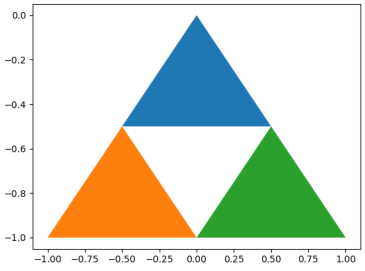
گزارش کار سری اول شبیه سازی رایانه ای در فیزیک

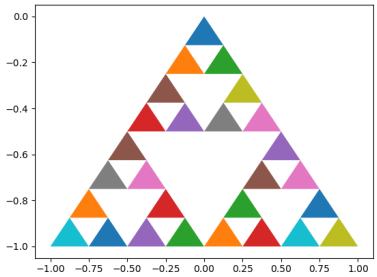
مشکات صدری
97100919

مثلث سرپینسکی با الگوریتم اصلی

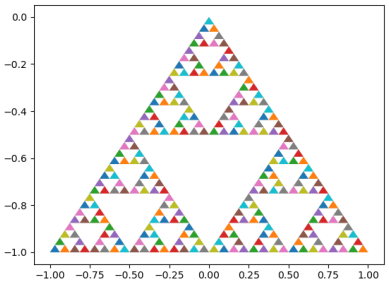
از الگوریتم ایجاد مثلث های جدید از مثلث های قبلی استفاده کرده ام، تابع newtr هر بار x یا y سه راس مثلث را گرفته و نه مختصات جدید میدهد (در واقع سه دسته ی سه تایی که کنار هم سیو شده اند) تابع ff هم تابع اصلی بوده و تمام مختصات ها را پروسس کرده و سه برابر نقطه میدهد. که نهایتا سه تا سه تا نقاط را به تابع رسم چند ضلعی می دهیم.



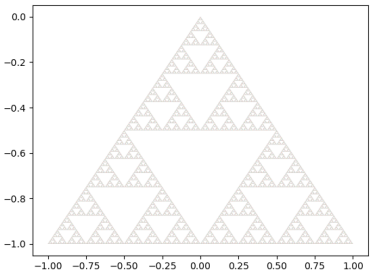
مرحله اول



مرحله سوم



مرحله پنجم



مرحله دهم (رنگاش رفت تو هم طوسی شد :))

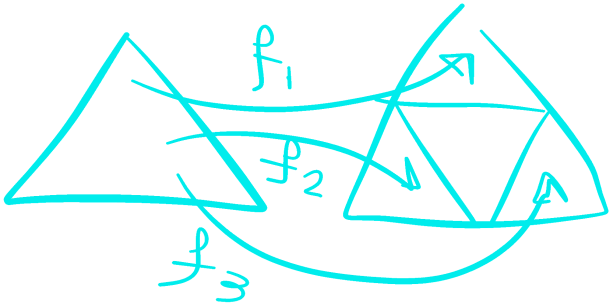
پی نوشت:
ابتدا مختصات ها را در یک ماتریس سه بعدی (n*2*3) ذخیره کرده بودم اما به مشکل میخورد، نهایتا چون کمی عجله داشتم تصمیم گرفتم تمام موارد را لیست های یک بعدی کنم، یکی برای ایکس و یکی برای ایگرگ که سه تا سه تا مختصات های یک مثلث باشند.

این دستور plt.fill گویا اگر رنگ ندهیم رندم رنگ میکند، ولی خب زیبا بود عوضش نکردم :)

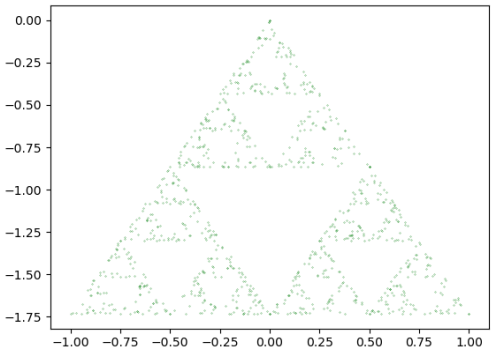
نام فایل این کد det sier fin است.

