

گزارش کار پروژه نهایی

سیگنال و سیستم ها

کسری کاشانی

810101490

قسمت اول :

```
clc , clearvars  
  
notes
```

با افزودن notes می توان به داده های موجود در این فایل دسترسی پیدا کرد.
برای راحتی بجای single quote از Double quote استفاده کردم که بتوان هر نت را به صورت یک استرینگ دریافت کرد.

```
silence = 0.025; %25 ms  
fs = 44100;  
one_silence = zeros(1,floor(silence*fs));
```

کد بالا هم برای قسمت های سکوت ایجاد شده و fs هم فرکانس نمونه برداری می باشد.

```
C = 16.352;  
CSharp = 17.324;  
D = 18.324;  
DSharp = 19.445;  
E = 20.602;  
F = 21.827;  
FSharp = 23.125;  
G = 24.500;  
GSharp = 25.957;  
A = 27.500;  
ASharp = 29.135;  
B = 30.868;
```

مقادیر موجود در ستون اول جدول داده شده را به صورت بالا در متغیر هایی ذخیره شده تا بعدا بتوان از آن ها استفاده کرد.

البته لازم نیست تمام نت ها را ذخیره کرد چرا که هر اکتاو که بالاتر می رویم فرکانس ها دوبرابر شده و می توان آن ها را در صورت نیاز ذخیره نمود. این ها فرکانس نت های اولین اکتاو می باشند.

```
finalSound = [];
```

قرار است صوت نهایی موسیقی در این آرایه ذخیره شود.

برای قرار دادن هر نت لازم است یک حلقه for درست شود تا هر عنصر آرایه داده شده را یک دور ملاقات کنیم.

در داخل حلقه از یک switch-case statement استفاده شده تا نت مورد نظر پیدا شود و برای ساخت سیگنال مربوط لازم است اکتاو نت و طول مدت زمان را داشته باشیم. سپس یک سینوسی با دامنه ثابت و فرکانس که محاسبه می شود و با مدت زمان داده شده ساخته می شود و در انتهای هر حلقه این سیگنال بعلاوه سیگنال سکوت به FinalSound که قرار است موسیقی نهایی باشد اضافه می شود.

```

for i = 1:61
    note = split(noteHarryPotter(i));
    noteType = note(1);
    octav = str2num(note(2));
    duration = str2double(note(3));
    switch noteType
    case "C"
        time = linspace(0,duration,duration*fs);
        soundofNote = sin(2*pi*C*(2^octav)*time);
    case "C#"
        time = linspace(0,duration,duration*fs);
        soundofNote = sin(2*pi*CSharp*(2^octav)*time);
    case "D"
        time = linspace(0,duration,duration*fs);
        soundofNote = sin(2*pi*D*(2^octav)*time);
    case "D#"
        time = linspace(0,duration,duration*fs);
        soundofNote = sin(2*pi*DSharp*(2^octav)*time);
    case "E"
        time = linspace(0,duration,duration*fs);
        soundofNote = sin(2*pi*E*(2^octav)*time);
    case "F"
        time = linspace(0,duration,duration*fs);
        soundofNote = sin(2*pi*F*(2^octav)*time);
    case "F#"
        time = linspace(0,duration,duration*fs);
        soundofNote = sin(2*pi*FSharp*(2^octav)*time);
    case "G"
        time = linspace(0,duration,duration*fs);
        soundofNote = sin(2*pi*G*(2^octav)*time);
    case "G#"
        time = linspace(0,duration,duration*fs);
        soundofNote = sin(2*pi*GSharp*(2^octav)*time);
    case "A"
        time = linspace(0,duration,duration*fs);
        soundofNote = sin(2*pi*A*(2^octav)*time);
    case "A#"
        time = linspace(0,duration,duration*fs);
        soundofNote = sin(2*pi*ASharp*(2^octav)*time);
    case "B"
        time = linspace(0,duration,duration*fs);
        soundofNote = sin(2*pi*B*(2^octav)*time);
    end


    finalSound = [finalSound soundofNote one_silence];
end

```

در انتها هم به صورت زیر صوت در یک فایل ذخیره می گردد.

