کورس udemy – 100 days of code

Day001

میتونیم برای کمک گرفتن در کورس با #100daysofcode توی تویتر مطلب پیدا کنیم.



میخوایم the band name generator بسازیم.

Hello world رو رفت و داره استرینگ توضیح میده .

توی pycharm داره پیش میره ، و جالبیش اینه که توی Pycharm میتونیم اون هایلایتی که زیر چیزای اشتباه میندازه رو هاور کنیم و نکته ش رو ببینیم. فکر کنم این توی vscode نیست.

-نکته جدید: خط آخر توی اکثر زبان ها بهتره که خالی باشه. وگرنه ai که چک میکنه اشکال تشخیص میده و Char آخر رو Highlight میکنه .

-یادآوری: \n برای رفتن به خط بعدی در خروجی

Concat با +

یه نرم افزار معرفی کرد به نام thonny که میتونیم باهاش کدی که نوشتیم رو ببینیم مرحله به مرحله توسط کامپیوتر چطور اجرا میشه. این جالب میتونه باشه و در ادامه استفاده های زیادی ازش بکنیم . شاید Pycharm و vscode هم داشته باشنشون .

تا 012 Variables

ادامه

همونطور که میتونیم باید از underline استفاده کنیم به جای camelCase .

یادآوری: len() برای گرفتن تعداد char ها.

-توی vscode توی اجرای کد هایی که توی هرجایی بجز c ذخیره شده به مشکل خوردیم . توی همون pycharm پیش میریم. یا عشقمون کشید با jupyter میریم .

تا انتهای day001

ادامه

Day002

Len() برای Int عمل نمیکنه و میگه که no length . باید conversion انجام بدیم روی Int ها وعدد ها اگر میخوام Len بگیریم .

با گذاشتن براکت جلوی هر چیزی که Length داره میتونه خروجی Index مربوطه رو داشته باشیم

Index منفی هم میتونیم استفاده کنیم و از آخر میشماره

Concatenation

نکته جالب: میتونیم بین اعداد Underline بزاریم به جای کاما تا خوندشون راحت تر بشه

Type() برای چک کردن data type

Type conversion در یسری حالتا غیر قابل انجامه.



یادآوری: اولی Float میده به صورت پیشفرض ، دومی ceil میکنه

\*\* توان

Round() روی اعداد >> گرد میکنه بسته به مقدار اعشار به بالا و یا به پایین 5 بشه به بالا

Round(x,x) مقدار دوم میتونیم اعلام کنیم تا چند رقم اعشار گرد کن.

Fstring : print(f”string{value}”

تمرین امروز



تا وسطای 009 Day 2 Project Tip Calculator

ادامه

ما برای تمرین روز دوم از number: .2f استفاده کردیم تا اعشارش رو محدود کنیم ولی خودش از round(x, x) استفاده کرده. اولی متغیری هست که میخوایم روند شه ، دومی مقدار روند کردنشه.

Day003 treasue-islad-end

تا 003 Control Flow with if else and Conditional Operators

ادامه از 003 Control Flow with if else and Conditional Operators

Modulo operator یا همون % خودمون. باقیمانده.

تمرین:



Nested if/else

تودرتو









-داخل pycharm یه سیستم تشخیص simplify داریم که میتونیم روش کلیک کنیم و همون کدی که نوشتیم رو ساده ترش رو بهمون بده .



<https://ascii.co.uk/art>

یادآوری: برای اینکه بتونیم یچیزی مثل صندوق ابتدای تمرین نهایی رو print() کنیم باید به جای “” از ‘’’ ‘’’ استفاده کنیم که 3 تا Single quotation میشه. این اجازه میده چنیدن خط رو یکجا پرینت بگیریم.

پروژه نهایی هم در انتها اینه که یه بازی ابتدا بر اساس diagram زیر بسازیم. بعدش اگر حست کشید بزرگتر و بسط داده ترش کن یا یکی دیگه بساز با یه diagram دیگه .



پایان day002

(پروژه day003 بازی سنگ کاغذ قیچیه. که بیشتر با مفهوم randomization و Python Lists قراره کار کنیم.)

ادامه

Day004

پروژه نهایی ساخت یک بازی سنگ کاغذ قیچی

Randomization

برای اینکه بتونیم از مقادیر random استفاده کنیم باید از module اضافه کنیم به برنامه :

Import random

که از.random() و .randint() استفاده میکنیم برای گرفتن یک مقدار random که البته فرق کوچیکی دارن که با یه سرچ در میاد (مثلا random خودِ صفر رو در نظر میگیره تا اون عددی که بهش دادی)

مفهوم module : میتونیم بخش هایی از برنامه رو



میتونیم ضرب هم کنیم که دو parameter رو ضرب میکنه از اونجایی که یکیشون صفره اونیکی فقط اعمال میشه عملا

یه برنامه نوشتیم واسه "شیر یا خط" که یه سکه پرت میکنیم.

Python list

یه data structure عه .



یادآوری: با Index میتونیم یک item رو return کنیم. با مقدار منفی از انتها شمرده میشه.

aList.add() و aList.extend() و...

میتونیم با random.choice() از item های یک List به صورت random returnداشته باشیم.

