1. Learning with Errors (LWE) Encryption: Parameters Generation

1.1. Private Key PrK Generationn

We consider computations performed **mod *p***, where ***p*** is prime and ***p***=11.
Let Alex, Bill and Cecilia are working in IT company earning x, y, z money per hour.
This reward we denote as an vector **w** of **unknown parameters** x, y, z.
Vector **w** in a transposed form can be written as a vector row having 3 components:

$\mathbf{w}^T$=(x, y, z).

For example, if x=1, y=2, z=3, then $\mathbf{w}^T$=(1, 2, 3).
For our application the vector **w** we use in non-transposed form as a vector column.
Then in Octave it can be defined by the following column:

```
>> w=[1;2;3]
w =
  1
  2
  3
```

Vector **w** is a **PrK** in LWE encryption system.

1.2. Public Parameters PP Generationn

Public Parameters **PP** consist of 2 components PP1 and PP2, i.e. **PP** = (PP1, PP2).
1.2.1. PP1 generation.
Let Alex, Bill and Cecilia were working in Monday, Tuesday, Wednesday and Thursday different times presented in table

	Alex	Bill	Cecilia
Monday	2 hours	2 hours	3 hours
Tuesday	2 hours	1 hours	2 hours
Wednesday	3 hours	2 hours	2 hours
Thursday	3 hours	3 hours	1 hours

This table of works in hours can be written in matrix form consisting of 4-rows and 3-columns

2	2	3
2	1	2
3	2	2
3	3	1

For our simulation this matrix we denote by **M** and create with Octave

```
>> M=[2,2,3;2,1,2;3,2,2;3,3,1]
M =
  2  2  3
  2  1  2
  3  2  2
  3  3  1
```

Further we will need 4 rows of matrix **M** denoted by **Mrow1**, **Mrow2**, **Mrow3**, **Mrow4**
represented in corresponding row vectors in Octave

```
> Mrow1=[2,2,3]
Mrow1 =    2   2   3

>> Mrow2=[2,1,2]
Mrow2 =    2   1   2

>> Mrow3=[3,2,2]
Mrow3 =    3   2   2

>> Mrow4=[3,3,1]
Mrow4 =    3   3   1
```

Then salaries per day can be computed by multiplying Mrows with the vector column earnings per hour.
Salary in Monday: **Mrow1*w** = Sal1
Salary in Tuesday: **Mrow2*w** = Sal2
Salary in Wedneday: **Mrow3*w** = Sal3
Salary in Thursday: **Mrow4*w** = Sal4

This multiplication corresponds the common rule of multiplication of vector row with the vector column of the same dimention which in our case this dimension is 3 (three components).
As the result of these vectors multiplication is the following linear system of 4 equations representing the total salaries per day denoted by vector column **Sal** with components (Sal1, Sal2, Sal3, Sal4).
The component (Sal1, Sal2, Sal3, Sal4) values can be found by having earnings per hour x=1, y=2, z=3.
All computations are performed **mod *p***.
We asumed ***p***=11.

Monday	2*x + 2*y + 3*z = 2*1 + 2*2 + 2*3 = 15 mod 11 = 4 = Sal1 (1)
Tuesday	2*x + 1*y + 2*z = 2*1 + 1*2 + 2*3 = 10 mod 11 = 10 = Sal2 (2)
Wednesday	3*x + 2*y + 2*z = 2*1 + 1*2 + 2*3 = 10 mod 11 = 2 = Sal3 (3)
Thursday	3*x + 3*y + 1*z = 3*1 + 3*2 + 1*3 = 12 mod 11 = 1 = Sal4 (4)

	Matrix M rows	Day Sal.Comp.	Day Sal.	Errors	Day Sal.w.Err.
Mrow1=[2,2,3]	2 2 3	Sal1=matmult(Mrow1,w,p)	4	-1	3
Mrow2=[2,2,3]	2 1 2	Sal2=matmult(Mrow2,w,p)	10	-1	9
Mrow3=[2,2,3]	3 2 2	Sal3=matmult(Mrow3,w,p)	2	1	3
Mrow4=[2,2,3]	3 3 1	Sal4=matmult(Mrow4,w,p)	1	1	2
		PuK has 2 components			
	PuK 1 Comp				**PuK 2 Comp**
	2 2 3				4
	2 1 2				9

	3 2 2				3
	3 3 1				2

2. LWE Encryption

For 1 bit encryption Encryptor Bob selects several equations from the system of linear equations.
For example, Bob selects ***at random*** 3 equations: (2), (3), (4), i.e. 3 vector rows:

2 1 2
3 2 2
3 3 1

Then Bob sumes components of selected vector rows for every column obtaining the followig row:
8 6 5

```
> Mrow23=matadd(Mrow2,Mrow3,p)
Mrow23 =   5   3   4
>> Mrow234=matadd(Mrow23,Mrow4,p)
Mrow234 =   8   6   5
```

Analogously Bob sumes Sal2, Sal3, Sal4 with errors, denoted by Sal2we=9, Sal3we=3, Sal4we=2.
As a result we obtain
Sal234we = Sal2we + Sal3we + Sal4we = 9 + 3 + 2 = 14 mod 11 = 3

In this case the ciphertext ***c*** consist of 2 parameters: ***c*** = (Mrow234, Sal234we)
c = (c_1, c_2) = (8 6 5 , 3)
If encrypted bit b=0, then ***c*** = (c_1, c_2) = (8 6 5 , 3) --> then c_2=3
If encrypted bit b=1, then ***c*** = (c_1, c_2) = (8 6 5 , 8) --> then c_2=8 = 3+(p-1)/2)) = 3+5;

3. LWE decryption

Alice after receiving ***c*** performes the following.
She takes **PrK** = **w** and computes
d = Mrow234***w mod *p*.**

```
>> d=matmult(Mrow234,w,p)
d = 2      % if c2=3, then Bob encrypted b=0 since the difference between c2=3 and d=2
           % is less than (p-1)/2=5
           % otherwise, if c2=8, then Bob encrypted b=1, since 8-2 = 6 >= (p-1)/2=5
```