Alexandria University

Faculty of Engineering

Electronics and Communication Department

Mini Project I

General signal generator

By

Ahmed Sameh Taha 21010083
Karen Mostafa El-Bardan 21010966
Amira Mohamed Mokhtar 21010308
Rewan Gamal AbdEl-Kader 21010540

Supervised by:

Dr. Mohamed Moselhy

Eng. Ahmed Mostafa

Eng. Esraa Ragab

Procedures

1) When the program starts, the user is asked for the following parameters of the signal:
   a) Sampling frequency of signal (Fs).
   b) Start and end of time scale (Start_time & End_time)
   c) Number of the break points (Breakpoints)
   d) The positions of the break points (Position): If there are no break points so, the position will be empty [].

```
Fs = input('Enter the sampling frequency of the signal:');
Start_time = input('Enter the start time of the time scale:');
End_time = input('Enter the end time of the time scale:');
Breakpoints = input('Enter the number of breakpoints: ');

Position = zeros(1, Breakpoints);
if (Breakpoints == 0)
    Position = [];
else
    for i = 1:Breakpoints
    Position(i) = input(['Enter position of breakpoint (', num2str(i), '):' ]);
   end
end
```

2) According to the number of break points, the program asks the user at each region to enter the type of the signal at this region:

a) **DC signal:** It stands for "direct current" and represents a signal that maintains a constant voltage or value over time.

b) **Ramp signal:** is a type of signal where the amplitude varies linearly with time.
The general mathematical form of a ramp signal:
r(t) = m*t + b, where:
m is the slope of the ramp (rate of change).
b is the intercept, representing the initial value of the ramp signal.

c) **General order polynomial:** refers to a signal that is represented by a polynomial function:
$p(t)=A * t^{k} + C$, where
A is the amplitude
k is the power or degree of the polynomial
C is the intercept, specifying the vertical shift (or offset) of the polynomial.

d) **Exponential signal:** is a type of signal that follows an exponential function

$y(t)=A* e^{bt}$

A is the amplitude

b is the exponent, which controls the rate of growth or decay of the exponential function.

e) **Sinusoidal signal:** A "sinusoidal signal" is a fundamental type of signal that follows a sinusoidal waveform, characterized by its periodic oscillation

$y(t)=A*\sin(2\pi ft+\phi)$

A is the amplitude, which controls the peak value of the sinusoidal wave.

f is the frequency, representing the number of cycles per unit of time (often measured in Hertz, Hz).

ϕ is the phase shift, determining the horizontal displacement of the waveform.

f) **Sinc function:** is a well-known mathematical function that is commonly used in signal processing and communications

$\text{sinc}(t)=A* \sin(\pi(t-c)) / \pi(t-c)$

A is the amplitude.

c is the center shift, representing the horizontal shift or displacement of the sinc function along the time axis.

g) **Triangle pulse:** A "triangle pulse" is a specific type of signal characterized by its triangular shape, typically used in signal processing and waveform generation

Its parameters:

A is the amplitude, representing the peak value of the triangle pulse.

c is the center shift, determining the horizontal position of the pulse.

w is the width of the pulse, specifying the duration over which the pulse rises and falls symmetrically around its center.

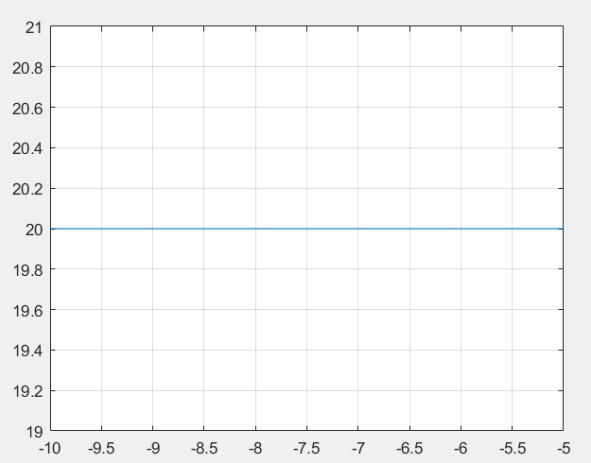
```
14 -    disp('1-DC signal');
15 -    disp('2-Ramp signal');
16 -    disp('3-General order polynomial');
17 -    disp('4-Exponential signal');
18 -    disp('5-sinusoidal signal');
19 -    disp('6-sinc function');
20 -    disp('7-triangle pulse');
```

```
for i = 1: Breakpoints + 1
    signal_type = input(['Enter the number of the type of signal (' num2str(i) '): ']);

    t = linspace(z(i),z(i+1),(z(i+1)-z(i))*Fs);

    switch signal_type
        case 1
            amplitude = input('Enter the amplitude: ');
             y = amplitude*ones(1,(z(i+1)- z(i))*Fs);

        case 2
            slope = input('Enter the slope: ');
            intercept = input('Enter the intercept: ');
            y = slope*t + intercept;

        case 3
            amplitude = input('Enter the amplitude: ');
            power = input('Enter the power: ');
            intercept = input('Enter the intercept: ');
             y = amplitude*(t.^power)+ intercept;

        case 4
            amplitude = input('Enter the amplitude: ');
            exponent = input('Enter the exponent: ');
            y = amplitude*exp(exponent*t);
        case 5
             amplitude = input('Enter the amplitude: ');
            frequency = input('Enter the frequency: ');
            phase = input('Enter the phase: ');
            y = amplitude*sin((2*pi*frequency*t)+phase);
        case 6
            amplitude = input('Enter the amplitude: ');
            center_shift = input('Enter the center shift: ');
             y =amplitude * sinc(t - center_shift);
        case 7
            amplitude = input('Enter the amplitude: ');
            center_shift = input('Enter the center shift: ');
            width = input('Enter the width: ');
            y = amplitude * tripuls(t- center_shift, width);

        otherwise
            error('Invalid signal type');
    end
        Y = [Y y] ;
end
```