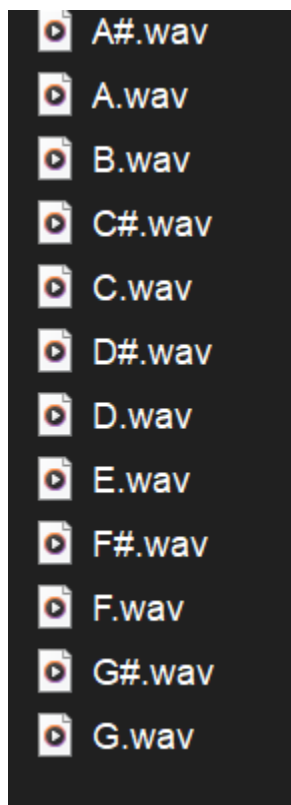
```
end  
audiowrite("noteHarryPoter.wav",finalSound,fs)
```

فایل تولید شده :

 noteHarryPoter.wav

قسمت دوم :

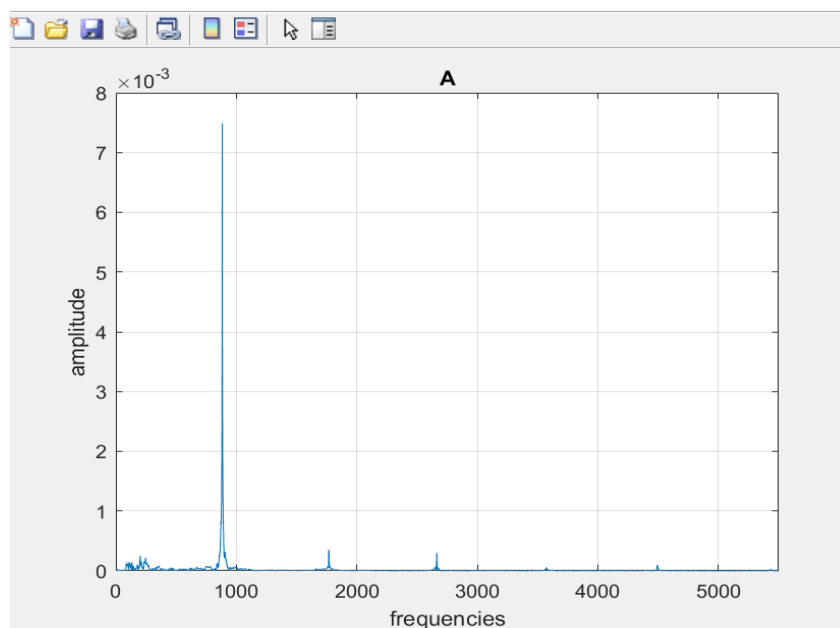
در ابتدا لازم بود صوت های مربوط به صدای پیانو برای تمام نت های یک اکتاو ، جمع آوری بشود.
برای این کار از شبیه ساز های پیانو استفاده شد.
این صوت ها در پوشه notes قرار دارند.