مثلث سرپینسکی با الگوریتم تصادفی

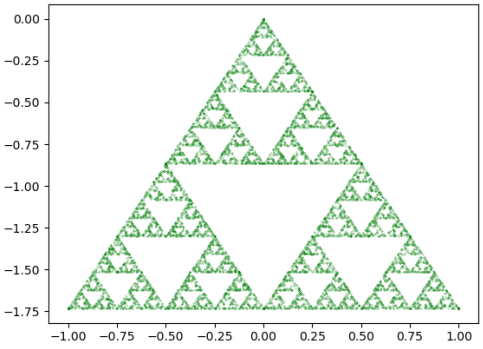
سه تابع تجانسی-انتقالی تعریف شده است، به فرم شکل زیر:



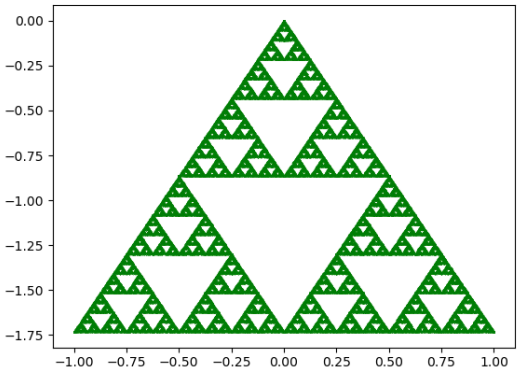
ابتدا به طور رندوم یکی از سه راس را بر میداریم، سپس عدد رندومی انتخاب میکنیم برای اینکه آن راس چند بار مورد عملیات تجانسی قرار بگیرد، سپس دوباره (سه باره :p) به طور رندوم پر میکنیم که آن تعداد عملیات با چه ترتیبی از چه توابعی پر شوند.



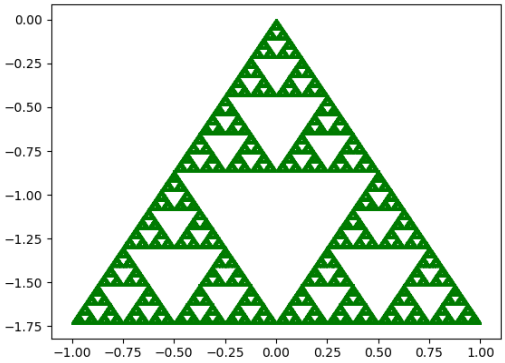
هزار نقطه



ده هزار نقطه



صد هزار نقطه



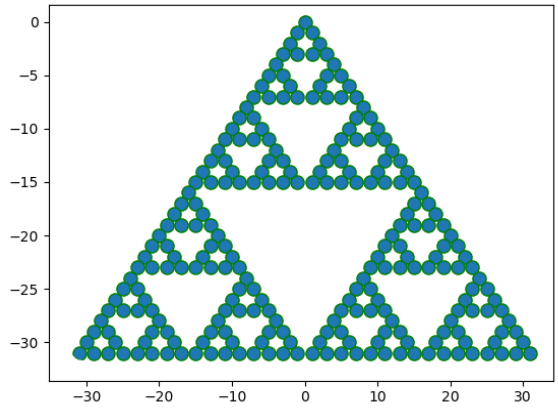
یک میلیون نقطه

نام فایل rand sier است.

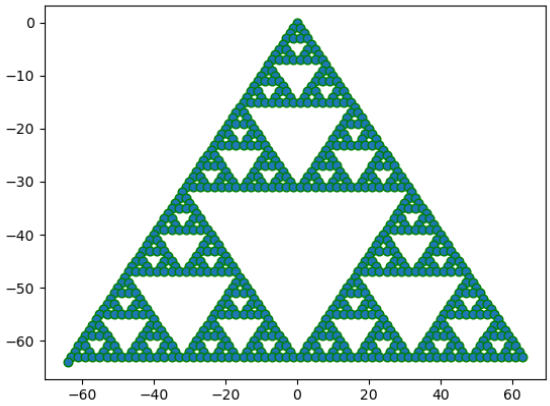
مثلث سرپینسکی با الگوریتم خیام پاسکال

برای محاسبه ی خود مثلث خیام پاسکال دو راه به نظرم به درد کد زدن میخورد، یکی الگوریتم جمع زدن و دیگری الگوریتم محاسبه ی هر کدام به فرم ترکیبیاتی(انتخاب). راه دوم را انتخاب کردم، از طرفی میدانم دستور فاکتوریل دستور سنگینی است اما خب کد ساده تر و قابل اعتماد تر میشد.
نهایتا پس از ایجاد مثلث (سه لیست شامل مختصات و مقدار هر نقطه) دو لیست ثانویه تشکیل دادم از x و y نقاطی که محتوای فرد دارند. همان ها را رسم کردم، به تناسب تعداد سایز ها را تنظیم کردم و به فرم دایره کشیدم که به چشم هموار تر باشد :)

