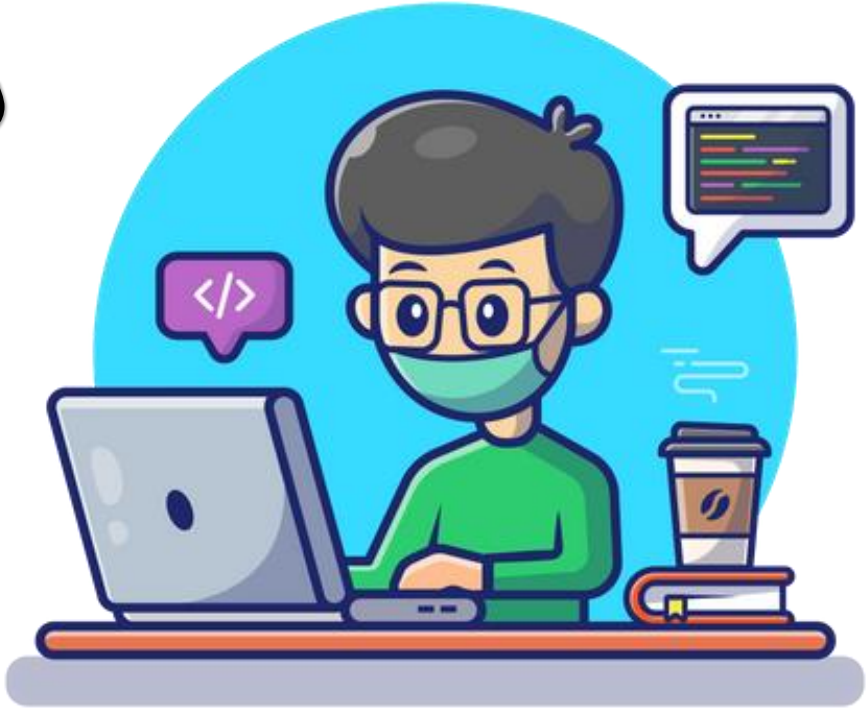
جزوه دستنویس پایتون

ریحانه محمدی

علی نظری زاده



فصل اول:

1 - تفکر برنامه نویسی و حل مسئله

2 - مسیر پادگیری پایتون و ابزارهای مورد نیاز

فصل دوم:

جلسه اول:8ص

جلسه دوم:3ص

جلسه سوم:2ص

جلسه پنجم:3ص

جلسه هفتم:5ص

جلسه نهم:2ص

جلسه دهم:4ص

جلسه یازدهم:8ص

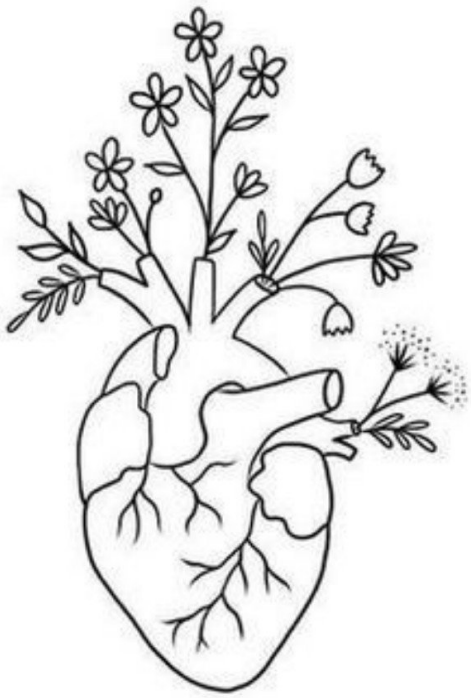
جلسه چهاردهم:5ص

جلسه پانزدهم:3ص

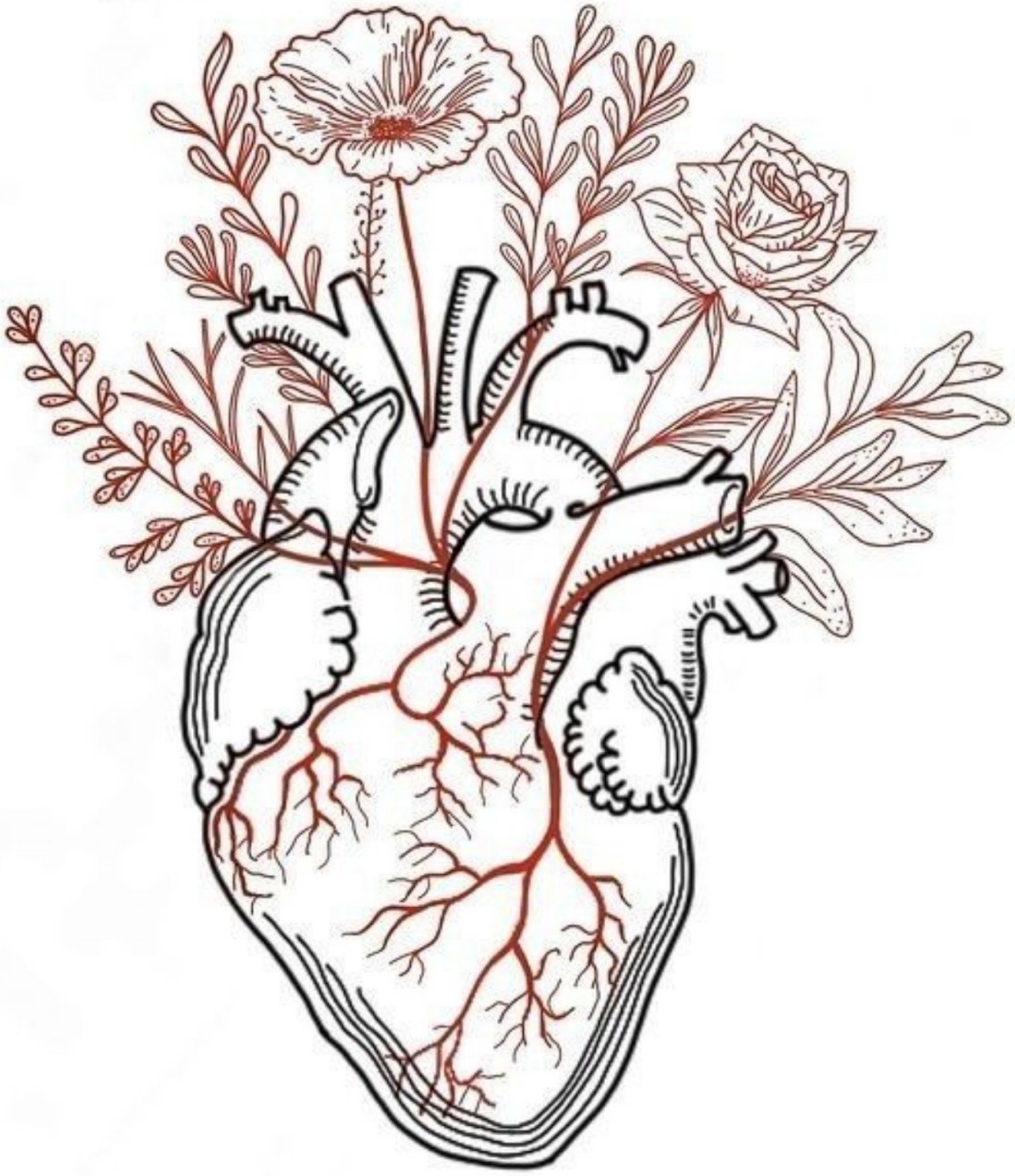
جلسه شانزدهم:6ص

جلسه بیستم:2ص

جلسه بیست و یکم:2ص



فصل اول



Subject ... Year 1400 _ Month 12 _ Date 4 _

فصل اول: آشنایی با پایتون و مفهوم برنامه نویسی

بخش ها:

1 تفکر برنامه نویسی و حل مسئله
2 مسیر یادگیری پایتون و ابزارهای مورد نیاز

برنامه نویسی و برنامه نویس

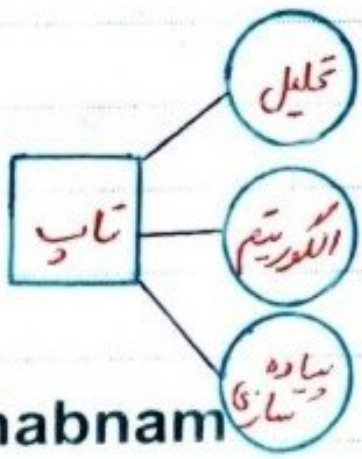
برنامه نویسی: حل مسائل دنیای واقعی با استفاده از ریاضیات و به کمک کامپیوتر.

برنامه نویس: فردی که با استفاده از زبان برنامه نویسی، قدرت حل مسئله، دانش ریاضیات، آشنایی به تکنولوژی های روز دنیا، قدرت ریسرچ می تواند نرم افزار طراحی کند.

الگوریتم

الگوریتم: ارائه راهکارهایی جهت حل مسئله.

سه گام اصلی در برنامه نویسی با قانون «تاپ»



تاپ فکر کن

ت: تحلیل

ا: الگوریتم

پ: پیاده سازی

Subject — Year 1400 — Month 12 — Date 4

مسئله: تشخیص زوج یا مزد بودن:

ت: دریافت عدد a جهت بررسی

ا: اگر با قیمانده a تقسیم بر ۲ برابر صفر شد: زوج و اگر با قیمانده a تقسیم بر ۲ مخالف صفر شد: مزد

ب: پیاده سازی

تاب د

```
If a %2 ==0:
    print("even")
Else:
    print("odd")
```

even : زوج

odd : مزد

بسترهای مختلف برنامه های کامپیوتر

1 اپلیکیشن های موبایل

2 نرم افزارهای تحت ویندوز

3 نرم افزارهای تحت وب

4 برنامه های روی سخت افزارهای متفاوت

5 برنامه های مخصوص اینترنت اشیا

و...

مدل های مختلف برنامه نویسی — نقش ریاضیات در برنامه نویسی

محاسباتی و ریاضی بسیار زیاد

پایگاه داده ای کم

طراحی رابط های کاربری اندک

بردهای مختلف سخت افزاری مانند... AVR, ARM, VHDL, Raspberry PI کم

تحلیل داده بسیار زیاد

هوش مصنوعی بسیار زیاد

و...

Shabnam

Subject Year 1400 – Month 12 – Date 4 –

زبان های برنامه نویسی (استفاده رایج)

python: web, mobile, AI, Hardwate, Game
C#: web, mobile, AI, Hardwate, Game
Java: web, mobile, AI, Hardwate, Game
C++: web, mobile, AI, Hardwate, Game
Matlab: web, mobile, AI, Hardwate, Game

از بیشتر به کمتر

python: very Easy
C#: Easy
Java: Medium
C++: Hard
matlab: Easy

تعریف فریمورک

یک محیط نرم افزاری که در آن با استفاده از زبان برنامه نویسی قادر به تولید نرم افزار هستیم.

Shabnam

Subject ... Year . 1400 ... Month . 12 ... Date . 4 ...

کتابخانه یا Library

مسائل سخت و الگوریتم های مختلف با استفاده از کتابخانه های مختلف پیاده سازی شده اند و جهت حل مسائل سخت، دیگر نیازی نیست که درگیر کدنویسی و طراحی الگوریتم شد.
تنها کافی است از کتابخانه های مختلف جهت استفاده از الگوریتم های پیچیده استفاده کرد.

برنامه نویس Back_End و برنامه نویس Front_End

برنامه نویس بک اند Back_End: کسی که بخش منطق، پایگاه داده و هسته اصلی یک نرم افزار را طراحی می کند.