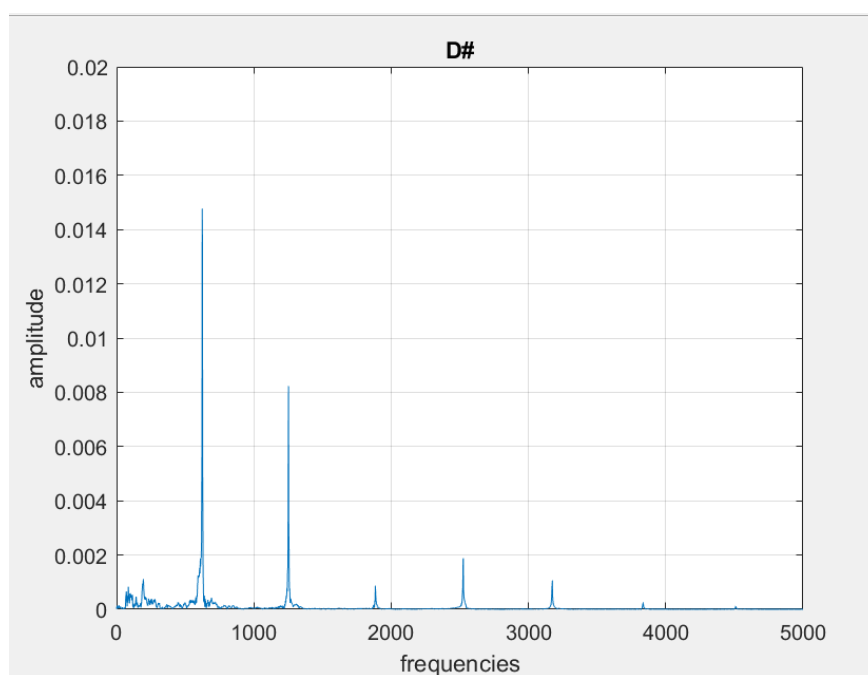
برای هر یک از این فایل ها یک کد متلب قرار دارد که تبدیل فوریه (در اصل FFT) صوت هایشان را plot میکند.

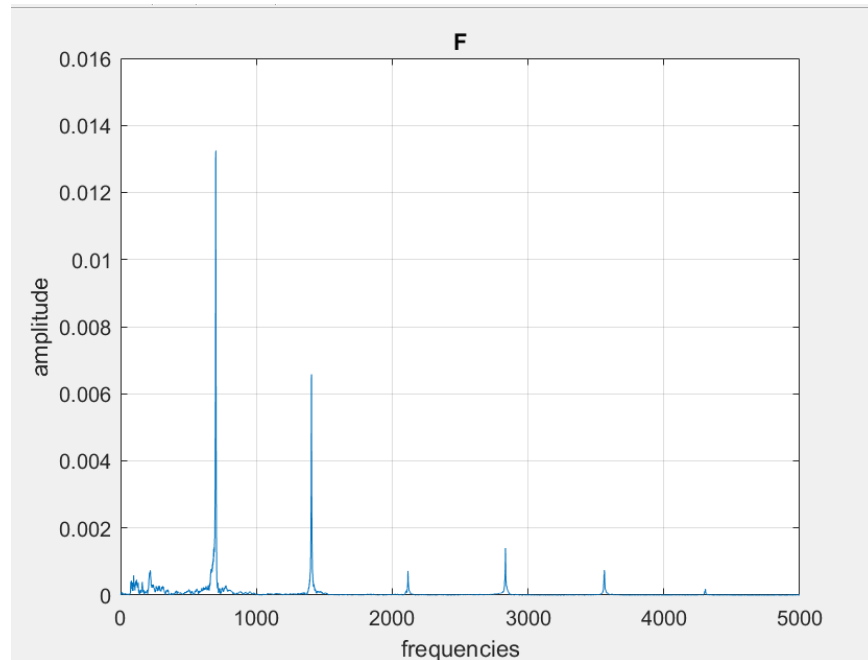
برای نمونه :

فایل a.m که صوت مربوط به نت A5 را پردازش میکند چنین خروجی دارد:



چند نمونه دیگر:





برای هر یک از نت ها مشاهده می شود که علاوه بر یک مولفه غالب فرکانسی هارمونی های دیگر هم وجود دارد.