Nested list : میتونیم داخل یک List ، List تعریف کنیم (داخل index ها)

تا 007 day 004 03:17

ادامه

برای اینکه بتونیم برنامه ای که نوشتیم رو کاری کنیم دائم در حال اجرا باشه و در صورت اینکه کاربر قصد پایان داشت خاتمه پیدا کنه باید از ساختار کلی زیر در پایتون استفاده کنیم (که روی پروژه روز 4 سنگ کاغذ قیچی پیاده ش کردیم) :



حالا باید بریم ببینیم روش خودش چیه

1-ascii art هارو داخل یک List ذخیره کرد که با مقادیر وارد شده توسط user مطابقت داره که باعث میشه فقط بنویسه List\_name [user\_choice] که برای پرینت کردن شکل ها خیلی راحت تره. (کاری که ما کردیم این بود که یه module براش ساختیم و هر دفعه برای پرینت کردن هرکدوم باید این رو مینوشتیم: module\_name.scissors و یدور هم نیاز به Import module\_name داشت. که خب کاری که خودش کرده راحت تره)

ادامه

جلسه قبل در طی ور رفتن با گیت فایل هایی که نوشته بودیم پرید ، ولی از دوباره یه repository ساختیم که اونجا Push کنیم و نیاز نیست از دوباره کد هارو بزنیم مربوط به ابتدا رو ، فقط از اینجا به بعد رو انجام میدیم .

Day005

پروژه امروز PyPassword generator هست .



جالب: یه سایت معرفی کرد برای اینکه چک کنیم email رو و اگر شرکتی مورد هک واقع شده و اطلاعات شما هم اونجا بوده و رفته بهتون بگه. ([link](https://haveibeenpwned.com/))

خب ما دوتا راه داریم این کارو انجام بدیم: easy version و hard version

ابتدا میریم سراغ easy .

در easy به ترتیبِ حروف ، سمبل، عدد رفتیم ولی در hard باید بدون ترتیب مشخص پسوورد ساخته میشد. که ما از خودمون با random درستش کردیم. ولی داخلِ خودِ random یه method هست به نام shuffle() که همین کار رو انجام میده .

Random.shuffle(List\_name) یک Object رو به صورت درهم خروجی میده.

در hard version میتونستیم از روش های مختلفی بریم. از اونجایی که easy version رو رفته بودیم پس یعنی تعداد در اون درسته ، فقط میمونه ترتیب رو به هم بریزیم که با random.shuffle انجام میشه . چیزی که ابتدا تو ذهنم بود این بود که برای هر جایگاه به صورت تصادفی از یکی از لیست ها انتخاب بشه و بعد به صورت اتفاقی از داخل همون لیست انتخاب بشه و این بره تا جایی که ورودی های کاربر تشکیل بشه. یعنی وقتی تعداد Number هایی که کاربر میخواست touch شد دیگه اون لیست رو داخل امکانات نزاره.

ادامه

قبل از اینکه ادامه بدیم برای اینکه این چیزایی که بلدیم رو سریع بزنیم بره بعدی، چطوره همون ابتدا سعی کنیم پروژه پایانی رو بسازیم. اگر تونستیم میتونیم اون روز رو Skip کنیم.

Day006

با یه تمرین باحال روبروییم اینجا:

<https://reeborg.ca/reeborg.html?lang=en&mode=python&menu=worlds%2Fmenus%2Freeborg_intro_en.json&name=Hurdle%201&url=worlds%2Ftutorial_en%2Fhurdle1.json>

که باید با کد های پایتون این رباته رو ببریم جایی که پرچم هست.



تا مرحله hurdle4 میره و بعدش میره سراغ maze که همه رو حل کردیم کداش هست. (برای مرور خوبه یه دور چک کنی)

3 تا مسئله هم برای debug داره که اونا هم حل کردیم ولی یه راه خودش رفت که برای سه تاشون جواب بود و اون رو نمینویسم اینجا توی کد ها هم نیست. سعی کن بنویسیش توی مرور.

(



این قطعه کد رو میزاره قبل از تابع اصلی تا قبل از اینکه شروع به هرکاری کنه، مطمئن بشه دیوار رو پیدا میکنه چون الگوریتم بر این اساسه که دیوار سمت راستشو بگیره و بره. لازم به ذکره داریم فقط و فقط راجب maze صحبت میکنیم.

)

ادامه

Day007

جلسه قبلی project نداشت و فقط درس بود. اما این جلسه Hangman رو داریم.

طبق چیزی که جلسه قبلی گفتیم سعی میکنیم خودمون بزنیمش اگر نتونستیم میریم با خودش پیش میریم.

خب چیزی که میبینم اینه که به تنهایی از این یکی بر نمیایم. با خودش پیش میریم. (روی این با خودش پیش میریم خیلی تاکید میکنم، چیزی رو skip نکن اگر مردی سرعت ببخش بهش. توی این تمرین ها قراره مسیر رو یادمون بده و راه و روش رو. اگر یه بخشی رو بلدی دلیل نمیشه بقیه رو هم بلد باشی. )

جالب اینه که Flow chart کشیدن رو یه مرحله از برنامه نویسی میدونه و باید قبل از کد زدن انجامش داد. با ابزار draw.io آنلاین میتونیم این کار رو بکنیم.

توضیح بازی:

اینطوریه که یه کلمه باید حدس زده بشه. تعداد حدس ها هم یه تعداد محدودیه (هربار که اشتباه حدس زده بشه یه قسمت از شکل hangman (انسانی که به دار آویزونه) رو میکشه و هروقت شکل کامل بشه یعنی ما باختیم و اگر قبل از اینکه جونامون تموم بشه کلمه رو حدس بزنیم ، بردیم .