برنامه نویس فرانت اند Front_End: کسی که بخش ظاهر سایت و هر آن چیزی که کاربر با آن تعامل دارد را طراحی می کند.

چرخه تولید نرم افزار

رفع مشکل و توسعه نسخه → بازخورد گرفتن از کاربران → تولید نسخه اولیه → تست → تاب دهید

Subject — Year 1400 — Month 12 — Date 5 —

دلایل استفاده از پایتون

سادگی
منابع آموزش زیاد
جامعه آماری بالا
کتابخانه های بی شمار
محبوبیت بالا
رشد زیاد طی سال های گذشته

تخصص ها

متخصص بک اند	متخصص هوش مصنوعی
〃 فرانت اند	〃 دیتا ساینس
〃 بازی سازی	〃 امنیت و شبکه
〃 Android	〃 سخت افزار
〃 iOS	〃 مهندس داده
〃 رایانش ابری	〃 تست نرم افزار

نقشه راه متخصص شدن

انتخاب یک تخصص ← آشنایی با ابزارها، کتابخانه ها، تکنولوژی ها ← شناسایی منابع وب سایت های مرجع ← کوچک جمع بزرگان باش نه بزرگ جمع کوچکان ← آموزش به دیگران

آموزش به دیگران ← تقویت ریاضیات و خلاقیات حل مسئله ← تقویت زبان انگلیسی ← یادگیری زبان برنامه نویسی ← آشنایی با فریم ورک ها و ابزارهای موجود ← فعالیت در گروه های برنامه نویسی و شبکه اجتماعی تیک تین

Shabnam

Subject — Year — Month — Date —

داده ⟸ ورودی

ذخیره سازی

پردازش

خروجی

عملگرها :!

+	جمع	==	مساوی
-	تفریق	>=	بزرگتر مساوی
*	ضرب	<=	کوچکتر مساوی
/	تقسیم	!=	
//	تقسیم و گرد کردن	a += 5	a = a + 5
%	تقسیم کن و باقی مانده رو بده	b -= 2	b = b - 2
		c *= 4	c = c * 4
**	توان	d /= 3	d = d / 3

print → جواب رو نشون میده یا چاپ میکنه

" " هر چیزی بینش بنویسیم عین همون رو چاپ میکنه

اولویت ها

**

* / % //

+ -

♡

Shabnam

and → همه مقایسه گرها باید True باشد
تا True نشه حتی اگر بین 1000 تا یکی false باشه
جواب false هست.

or → اگر بین 100 تا حتی یکی True باشه
و همه False جواب True میشه.

