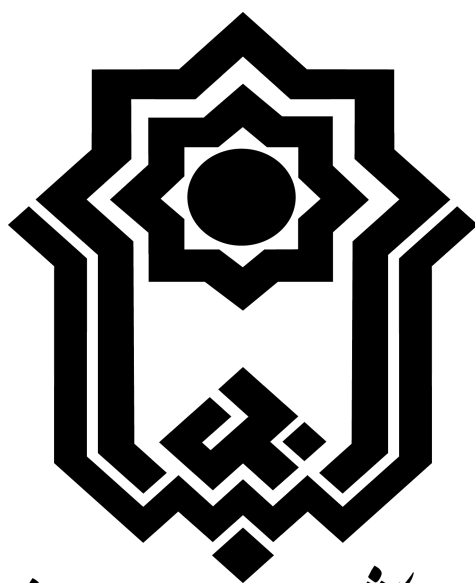


به نام خدا



دانشگاه بوعلی سینا

نام و نام خانوادگی: سید فرهاد حسینی

شماره دانشجویی: ۹۶۱۲۳۵۸۰۱۶

نام درس: سیگنال ها و سیستم ها

استاد مربوطه: دکتر مهدی عباسی

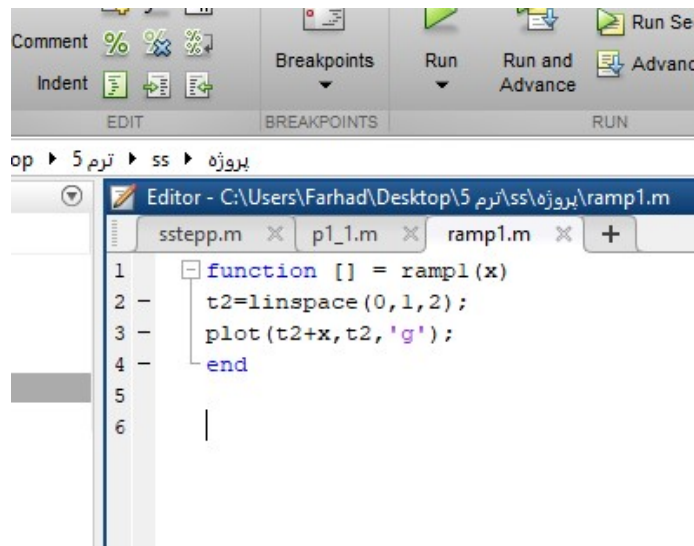
۳	پاسخ سوال ۱ :	•
۳	بخش اول سوال ۱ :	✓
۱۱	بخش دوم سوال ۱ :	✓
۱۲	بخش سوم سوال ۱ :	✓
۱۳	بخش چهارم سوال ۱ :	✓
۱۴	پاسخ سوال ۲ :	•
۱۴	بخش اول سوال ۲ :	✓
۱۸	بخش دوم سوال ۲ :	✓
۱۹	بخش سوم سوال ۲ :	✓
۲۱	پاسخ سوال ۳ :	•
۲۱	بخش اول سوال ۳ :	✓
۲۱	بخش دوم سوال ۳ :	✓
۲۱	بخش سوم سوال ۳ :	✓
۲۱	پاسخ سوال ۴ :	•
۲۱	بخش اول سوال ۴ :	✓
۲۱	بخش دوم سوال ۴ :	✓
۲۲	پاسخ سوال ۵ :	•
۲۲	بخش اول سوال ۵ :	✓
۲۲	بخش دوم سوال ۵ :	✓

## • پاسخ سوال ۱ :

### ✓ بخش اول سوال ۱ :

در بخش اول این سوال بایستی سیگنال های رمپ و پله واحد را رسم کنیم . روند کاری پروژه تماما با نرم افزار متلب میباشد . در تصویر زیر کد نوشته شده برای رسم نمودار رمپ میباشد . ورودی تابع مقدار شیفت زمانی را تعیین میکند .

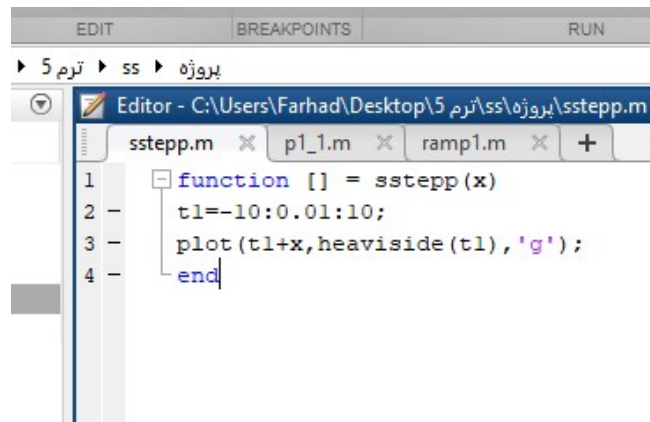
نکته : ورژن دوم سیگنال های پله و رمپ به ترتیب در فایل های `u_step.m` و `ramp2.m` نوشته شده اند .



```
1 function [] = ramp1(x)
2     t2= linspace(0,1,2);
3     plot(t2+x,t2,'g');
4     end
5
6
```

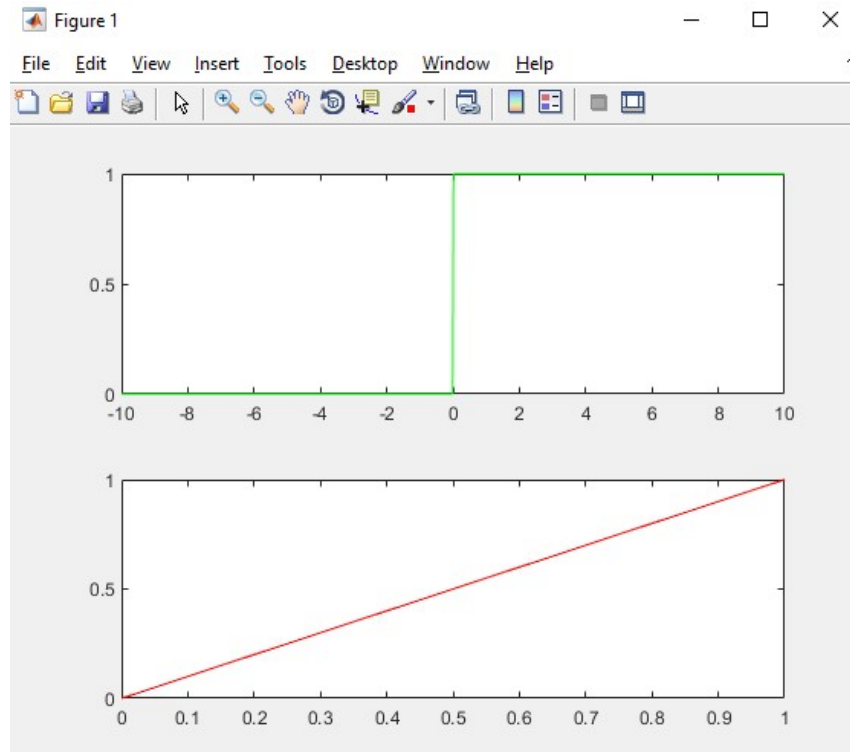
در تصویر بالا `t2` همان محور افقی (زمان) میباشد . این تابع در فایل با نام `ramp1.m` ذخیره شده است.