حالا باید با این توضیحات و چیزی که دیدیم یه flow chart بکشیم که این پروژه چطور کار میکنه Logically

Enumerate() میتونیم str بفرستیم داخلش و به عنوان خروجی بهمون index و char رو میده میتونیم استفاده کنیم . (که با استفاده از اون داخل یک for میتونیم iterate کنیم و برای هرکدوم داشته باشیمش. )

با ذهن خسته نمیشه کار کرد ، از یه مسیر خیلی ساده ای میشد مشکل حل بشه که داخل ویدیو خودش رفت و ما مدتی روش فکر کردیم ، نکته اصلی اینه که نباید سعی کنیم جلو جلو بریم ، وقتی داریم step step میریم باید باهاش پیش بریم و کاری که خواسته رو بکنیم.

باقیش برای فردا

تا انتهای 004 از day007

ادامه

-نکته: میتونیم برای اینکه در مراحل مختلف برنامه cls بگیریم داخل کد از :



استفاده کنیم. البته باید import subprocess هم بکنیم.

پیشنهاد: میتونیم برای تمرین یا مرور سعی کنیم از ابتدا project های هر روزی که هست میخوایم مرورش کنیم رو بزنیم از ابتدا. دفعه بعدی این کار رو بکن ببین efficient هست از لحاظ زمانی یا نه

تا انتهای day007

ادامه

باید بدو بدو بریم تا جلسه 15. در حداقلی ترین و سریع ترین حالت ممکن پیش میریم. بجز نکات مهم چیزی نمینویسیم اینجا .

Day008

Encode decode کردن با shift کردن به تعداد مشخص (Caesar cypher )

یادآوری: میشه در تعریف function به جای Positional arg از حالت parameter=”value” استفاده کرد که دیگه ترتیب مهم نباشه. ولی خب باید Parameter هارو بدونیم.



(عکس بالا برای شیف کردن استفاده شد که به جای اینکه چندین سری از alphabet رو اد کنیم توی List که جا داشته باشه و ادامه بده به میزان اون عدد ، اینکار باعث میشه user هر عددی خواست وارد کنه و معادلش در حالت شیف شده بدست بیاد. )

تا day008 007 12:31

ادامه

برای نوشتن Caesar() یه ترفند جالبی زد که با توجه به فرایند فقط نیاز بود shif \*= -1 بشه تا decrypt انجام بشه. همین. لازم نبود مثل چیزی که خودمون نوشتیم یه func جدا تعریف کنیم.

پایان day008

Day009

Dictionaries and nesting

Silent auction program

یادآوری dictionary\_name = {“key” : “value”, “key2”: “value”, … }

برای return کردن: dictionary\_name[“key”]

برای مقدار اضافه کردن : dictionary\_name[“new\_key\_name”]= “value”

برای تغیر یک key : dictionary\_name[“key\_for\_change”] = “changed value”

-در iteration ها key return میشه

-برای بیشترین مقدار max(dictionary\_name)

تا ابتدای day010

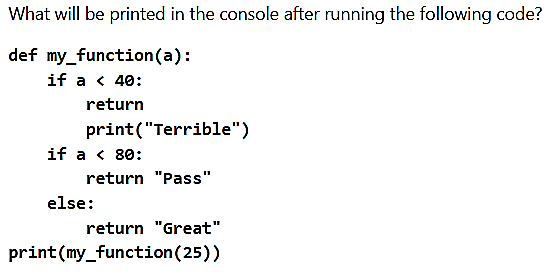
ادامه

‘string’.title() کاری میکنه string مون هرجوری از لحاظ بزرگی و کوچیکی بود اولین char بزرگ بشه بقیش کوچیک

-یادآوری: وقتی return انجام میشه func تموم میشه.

-یادآوری: برای function های داخلی python ما docstring داریم که وقتی روش hover میکنیم اون رو نشون میده و بهمون میگه که این function در واقع چکار میکنه چیا میگیره با چه data type ای و چی پس میده و ... ، ما برای Function هایی که خودمون میسازیم هم میتونیم این رو ایجاد کنیم با تنها اضافه کردن یک string به خط اول function مون. که اگر تعداد خطوط زیاده میتونیم از ‘’’’’’’ استفاده میکنیم. بعد وقتی Hover میکنیم اون چیزی که تو توضیحات نوشتیم نشون داده میشه. توی تمرینات exercism هم داشتیم میتونیم بریم اونجا نکات بیشتری یادمه داریم./

-نکته: در ازای این کد ها None میگیریم:



چرا؟ چون 25 که وارد میشه اولین return که جلوش هیچی نیست و میشه None اجرا میشه چراکه شرطش برقراره و سراغ باقی کد ها نمیره.

یادآوری: تغیر نام function چه داخلی چه چیزی که خودمون نوشتیم فقط باید یک variable بنویسیم و اسم اون function رو بدون پرانتز (چون پرانتز بزاریم call میشه) بزاریم اون داخل. از این به بعد با اسمی که برای اون variable گذاشتیم هم میتونیم function call کنیم که میشه توی ساختاری مثل این ازش استفاده کرد:



(فقط باید مواظب باشیم که نباید در حین این کار Function call انجام بشه ، یعنی نباید پرانتز بزاریم وقتی میخوایم تغیر نام بدیم . بدون پرانتز باید استفاده کنیم .)