not → برعکس می کند
False ⇐ True
True ⇐ False

& → and

| → or

^ → باید فرد تا ۲ داشته باشد → تابع فرد است

int → اعداد رو از اعشار درمیاره — عدد اعشاری ⇐ عدد صحیح

float → اعداد رو اعشاری میکنه — عدد صحیح ⇐ عدد اعشاری

type → نوع عدد رو میگه

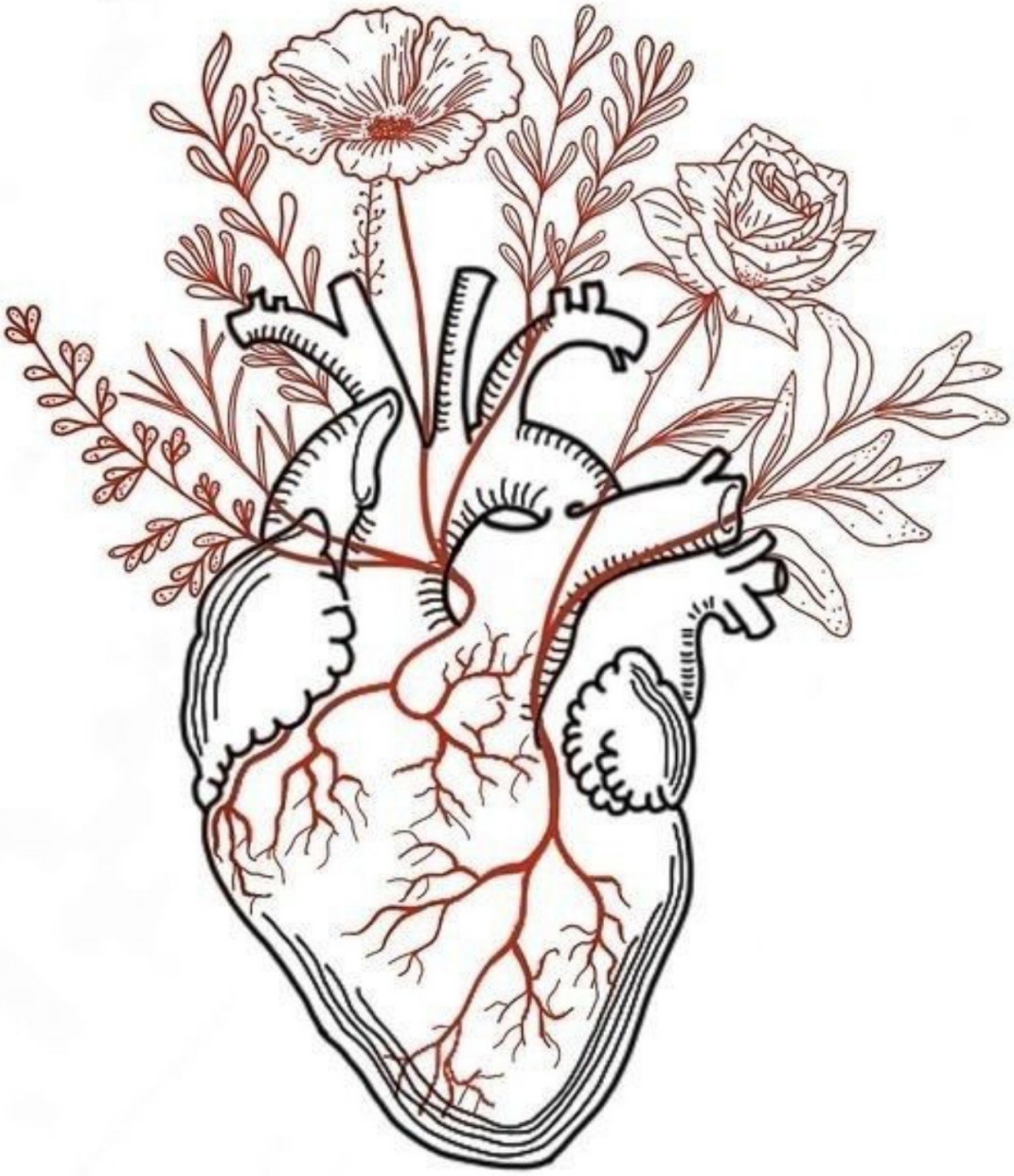
bool → یعنی صفر و یک — یا صفر می گیرند یا یک

Shabnam str → string

فصل اول

جلسه اول:

عملگرها، رشته ها و ریاضیات



بسمه تعالی

Subject جلسه اول Year. 17 – Month. 4 – Date. 1401 –

ژوپیتر نوتبک : یک محیط برنامه نویسی است. (IDL)
سلول به سلول است. ← خوبی که داره می‌تونیم جداگانه کدها را اجرا کنیم و اگر اشکالی داشت بفهمیم.

زوم کردن صفحه ژوپیتر: ctrl + +

اجرا کردن کدها در ژوپیتر: (Run) (ctrl + enter) (shift + enter)
فقط اجرا — اجرا + میره سلول بعد

* سمت چپ یکسری اعداد ظاهر می‌شه ← نشان دهنده ترتیب اجرا کدها.

code : وقتی می‌خوایم یک کد اجرا بشه باید روی code تنظیم بشه.