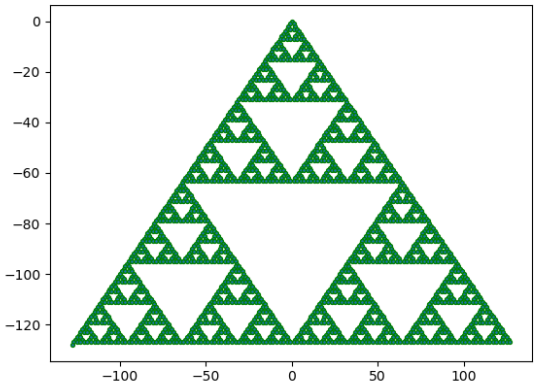
تا ردیف 32 ام



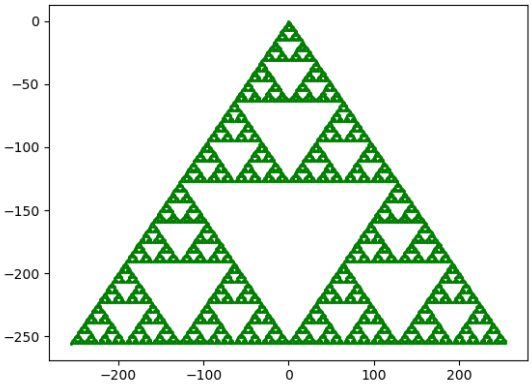
64 ام



128 ام



526 ام

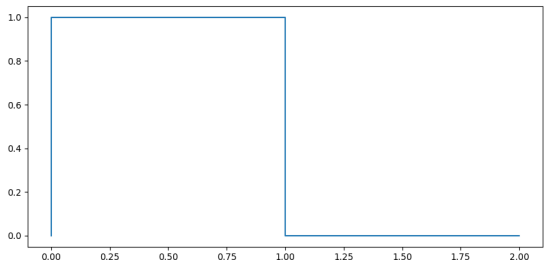


نام فایل این کد sier pascal است.

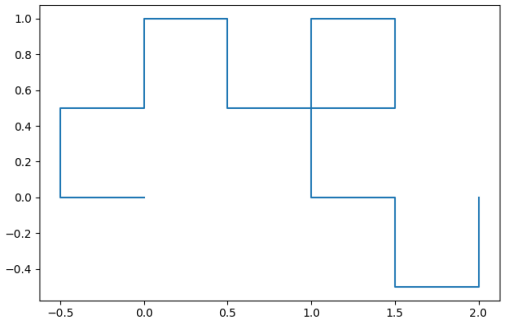
اژدهای هیوی

دو تابع تجانس-انتقال لازم بود، یکی برای نیمه ی چپ که تنها لازم بود 45 درجه دوران کند، اما تابع دوم برای نیمه راست علاوه بر اینکه 135 درجه بچرخد لازم بود از جهت ترتیب لیست نقاط سر و ته آن برعکس شود و هم اینکه منتقل شود.