این هارمونی ها را با ابزار brush data موجود در همین پنجره نمودار فرکانس و دامنه شان جدا کردیم و در نرم افزار اکسل به صورت زیر قرار دادیم.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	harmonies	1		2		3		4		5		6		note
2	freq - amplitude	525.1525	0.030178	1050.305	0.002209435	1579.117	0.0019434	2113.419	0.001235	2653.209	0.00107	3203.979	0.000571	C
3	freq - amplitude	555.5566	0.039337	1111.113	0.001487067	1673.13	0.0027623	2239.453	0.000719	2812.236	0.001049	3393.633	0.00079	C#
4	freq - amplitude	588.3347	0.015736	1177.673	0.001037645	1773.036	0.0008245	2978.821	0.000496	2978.821	0.000496	3595.267	0.000291	D
5	freq - amplitude	623.8028	0.01477	1252.65	0.008228524	1884.86	0.0008699	2525.477	0.001884	3172.819	0.000881	3835.294	0.000256	D#
6	freq - amplitude	662.0711	0.010374	1326.164	0.005686177	1997.332	0.0006188	2675.576	0.001039	3362.917	0.000625	4063.398	0.000143	E
7	freq - amplitude	701.6824	0.013249	1404.706	0.006415572	2115.78	0.0007207	2833.563	0.001401	3562.078	0.000746	4305.351	0.000182	F
8	freq - amplitude	742.1968	0.011951	1485.198	0.000534031	2239.456	0.0004943	3004.168	9.74E-05	3777.726	0.000159	4571.386	4.78E-05	F#
9	freq - amplitude	786.9527	0.010764	1573.905	0.000551388	2373.118	0.0004483	3183.059	8.84E-05	4002.195	0.000146	4842.786	4.40E-05	G
10	freq - amplitude	833.5312	0.017527	1667.062	0.000781906	2514.507	0.0006425	3372.07	1.33E-04	4239.752	0.000217	5131.466	6.47E-05	G#
11	freq - amplitude	882.872	0.007493	1766.318	0.000348604	2663.531	0.0003008	3572.792	5.63E-05	4492.378	9.30E-05	5436.059	2.83E-05	A
12	freq - amplitude	935.3041	0.013992	1878.885	0.001138589	2834.291	0.0002533	3815.709	8.32E-05	4823.142	1.21E-05	5746.622	8.54E-06	A#
13	freq - amplitude	991.4735	0.014172	1990.911	0.001150431	3002.293	0.0002475	4041.549	8.02E-05	5108.677	1.21E-05	6214.296	4.27E-06	B
14														

برای هر هارمونی در سمت چپ فرکانس و در سمت راست دامنه اش قرار دارد.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	harmonies	1		2		3		4		5		6		note
2	freq - amplitude	525.1525	0.030178	1050.305	0.002209435	1579.117	0.0019434	2113.419	0.001235	2653.209	0.00107	3203.979	0.000571	C
3	freq - amplitude	555.5566	0.039337	1111.113	0.001487067	1673.13	0.0027623	2239.453	0.000719	2812.236	0.001049	3393.633	0.00079	C#
4	freq - amplitude	588.3347	0.015736	1177.673	0.001037645	1773.036	0.0008245	2978.821	0.000496	2978.821	0.000496	3595.267	0.000291	D
5	freq - amplitude	623.8028	0.01477	1252.65	0.008228524	1884.86	0.0008699	2525.477	0.001884	3172.819	0.000881	3835.294	0.000256	D#
6	freq - amplitude	662.0711	0.010374	1326.164	0.005686177	1997.332	0.0006188	2675.576	0.001039	3362.917	0.000625	4063.398	0.000143	E
7	freq - amplitude	701.6824	0.013249	1404.706	0.006415572	2115.78	0.0007207	2833.563	0.001401	3562.078	0.000746	4305.351	0.000182	F
8	freq - amplitude	742.1968	0.011951	1485.198	0.000534031	2239.456	0.0004943	3004.168	9.74E-05	3777.726	0.000159	4571.386	4.78E-05	F#
9	freq - amplitude	786.9527	0.010764	1573.905	0.000551388	2373.118	0.0004483	3183.059	8.84E-05	4002.195	0.000146	4842.786	4.40E-05	G
10	freq - amplitude	833.5312	0.017527	1667.062	0.000781906	2514.507	0.0006425	3372.07	1.33E-04	4239.752	0.000217	5131.466	6.47E-05	G#
11	freq - amplitude	882.872	0.007493	1766.318	0.000348604	2663.531	0.0003008	3572.792	5.63E-05	4492.378	9.30E-05	5436.059	2.83E-05	A
12	freq - amplitude	935.3041	0.013992	1878.885	0.001138589	2834.291	0.0002533	3815.709	8.32E-05	4823.142	1.21E-05	5746.622	8.54E-06	A#
13	freq - amplitude	991.4735	0.014172	1990.911	0.001150431	3002.293	0.0002475	4041.549	8.02E-05	5108.677	1.21E-05	6214.296	4.27E-06	B
14														