Markdown : وقتی اجرا بشه اون رو به صورت کامنت (متن) چاپ میکنه. نشانه گذاری ← برای یکسری توضیحات درباره کدها.

Raw NBconvert : مثل Markdown عمل میکنه ولی قالب سلول ازین نمیره.

Heading : مانند یک تیتر

Shabnam

بسمه تعالی

Subject جلسه اول Year 17 Month 4 Date 1401

↑ ↓ : انتقال سلول ها

علامت قیچی : برای حذف سلول Edit ← Delete cells Undo :
برای برگردوندن

Restart & clear output ← Kernel : برای پاک کردن تمام خروجی ها

Notebook ← Download as ← File : برای ذخیره کردن
Toggle Header ← View : مخفی یا پیدا بودن منوی بالای ژوپیتر

* چهار مورد مهم در برنامه نویسی :
داده ـ ذخیره سازی ـ پردازش ـ خروجی

داده : میتونه عدد ، رشته ، متن باشه .

داده ساختار : مجموعه‌ای از اعداد ← داخل متغیر ها ذخیره می کنیم .

متغیر داده

$a = 2$
$b = 3.2$
$a, b = 2, 3.2$

$\Longrightarrow a + b \Longrightarrow x1 = a + b$

5.2

Shabnam

بسمه تعالی

Subject جلسه اول Year 1401 – Month 4 – Date 27 –

عملگرها:

+ جمع ، - تفریق ، * ضرب ، / تقسیم ، // گرد کردن

** توان ، % باقی‌مانده

Print → خروجی رو چاپ میکنه → Print ("....")

* عینا چاپ میکنه: هیچ پردازشی روش انجام نمیشه *

Print ("x1:", x1)

x1 مقدار ، خوده: x1: چاپ میشه

تقدم عملگرها:

() ، * ، / ، % ، // ، + ، -

→ کامنت → چیزی چاپ نمیکنه

a = a+5 → a += 5

b -= 6

c *= 2

d /= 3

Shabnam

بسمه تعالی

Subject: جلسه اول Year: 1401 – Month: 4 – Date: 27 –

عملگرهای مقایسه ای:

== → برابر != → مخالف

x = 0
y = 1

x == y → False: غلط یا 0
x != y → True: درست یا 1

> → بزرگتر ، >= → بزرگتر مساوی ، < → کوچکتر ، <= → کوچکتر مساوی

& : and

همه شرایط باید 1 یا True باشه، تا من 1 یا True برگردونم.
اگه حتی یکی از شرایط 0 یا False باشه، من 0 یا False چاپ می کنم.

| : or

برعکس and عمل می کند.
or میگه حتی اگه یکی از شرایط 1 یا True بود، من True می شوم.

: not

شرایط رو برعکس میکنه.
اگه 1 یا True باشه، میشه 0 یا False و برعکس.

False
(not (2 == 2))
True

ما می تونیم عملگرهای منطقی رو داخل متغیرها ذخیره کنیم
a = True b = False

Shabnam

بسمه تعالی

Subject جلسه اول Year 1401 – Month 4 – Date 27 –

int: اعداد رو از اعشار در میاره

float: اعداد صحیح رو به اعشاری تبدیل میکنه

---

type: نوع class رو میگه
x: 7.0
type (x) <class 'float'>

متن ها از نوع ← str هستند.

رشته ها: می تونیم اون ها رو هم داخل متغیرهایی ذخیره کنیم.

```
str1: "Hi friends,"
str2: "my name is"
str3: "Reyhane Mohammadi"
str4= str1 + str2 + str3
```

می تونیم روشون پردازش هایی هم انجام بدیم.

```
Print (2 * str4)
```

↓
بهاین تعداد تکرار میکنه

Shabnam

بسمه تعالی

Subject: جلسه اول Year: 1401 – Month: 4 – Date: 27 –

input: یک ورودی از کاربر می خواد

ما می تونیم رشته ها رو تکه تکه کنیم.