برای رسم سیگنال پله واحد نیز برای آن تابعی در نظر میگیریم . این تابع نیز تنها یک ورودی دارد و این ورودی شیفت زمانی را مشخص میکند . تصویر صفحه بعد کد این تابع را نشان میدهد .



```
EDIT BREAKPOINTS RUN
پروژه ss ترم 5
Editor - C:\Users\Farhad\Desktop\ترم 5\پروژه\ss\sstepp.m
sstepp.m x p1_1.m x ramp1.m x +
1 function [] = sstepp(x)
2 t1=-10:0.01:10;
3 plot(t1+x,heaviside(t1),'g');
4 end
```

این تابع نیز در فایل جداگانه با نام `sstepp.m` ذخیره شده است. در عکس پایین دو نمودار نمایش داده شده . نمودار سبز رنگ سیگنال پله و نمودار قرمز رنگ سیگنال رمپ میباشد(فایل کد `p1_1.m` میباشد). در این مثال مقدار شیفت برای هر دو نمودار صفر در نظر گرفته شده است.



نکته : ورژن دوم سیگنال های پله و رمپ بترتیب در فایل های `u_step.m` و `ramp2.m` نوشته شده اند .

کد ورژن دوم این توابع را در دو شکل زیر میبینید .

```

Editor - C:\Users\ramad\Desktop\پروژه درس سیگنال ها و سیستم ها\u_step.m
u_step.m x ramp2.m x p1_2.m x p1_3.m x p1_4.m x +
1 function [out1]=u_step(t0)
2     t=-10:0.01:10;
3     out1= 1 .* (t>=t0) + 0 .* (t<t0);
4     end
5

```

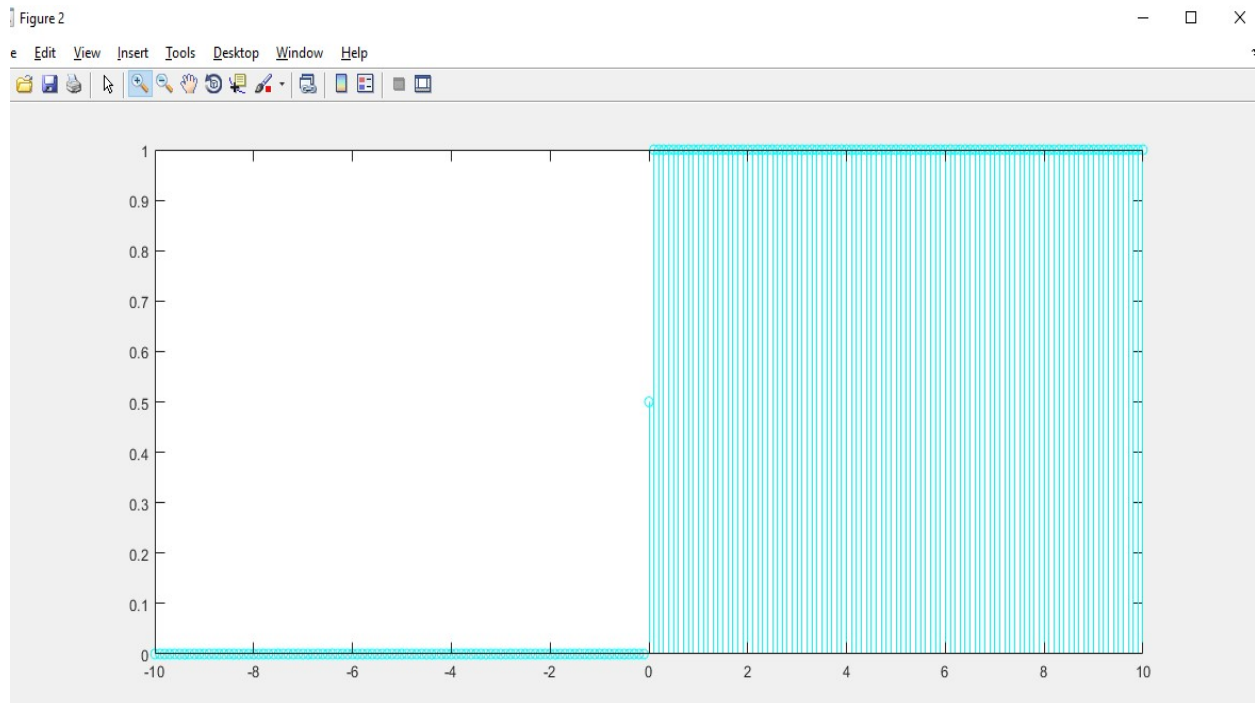
```

Editor - C:\Users\Farnad\Desktop\پروژه\ss\برم\p1_2.m
u_step.m x ramp2.m x p1_2.m x p1_3.m x p1_4.m x +
1 function [out1]=ramp2(t0)
2   t=-10:0.01:10;
3   out1= (t-t0) .* (t>=t0 & t<t0+1) + 0 .* (t<=t0 & t>t0+1);
4   end
5

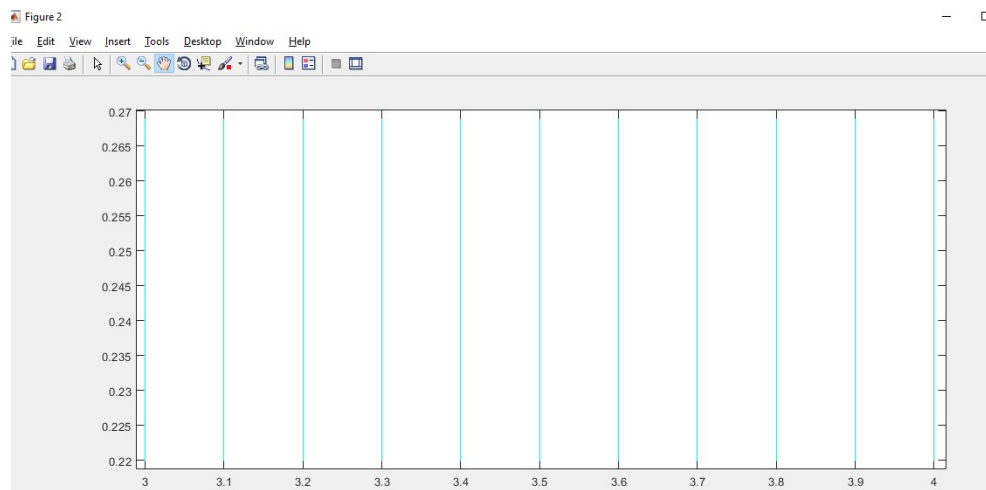
```

در قسمت آخر اولین بخش از پروژه خواسته شده که از سیگنال های پله و رمپ با فرکانس های ۱۰ و ۱۰۰ نمونه برداری شود (یعنی در هر ثانیه ۱۰ یا ۱۰۰ نمونه). چون سیگنال ها بعد از نمونه برداری گسسته میشوند برای رسم در متلب از تابع **stem** استفاده میکنیم.