Output

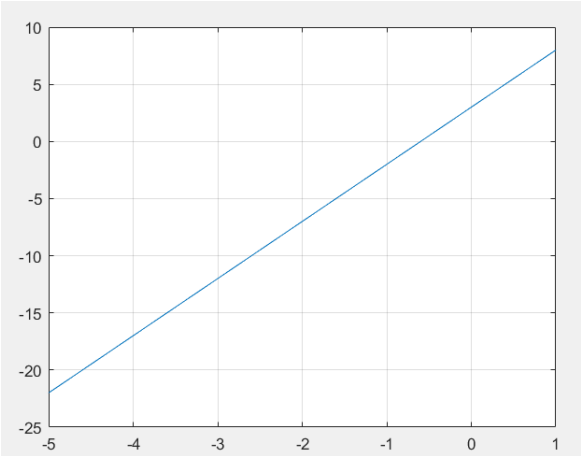
DC signal: $y(t) = 20$

```
Command Window
>> Mini_Project
Enter the sampling frequency of the signal:100
Enter the start time of the time scale:-10
Enter the end time of the time scale:-5
Enter the number of breakpoints: 0
1-DC signal
2-Ramp signal
3-General order polynomial
4-Exponential signal
5-sinusoidal signal
6-sinc function
7-triangle pulse
Enter the number of the type of signal (1): 1
Enter the amplitude: 20
```



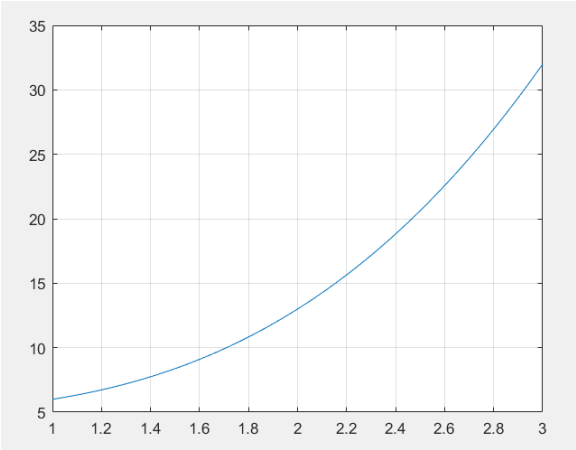
Ramp signal: $y(t) = 5t + 3$

```
Command Window
>> Mini_Project
Enter the sampling frequency of the signal:100
Enter the start time of the time scale:-5
Enter the end time of the time scale:1
Enter the number of breakpoints: 0
1-DC signal
2-Ramp signal
3-General order polynomial
4-Exponential signal
5-sinusoidal signal
6-sinc function
7-triangle pulse
Enter the number of the type of signal (1): 2
Enter the slope: 5
Enter the intercept: 3
```



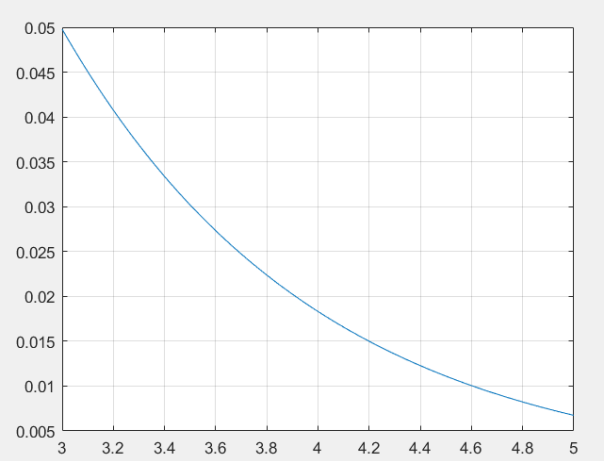
General order polynomial: $y(t) = t^3 + 5$

```
Command Window
>> Mini_Project
Enter the sampling frequency of the signal:100
Enter the start time of the time scale:1
Enter the end time of the time scale:3
Enter the number of breakpoints: 0
1-DC signal
2-Ramp signal
3-General order polynomial
4-Exponential signal
5-sinusoidal signal
6-sinc function
7-triangle pulse
Enter the number of the type of signal (1): 3
Enter the amplitude: 1
Enter the power: 3
Enter the intercept: 5
```



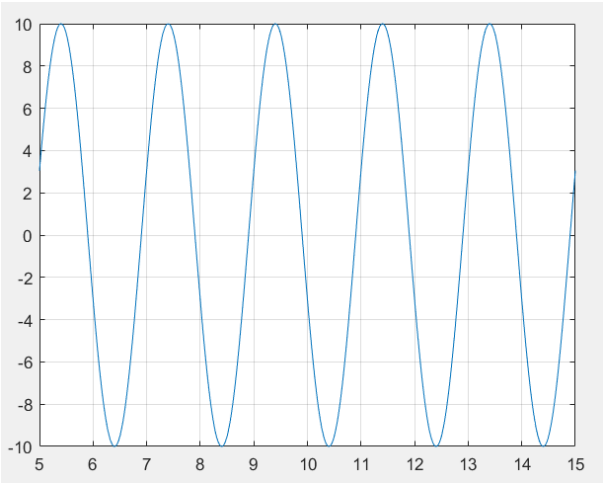
Exponential signal: $y(t) = e^{-t}$

```
Command Window
>> Mini_Project
Enter the sampling frequency of the signal:100
Enter the start time of the time scale:3
Enter the end time of the time scale:5
Enter the number of breakpoints: 0
1-DC signal
2-Ramp signal
3-General order polynomial
4-Exponential signal
5-sinusoidal signal
6-sinc function
7-triangle pulse
Enter the number of the type of signal (1): 4
Enter the amplitude: 1
Enter the exponent: -1
```