[:] ⇐ کلش [23:] ⇐ از 23 به بعد

[23:26] ⇐ از کاراکتر 23 تا یکی کمتر از 26

[-1] ⇐ از آخر یک کاراکتر میده [:-5] ⇐ 5تای آخر رو حذف میکنه

len(str6):

len تعداد یا طول کاراکترهای رشته رو حساب میکنه.

بسمه تعالی

Subject جلسه اول Year 1401 – Month 4 – Date 27 –

Print:

Print(f"...:{...}")

Print("...:", ...)

str دارای توابع است:

Find: وقتی که لازمه یک کاراکتر رو در رشته پیدا میکنه.
اگر پیدا کرد اندیسش رو برمی‌گردونه.
اگه نباشه ← 1-: نشون دهنده اینه که وجود نداره.

lower: اسمال رو به کپتال تبدیل میکنه.

islower: آیا این حرف کوچک؟ → True یا False

upper: تبدیل به حرف بزرگ

isupper: آیا این حرف بزرگ؟ → True یا False

isspace: آیا فاصله وجود داره؟ → True یا False

Shabnam

بسمه تعالی

Subject: جلسه اول Year: 1401 – Month: 4 – Date: 29

startswith: آیا با این حروف شروع شده؟

endswith: آیا با این مقدار تموم شده؟

replace: یک مقدار موجود در متن رو با یک مقدار جدید جایگزین میکنه.

isdigit: آیا عددی است؟

import math :

math + . + Tab → دسترسی به توابع

pow: توان (2,3) 2^3

sqrt: رادیکال (25)

sin: سینوس (30)

cos: کسینوس (10)

factorial: فاکتوریل (4)

pi: عدد پی

log10: لگاریتم

log2: لد

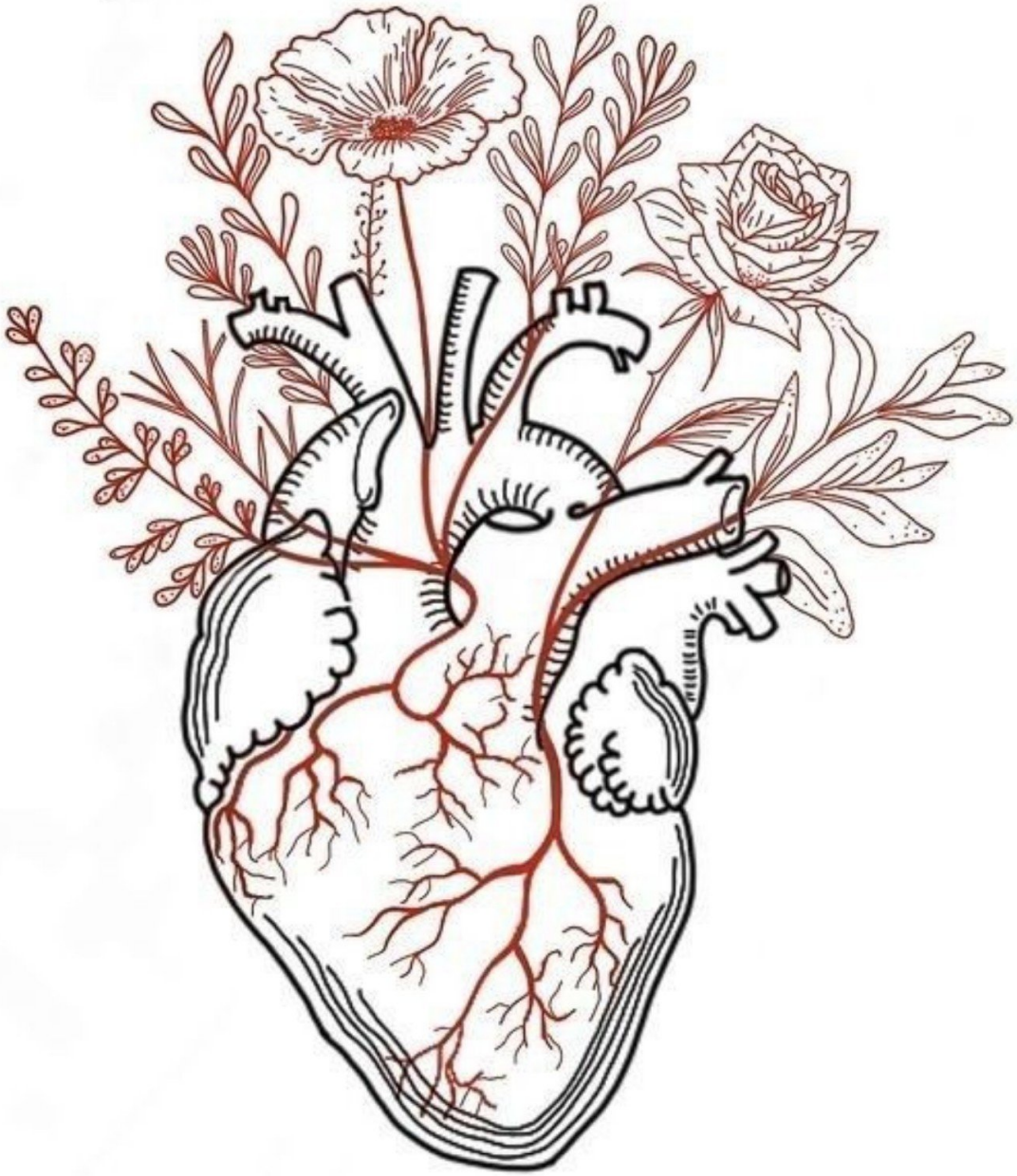
fabs: قدر مطلق (-3)