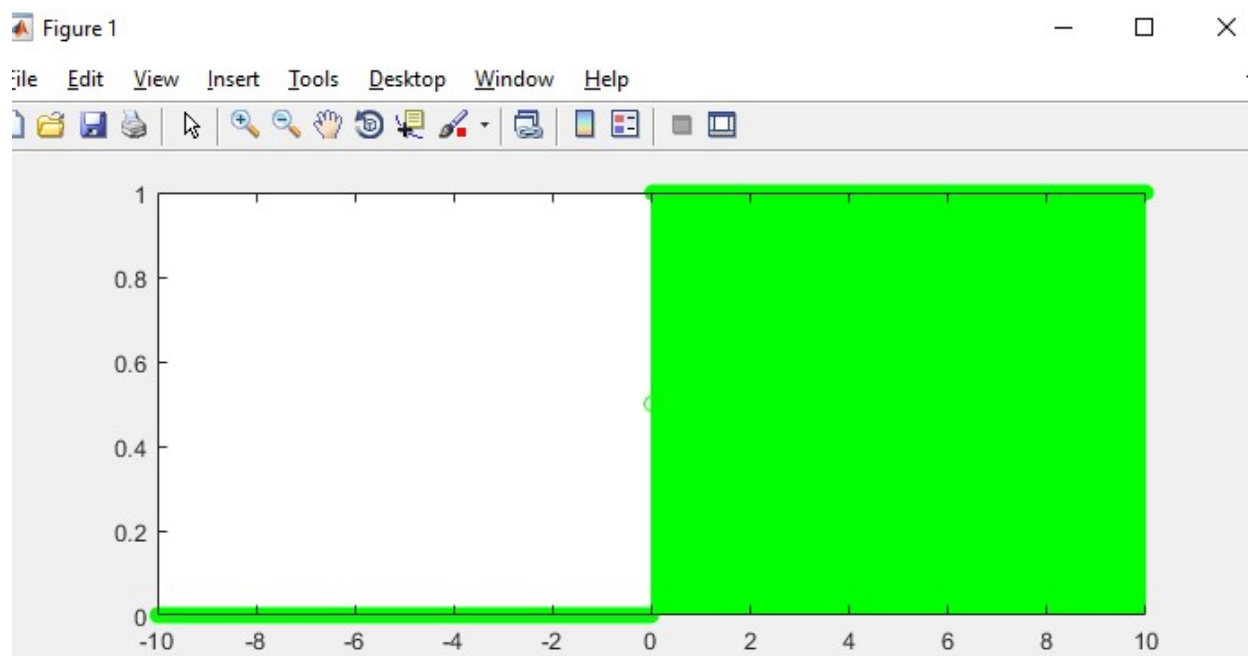
در تصویر زیر نمونه برداری از سیگنال پله واحد با فرکانس ۱۰ انجام شده است.



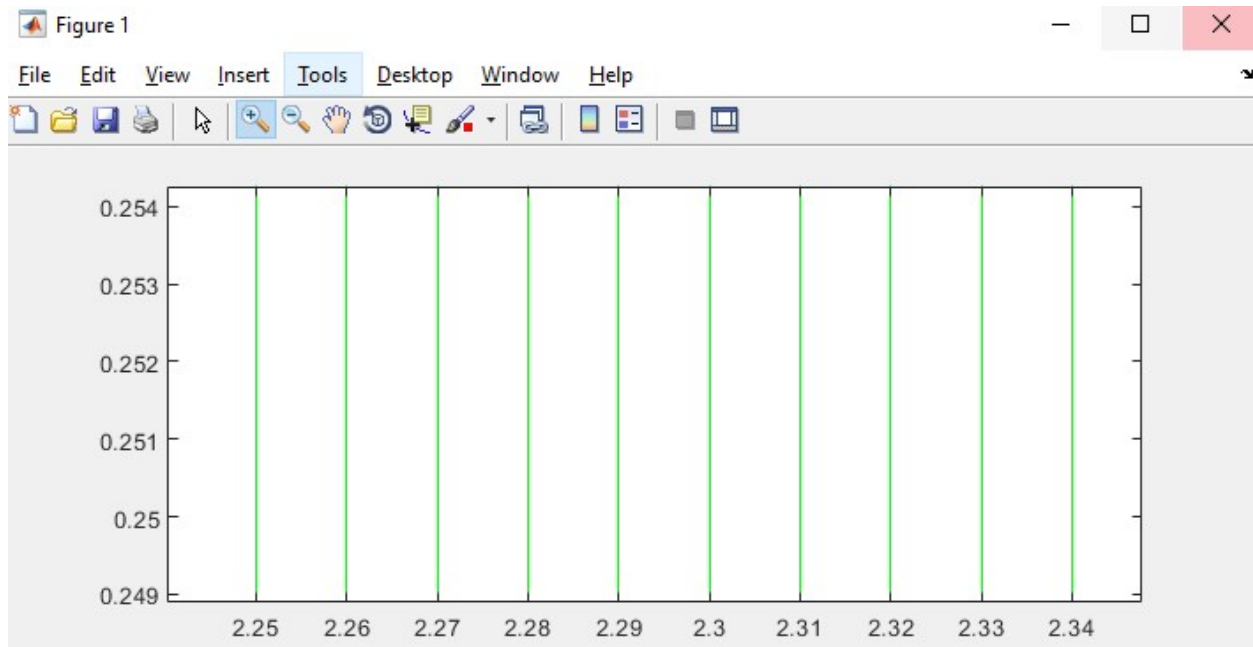
برای اینکه مشخص شود که در هر ثانیه ۱۰ نمونه گرفته شده تصویر را بزرگنمایی میکنیم.



تصوی زیر نشاندهنده نمونه برداری از سیگنال پله با فرکانس ۱۰۰ میباشد



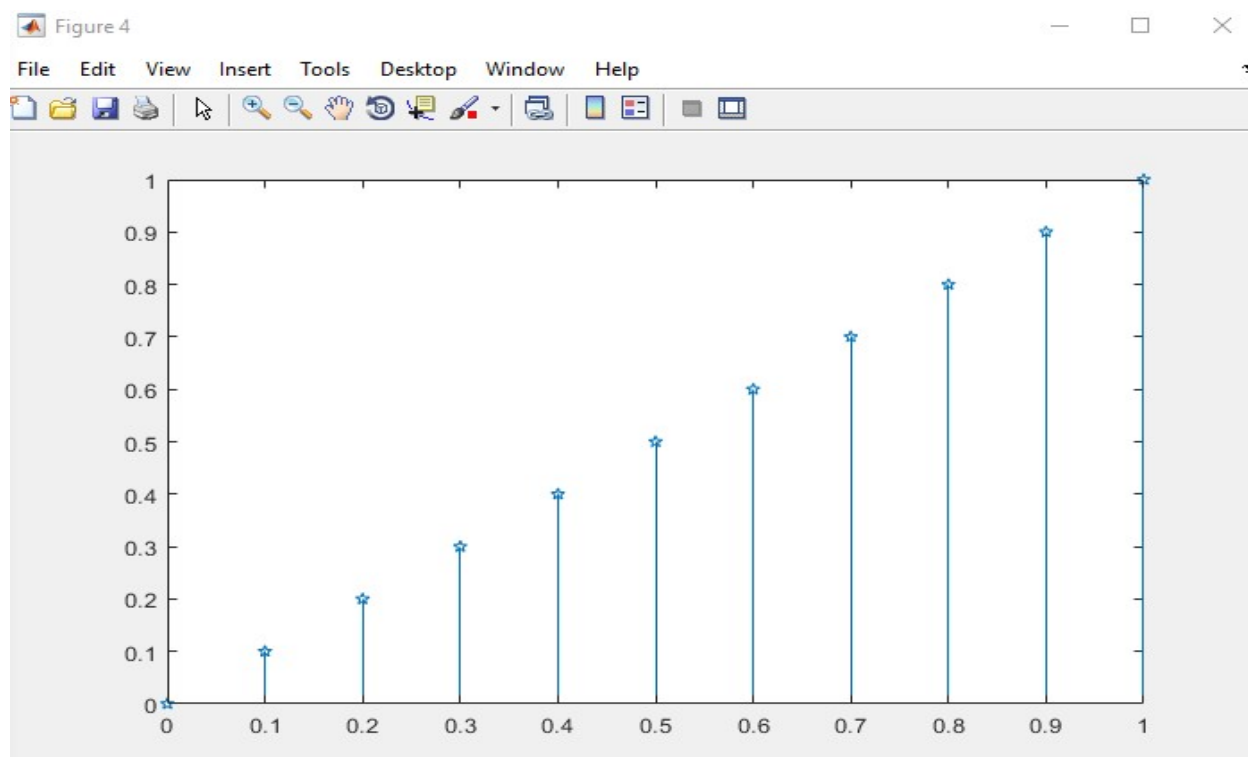
برای اینکه مشخص شود که در هر ثانیه ۱۰۰ نمونه گرفته شده تصویر را بزرگنمایی میکنیم.