تا day010-007-12:13

ادامه

Day011

Black jack game

این بازی رو قبلا ساختیم . ولی سراغ چیزایی که قبلا نوشتیم نمیریم و از ابتدا باید بسازیم

Flow chart رو ساختیم

تا day011 002 end

جلسه بعد از روی flow chart کد بزن

ادامه

الان میخوایم از روی flow chart خودمون کدش رو بزنیم

نکته: 

این رو AI داده برای ascii art هر ورق، شاید بعدا ادش کردم توش:



-برای اینکه جمع همۀ عناصر یک list رو داشته باشیم sum()

ادامه



ما به ترتیب به هردو کارت میدادیم ولی اینجا توی دوره اول به user کارت میده بعد شروع میکنه به pc کارت میده تا زمانی که pc < 17 باشه کارت دادن ادامه داره. البته واسه زمانیه که ما گفتیم دیگه کارت نمیخوایم به عنوان user. یعنی کارت دادن به pc بعد از user اتفاق میوفته.

ادامه

Day012

Number guessing project

درمورد Scope و اهمیتش گفت.

نکته یادآوری مهم : برای اینکه از داخل یک function بخوایم یک global var رو Modify کنیم باید یبار داخل Scope ای که برای function داریم ذکر کنیم که یک متغیری هست به این نام و global هست و ما میخوایم اون رو تغیر بدیم. که میشه global variable\_name و این باید داخل scope باشه.

-البته میگه که اینکارو نکنیم بهتره، چون باعث باگ میشه و دیباگ کردنش سخته . اگر خواستیم Modify انجام بدیم روی یچیز global با Function بهتره از return استفاده کنیم و یجوری return کنیم که وقتی برابر با اون متغیر قرارش میدیم اون Functionality ای رو داشته باشه که میخوایم . ولی درمورد const variable فرق میکنه .

یادآوری: Const چیزیه که define میکنیم و نمیخوایم که تغیرش بدیم . در این صورت میتونیم global رو اعلام کنیم ولی باید این رو یادمون باشه که تغیر دادنش ممکنه، یعنی مثل باقی زبان ها نیست که نشه تغیرش داد، عملا همون متغیر عادیه. ولی با حروف بزرگ نوشته میشه که یادِ خودمون بمونه که این const عه . (و اینطوری هم هست که const میسازی و داخل function ها اگر خواستی استفاده کنی global میزاری پشتش و تمام. هر بلایی هم داخل سرش بیاری هرجایی برای ابتدا نوشتی ایجادش کردی اونجا هم در نظر گرفته میشه. )

-علاوه بر اون سایت که برای ascii art معرفی کرد که حاضر آماده بود ، این سایته هم هست:

<https://patorjk.com/software/taag/#p=display&f=Ghost&t=Type%20Something%20>

که میشه بری هرچیزی که مینویسی رو برات ascii art کنه بهت بده

انتهای day012

ادامه

Day013

درمورد روش های debug داره صحبت میکنه. چطور مشکل رو حل کنیم.

1.اگر مشکل دارید، اول باید سعی کنیم مشکل رو توضیح بدیم برای خودمون و برای خودمون حداقل روشن کنیم که قضیه از چه قراره. جمله ای که گفت : (تقریبا غیر ممکنه وقتی که خودتون از مشکل خبر ندارید بتونید حلش کنید)

2. چیزی که نوشتید ببینید دقیقا چیکار میکنه، چیکار میخواستید بکنه باید چه چیزی رو تغیر بدید

نکته مهمی هم که وجود داره اینه که مشکلی رو میشه حل کرد که بشه از دوباره خلقش کرد. به تعبیری بدونی در ازای چی ارور میگیری یا مشکل داری دراون صورت میتونی حلش کنی. با تغیر دادن تصادفی و بدون حساب کتاب گاها شاید مشکل حل بشه ولی توی روند فرسایشی میوفتیم .

یادآوری ساختار try except ، وقتی که یه ارور توی یک بخش مشخص از کد پیش میاد و میدونیم ارور چیه (مثلا ValueError) اون رو بگیریم و تبدیل کنیم به یه پیام مشخص تر یا یه عملیات کاربردی در طی پیش اومدن اون مشکل: (این نکته هم قابل توجهه که یک function داخلی هم از همین قضیه برای هندل کردن ارور های مختلف استفاده میکنه این رو توی step into فهمیدیم)



و این هم هست که در طی این try except کد در محل error بلاک نمیشه و روند تا انتها ادامه پیدا میکنه .

نحوه کار با debugger هم اینطوریه که شما ران میکنی و این خط به خط اجرای کد رو بهت نشون میده. یه step over داریم و step in و توی Pycharm یدونه داریم که داخل Module های Inner نمیره و فقط module های خودمون رو برسی میکنه . اینطوری میتوینم خط به خط ببینیم چی میشه و اگر یسری خطوط رو میخوایم بزنیم بره جلو از breakpoint همون نقطه قرمزه که بقل عدد های خطوط کد میشه اضافه ش کرد قرار میشه داد. از اونجایی که breakpoint هست Debug شروع میشه .

