Experiment 1

Shashwat Shah
60004220126
BE comps C22

Aim: Construction of standard DT signals (sine, cos, step, unit, unit impulse, ramp), and single operations.

Theory: Discrete time (DT) signals are representations of continuous signals in digital form, sampled at specific intervals.

→ Sine and cosine signals.
These are periodic waveforms used extensively in signal processing

→ Unit step signals.
The unit step signal $u(n)$ is a discrete signal that starts from zero and stops to a certain point

→ Unit impulse signal
Also known as delta function $\delta(n)$ its zero everywhere, except at $n=0$ where it is 1

→ Ramp signal.
It increases linearity with time, useful in control systems and for simulating acceleration over time.

→ Signal operators
Addition and subtraction

→ Multiplication.

→ Time Scaling - It stretches or compresses the signal in the time domain.

onclusion - By constructing and visualizing these DT signals, we gain and understand of how each signal behaves over discrete intervals and how they can be combined or altered through basic signal operations.

NAME: Shashwat Shah SAP ID: 60004220126
DIV/BATCH:C22

DIGITAL SIGNAL PROCESSING (DSP) EXPERIMENT 01

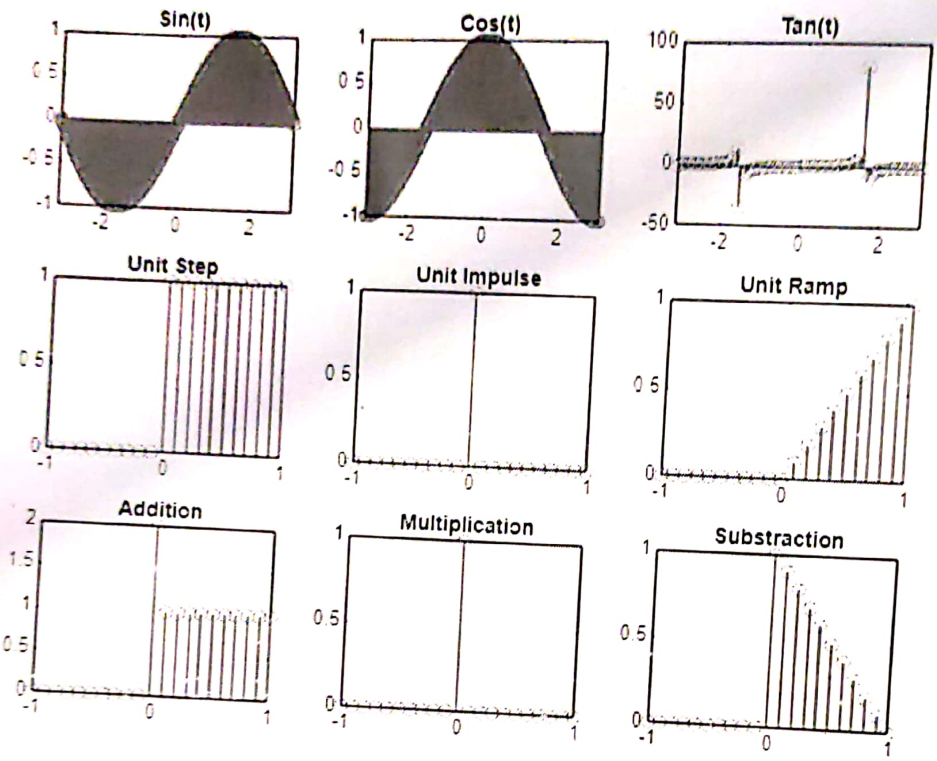
CODE:

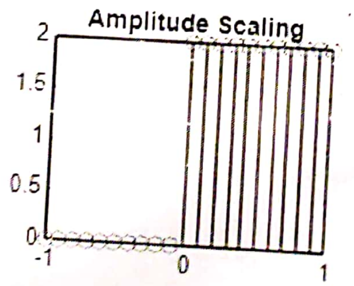
```
t=-pi:0.01:pi; y =
sin(t);
subplot(3,3,1),stem(t,y);
title('Sin(t)');

t=-pi:0.01:pi; y =
cos(t);
subplot(3,3,2),stem(t,y);
title('Cos(t)');

t=-pi:0.1:pi; y =
tan(t);
subplot(3,3,3),stem(t,y);
title('Tan(t)');

t = -1:0.1:1;
unitstep = t>=0;
subplot(3,3,4),stem(t,unitstep); title('Unit
Step');

impulse = t==0;
subplot(3,3,5),stem(t,impulse); title('Unit
Impulse');

ramp = t.*unitstep;
subplot(3,3,6),stem(t,ramp); title('Unit
Ramp');

add = impulse+unitstep;
subplot(3,3,7),stem(t,add); title('Addition');

mul = impulse.*unitstep;
subplot(3,3,8),stem(t,mul);
title('Multiplication');

sub = unitstep-ramp;
subplot(3,3,9),stem(t,sub); title('Substraction');
```

Amplitude Scaling

```
mul = 2.*unitstep; subplot(3,3,8),stem(t,mul);
title('Multiplication');
```

OUTPUT:





DE:

```
)efine the original signal x(n) n = 0:8; % n
ges from 0 to 8
1:9; % x(n) is defined from 1 to 9

ake input for the expansion factor b
nput('Enter the expansion factor b: ');

ompute the expanded signal x(n/b) expanded_n = n / b;

terpolate to find the expanded x values
nded_x = interp1(n, x, expanded_n, 'linear', 'extrap');

t x(n/b) to 0 when n/b is not an integer expanded_x(mod(expanded_n, 1) ~= 0) = 0;
```