همانطور که در تصویر صفحه قبل مشخص است از سیگنال پله در هر  $0.01$  ثانیه یه نمونه گرفته شده است به عبارت دیگر فرکانس نمونه برداری  $100$  میباشد.

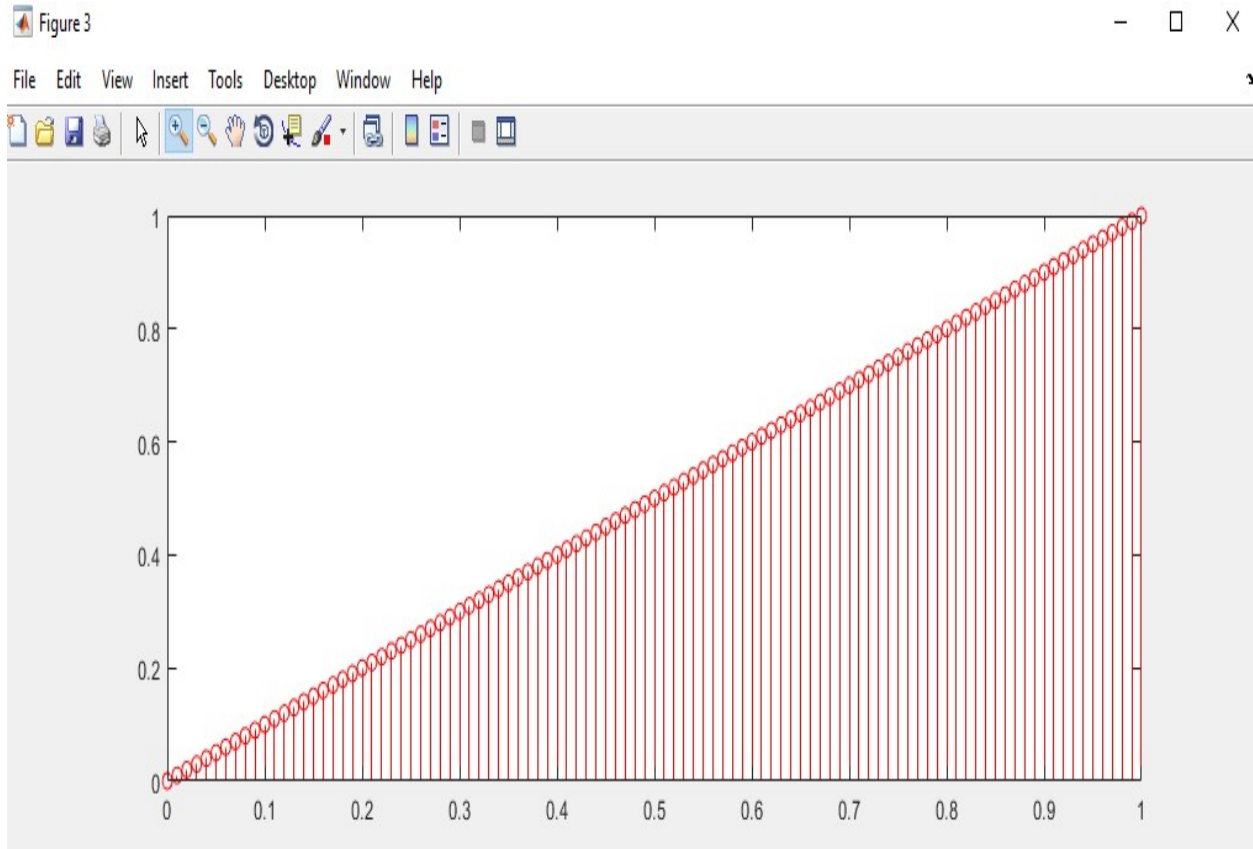
برای سیگنال رمپ نیز همانند سیگنال پله عمل میکنیم. در تصویر زیر نمونه برداری از سیگنال رمپ با فرکانس  $10$  انجام شده است.





همانطور که در تصویر مشخص است هر ۰,۱ ثانیه یکبار یک نمونه برداری انجام میشود .

عکس زیر نیز نمونه برداری از سیگنال رمپ با فرکانس ۱۰۰ می باشد .



کدهای مربوط به این بخش (نمونه برداری از سیگنال های رمپ و پله) در فایل با نام `sampling.m` می باشد.

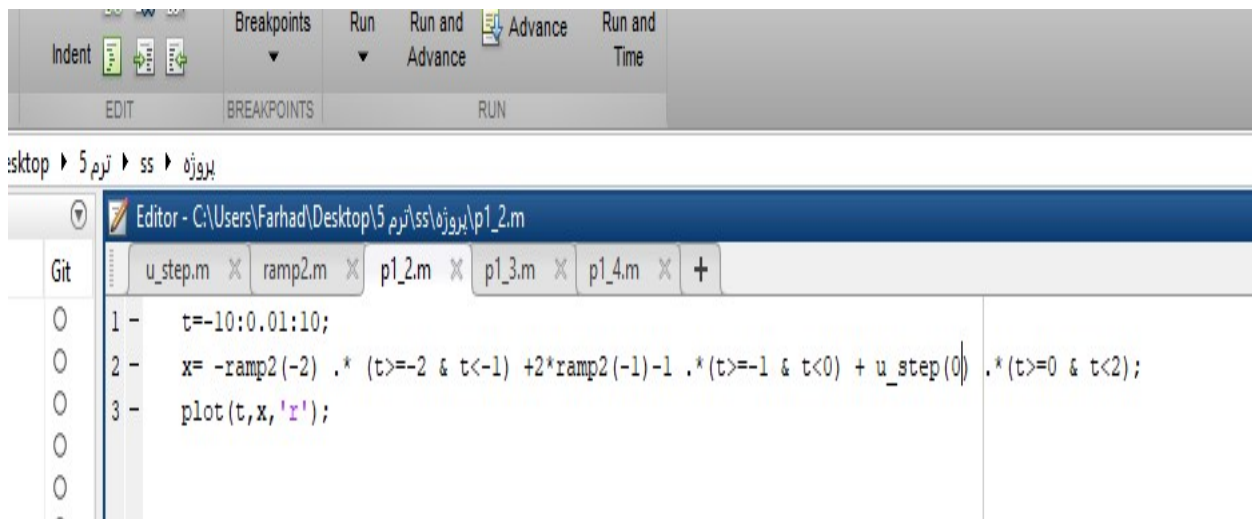
## ✓ بخش دوم سوال ۱ :

برای انجام بخش های ۲ و ۳ و ۴ از سوال ۱ توابع رمپ و پله برای سادگی کار کمی یافته اند. تغییر یافته ی آنها در فایل های `u_step.m` و `ramp2.m` میباشند.

تابع  $x(t)$  از ۵ قسمت تشکیل شده است اگر بخواهیم این سیگنال را بصورت دستی با استفاده از سیگنال های رمپ و پله پیاده سازی کنیم به شکل زیر در می آید.

$$X(t) = -r(t+2) + 2 \cdot r(t+1) - 1 + u(t) - u(t-2)$$

پیاده سازی این سیگنال در نرم افزار متلب بطور زیر میباشد.