دامنه / فرکانس

نت ها

مربوط به هارمونی 5

چون طول زمان هر صوت یکسان نبود و شدت ها ممکن است متفاوت باشد تصمیم گرفتیم که هارمونی اول هر نت را 1 کنیم و بقیه را اسکالی که هارمونی اول گرفت رویش انجام دهیم تا داده ها normalize بشوند.

مقادیر نرمال شده:

normalized:														
harmonies	1		2		3		4		5		6		note	
freq - amplitude	525.1525	1	1050.305	0.073214087	1579.117	0.0644	2113.419	0.040931	2653.209	0.035446	3203.979	0.018936	C	
freq - amplitude	555.5566	1	1111.113	0.037803069	1673.13	0.0702219	2239.453	0.018286	2812.236	0.026664	3393.633	0.020092	C#	
freq - amplitude	588.3347	1	1177.673	0.065942653	1773.036	0.0523991	2978.821	0.031521	2978.821	0.031534	3595.267	0.01852	D	
freq - amplitude	623.8028	1	1252.65	0.557124672	1884.86	0.0588947	2525.477	0.127581	3172.819	0.059642	3835.294	0.017356	D#	
freq - amplitude	662.0711	1	1326.164	0.548102853	1997.332	0.0596453	2675.576	0.100175	3362.917	0.060257	4063.398	0.013777	E	
freq - amplitude	701.6824	1	1404.706	0.484218382	2115.78	0.0543963	2833.563	0.105728	3562.078	0.056341	4305.351	0.013705	F	
freq - amplitude	742.1968	1	1485.198	0.044685771	2239.456	0.0413649	3004.168	0.008153	3777.726	0.013285	4571.386	0.003997	F#	
freq - amplitude	786.9527	1	1573.905	0.051223794	2373.118	0.0416458	3183.059	0.008211	4002.195	0.013589	4842.786	0.004092	G	
freq - amplitude	833.5312	1	1667.062	0.044612578	2514.507	0.0366573	3372.07	0.007565	4239.752	0.0124	5131.466	0.003692	G#	
freq - amplitude	882.872	1	1766.318	0.046526706	2663.531	0.0401442	3572.792	0.007512	4492.378	0.012419	5436.059	0.003779	A	
freq - amplitude	935.3041	1	1878.885	0.081371576	2834.291	0.0181037	3815.709	0.005945	4823.142	0.000866	5746.622	0.00061	A#	
freq - amplitude	991.4735	1	1990.911	0.081175441	3002.293	0.0174605	4041.549	0.005658	5108.677	0.000851	6214.296	0.000301	B	

همین مقادیر بالا را در یک فایل با فرمت mat. قرار دادیم تا در متلب از آن ها استفاده کنیم.

