

Faculty of Engineering & Technology

Department of Information and Communication Technology

Subject: Programming With Python (01CT1309)

Aim: Analysis of Discrete-Time Signals Using Z-Transform

Experiment No: 17 Date: Enrollment No:92510133011

<u>Aim:</u> Analysis of Discrete-Time Signals Using Z-Transform

IDE:

Install Library

pip install sympy

Example 1:

import sympy as sp

Define symbols

n, z, a = sp.symbols('n z a')

Define the signal $x[n] = a^n * u[n]$

 $x n = a^{**}n$

Compute the Z-transform

X z = sp.summation(x n * z**(-n), (n, 0, sp.oo))

Print the result

print("Z-transform of $x[n] = a^n u[n]$:")

sp.pprint(X_z, use_unicode=True)

output:

Example 2:

Define symbols

Faculty of Engineering & Technology

Department of Information and Communication Technology

Subject: Programming With Python (01CT1309)

Aim: Analysis of Discrete-Time Signals Using Z-Transform

Experiment No: 17 Date: Enrollment No:92510133011

```
n, z, a = sp.symbols('n z a')
# Define the signal x[n] = a^n * u[n]
x_n = 2**n
# Compute the Z-transform
X_z = sp.summation(x_n * z**(-n), (n, 0, sp.oo))
# Print the result
print("Z-transform of x[n] = a^n u[n]:")
sp.pprint(X_z, use_unicode=True)
output:
```

```
Example 3:
```

import sympy as sp

Define symbols

n, z = sp.symbols('n z')

Define the unit step signal u[n]

u n = 1

Compute the Z-transform

U z = sp.summation(u n * z**(-n), (n, 0, sp.oo))

Print the result

print("Z-transform of the unit step signal u[n]:")

sp.pprint(U z, use unicode=True)

output:



Faculty of Engineering & Technology

Department of Information and Communication Technology

Subject: Programming With Python (01CT1309)

Aim: Analysis of Discrete-Time Signals Using Z-Transform

Experiment No: 17 Date: Enrollment No:92510133011

Example 4:

import sympy as sp

Define symbols

n, z, alpha = sp.symbols('n z alpha')

Define the signal $x[n] = \exp(alpha * n) * u[n]$

 $x_n = sp.exp(alpha * n)$

Compute the Z-transform

 $X z = sp.summation(x_n * z**(-n), (n, 0, sp.oo))$

Print the result

print("Z-transform of x[n] = exp(alpha * n) u[n]:")

sp.pprint(X z, use unicode=True)

output:



Faculty of Engineering & Technology

Department of Information and Communication Technology

Subject: Programming With Python (01CT1309)

Aim: Analysis of Discrete-Time Signals Using Z-Transform

Experiment No: 17 Date: Enrollment No:92510133011

```
Example 5:
import sympy as sp
# Define symbols
n, z = sp.symbols('n z')
# Define the finite sequence x[n] = {1, 2, 3}
x_n = [1, 2, 3]
# Compute the Z-transform manually
X_z = sum(x_n[i] * z**(-i) for i in range(len(x_n)))
# Print the result
print("Z-transform of the finite sequence {1, 2, 3}:")
sp.pprint(X_z, use_unicode=True)
output:
```

```
PS C:\Users\trupa\OneDrive\Documents\PWP> python -u "c:\Users\trupa\OneDrive\Documents\PWP\PWP EXP 17.py"

Z-transform of the finite sequence {1, 2, 3}:

2  3

1 + - + --

z  2

z
```

```
Example 6
```

import sympy as sp

Define symbols

n, z, omega = sp.symbols('n z omega')

Define the sinusoidal sequence x[n] = sin(omega * n) * u[n]

x n = sp.sin(omega * n)

Compute the Z-transform

X z = sp.summation(x n * z**(-n), (n, 0, sp.oo))

Print the result

print("Z-transform of x[n] = sin(omega * n) u[n]:")

sp.pprint(X z, use unicode=True)

output:

```
PS C:\Users\trupa\OneDrive\Documents\PWP> python -u "c:\Users\trupa\OneDrive\Documents\PWP\PWP EXP 17.py"

Z-transform of x[n] = sin(omega * n) u[n]:

α

-n

z · sin(n·ω)

n = 0
```

Post Lab Exercise:

• Using Python, compute the Z-transform of the sequence $x[n] = 3^n u[n]$.

```
PS C:\Users\trupa\OneDrive\Documents\PWP> python -u "c:\Users\trupa\OneDrive\Documents\PWP\PWP POST LAB 17.py"

Z-transform of x[n] = 3^n u[n]:

\[ \frac{z}{-} \] for z > 3 \ v z < -3 \\
    z - 3 \\
    \times \]

\[ \text{n - n} \]

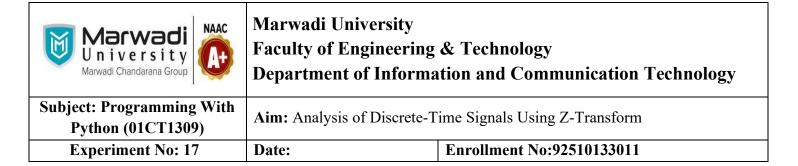
\[ \text{n - n} \]

\[ 3 \cdot z \]

\[ \text{otherwise} \]

\[ \text{n = 0} \]
```

• Using Python, compute the Z-transform of the sequence $x[n] = \cos(wn)u[n]$.



Github link:

https://github.com/trupalijasani05/trupali-jasani