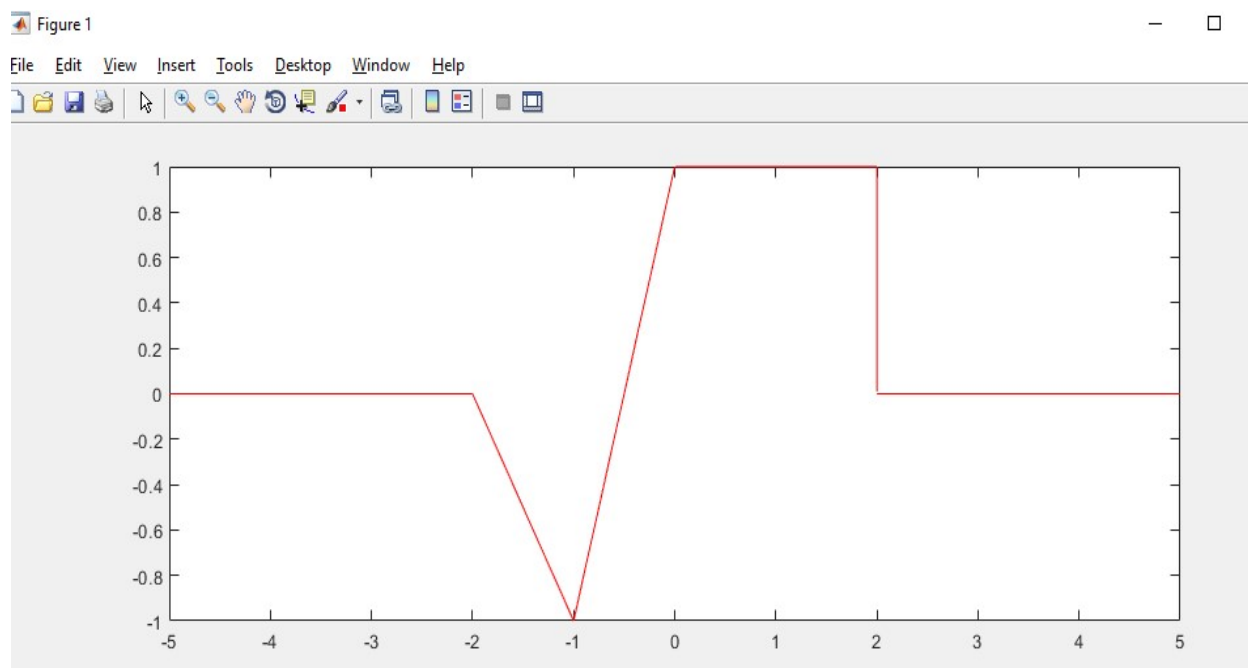


```

Editor - C:\Users\Farhad\Desktop\5\ترم 5\پروژه\ss\p1_2.m
u_step.m x ramp2.m x p1_2.m x p1_3.m x p1_4.m x +
1 - t=-10:0.01:10;
2 - x= -ramp2(-2) .* (t>=-2 & t<=-1) +2*ramp2(-1)-1 .* (t>=-1 & t<0) + u_step(0) .* (t>=0 & t<2);
3 - plot(t,x,'r');
  
```

روش رسم شکل بدین گونه است که شکل به ۳ قسمت تقسیم شده و هر یک از این قسمت ها رسم شده و همه این شکل ها در یک صفحه مختصات نگهداری شده اند.

اجرا این کد و رسم شکل در متلب بصورت زیر است.



کد این سوال در فایل به نام p1\_2.m قرار دارد .

✓ بخش سوم سوال ۱ :

در این بخش از ما خواسته شده تا سیگنال  $y_1(t)$  را بر حسب  $x(t)$  با استفاده از شیفت و مقیاس زمانی بنویسیم .

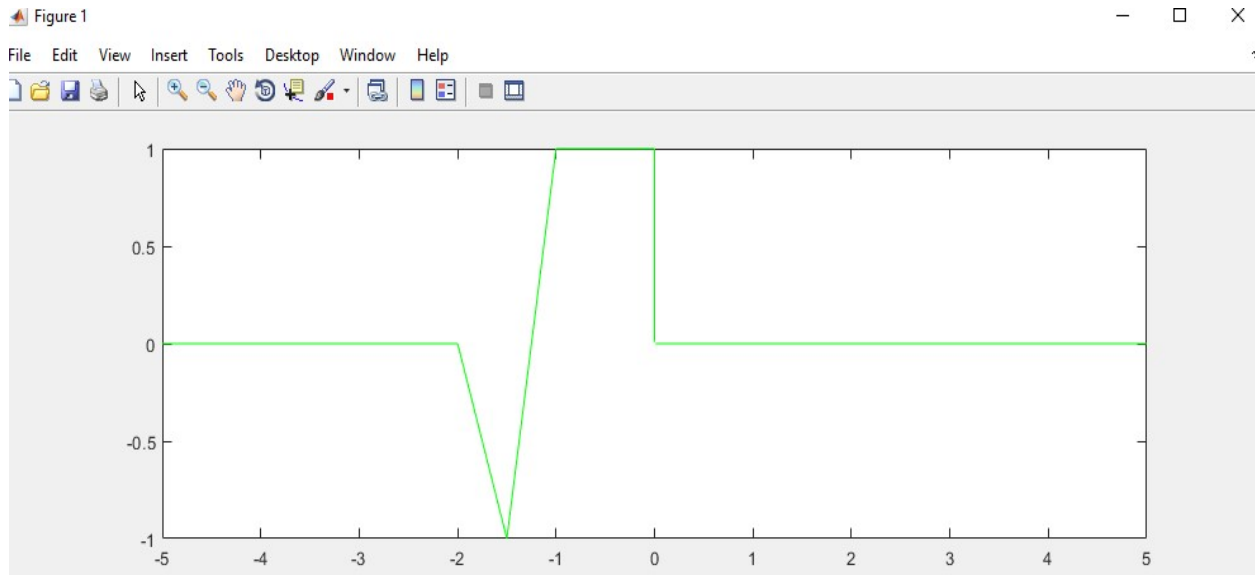
ابتدا نمودار  $x(t)$  را ۲ واحد به سمت چپ شیفت می‌دهیم یعنی :

$$Y(t) = x(t+2)$$

سپس حاصل را بر حسب زمان تا دو برابر فشرده می‌کنیم به صورت زیر :

$$Y_1(t) = Y(2t) = x(2 \cdot (t+2)) = x(2t + 4)$$

سیگنال  $y_1(t)$  را نیز در متلب رسم می‌کنیم . کد نمودار زیر در فایل p1\_3.m میباشد .



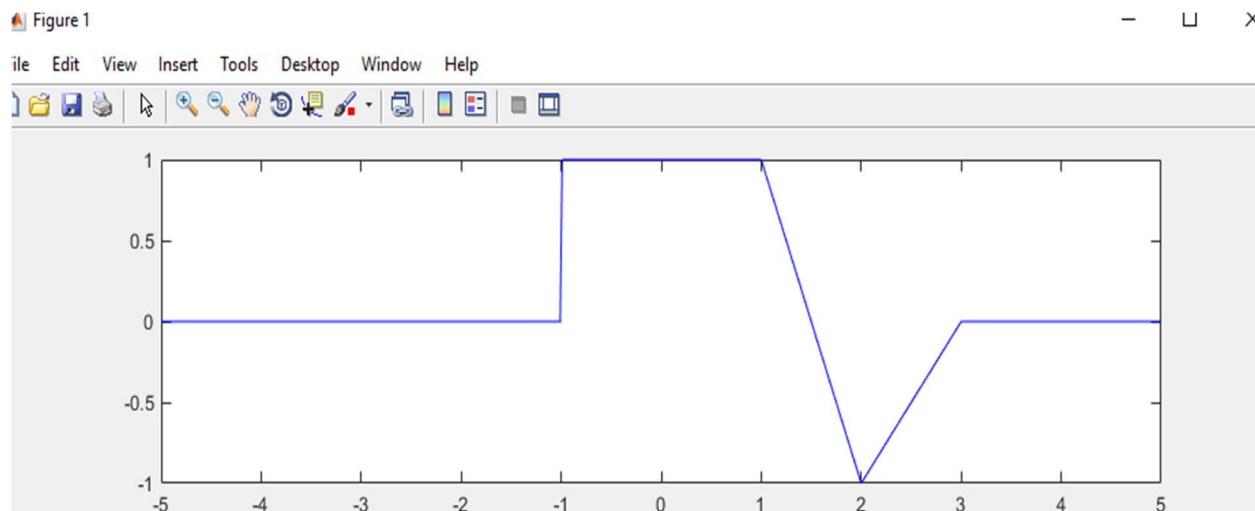
✓ بخش چهارم سوال ۱ :

این بخش نیز همانند بخش قبلی میباشد و بایستی ابتدا بصورت دستی سیگنال  $y_2(t)$  را بر حسب  $x(t)$  بنویسیم سپس بوسیله نرم افزار متلب آنرا رسم کنیم .