(پس این رو میتونیم به عنوان یه نکته از این به بعد داشته باشیم که 1.همیشه کامنت کن کاری که قراره توی اون بخش کد انجام بدی و قراره بنویسیش 2.اگر به مشکل خوردی از debugger استفاده کن تا مشکل رو پیدا کنی. )

اهمیت Debugger از اهمیت print() میاد، که وقتی خروجی میگیری از چیزی که نوشتی بهتر میتونی بفهمی چه اتفاقی داره میوفته. حالا debugger در عمل داره برای همه چی Print() استفاده میکنه و خط به خط نشونت میده چه اتفاقی داره میوفته. با استفاده کردن مکرر از debugger که در IDE های مختلف ساختارش یکیه، میتونیم مهارتمون رو هی بهتر و بهتر کنیم. سر کلاس های c++ , java بهش میگفتیم تحلیلِ برنامه ، که خط به خط میرفتیم و برسی میکردیم چه اتفاقی داره میوفته. جالبه

نکته بعد هم درمورد رفع مشکل و ارور، اینه که اگر هرکاری میکنی نمیشه ، شاید اون روز روزش نیست و باید استراحت کنی و یه مقدار زمان بدی بهش و بعد بیای سراغش. ما این رو میدونیم که بعد از گذروندن زمان و چرت زدن و خوابیدن و ... مغز ما این مهلت رو پیدا میکنه که بین اطلاعات و بخش های مختلف ارتباط برقرار کنه و اینطوریه که ما چیز هارو میفهمیم و به خاطر میسپاریم، این که کد زدنه درمورد ورزش و فعالیت های فیزیکی هم قضیه همینه، پس زمان دادن به مغز برای رفع مشکل خیلی مهمه.

در مرحله بعدی هم اگر نتونستی باید از یه ذهنِ انسانی دیگه بپرسی

نکته بعدی هم این که بعد از اینکه بخشی اضافه شد به کد و نوشتیم و پیاده ش کردیم ، بهترین کار همینه که همون لحظه یه اجرا ازش بگیریم و ببینیم چی پیش میاد.

باگ ها پیش میان، نکته اصلی اینه که حلش کنی و مهمتر اینکه راه های مختلف برای حل کردنش رو بلد باشی، چه مستقیم چه غیر مستقیم .

انتهای Day013

Day014

Higher lower game

این یه بازیه که باید خودمون از صفر تا صد بسازیم.

تا الان flow chart رو کشیدیم ، کدش هم زدیم ، مونده debug کردنش، مشکلش هم نوشتم توی calendar در مراحل پایانی مشکل داریم ، دونه دونه هم مرحله هایی که درست انجام دادیم رو commit کردیم . از این جهت مشکلی نداریم.

جلسه بعدی اول باید کاری کنیم این بازی درست کار کنه که کار زیادی نمیبره و بعدش ویدیو هاشو میبینیم.

ادامه

الان قراره که debug کنیم. مشکل چیست؟

مشکل هایی که حل کردیم اولیش این بود که debug کردیم مشکلِ حذف شدن score درست بود. یعنی نمیرفت بشینه جای A و در عمل از ابتدا تشکیل میشد. و یسری قابلیت های کلی به برنامه اضافه کردیم و عملا نکته خاصی نبود و در انتها بازی باحالی شد (البته اگر خودمون بریم یسری data درست و درمون بزاریم توی database ش . )

Day015

Coffee machine project

001 رو دیدیم توضیحات اولیه بود برای برنامه coffee machine .

جلسه بعدی pdf رو چک میکنیم، flowchart میکشیم و کد میزنیم.

ادامه

یه وقفه طولانی مدت داشتیم سر مسائل درسی و کاری و ...

قرار بر این بوده که بریم یه سر سراغ Pdf و خودمون Flowchart بکشیم و ادامه ماجرا ولی چون فاصله افتاده قصد دارم قسمت اول رو دوباره ببینم.

این اطلاعات یه Coffee machine ساده س:



این اطلاعات 3 حالت مختلفیه که باید تولید کنه:



منابعی که در ابتدا در اختیار داره:



ارزش سکه ها:



طبق اطلاعات بالا و چیزایی که اد کردیم تو کدش همین الان باید جلسه بعدی پیش بریم و سعی کنیم که کدی بنویسیم که وقتی با PDF این قسمت چکش میکنیم درست کار کنه. ادامه ش جلسه بعد.

ادامه

درحال تکمیل پروژه coffee machine هستیم ، خیلی طولانی شده ، ولی بالاخره انجاامش میدیم.

خب در حال حاضر coffee machine رو انجام دادیم تا حدی که میخواست داره کار میکنه، البته خودم شاید بعدا یسری چیزا بهش اضافه کنم.

در ادامه باید با Pdf ای که گذاشته برنامه رو چک کنیم و چیزایی که میخواد رو اضافه کنیم و چیزایی که نداره رو اصلاح کنیم. که باشه برای جلسه بعد.

ادامه

باید مرحله به مرحله گزینه های pdf این قسمت (برای یادآوری الان day015 ایم. ) رو چک کنیم و باید طوری باشه که برنامه طبق اون چیزی که توی هر مرحله گفته درست عمل کنه .

Off رو اضافه کردیم.

درحال اضافه کردن کد های درست برای پیام out of resource بودیم که نت رفت. باقیش بعدا.

ادامه

درحال درست کردن این هستیم که وقتی یچیزی از منابع کمه اون رو توی پیام اطلاع بده، نه اینکه مثل برنامه خودمون سر راست بگه که منابع کافی نیست، یعنی بنویسه water کافی نیست. در طی این میخواستیم که عناصر یک list رو کنار هم پرینت کنیم که باید از

استفاده کنیم تا بدون داشتن [()] پرینت بشن . (کاری که این میکنه اینه که عناصر رو به هم دیگه متصل میکنه ، چیزی که در ابتدا به عنوان string گذاشتیم رو بینشون قرار میده و همه رو در قالب یک string ریترن میکنه. )

نکته: یه راه حل ai گفت متاسفانه نتونستیم انجامش بدیم. مثل اینکه با گذاشتن یک \* پشت اسمِ list میتونیم تمامی عناصرش رو پرینت بگیریم. ولی خب چرا توی اون فرمتی که ما میخواستیم نشد رو نمیدونم (الان تستش کردم ، انجام میده ولی با f-string مشکل داره، دلیلشم اینه که میشه String رو با string کانکت کرد ولی نمیشه با یه list این کار رو انجام داد از همین جهته که Python مشکل داره باهاش. باید با همون join ابتدا خروجی str بگیریم و بعد استفاده کنیم. )



(این هم در نوع خودش چیز جالبیه، الان به کارمون نمیاد ولی در آینده شاید یکی از راه حل های ساخت class باشه برامون. )

-نکته: یه چیز جالب یاد گرفتیم. توی vscode و البته pycharm میتونیم با نگه داشتن shift + alt و تکون دادن موس از یه ستون مشخص انتخاب کنیم و همشون رو یکجا تغیر بدیم. جالب بود.

