
 <b>Marwadi University</b> Marwadi Chandarana Group 	<b>Marwadi University</b> <b>Faculty of Engineering &amp; Technology</b> <b>Department of Information and Communication Technology</b>	
<b>Subject: Programming With Python (01CT1309)</b>	<b>Aim:</b> Analysis of Discrete-Time Signals Using Z-Transform	
<b>Experiment No: 17</b>	<b>Date:</b>	<b>Enrollment No:</b>

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

**IDE:**

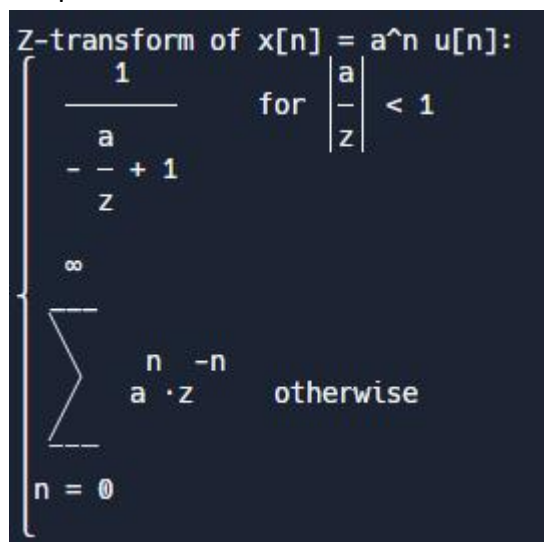
Install Library

```
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 :



The screenshot shows the output of the SymPy code. It displays the Z-transform of the signal  $x[n] = a^n u[n]$ . The result is a piecewise function: a rational expression  $\frac{1}{1 - \frac{a}{z}}$  for  $\left| \frac{a}{z} \right| < 1$ , and an infinite summation  $\sum_{n=0}^{\infty} a^n \cdot z^{-n}$  otherwise.

$$\text{Z-transform of } x[n] = a^n u[n]:$$

$$\left\{ \begin{array}{ll} \frac{1}{1 - \frac{a}{z}} & \text{for } \left| \frac{a}{z} \right| < 1 \\ \sum_{n=0}^{\infty} a^n \cdot z^{-n} & \text{otherwise} \end{array} \right.$$

Example 2:

```
# Define symbols
n, z, a = sp.symbols('n z a')
```

**Subject: Programming With Python (01CT1309)**

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

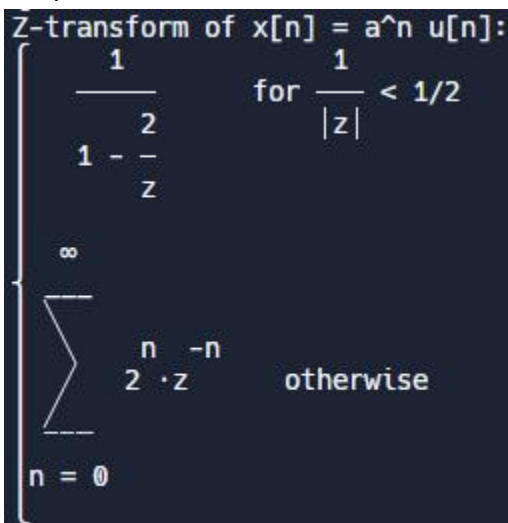
**Experiment No: 17**

**Date:**

**Enrollment No:**

```
# 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 :





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

$$\left\{ \begin{array}{ll} \frac{1}{1 - \frac{2}{z}} & \text{for } \frac{1}{|z|} < 1/2 \\ \sum_{n=0}^{\infty} \left(\frac{2}{z}\right)^n & \text{otherwise} \end{array} \right.$$