برای بدست آوردن سیگنال  $y_2(t)$  ابتدا  $x(t)$  را یک واحد به چپ شیفت داده و سپس حاصل را وارون زمانی میکنیم .

$$Y_2(t) = X(-1 - (t+1))$$

نمودار آن نیز به شکل زیر میباشد و در فایل p1\_4.m کد آن قرار دارد .



کد آن نیز به شکل زیر میباشد.

```
Editor - C:\Users\Farnad\Desktop\برم\پروژه\p1_4.m
u_step.m x ramp2.m x p1_2.m x p1_3.m x p1_4.m x +
1 - t=-10:0.01:10;
2 - y2= u_step(-1) .* (t>=-1 & t<1) + -2*ramp2(1)+1 .* (t>=1 & t<2) + ramp2(2)-1 .* (t>=2 & t<3);
3 - plot(t,y2,'b');
```

## • پاسخ سوال ۲ :

### ✓ بخش اول سوال ۲ :

در بخش اول این سوال از ما خواسته شده تا ۴ سیگنال داده شده در صورت سوال را پیاده سازی کرده و برای هر کدام تابع رسم بنویسیم .

رسم تابع  $xo[n]$  :

این تابع برای سیگنال هایی که زوج هستند مقدار صفر را برمیگرداند و برای سیگنال های فرد خود سیگنال را برمیگرداند .

کد بدست آوردن این سیگنال در فایل `xo.m` میباشد و بصورت زیر است .

```

1 function [out1]=xo(x)
2     n=-10:0.01:10;
3     y1= x .* (n>-10 & n<10);
4     y2=fliplr(y1);
5     out1= (y1 - y2)/2;
6     end

```

رسم تابع  $xe[n]$  :

این تابع برای سیگنال هایی که فرد هستند مقدار صفر را برمیگرداند و برای سیگنال های زوج خود سیگنال را برمیگرداند .

کد بدست آوردن این سیگنال در فایل `xe.m` میباشد و بصورت زیر است

```

1 function [out1]=xe(x)
2     n=-10:0.01:10;
3     y1= x .* (n>-10 & n<10);
4     y2=fliplr(y1);
5     out1= (y1 + y2)/2;
6     end

```

رسم تابع  $xl[n]$  :

این تابع فقط مقادیر قسمتهایی که  $n$  منفی دارند را نشان میدهد و برای بقیه قسمت ها مقدار صفر دارد . کد بدست آوردن این سیگنال در فایل `xl.m` میباشد و بصورت زیر است:

```

1 function [out1]=xl(x)
2     n=-10:0.01:10;
3     out1= x .* (n<0) ;
4 end
5

```

رسم تابع  $xr[n]$  :

این تابع فقط مقادیر قسمتهایی که  $n$  مثبت دارند را نشان میدهد و برای بقیه قسمت ها مقدار صفر دارد. کد بدست آوردن این سیگنال در فایل  $xr.m$  میباشد و بصورت زیر است:

```

1 function [out1]=xr(x)
2     n=-10:0.01:10;
3     out1= x .* (n>0) ;
4 end
5

```

یک فایل نیز به نام  $test.m$  قرار داده شده که ای چهار تابع در آن تست شده است .

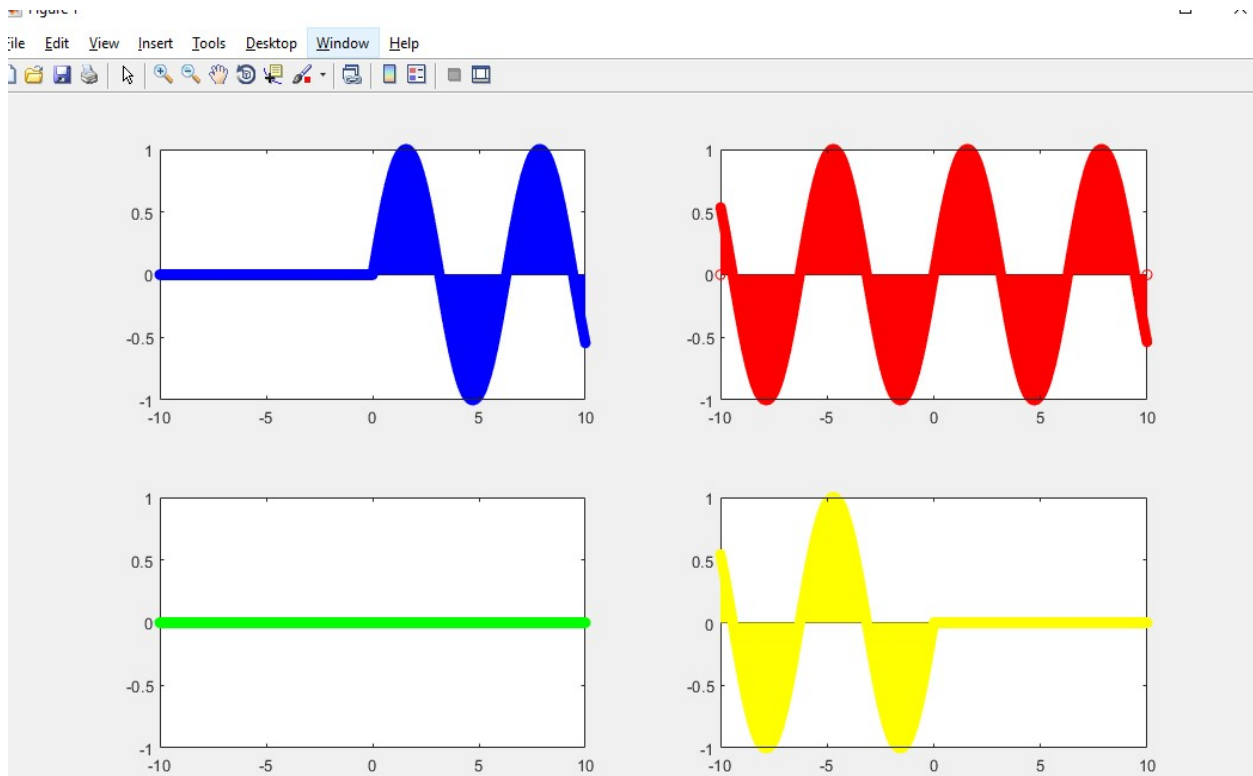
ورودی که برای این توابع در نظر گرفته شده تابع  $\sin(x)$  میباشد . این تابع تابعی فرد میباشد و انتظار میرود که تابع  $xr[n]$  فقط برای مقادیر مثبت  $n$  مقدار داشته باشد . (شکل آبی رنگ)

تابع  $xl[n]$  فقط برای مقادیر منفی  $n$  مقدار داشته باشد . (شکل زرد رنگ)

تابع  $xo[n]$  همان تابع  $\sin$  را به خروجی برگرداند . ( شکل قرمز رنگ)

تابع  $xe[n]$  مقدار تماما صفر را به خروجی برگرداند . (شکل سبز رنگ )