کد این قسمت بسیار شبیه قسمت اول است و از یک حلقه for و یک switch-case statement تشکیل شده با این تفاوت که هر صوت بجای یک سینوسی خالص از جمع چندین سینوسی با فرکانس و دامنه های استخراج شده تشکیل شده است.

```

case "C"
    time = linspace(0,duration,duration*fs);
    soundofNote = harmony(1,2)*sin(2*pi*harmony(1,1)*(2^(octav-5))*time);
    soundofNote = soundofNote + harmony(1,4)*sin(2*pi*harmony(1,3)*(2^(octav-5))*time);
    soundofNote = soundofNote + harmony(1,6)*sin(2*pi*harmony(1,5)*(2^(octav-5))*time);
    soundofNote = soundofNote + harmony(1,8)*sin(2*pi*harmony(1,7)*(2^(octav-5))*time);
    soundofNote = soundofNote + harmony(1,10)*sin(2*pi*harmony(1,9)*(2^(octav-5))*time);
    soundofNote = soundofNote + harmony(1,12)*sin(2*pi*harmony(1,11)*(2^(octav-5))*time);
case "C#"
    time = linspace(0,duration,duration*fs);
    soundofNote = harmony(2,2)*sin(2*pi*harmony(2,1)*(2^(octav-5))*time);
    soundofNote = soundofNote + harmony(2,4)*sin(2*pi*harmony(2,3)*(2^(octav-5))*time);
    soundofNote = soundofNote + harmony(2,6)*sin(2*pi*harmony(2,5)*(2^(octav-5))*time);
    soundofNote = soundofNote + harmony(2,8)*sin(2*pi*harmony(2,7)*(2^(octav-5))*time);
    soundofNote = soundofNote + harmony(2,10)*sin(2*pi*harmony(2,9)*(2^(octav-5))*time);
    soundofNote = soundofNote + harmony(2,12)*sin(2*pi*harmony(2,11)*(2^(octav-5))*time);
case "D"
    time = linspace(0,duration,duration*fs);
    soundofNote = harmony(3,2)*sin(2*pi*harmony(3,1)*(2^(octav-5))*time);
    soundofNote = soundofNote + harmony(3,4)*sin(2*pi*harmony(3,3)*(2^(octav-5))*time);
    soundofNote = soundofNote + harmony(3,6)*sin(2*pi*harmony(3,5)*(2^(octav-5))*time);
    soundofNote = soundofNote + harmony(3,8)*sin(2*pi*harmony(3,7)*(2^(octav-5))*time);
    soundofNote = soundofNote + harmony(3,10)*sin(2*pi*harmony(3,9)*(2^(octav-5))*time);
    soundofNote = soundofNote + harmony(3,12)*sin(2*pi*harmony(3,11)*(2^(octav-5))*time);
case "D#"
    time = linspace(0,duration,duration*fs);
    soundofNote = harmony(4,2)*sin(2*pi*harmony(4,1)*(2^(octav-5))*time);
    soundofNote = soundofNote + harmony(4,4)*sin(2*pi*harmony(4,3)*(2^(octav-5))*time);
    soundofNote = soundofNote + harmony(4,6)*sin(2*pi*harmony(4,5)*(2^(octav-5))*time);
    soundofNote = soundofNote + harmony(4,8)*sin(2*pi*harmony(4,7)*(2^(octav-5))*time);
    soundofNote = soundofNote + harmony(4,10)*sin(2*pi*harmony(4,9)*(2^(octav-5))*time);
    soundofNote = soundofNote + harmony(4,12)*sin(2*pi*harmony(4,11)*(2^(octav-5))*time);
case "E"
    time = linspace(0,duration,duration*fs);
    soundofNote = harmony(5,2)*sin(2*pi*harmony(5,1)*(2^(octav-5))*time);
    soundofNote = soundofNote + harmony(5,4)*sin(2*pi*harmony(5,3)*(2^(octav-5))*time);
    soundofNote = soundofNote + harmony(5,6)*sin(2*pi*harmony(5,5)*(2^(octav-5))*time);
    soundofNote = soundofNote + harmony(5,8)*sin(2*pi*harmony(5,7)*(2^(octav-5))*time);
    soundofNote = soundofNote + harmony(5,10)*sin(2*pi*harmony(5,9)*(2^(octav-5))*time);
    soundofNote = soundofNote + harmony(5,12)*sin(2*pi*harmony(5,11)*(2^(octav-5))*time);

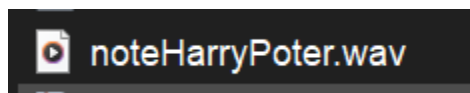
```