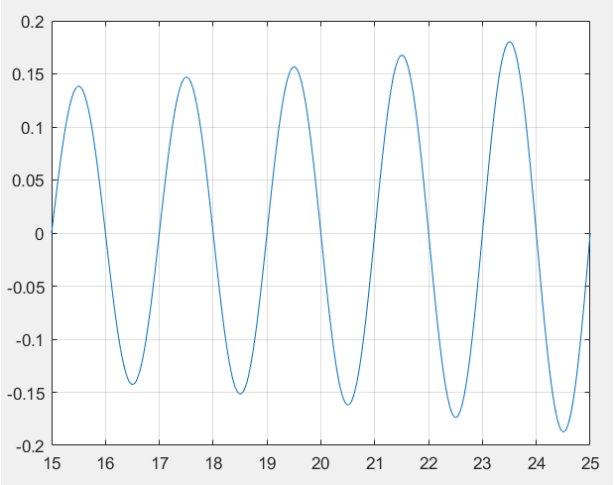
Sinusoidal signal: $y(t) = 10\sin(0.5 * 2\pi + 60)$

```
Command Window
>> Mini_Project
Enter the sampling frequency of the signal:100
Enter the start time of the time scale:5
Enter the end time of the time scale:15
Enter the number of breakpoints: 0
1-DC signal
2-Ramp signal
3-General order polynomial
4-Exponential signal
5-sinusoidal signal
6-sinc function
7-triangle pulse
Enter the number of the type of signal (1): 5
Enter the amplitude: 10
Enter the frequency: 0.5
Enter the phase: 60
```



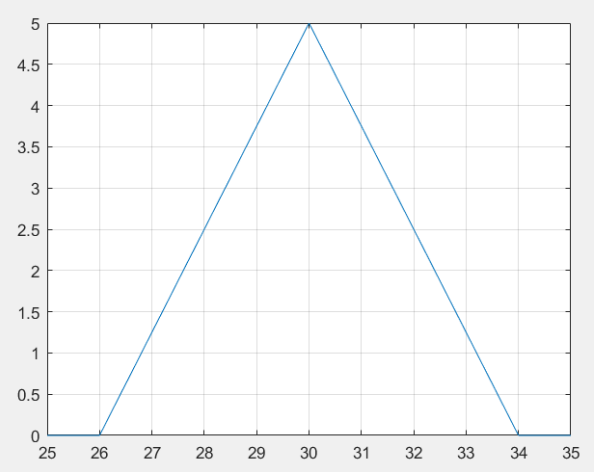
Sinc function: $y(t) = 15sinc(t - 50)$

```
Command Window
>> Mini_Project
Enter the sampling frequency of the signal:100
Enter the start time of the time scale:15
Enter the end time of the time scale:25
Enter the number of breakpoints: 0
1-DC signal
2-Ramp signal
3-General order polynomial
4-Exponential signal
5-sinusoidal signal
6-sinc function
7-triangle pulse
Enter the number of the type of signal (1): 6
Enter the amplitude: 15
Enter the center shift: 50
```



Triangle pulse: $y(t) = 5tripuls(t - 30,4)$

```
Command Window
>> Mini_Project
Enter the sampling frequency of the signal:100
Enter the start time of the time scale:25
Enter the end time of the time scale:35
Enter the number of breakpoints: 0
1-DC signal
2-Ramp signal
3-General order polynomial
4-Exponential signal
5-sinusoidal signal
6-sinc function
7-triangle pulse
Enter the number of the type of signal (1): 7
Enter the amplitude: 5
Enter the center shift: 30
Enter the width: 4
```



Total Function:

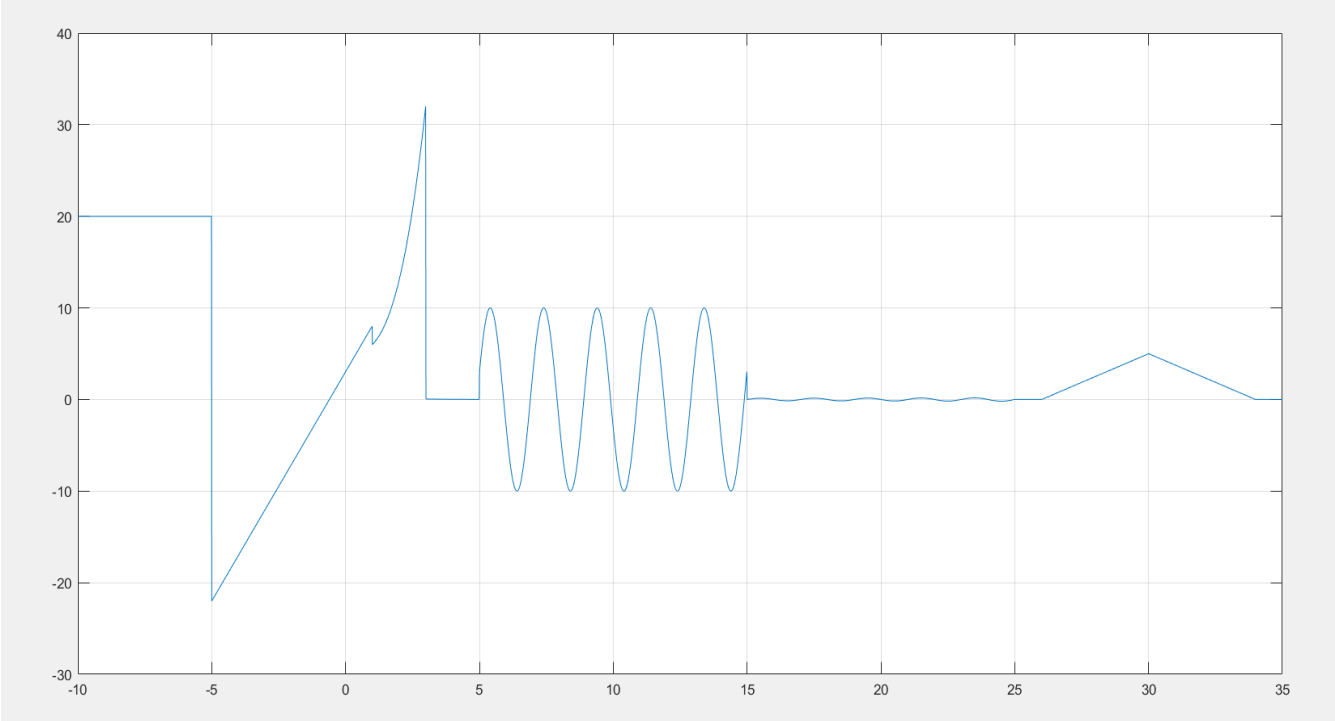
$$y(t) = \begin{cases} 20 & -10 \le t < -5 \\ 5t + 3 & -5 \le t < 1 \\ t^3 + 5 & 1 \le t < 3 \\ e^{-t} & 3 \le t < 5 \\ 10\sin(0.5 * 2\pi + 60) & 5 \le t < 15 \\ 15sinc(t - 50) & 15 \le t < 25 \\ 5tripuls(t - 30,4) & 25 \le t < 35 \end{cases}$$

User interface:

Command Window

```
>> Mini_Project
Enter the sampling frequency of the signal:100
Enter the start time of the time scale:-10
Enter the end time of the time scale:35
Enter the number of breakpoints: 6
Enter position of breakpoint (1):-5
Enter position of breakpoint (2):1
Enter position of breakpoint (3):3
Enter position of breakpoint (4):5
Enter position of breakpoint (5):15
Enter position of breakpoint (6):25
1-DC signal
2-Ramp signal
3-General order polynomial
4-Exponential signal
5-sinusoidal signal
6-sinc function
7-triangle pulse
Enter the number of the type of signal (1): 1
Enter the amplitude: 20
Enter the number of the type of signal (2): 2
Enter the slope: 5
Enter the intercept: 3
Enter the number of the type of signal (3): 3
Enter the amplitude: 1
Enter the power: 3
Enter the intercept: 5
Enter the number of the type of signal (4): 4
Enter the amplitude: 1
Enter the exponent: -1
Enter the number of the type of signal (5): 5
Enter the amplitude: 10
Enter the frequency: 0.5
Enter the phase: 60
Enter the number of the type of signal (6): 6
Enter the amplitude: 15
Enter the center shift: 50
Enter the number of the type of signal (7): 7
Enter the amplitude: 5
Enter the center shift: 30
Enter the width: 4
```

Output signal:



3) The user is asked to enter the number(s) of the operation(s) to be performed on the signal
   a) **Amplitude Scaling: scale value:** involves adjusting the magnitude of a signal without changing its shape or waveform
      Y(t)=a*y(t)
      a is the scale value
   b) **Time reversal**: where the temporal order of a signal's samples or events is reversed
      Y(T)=y(−t) is the time-reversed counterpart
   c) **Time shift: shift value:** refers to the displacement of a signal either forwards or backwards along the time axis
      $Y(T)=y(t-t_o)$
      t_o is the shift value, can be positive (moving to the right) or negative (moving to the left).
   d) **Expanding & compressing the signal:** the duration of the signal is increased. This operation involves stretching the signal along the time axis
      Y(T)=y(αt)

α can be greater than 1 (compressing the signal), or smaller than 1 (expanding the signal).

e) **Clipping the signal: upper and Lower clipping values:** Clipping the signal involves limiting the amplitude of a signal to a specified range.

$$Y(T)=\begin{cases} I_{upper} & \text{if } y(t) > I\text{upper} \\ I_{lower} & \text{if } y(t) < I\text{lower} \end{cases}$$

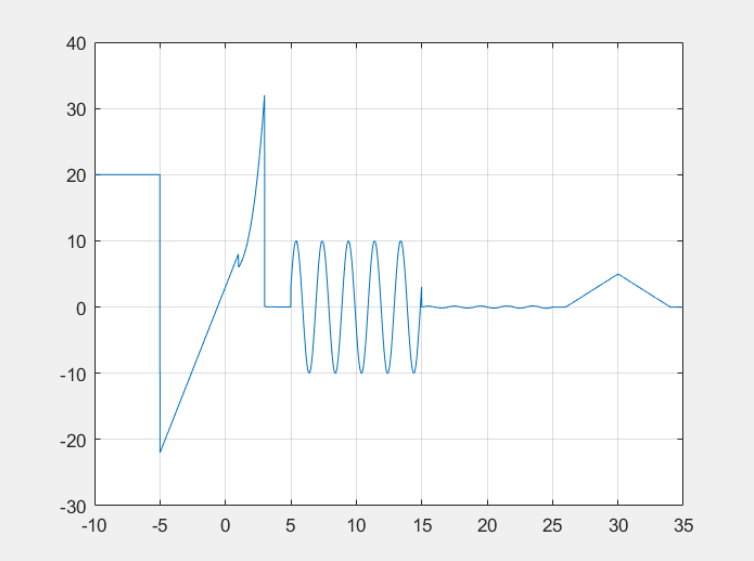
f) **The first derivative of the signal:** it represents the rate of change of the signal with respect to time