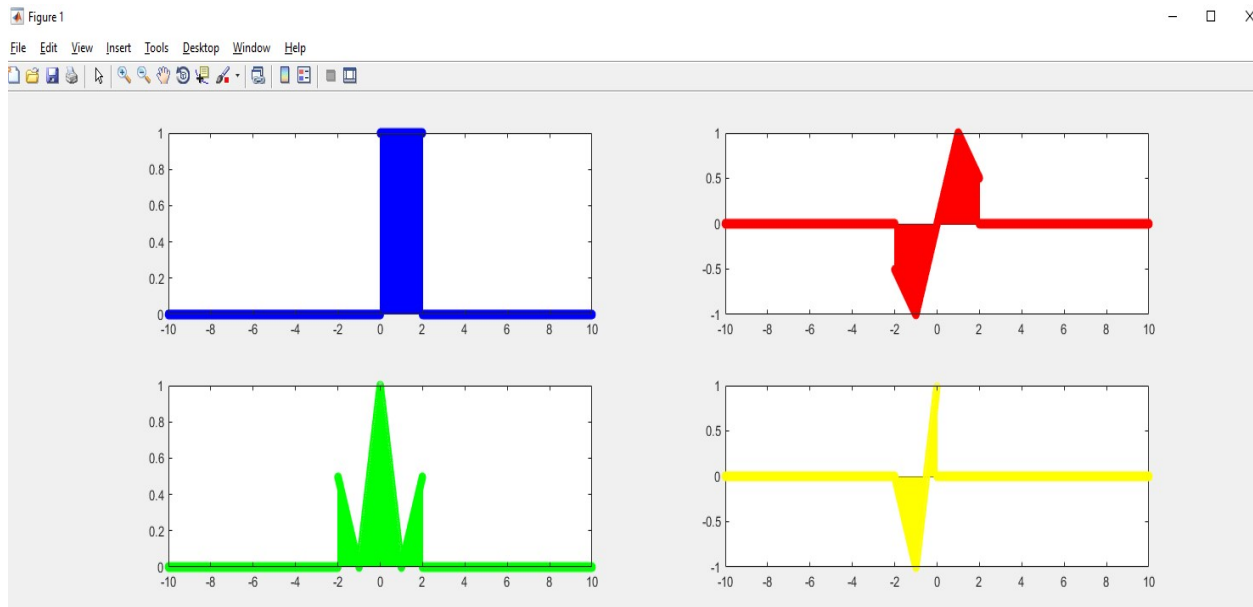
در قسمت بعدی سوال از ما خواسته شده تا سیگنال  $x(t)$  که در سوال اول به آن پرداختیم را به این توابع پاس دهیم و با فرکانس ۱۰۰ نمونه بر ثانیه از آن نمونه برداری کنیم.

شکل زیر کد مربوط به این بخش است که در فایل `p2_1_xt.m` ذخیره شده است.

```

Editor - C:\Users\Farhad\Desktop\5 پروژه\p2_1_xt.m
p1_3.m x p1_4.m x p1_1.m x sampling.m x xr.m x xo.m x xe.m x xl.m x test.m x p2_1_xt.m x +
1 - t=-10:0.01:10;
2 - xt= -ramp2(-2) .* (t>=-2 & t<-1) +2*ramp2(-1)-1 .* (t>=-1 & t<0) + u_step(0) .* (t>=0 & t<2);
3 - y= xt;
4 - subplot(2,2,1); |
5 - stem(t,xr(y),'b');
6 - subplot(2,2,2);
7 - stem(t,xo(y),'r');
8 - subplot(2,2,3);
9 - stem(t,xe(y),'g');
10 - subplot(2,2,4);
11 - stem(t,xl(y),'y');
    
```

کد بالا شکل زیر را حاصل میکند. در شکل زیر ترتیب رنگها به همان گونه است که در قسمت قبل توضیح داده شد.



این تصویر سیگنال های چهارگانه در سوال ۲ است که به ورودی آن سیگنال  $x(t)$  مربوط به سوال ۱ را داده ایم .

### ✓ بخش دوم سوال ۲ :

برای  $n$  های بزرگتر از صفر از تابع  $xr[n]$  در ناحیه مثبت ها استفاده میکنیم .

اما برای ناحیه ی  $n$  های کوچکتر از صفر طبق رابطه ی زیر عمل میکنیم :

$$2. \quad x_e[n] = x[n] + x[-n]$$

$$X[n] = 2.x_e[n] - x[-n]$$

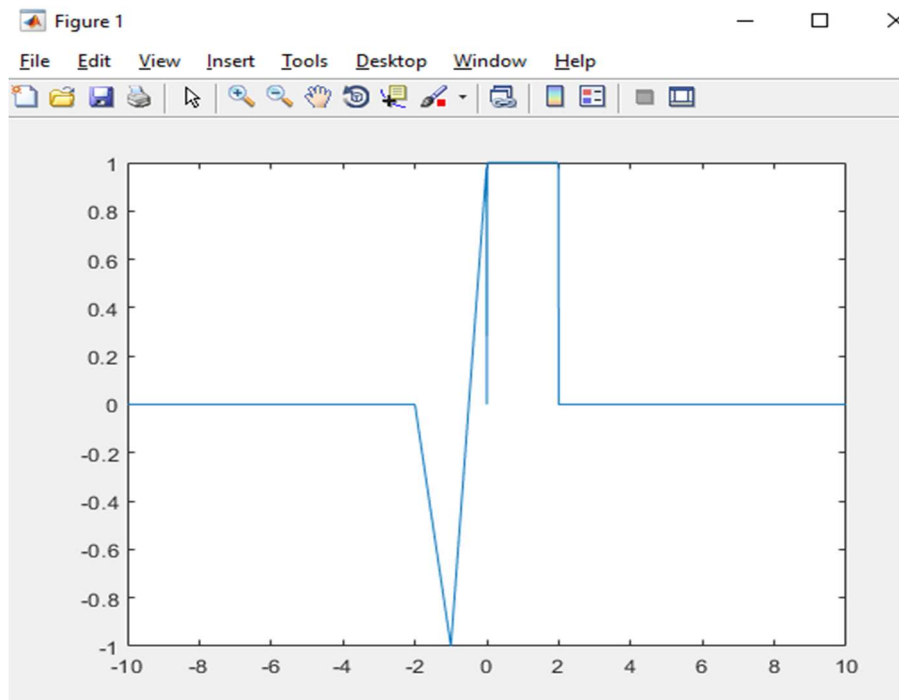
کد زیر بیان همین مطلب میباشد که در فایل p2\_2.m ذخیره شده است .

```

Editor - C:\Users\Farhad\Desktop\5 پروژه\ss\آرم\p2_2.m
p1_3.m  p1_1.m  sampling.m  xr.m  xo.m  xe.m  xl.m  test.m  p2_1_xt.m  p2_2.m  p2_3.m  +
1 -   t=-10:0.01:10;
2 -   xt= -ramp2(-2) .* (t>=-2 & t<-1) +2*ramp2(-1)-1 .* (t>=-1 & t<0) + u_step(0) .* (t>=0 & t<2);
3 -   v=fliplr(xr(xt));
4 -   y1=2*xe(xt)-v;
5 -   y2=xr(xt);
6 -   y=y1.*(t<0) +y2 .* (t>0);
7 -   plot(t,y);
8

```

رسم این سیگنال بازسازی شده بصورت زیر میباشد .



دلیل اینکه کمی شکل با تابع اصلی متفاوت است این است که این تابع از دوبخش تشکیل شده و در نقطه صفر شکل برای هر دو قسمت چپ و راست رسم میشود .

✓ بخش سوم سوال ۲ :

برای  $n$  های کوچکتر از صفر از تابع  $xr[n]$  در ناحیه منفی ها استفاده میکنیم .