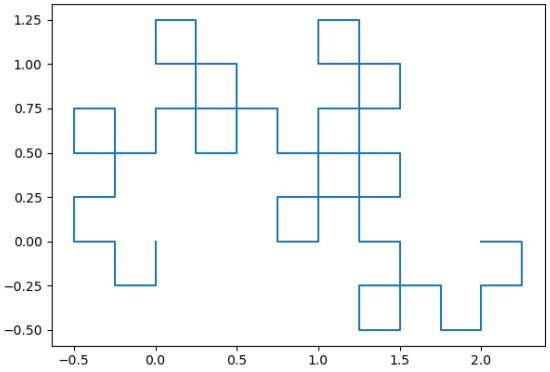
مرحله 3



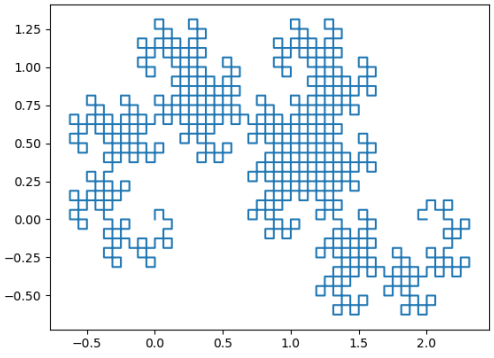
مرحله 5



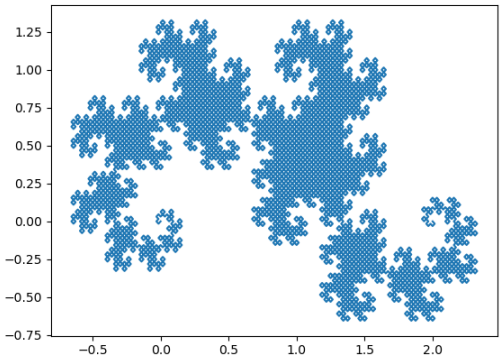
مرحله 7



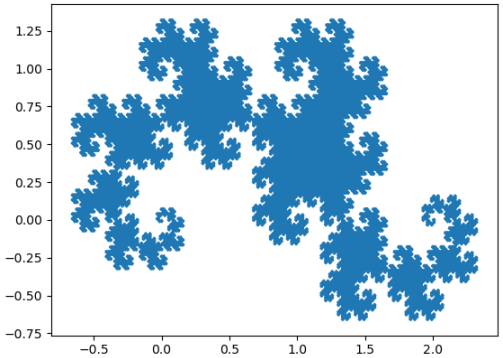
مرحله 11



مرحله 14



مرحله 18

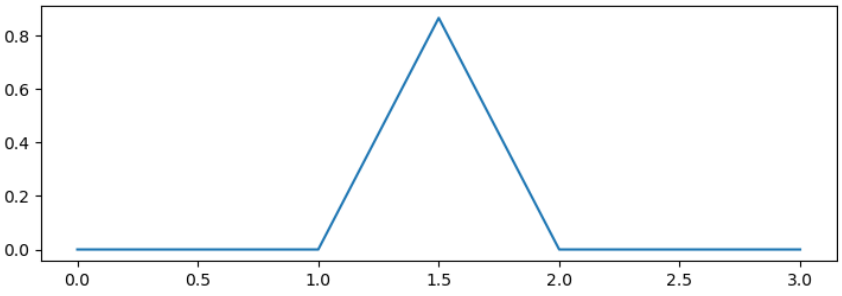


نام این فایل heighway dragon است.

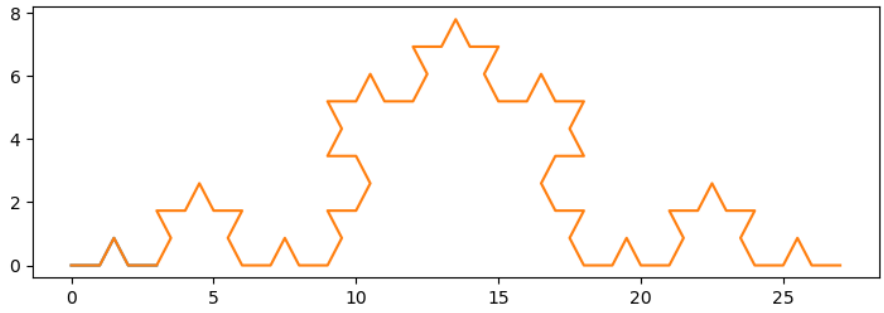
برفدانه کخ

چهار تابع تجانس-انتقال لازم است، اولی از چپ یک سوم میکند (من چون شکل اتوماتیک توی کادر جا میشد از تجانس استفاده نکردم و فقط چرخش ها را انجام دادم) دومی چهل و پنج درجه پادساعتگرد می چرخاند و منتقل میکند، سومی همان اما ساعت گرد و چهارمی تنها انتقال ایکس است.