$$Y(T)=\frac{1}{Ts} * \text{diff(y)}$$

```
while true
    disp('1-amplitude Scaling');
    disp('2-time reversal');
    disp('3-time shift');
    disp('4-time Scale (Expanding & Compressing the signal)');
    disp('5-clipping the signal');
    disp('6-the first derivative of the signal');
    disp('7-None');


    operation_numbers =str2num( input('Enter the number(s) of the operation(s) to be performed on the signal

    for operation = operation_numbers
     switch operation
        case 1
            Amplitude_scaling = input('Enter the scale value: ');
            Y = Y*Amplitude_scaling;
        case 2
            T = T*-1;
        case 3
            Time_shift = input('Enter the shift value: ');
              T = T+Time_shift;
        case 4
            scaling_value = input('Enter the scaling value: ');
            T = T/scaling_value;
        case 5
            Upper = input('upper clipping value : ');
            Lower = input('lower clipping value : ');
            IU=find(Y>Upper);
            Y(IU)= Upper;
            IL=find(Y<Lower);
            Y(IL)= Lower;
        case 6
            Y = Fs*diff(Y);
            T = T(1:end-1);

     end
    end


figure;
plot(T, Y)
grid on

    if operation_numbers == 7
        break;
    end

    if any(operation_numbers) > 7 || any(operation_numbers) <1
        error('Invalid operation');
    end
end
```

Output

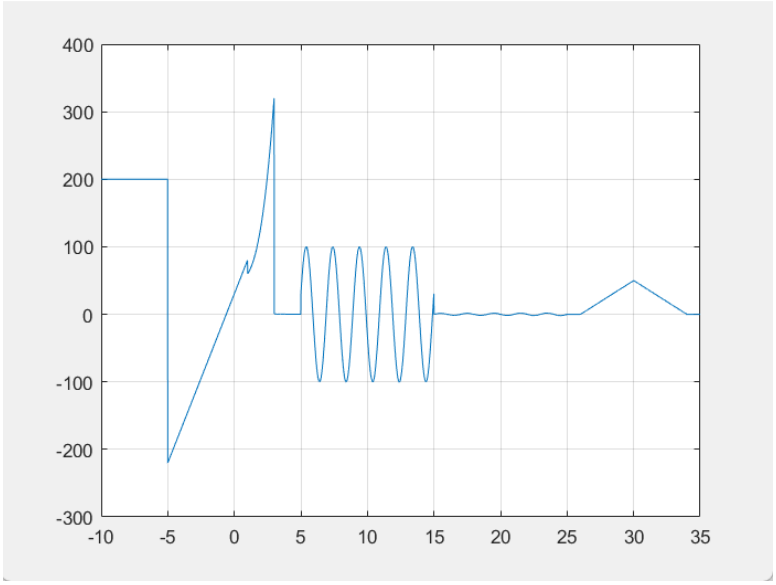
Original signal:



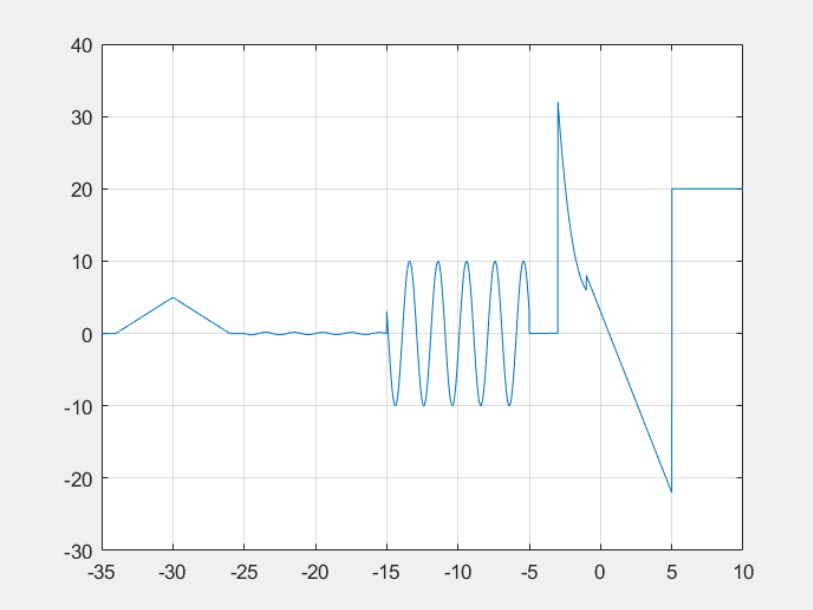
Output signal after performing each of the operations:

1.Amplitude Scaling: Y(t)=10*y(t)

```
1-amplitude Scaling
2-time reversal
3-time shift
4-time Scale (Expanding & Compressing the signal)
5-clipping the signal
6-the first derivative of the signal
7-None
Enter the number(s) of the operation(s) to be performed on the signal (separated by spaces): 1
Enter the scale value: 10
```