-الان داریم برسی میکنیم که ببینیم اون چطوری نوشته و ما چطوری نوشتیم و کد زدن و کارای خودمون تموم شده، همش هم با Pdf چک کردیم درست بوده .

یه فرقی که داره اینه که من برای پرینت کردن report حلقه نوشتم و for و همه چی رو با منطق چفت کردم که هر resource دیگه ای هم اضافه کنی بهش درست کار میکنه، ولی خودش فقط از Print() کردنشون استفاده کرده. که خب راحت تر و ساده تره ولی کاربردی نیست. بگذریم . (البته الان که دارم فکر میکنم همچین کار شاقی هم نکردیم، چراکه فقط برای water, coffee, milk, money درست عمل میکنه، خب در این حالت روش دوره راحت تر و درست تره)

فرق دیگه اینه که ما تلاش کردیم کد برسی کنه کدوم resource کمه و اون رو دقیق توی همون خط پرینت کنه که توی خودِ دوره خط به خط پرینت میکنه مثل اینکه. چراکه توی loop گذاشتش. بگذریم.

فرق دیگه: برای Process coins یدونه function قرار داد. ما توی خودِ کد انجامش دادیم. هرچند منتقل کردنش راحته. (تا به اینجا لزومی نمیبینم بخوایم function ش کنیم. البته در OOP و دیدگاهش باید این کار رو بکنیم تا بعدا هم بتونیم استفاده کنیم ولی خب)

فرق دیگه: برای اینکه in\_change رو نشون بده از round(,) استفاده کرده که ما از در نظر نگرفتن یسری از واحد های اعشار در حالت float استفاده کردیم :

که خط کد بالا باعث میشه فقط دو عدد اعشارش نمایش داده بشه و وقتی هم که عدد اعشار نداره صفر میزاره به جاش (که در این حالت روند میشه به سمت پایین و عملا floor() انجام میشه. )

Day016 – OOP programming

الان میخوایم بریم سراغ oop .

به نوعی از برنامه نویسی که تا اینجای کار انجام دادیم میگن procedural programming ، که فکر میکنم فارسیش میشه ساخت یافته. که میشه ترکیبی از Process function که وسطاش call میشه و ... و باعث میشه در برنامه های complex شبیه کلاف تو در تو بشه و به یادآوردن ارتباط function ها و کلا همه چی سخت بشه. چیزی نیست که امروزه روز استفاده بشه.

برای یادآوری باید بگم که مفهوم OOP برای این به کار میره که قابلیت scale خیلی زیادی داره، که در نوع قبلی این وجود نداره به اون صورت. به صورت بخش بخش و Object object وقتی به یک مسئله بزرگ نگاه کنی میتونی بهتر عمل کنی.

یادآوری: attributes همون variable هست که مربوط به یه object بخصوصه . method همون function عه که مخصوصه یه object خاصه.

یادآوری: class به حالت کلی برای Object های مختلف میگن. در عمل object ها از روی class و با پر شدن اونا به طرق مختلف بسته به نوع نیاز از اون class ساخته میشن. که Object های ساخته شده از یک class دارای attribute و method های یکسان هستن ولی با data متفاوت

مثل اینکه اسمِ class هارو در پایتون به صورت camelCase نام گذاری میکنیم. برای اینکه یادمون باشه این class عه . (این درسته کاملا با ai چک کردم، همچنین کامل با حروف بزرگ نوشتن برای const ها و همون حالت عادی برای variable ها و function های عادیه. camelCase هم برای class ها. )

یه چیز جالب و هیجان انگیز اینه که ما قراره از اینجا به بعد حالت های گرافیکی داشته باشیم و از کدِ خالص در بیایم. با استفاده از یه library به نام turtle graphic که گفته شد به صورت پیش فرض روی همه Python هایی که نصب میکنیم هست.

برای ساخت Object، میریزیمش توی یک متغیر دلخواه . (درست ترش اینه که MyClassName() رو میریزیم داخل یک متغیر و داخل پرانتزش بهش مقادیر لازم برای اون Object بخصوص رو میدیم. )

وقتی پرانتز داریم یعنی method داریم.

این یه window باز میکنه که توش یه فلش هست که میشه بهش شکل داد رنگ داد و کلا همه بلایی سرش اوورد و طوری که توی خود vscode نوشته بود یچیزیه که برای آموزش کد نویسی به بچه ها درست شده ، ولی خب جالبه دیگه

میتونیم از آدرس : docs.python.org/3/library/turtle.html

بیشتر وارد جزئیات این library بشیم ولی فعلا همین چیزاییه که گفتم . اینم قبلتر گفت ولی من فکر کردم داره برای ساختنش میگه نگو کلا اینطوریه، گفت فکر کنید یه لاکپشته که میشه شکلشو تغیر داد به کلی و یه قلمو بسته شده به پشتش و توی صفحه بالا میاد و میتونیم بهش دستور بدیم و چیز میزاشو تغیر بدیم و این حرکت کنه و در انتها یچیزی روی اون صفحه ای که بالا اومده بکشه.