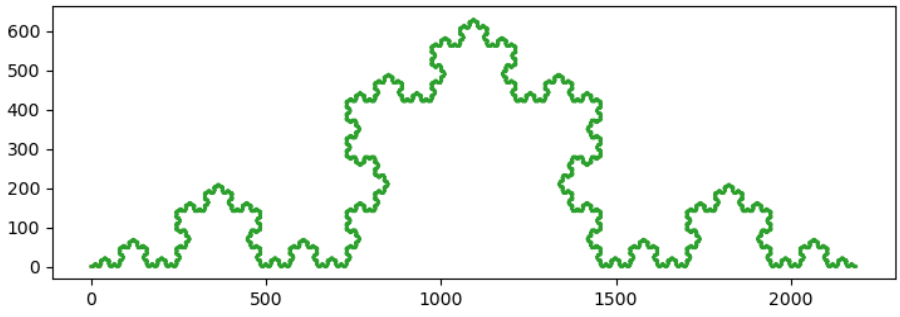
مرحله 1



مرحله 3



مرحله 8

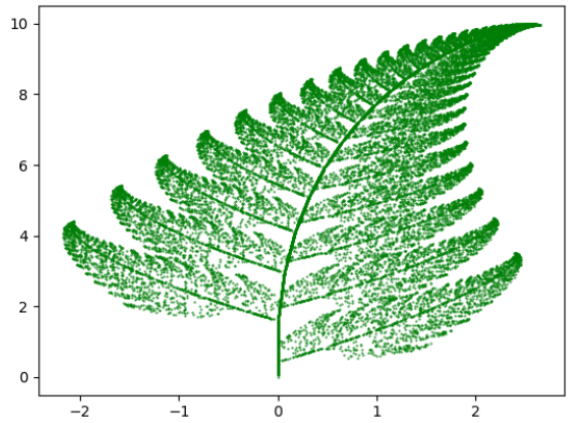


نام این فایل koch final است

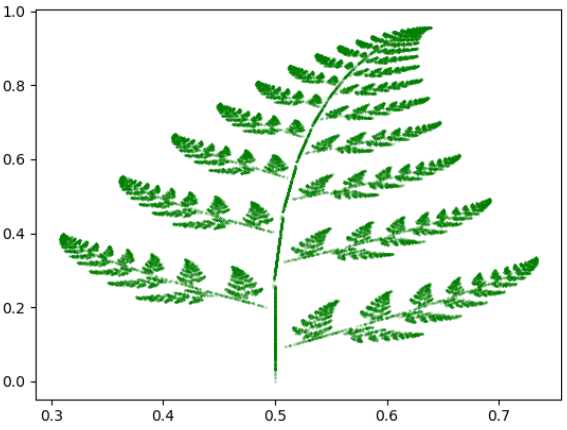
فرکتال سرخس

چهار تابع لازم داریم، یکی برای خطوط، دو تا برای دو برگ طرفین و چهارمی برای بقیه ی سرخس
از آنجایی که با توابع بازی کردم عکس هر سرخس را در کنار تابعش میگذارم؛ تعداد نقاط به کار رفته در تمام سرخس ها 30000 است.

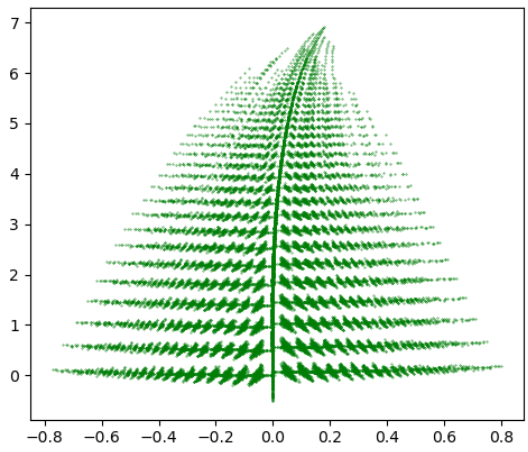
```
if z < 4:
    Arx.append(0)
    Ary.append(0.16*(Ary[count]))

if z>= 2 and z<= 86:
    Arx.append(0.85*(Arx[count]) + 0.04*(Ary[count]))
    Ary.append(-0.04*(Arx[count]) + 0.85*(Ary[count])+1.6)


if z>= 87 and z<= 93:
    Arx.append(0.2*(Arx[count]) - 0.26*(Ary[count]))
    Ary.append(0.23*(Arx[count]) + 0.22*(Ary[count])+1.6)


if z>= 94 and z<= 100:
    Arx.append(-0.15*(Arx[count]) + 0.28*(Ary[count]))
    Ary.append(0.26*(Arx[count]) + 0.24*(Ary[count])+0.44)

count += 1
```



