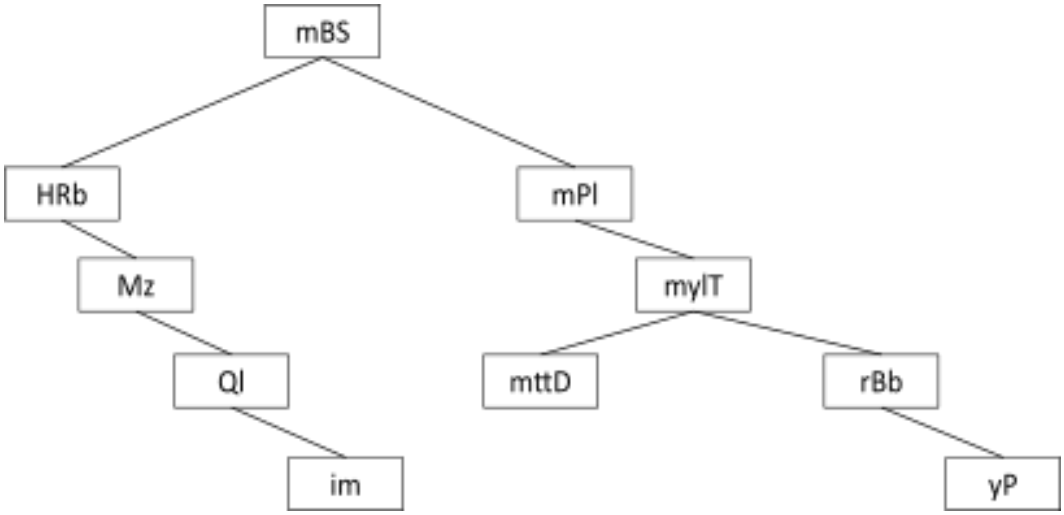
```
LET
  F : FILE OF STRINGS BUFFER S ;
  I : INTEGER ;

BEGIN
  OPEN ( F , 'F.z' , 'N' ) ;
  FOR I := 1 , 101
    S := RANDSTRING ( RANDNUMBER ( 5 ) + 1 ) ;
    WRITESEQ ( F , S )
  EFOR ;
  CLOSE ( F ) ;
  OPEN ( F , 'F.z' , 'O' ) ;
  I := 0 ;
  WH NOT ENDFILE ( F )
    READSEQ ( F , S ) ;
    WRITE ( S ) ;
    I := I + 1
  EWH ;
  CLOSE ( F ) ;
  WRITE ( 'Count = ' , I ) ;

END
```

3. In Section C (Simulation Program)

A. Counting Operations in Search and Range Search

The following tree is obtained after inserting the words in this order:

'mBS', 'HRb', 'Mz', 'QI', 'im', 'mPI', 'myIT', 'mttD', 'rRb', 'yP'



Example of a String Binary Search Tree.

For a single search (Word) :

- Count left child (LC) and right child (RC) operation to reach **Word**.

Examples:

Searching for **'mttD'** results in a total search path length of 4.

Searching for **'Ma'** results in a total search path length of 3.

For a range search ([Word1, Word2]):

- Count left child (LC) and right child (RC) operation to reach the first word greater than or equal to **Word1.**
- Count left child (LC), right child (RC), and the operation (Parent or Pop) to reach all the next words less than or equal to **Word2**.

Example for range search ['Ma' , 'mz'] using the **Parent** operation:

- Search phase: LC , RC(Visit 'Mz')
- Subsequent steps:
  - RC (Visit 'Ql')
  - RC (Visit 'im')
  - Parent, Parent, Parent, Parent (Visit 'mBS')
  - RC (Visit 'mPl')
  - RC, LC (Visit 'mttD')

- Parent (Visit 'myIT')
- RC (Visit 'rBb' and stop)

The range search visits a total of 13 nodes.

Alternatively, using a **Stack-based** approach:

- Search phase: LC , RC (Visit 'Mz')
- Subsequent steps:
  - RC (Visit 'Ql')
  - RC (Visit 'im')
  - Pop (Visit 'mBS')
  - RC (Visit 'mPl')
  - RC , LC (Visit 'mttD')
  - Pop (Visit 'myIT')
  - RC (Visit 'rBb' and Stop)

The range search visits a total of 9 nodes.

B. Table for Storing Results

- E denotes the number of existing elements in the tree.
- !E denotes the number of non-existing elements in the tree.
- All = E + !E.

Use the following table in the **single search** simulation:

Simulation		**BST0**	**(BST1, BST2, BST3)**
S1	E	Total Length of Search Paths Traversed	Total Length of Search Paths Traversed
	!E		
	All		
S2	E	Total Length of Search Paths Traversed	Total Length of Search Paths Traversed
	!E		
	All		
.....			
S10	E	Total Length of Search Paths Traversed	Total Length of Search Paths Traversed
	!E		
	All		

Use the following table in the **Range Search** simulation:

Simulation	**BST0**	**(BST1, BST2, BST3)**
S1	*Total Number of Nodes Visited*	*Total Number of Nodes Visited*
S2	*Total Number of Nodes Visited*	*Total Number of Nodes Visited*
	...	
S10	*Total Number of Nodes Visited*	*Total Number of Nodes Visited*