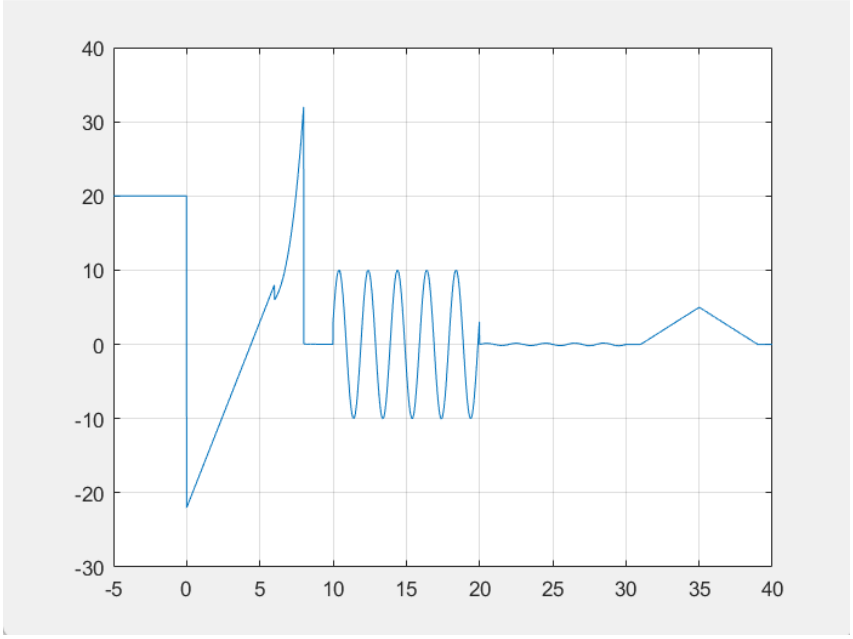


2.Time reversal: Y(T)=y(−t)



3.Time shift: Y(T)=y(t−5)

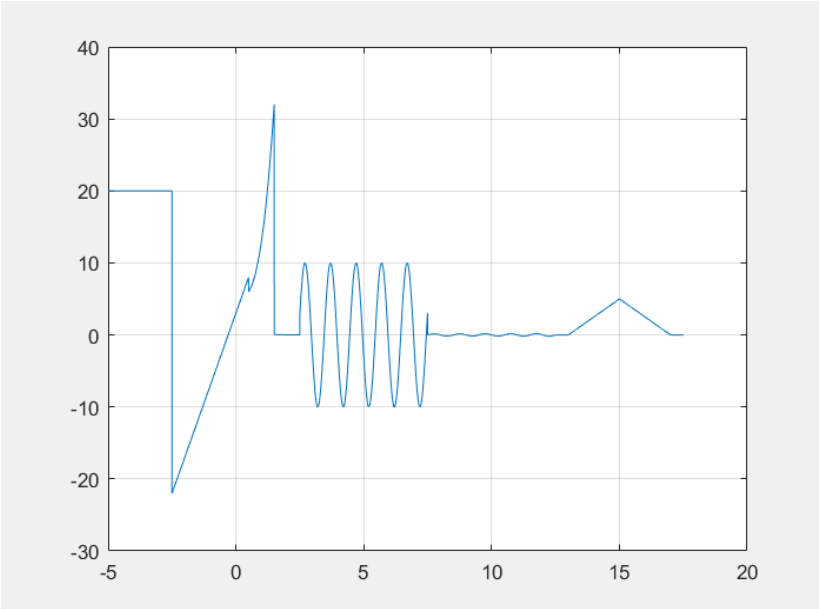
```
Enter the number(s) of the operation(s) to be performed on the signal (separated by spaces): 3
Enter the shift value: 5
```



4.Expanding & compressing the signal:

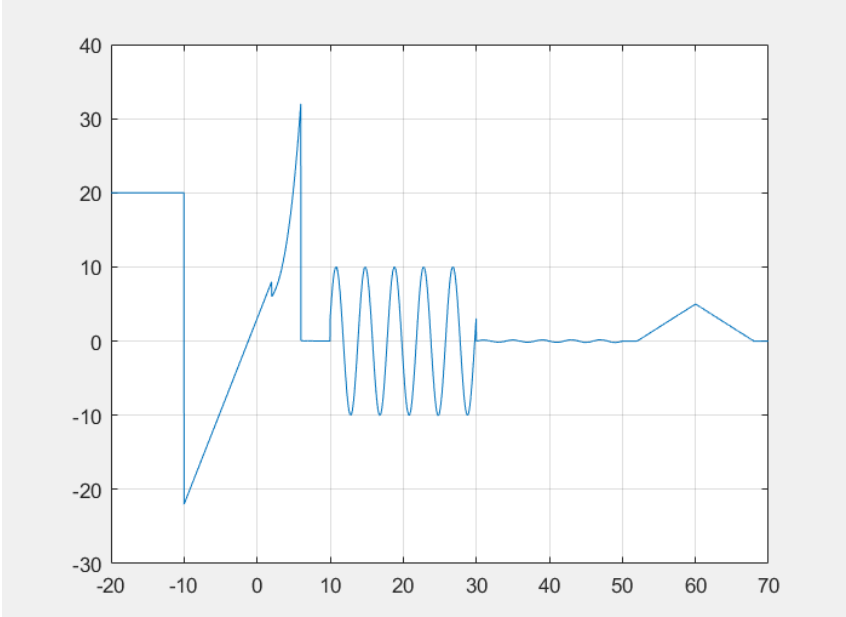
1st: Compression: Y(T)=y(2t)

```
Enter the number of the operation to be perform on the signal4
Enter the scaling value:  2
```