باقیش جلسه بعد. (فکر میکنم باید تلاش میکردیم رنگ قرمز بدیم به لاکپشته.)

ادامه

نکته: درمورد module, package, library : یک فایل از کد ها و function ها و variable ها میشه یک module. که میشه در باقی برنامه ها Import کرد. یک package میشه یک پوشه یا بهتره بگیم یه مجموعه ای از Module های به هم مربوط برای مصرفی خاص. که معمولا یک فایل با نام:

داره که به خودِ python میفهمونه این باید به عنوان یک Package شناخته بشه. و در انتها Library هم میشه مجموعه ای package ها و Module های به هم مربوط برای یک مصرف خاص که از قبل آماده شده میشه import کرد و استفاده کرد. به نحوی باید گفت که کدیه که زده شده دیگه لزومی نداره باقی افراد هم بزنن . نکته دیگه ای هم که هست اینه که مثلا این رابطه در همه زبان ها وجود داره ولی شاید اسم های مختلفی بهشون اطلاق کنن مثلا در ruby به package ها میگن gems . جالبه.

-نکته بعدی اینکه داره درمورد یه موضوعی صحبت میکنه که بتونه باهاش استفاده از یک Package رو توضیح بده . مثلا پرینت کردن یک table به شکل ascii که اگر بخوایم خودمون این کار رو انجام بدیم خیلی نفس گیر خواهد بود. رسید به اینکه یه library یا package معرفی کرد به نام pypi که در اون علاوه بر اینکه میشه package های مختلف با کاربرد های مختلف پیدا کرد و document شو خوند و استفاده کرد به رایگان، یدونه برای این کار وجود داره به اسم PrettyTable . چون داره از روی pycharm پیش میره روش pycharm ش رو گفت که میری از تو تنظیمات و سرچش میکنی و Install رو میزنی و ... ولی ما که روی vscode ایم و با Pip اشنایی داریم تست کردیم و بله ... با کد زیر نصب کردیم آخرین حالت رو:

و تمام. پس روش نصب کردن یک package اینه که اسمشو بدونیم و pip install packagename

-(درمورد Pypi باید بگیم که این یه Library محسوب میشه که عمومیه و همه داخلش کدهاشونو اضافه میکنن و همه هم میتونن بردارن و استفاده کنن، و مستقیم مثل اینکه وصله به Pip ، این رو نمیدونم که Pip به چندتا Library دیگه وصله.)

خب الان تا انتهای قسمت 7 از روز 16 پیش رفتیم، و الان قراره با فایل های آماده ای که خودِ امروز داره و استفاده از oop ، همون پروژه دیروزی رو بسازیم. بریم

تا ابتدای 008

ادامه

کداشو به اصطلاح refactor کردیم، همونطوری کار میکنه (که در روش procedural programing کار میکرد) فقط با کدهای خودِ دوره انجام شده .

نکته خاصی نداره به اون صورت، بیشتر نکته همون استفاده از کدها و package های ساخته شده توسط دیگری با توجه به document (این کدهایی که نوشته بود یه فایل pdf به عنوان document داشت که از روش بتونی بسازی) ای هست که درموردش در اختیار داریم. یعنی اینطوریه که مثل یه ماشین لزومی نداره بدونیم چه اتفاقی اون داخل میوفته، فقط لازمه ورودی و نوعش رو بدونیم و بدونیم در انتها چه خروجی ای در ازای ورودی و با چه نوعی بهمون میده (نحوه کار کردن با اون رو باید یاد بگیریم.)

پایان روز 16

Day017

Creating classes

Project : quiz game

قراره بریم سراغ ساخت class های خودمون. در day قبلی داشتیم استفاده کردن از class های آماده رو یاد میگرفتیم که حالا یا built-in بودن یا با package بودن و Install شون کردیم. الان قراره خودمون بسازیم. هرچند بلدیم ولی خب باید ببینیم و یادآوری بشه.

-برای اینکه یه function یا class رو خالی بزاریم و همچنان بتونیم ادامه بدیم و بعدا پرش کنیم روش درست گذاشتن pass هست. خودمون یه استرینگ خالی میزاشتیم.

-یه چیزی قبلتر درمورد نوع نام گذاری گفتیم که تنها یچیزش اشتباه بود، اونم اینه که برای class ها از camelCase استفاده نمیکنن از PascalCase استفاده میکنن. که اولش بزرگه حتما . گفت از camelCase اصلا در پایتون استفاده نمیشه. (که خب این خودش جالبه میتونه برای متغیر های خاصی که فقط خودمون باید بدونیم استفاده بشه. بگذریم)

یادآوری: همونطور که گفتیم، یک class دارای attribute و method() هست. یه راه اضافه کردن attribute اینه که زارت بهش اشاره کنیم و مقدار بدیم، اون attribute ساخته میشه. فکر میکنم برای method هم همین باشه. ولی یه راه دیگش constructor عه.

یادآوری: برای ساخت یک class :

و داخلش هرچی بنویسیم جزو اون کلاس میشه.

برای ساخت constructor :



که این تابع همون constructor رو میسازه. دوتا underscore ش مهمه.

نکته ش هم اینه که هربار که class صدا زده بشه این تابع کال میشه.