در انتهای حلقه هم قبل از اینکه به موسیقی اضافه شود در یک نمایی با ضریب میرایی 7.5 ضرب می شود و اسکیل می شود تا مطمئن شویم از حد بالای خود تجاوز نکند. دلیل استفاده از 7.5 هم این بود که مقادیر مختلفی امتحان شد و به نظر این مقدار مناسبی می آمد.

بعد از آن هم سیگنال مربوط به سکوت اضافه می شود و به موسیقی تشکیل شده تا قبل از این اضافه می شوند.

سپس در فایل جدید به نام noteOptimized.wav نوشته می شود. برای موسیقی این قسمت هم از همان نت های بخش قبل استفاده کردیم تا چیزی قابل تحمل را گوش کنیم.

فایل تولید شده:



قسمت امتیازی:

نحوه عمل در این قسمت به این صورت بود که از این فرض استفاده کردیم که هر نت حداکثر 0.1 ثانیه طول خواهد کشید. بنابراین قسمت های سیگنال به طول 0.1 را جدا کردیم و تبدیل فوریه آن را محاسبه کردیم. به شکل زیر:

```
Frame = abs(fft(Y(i:i + fram_samples)));  
[A , indexhelper] = max(Frame(1:floor(fram_samples)));  
  
f_note = frequency(indexhelper);
```

از یک اصل در رابطه با نت های موسیقی کمک گرفتیم. فرکانس هر نت در اکتاو بالاتر دو برابر اکتاو قبلی است. علاوه بر آن هم نیم پرده هم نسب به نیم پرده قبلی $2^{1/12}$ فرکانسش چند برابر میشود. بنابراین فرکانس پایه نت را اگر داشته باشیم، می توانیم اختلاف تعداد نیم پرده های آن با یک نت پایه که C4 را به عنوان نت پایه گرفتیم به صورت زیر بدست می آید:

```
noteOffset = round(log(f_note/C4)/(log(2)/12));
```

بنابراین با داشتن این آفست می توانیم نوع نت و اکتاو آن را بدست بیاوریم. چون ممکن است مقداری اختلاف بین این فرکانس ها داشته باشیم 8 هرتز با به عنوان مقدار خطای مجاز در نظر گرفتیم و اگر مقدار فرکانس در آن ناحیه افتاد قابل قبول خواهد بود.

```
else  
    f_prev = f_note;  
    t = round(duration , 1);  
    if t>=0.1  
  
        octav = floor(noteOffset/12)+4;  
        noteName = mod(noteOffset,12)+1;  
        notes = [notes , names(noteName)+" "+string(octav)+" "+string(t)];  
  
    end  
    duration = [0.1];
```

این قسمت از کد برای زمانی است که نواختن یک نت تمام شده و حالا باید آن را در یک آرایه ذخیره کرد.

همه این ها داخل یک حلقه `while` انجام می شود تا زمانی که مقدار اندیس از سائز خود سیگنال بیشتر شود که در آن جا از برنامه خارج می شویم.

نت های بدست آمده را در یک فایل `note.mat` ذخیره کردیم تا از آن در برنامه قسمت 1 استفاده کنیم.

برنامه استخراج نت ها از موسیقی برنامه قسمت 1 با تغییرات جزئی مربوط به نحوه وارد کردن اطلاعات در پوشه `optional` قرار گرفته است. در این پوشه همان فایل تولید شده در قسمت 1 قرار دارد. پس از اجرای برنامه قسمت 1 فایل جدیدی با نام `re_noteHarryPoter` تولید می شود که نشان می دهد برنامه این قسمت نیز به خوبی عمل کرده است.