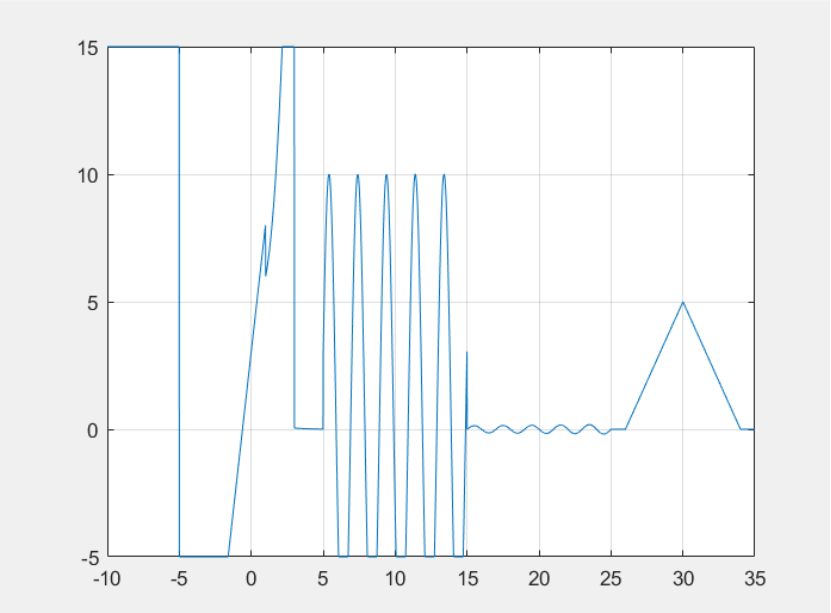
2nd: Expansion: Y(T)=y(0.5t)

```
Enter the number of the operation to be perform on the signal4
Enter the scaling value:  0.5
```

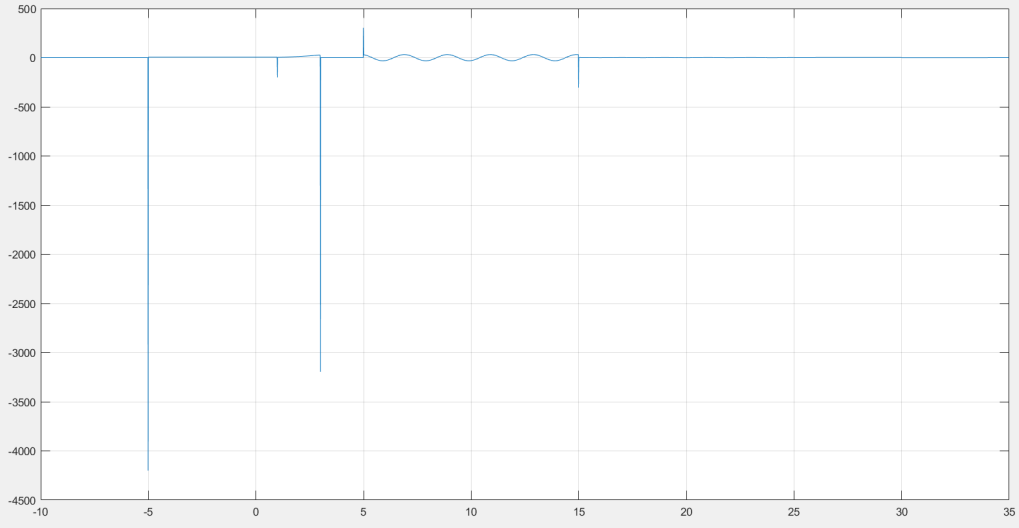


5.Clipping the signal: Y(T)=$\begin{Bmatrix} 15 & \text{if } y(t) > 15 \\ -5 & \text{if } y(t) < -5 \end{Bmatrix}$

```
Enter the number of the operation to be perform on the signal5
upper clipping value :  15
lower clipping value :  -5
```



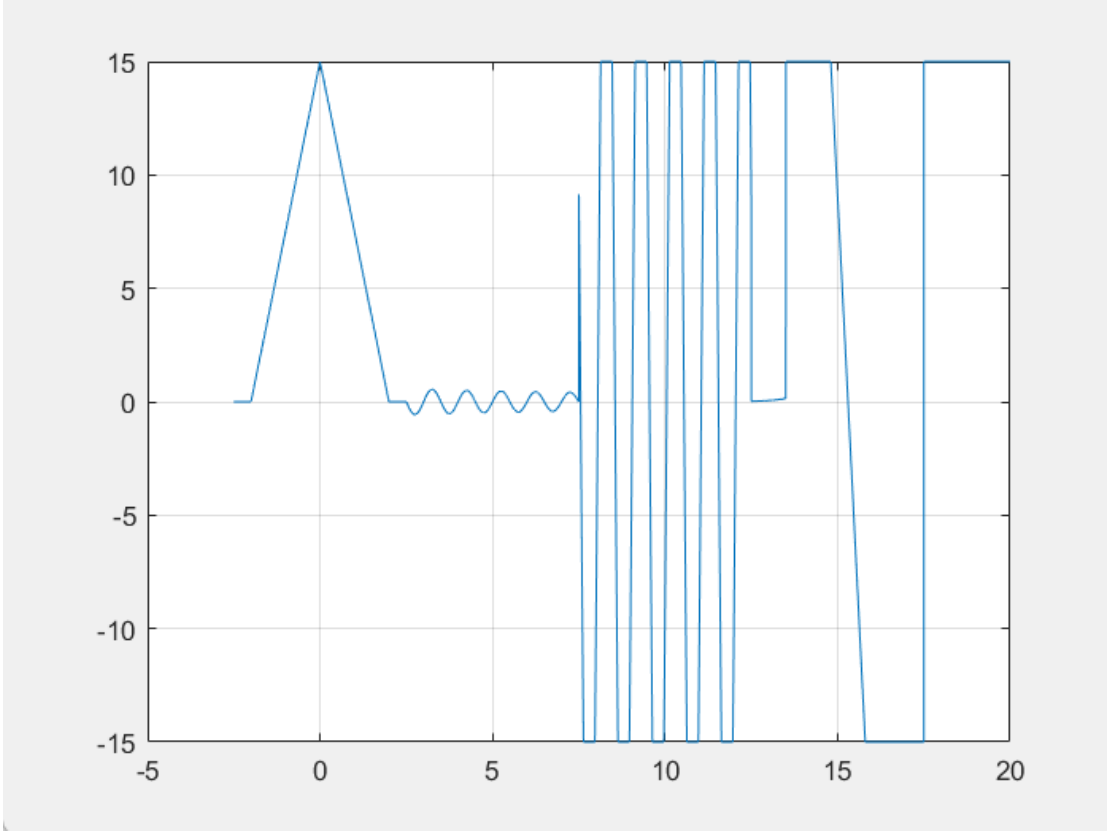
6.The first derivative of the signal: Y(T)= $\frac{1}{Ts} * \text{diff(y)}$