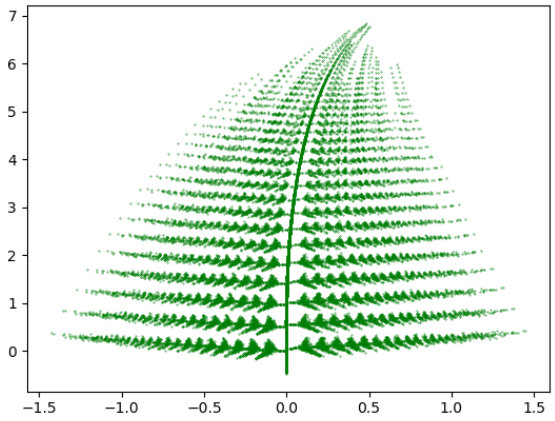

```
if z < 4:
    Arx.append(.5)
    Ary.append(0.25*(Ary[count]))

if z>= 2 and z<= 86:
    Arx.append(-0.14*(Arx[count]) + 0.26*(Ary[count])+.6)
    Ary.append(+0.25*(Arx[count]) - 0.22*(Ary[count])-.04)


if z>= 87 and z<= 93:
    Arx.append(0.2*(Arx[count]) - 0.21*(Ary[count])+.4)
    Ary.append(0.22*(Arx[count]) + 0.18*(Ary[count])+.09)


if z>= 94 and z<= 100:
    Arx.append(0.78*(Arx[count]) + 0.03*(Ary[count])+.11)
    Ary.append(-0.03*(Arx[count]) + 0.74*(Ary[count])+0.3)

count += 1
```



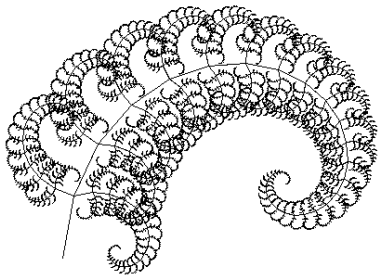

```
if z < 4:
    Arx.append(0)
    Ary.append(0.25*(Ary[count])-.4)

if z>= 2 and z<= 86:
    Arx.append(0.95*(Arx[count]) + 0.002*(Ary[count])-.002)
    Ary.append(-0.002*(Arx[count]) + 0.93*(Ary[count])+.5)


if z>= 87 and z<= 93:
    Arx.append(0.035*(Arx[count]) - 0.11*(Ary[count])-.05)
    Ary.append(0.27*(Arx[count]) + 0.01*(Ary[count])+.005)


if z>= 94 and z<= 100:
    Arx.append(-0.04*(Arx[count]) + 0.11*(Ary[count])+.047)
    Ary.append(0.27*(Arx[count]) + 0.01*(Ary[count])+.07)

count += 1
```



```
if z < 4:
    Arx.append(0)
    Ary.append(0.25*(Ary[count])-.4)

if z>= 2 and z<= 86:
    Arx.append(0.95*(Arx[count]) + 0.005*(Ary[count])-.002)
    Ary.append(-0.005*(Arx[count]) + 0.93*(Ary[count])+.5)


if z>= 87 and z<= 93:
    Arx.append(0.035*(Arx[count]) - 0.2*(Ary[count])-.09)
    Ary.append(0.16*(Arx[count]) + 0.04*(Ary[count])+.02)


if z>= 94 and z<= 100:
    Arx.append(-0.04*(Arx[count]) + 0.2*(Ary[count])+.1)
    Ary.append(0.15*(Arx[count]) + 0.04*(Ary[count])+.07)

count += 1

t.scatter(Arx, Ary, 0.1, edgecolor ='green')
```



در باب تفاوت دو تای آخر باید اشاره کنم که اولی تیغ هایش مثل هاشور است و دومی مثل استخوان ماهی.
خیلی وقت گذاشتم که الگویی پیدا کنم که سرخس **خیلی** پیچ بخورد مثل شکل زیر ولی آن طور که باید ست نمیشد، اگر کسی در آورده بود لطفا به اشتراک بگذارید *-*



نام فایل کد fern normal است