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)
```

 <b>Marwadi University</b> Marwadi Chandarana Group 	<b>Marwadi University</b> <b>Faculty of Engineering &amp; Technology</b> <b>Department of Information and Communication Technology</b>	
<b>Subject: Programming With Python (01CT1309)</b>	<b>Aim:</b> Analysis of Discrete-Time Signals Using Z-Transform	
<b>Experiment No: 17</b>	<b>Date:</b>	<b>Enrollment No:</b>

Output :

Z-transform of the unit step signal  $u[n]$ :

$$\begin{cases} \frac{1}{1 - \frac{1}{z}} & \text{for } \frac{1}{|z|} < 1 \\ \sum_{n=0}^{\infty} z^{-n} & \text{otherwise} \end{cases}$$



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 :

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

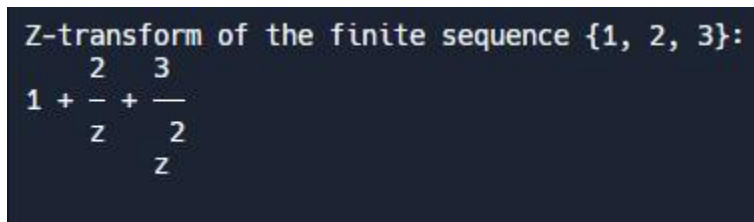
$$\sum_{n=0}^{\infty} z^{-n} e^{\alpha \cdot n}$$

 <b>Marwadi University</b> Marwadi Chandarana Group 	<b>Marwadi University</b> <b>Faculty of Engineering &amp; Technology</b> <b>Department of Information and Communication Technology</b>	
<b>Subject: Programming With Python (01CT1309)</b>	<b>Aim:</b> Analysis of Discrete-Time Signals Using Z-Transform	
<b>Experiment No: 17</b>	<b>Date:</b>	<b>Enrollment No:</b>

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 :





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

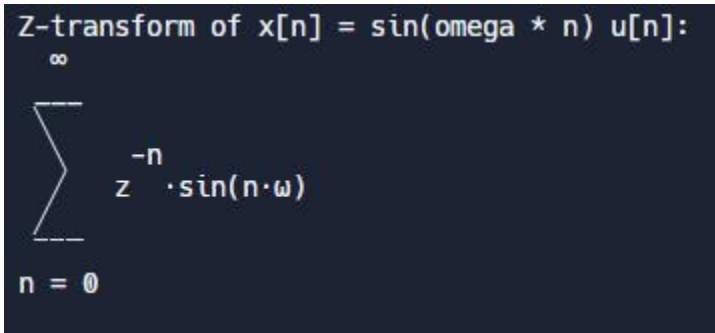
$$1 + \frac{2}{z} + \frac{3}{z^2}$$

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)
```

 <b>Marwadi University</b> Marwadi Chandarana Group 	<b>Marwadi University</b> <b>Faculty of Engineering &amp; Technology</b> <b>Department of Information and Communication Technology</b>	
<b>Subject: Programming With Python (01CT1309)</b>	<b>Aim:</b> Analysis of Discrete-Time Signals Using Z-Transform	
<b>Experiment No: 17</b>	<b>Date:</b>	<b>Enrollment No:</b>

Output:



Z-transform of  $x[n] = \sin(\omega \cdot n) u[n]$ :

$$\sum_{n=0}^{\infty} z^{-n} \cdot \sin(n \cdot \omega)$$

$n = 0$

#### Post Lab Exercise:

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

Code :

```
import sympy as sp

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

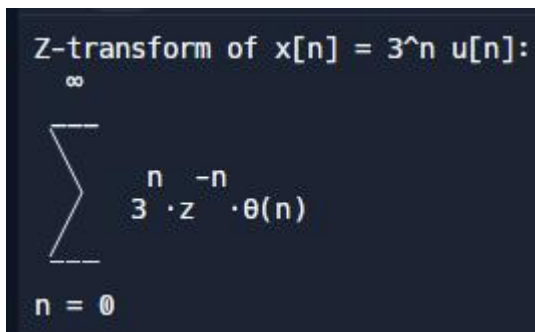
x_n = 3**n * sp.Heaviside(n) # Heaviside function u[n]

Z_transform = sp.Sum(x_n * z**(-n), (n, 0, sp.oo)).doit()

print("Z-transform of x[n] = 3^n u[n]:")

sp.pprint(Z_transform)
```



Output :



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

$$\sum_{n=0}^{\infty} 3^n \cdot z^{-n} \cdot \theta(n)$$

$n = 0$

 <b>Marwadi University</b> Marwadi Chandarana Group 	<b>Marwadi University</b> <b>Faculty of Engineering &amp; Technology</b> <b>Department of Information and Communication Technology</b>	
<b>Subject: Programming With Python (01CT1309)</b>	<b>Aim:</b> Analysis of Discrete-Time Signals Using Z-Transform	
<b>Experiment No: 17</b>	<b>Date:</b>	<b>Enrollment No:</b>

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

Code :

```
import sympy as sp

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

x_n = sp.cos(omega * n) * sp.Heaviside(n) # Heaviside function u[n]

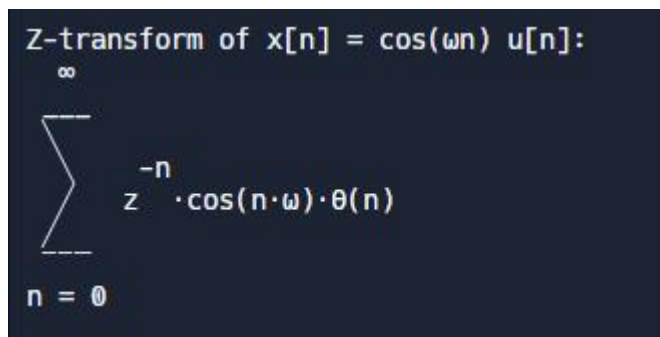
Z_transform = sp.Sum(x_n * z**(-n), (n, 0, sp.oo)).doit()

Z_transform_simplified = sp.simplify(Z_transform)

print("Z-transform of x[n] = cos(ωn) u[n]:")

sp.pprint(Z_transform_simplified)
```

Output:



Z-transform of  $x[n] = \cos(\omega n) u[n]$ :

$$\sum_{n=0}^{\infty} z^{-n} \cdot \cos(n \cdot \omega) \cdot \theta(n)$$

$n = 0$