7. performing more than one operation on the signal simultaneously:

$$Y(T)=\begin{cases} 15 & \text{if } y(t) > 15 \\ Y(T) = 3 * y\left(-(2t-5)\right) & \text{if } -5 < y(t) < 15 \\ -15 & \text{if } y(t) < -5 \end{cases}$$

```
Enter the number(s) of the operation(s) to be performed on the signal (separated by spaces): 1 2 3 4 5
Enter the scale value: 3
Enter the shift value: 5
Enter the scaling value:  2
upper clipping value :  15
lower clipping value :  -15
```



The Code

```
Fs = input('Enter the sampling frequency of the signal:');
Start_time = input('Enter the start time of the time scale:');
End_time = input('Enter the end time of the time scale:');
Breakpoints = input('Enter the number of breakpoints: ');

if (Breakpoints == 0)
    Position = [];
else
   for i = 1:Breakpoints
    Position(i) = input(['Enter position of breakpoint (', num2str(i), '):' ]);
   end
end

disp('1-DC signal');
disp('2-Ramp signal');
disp('3-General order polynomial');
disp('4-Exponential signal');
disp('5-sinusoidal signal');
disp('6-sinc function');
disp('7-triangle pulse');

T = linspace(Start_time,End_time,(End_time-Start_time)*Fs);
Y = [];

z = [Start_time Position End_time];

for i = 1: Breakpoints + 1
    signal_type = input(['Enter the number of the type of signal (' num2str(i) '): ']);

    t = linspace(z(i),z(i+1),(z(i+1)-z(i))*Fs);

    switch signal_type
        case 1
            amplitude = input('Enter the amplitude: ');
             y = amplitude*ones(1,(z(i+1)- z(i))*Fs);

        case 2
            slope = input('Enter the slope: ');
            intercept = input('Enter the intercept: ');
            y = slope*t + intercept;

        case 3
            amplitude = input('Enter the amplitude: ');
            power = input('Enter the power: ');
            intercept = input('Enter the intercept: ');
             y = amplitude*(t.^power)+ intercept;

        case 4
            amplitude = input('Enter the amplitude: ');
            exponent = input('Enter the exponent: ');
            y = amplitude*exp(exponent*t);
        case 5
             amplitude = input('Enter the amplitude: ');
            frequency = input('Enter the frequency: ');
            phase = input('Enter the phase: ');
             y = amplitude*sin((2*pi*frequency*t)+phase);
        case 6
            amplitude = input('Enter the amplitude: ');
            center_shift = input('Enter the center shift: ');
             y =amplitude * sinc(t - center_shift);
        case 7
            amplitude = input('Enter the amplitude: ');
            center_shift = input('Enter the center shift: ');
            width = input('Enter the width: ');
            y = amplitude * tripuls(t- center_shift, width);

        otherwise
            error('Invalid signal type');
    end
        Y = [Y y] ;
end

figure;
plot(T,Y)
grid on

while true
    disp('1-amplitude Scaling');
    disp('2-time reversal');
    disp('3-time shift');
    disp('4-time Scale (Expanding & Compressing the signal)');
    disp('5-clipping the signal');
    disp('6-the first derivative of the signal');
    disp('7-None');


    operation_numbers =str2num( input('Enter the number(s) of the operation(s) to be performed on the
signal (separated by spaces): ', 's'));

    for operation = operation_numbers
     switch operation
        case 1
            Amplitude_scaling = input('Enter the scale value: ');
            Y = Y*Amplitude_scaling;
        case 2
            T = T*-1;
        case 3
            Time_shift = input('Enter the shift value: ');
             T = T+Time_shift;
        case 4
            scaling_value = input('Enter the scaling value: ');
            T = T/scaling_value;
        case 5
            Upper = input('upper clipping value : ');
            Lower = input('lower clipping value : ');
            IU=find(Y>Upper);
            Y(IU)= Upper;
            IL=find(Y<Lower);
            Y(IL)= Lower;
        case 6
            Y = Fs*diff(Y);
            T = T(1:end-1);

     end
    end


figure;
plot(T, Y)
grid on

    if operation_numbers == 7
        break;
    end

    if any(operation_numbers) > 7 || any(operation_numbers) <1
        error('Invalid operation');
    end
end
```