Syntax برای ساختن attribute در class \_\_init\_\_() :



اینطوریه که self میشه همون object که داره ساخته میشه با این init ، خب اولیش که self عه هیچی . هرچی parameter اضافه کنیم در داخل function متصلش میکنیم به همین object درحال ساخت به صورتی که نوشته.

نکته: وقتی یه class کال میشه حتما اولش constructor ش کال میشه و بسته به اینکه چندتا arg میخواد باید حتما بهش ارسال بشه، اگر نشه ارور میگیریم و object ساخته نمیشه، پس میتونیم این طوری در نظر بگیریم که وقتی constructor مینویسیم داریم یچیز دائمی مینویسیم که باید حتما اون Object برای ساخته شدن داشته باشه و یجورایی اون اطلاعات پیش نیازش هستن. و زمانی که نمیخوایم حتما اون اطلاعات رو در ابتدا به صورت arg ارسال کنیم ، ولی نیازه که باشه از default value استفاده میکنیم و دیگه توی اون فرمت به عنوان parameter اضافه نمیکنیم، فقط self.myValue = defaultValue .

##عاها، یه سوالی هم که اینجا مطرحه اینه که فرض بگیریم یه class ساختیم و ازش استفاده کردیم و چندین object ازش ساختیم و داریم استفاده میکنیم ، بعد میخوایم به class مون یسری attribute و البته function اضافه کنیم، اونموقع چطور میشه؟ (در نگاه اول که میشه ، قطعا روش های خاصی هم وجود باید داشته باشه که این کارو راحت کنه، ولی راه حلی که خودم الان به ذهنم میرسه اینه که یه Loop بنویسیم که اون مقادیر رو برای Object های قبلی پیش فرض یه مقداری قرار بده یا با فرایندی پرشون کنه .)

برای ساختِ method() هم basic یکسانه، از اونجایی که ما یه self داریم باید همه چی رو به اون attach کنیم ، حالا یا با arg یا بصورت default. درکل باید با self کار کنیم.

برای مثال از ساخت و استفاده از class, object, method, attribute در عکس پایین فرایند فالو و فالوینگ اینستا رو تا حدی شبیه سازی کرده، خیلی جالبه:



که خب در basic ای که برای یک class مینویسیم خیلی مهمه که اینا دقیق نوشته بشه چون همه Object ها ازش تبعیت میکنن.

حالا که این هارو که فهمیدیم باید بریم سراغ پروژه امروز .

تا ابتدای day017 006

ادامه

-یه نکته ای که درمورد list وجود داره اینه که میشه بدون ساخت یک variable یک object رو بهش فرستاد، چراکه هر Index خودش یک variable عه . درصورتی که کاری که خودش کرد این بود که اول ریختش توی متغیر.

-todo های این پروژه بعد از اینکه data set رو درست کردیم:



نکته: چیزی که مهمه اینه که باید برای مرور یدور دیگه سعی کنیم این پروژه رو از صفر خودمون بزنیم. یا اصن یه پروژه دیگه مثل این رو.

ta Day017 009 03:35

ادامه

در ابتدا باید یه برداشت کلی ای از جلسه قبلی داشته باشیم، که فکر میکنم با run کردن بتونیم بدستش بیاریم.

-یه سوال دیگه ای هم که هست اینه که کدوم متغیر رو self در نظر بگیریم (یعنی بچسبونیمش به object و بتونیم بهش ارجاع بدیم و return بگیریم، و کدوم رو نکنیم، وقتی self میزاریم هم میشه ولی یسری جاها توی دوره self نزاشتیم، این رو میدونیم که قطعا باید توی \_\_init\_\_() از self استفاده کنیم. ولی بقیه جاها چی؟

همونطور که خودم فکر میکردم:



-در ادامه رسیدیم به اینکه گفت توی یه همچین بازی هایی یدونه data set منطقی نیست و بهتره یدونه دیگه داشته باشیم، یه سایت معرفی کرد opentbn فکر میکنم بود تو دوره لینکش هست از اونجا میشه سوال های verify شده دریافت کرد که البته json تحویل میده که همون dictionary در پایتونه و ممکنه key:value های متفاوتی داشته باشه که ما question و answer رو نیاز داریم. دو راه داریم یکی اینکه data set رو تغیر بدیم به چیزی که باید. یا کد برنامه رو تغیر بدیم به چیزی که باید .

-در IDE ها یه قابلیت وجود داره به نام reformat code که Indent هارو مرتب میکنه، کد رو خوانا تر میکنه و استفاده ما ازش در این قسمت این بود که data set ای که گرفتیم رو مرتب میکرد تا ازش سر دربیاریم که به عنوان یه dictionary داریمش یا list آیا یه List از dict هاست یا یه dict از List ها و کلی ترکیب دیگه. که در vscode باید یچیزی براش دانلود کنی. که با shift+alt+F کار میکنه، use that!

-نکته: اینجا دقیقا یه برنامه OOP داریم، که میتونیم به عنوان الگو بهش نگاه کنیم برای نوشتن برنامه های دیگه:

1. اینکه class ها و data set ها و برنامه اصلی بهتره جدا باشن، اینطوری واسه فهمیدن راحت ترن، واسه کامپیوتر فرقی نداره عملا.

2. در main.py بهتره طوری بنویسیم و در class ها طوری نام گذاری کنیم که در نگاه اول بدون اینکه یادمون باشه کدش چطوری نوشته شده بشه فهمید فرایند برنامه چطوری انجام میشه.

پایان day017