اما برای ناحیه  $n$  های بزرگتر از صفر طبق رابطه ی زیر عمل میکنیم :

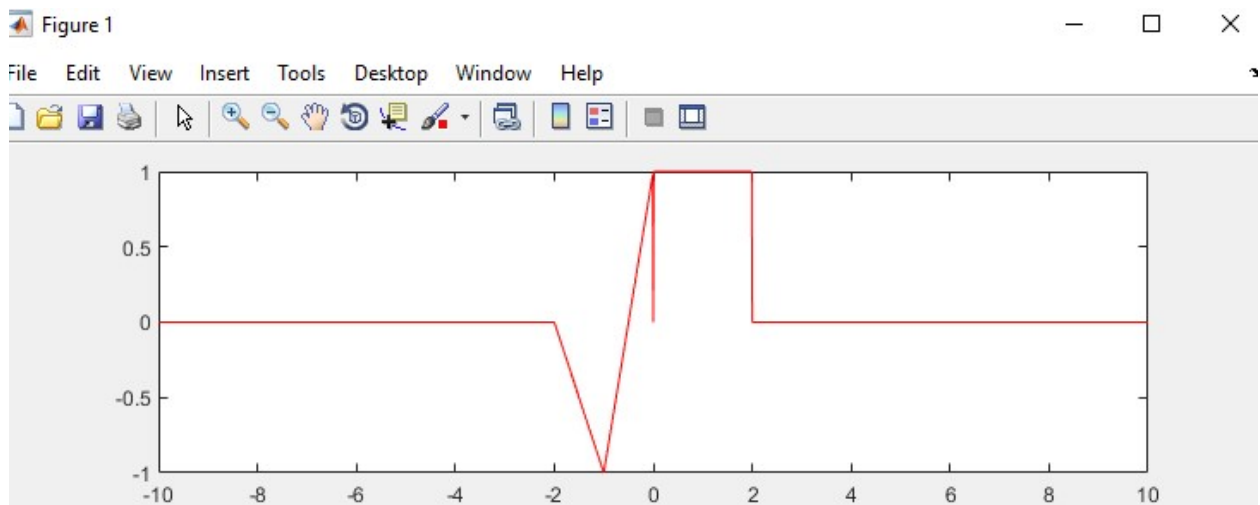
$$2. \ xo[n] = x[n] - x[-n]$$

$$X[n] = 2.xo[n] + x[-n]$$

کد زیر بیان همین مطلب میباشد که در فایل p2\_3.m ذخیره شده است .

```
p1_3.m x p1_1.m x sampling.m x xr.m x xo.m x xe.m x xl.m x test.m x p2_1_xt.m x p2_2.m x p2_3.m x +
- t=-10:0.01:10;
- xt= -ramp2(-2) .* (t>=-2 & t<-1) +2*ramp2(-1)-1 .* (t>=-1 & t<0) + u_step(0) .* (t>=0 & t<2);
- v=fliplr(xl(xt));
- y1=2*xo(xt)+v;
- y2=xl(xt);
- y=y1.* (t>0) +y2 .* (t<0);
- plot(t,y);
```

رسم این سیگنال بازسازی شده بصورت زیر میباشد .



دلیل اینکه کمی شکل با تابع اصلی متفاوت است این است که این تابع از دوبخش تشکیل شده و در نقطه صفر شکل برای هر دو قسمت چپ و راست رسم میشود .

• پاسخ سوال ۳ :

✓ بخش اول سوال ۳ :

✓ بخش دوم سوال ۳ :

✓ بخش سوم سوال ۳ :

• پاسخ سوال ۴ :

✓ بخش اول سوال ۴ :

این سوال در رابطه با بررسی انرژی سیگنال میباشد . که در قسمت اول سوال بایستی تابعی برای بدست آوردن انرژی سیگنال پیاده سازی کنیم .

```
function [e]=energy(x)
    e=sum(abs(x).^2);
end
```

این تابع در فایل energy.m ذخیره شده است .

✓ بخش دوم سوال ۴ :

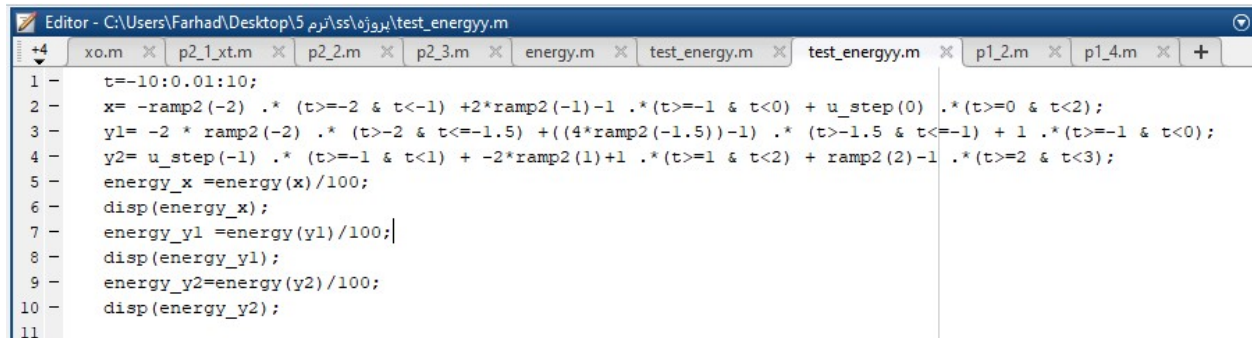
تابعی که در قسمت اول این سوال پیاده سازی شد برای سیگنال های گسسته کاربرد دارد . سیگنال های  $x(t)$  و  $y_1(t)$  و  $y_2(t)$  پیوسته اند و اگر اینها را به تابع انرژی پاس دهیم مقادیر را با فرکانس ۱۰۰ نمونه در ثانیه (فرکانس نمونه برداری) با هم جمع میکند و مقدار درستی را بعنوان خروجی به ما میدهد . برای رفع این مشکل بایستی خروجی تابع را بر فرکانس نمونه برداری تقسیم کنیم چون در تابعی که ما پیاده سازی کرده ایم

فرکانس نمونه برداری ۱۰۰ در نظر گرفته شده پس ما باید حاصل را بر ۱۰۰ تقسیم کنیم. در نتیجه جواب نهایی برای انرژی این سه سیگنال برابر است با :

$$\text{Energy}(x(t)) = 2.6618$$

$$\text{Energy}(y_1(t)) = 1.3585$$

$$\text{Energy}(y_2(t)) = 2.6718$$



```

1 - t=-10:0.01:10;
2 - x= -ramp2(-2) .* (t>=-2 & t<=-1) + 2*ramp2(-1)-1 .* (t>=-1 & t<0) + u_step(0) .* (t>=0 & t<2);
3 - y1= -2 * ramp2(-2) .* (t>=-2 & t<=-1.5) + ((4*ramp2(-1.5))-1) .* (t>=-1.5 & t<=-1) + 1 .* (t>=-1 & t<0);
4 - y2= u_step(-1) .* (t>=-1 & t<1) + -2*ramp2(1)+1 .* (t>=1 & t<2) + ramp2(2)-1 .* (t>=2 & t<3);
5 - energy_x =energy(x)/100;
6 - disp(energy_x);
7 - energy_y1 =energy(y1)/100;
8 - disp(energy_y1);
9 - energy_y2=energy(y2)/100;
10 - disp(energy_y2);
11

```

تصویر بالا کد مربوط به این بخش است که در فایل test\_energy.m ذخیره شده است. خروجی مربوط به این بخش نیز در تصویر زیر مشاهده میکنید .

```

>> test_energy
2.6618

1.3585

2.6718

```

• پاسخ سوال ۵ :

✓ بخش اول سوال ۵ :

✓ بخش دوم سوال ۵ :