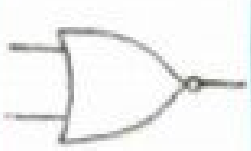
floor: رند کردن به سمت عدد پایین تر (3.8)

ceil: رند کردن به سمت عدد بالاتر (3.3)

Shabnam

جلسه دوم:
حل مسائل عملگردها، رشته ها و پاضبات



نام گیت	NOT	AND	NAND	OR	NOR	XOR	XNOR
تابع جبری	$\overline{A}$	AB	$\overline{AB}$	$A+B$	$\overline{A+B}$	$A \oplus B$	$\overline{A \oplus B}$
نماد							
جدول درستی	A | x 0 | 1 1 | 0	B A | x 0 0 | 0 0 1 | 0 1 0 | 0 1 1 | 1	B A | x 0 0 | 1 0 1 | 1 1 0 | 1 1 1 | 0	B A | x 0 0 | 0 0 1 | 1 1 0 | 1 1 1 | 1	B A | x 0 0 | 1 0 1 | 0 1 0 | 0 1 1 | 0	B A | x 0 0 | 0 0 1 | 1 1 0 | 1 1 1 | 0	B A | x 0 0 | 1 0 1 | 0 1 0 | 0 1 1 | 1

بسمه تعالی

Subject: جلسه دوم Year: 1401 – Month: 4 – Date: 29

NOT: کافیه یک not بیشتر اون بیت بیاریم.

AND: به صورت کلی میاد عملیات minimom رو انجام میده.

NAND: برعکس AND ← AND رو not میکنه.

OR: به صورت کلی میاد عملیات moximam رو انجام میده.

NOR: برعکس OR ← OR رو not میکنه.

XOR: اگر فرد تا از ورودی ها ① باشه خروجی ② میشه.

XNOR: برعکس XOR ← XOR رو not میکنه.

می خوام یک عددی رو نگاشت بدیم به یک بازه دیگر:

$x = \Lambda$

$a, b = 0, 20$

$c, d = 0, 1$

Shabnam $((x\_a) / (b\_a)) * ((d\_c) + c)$

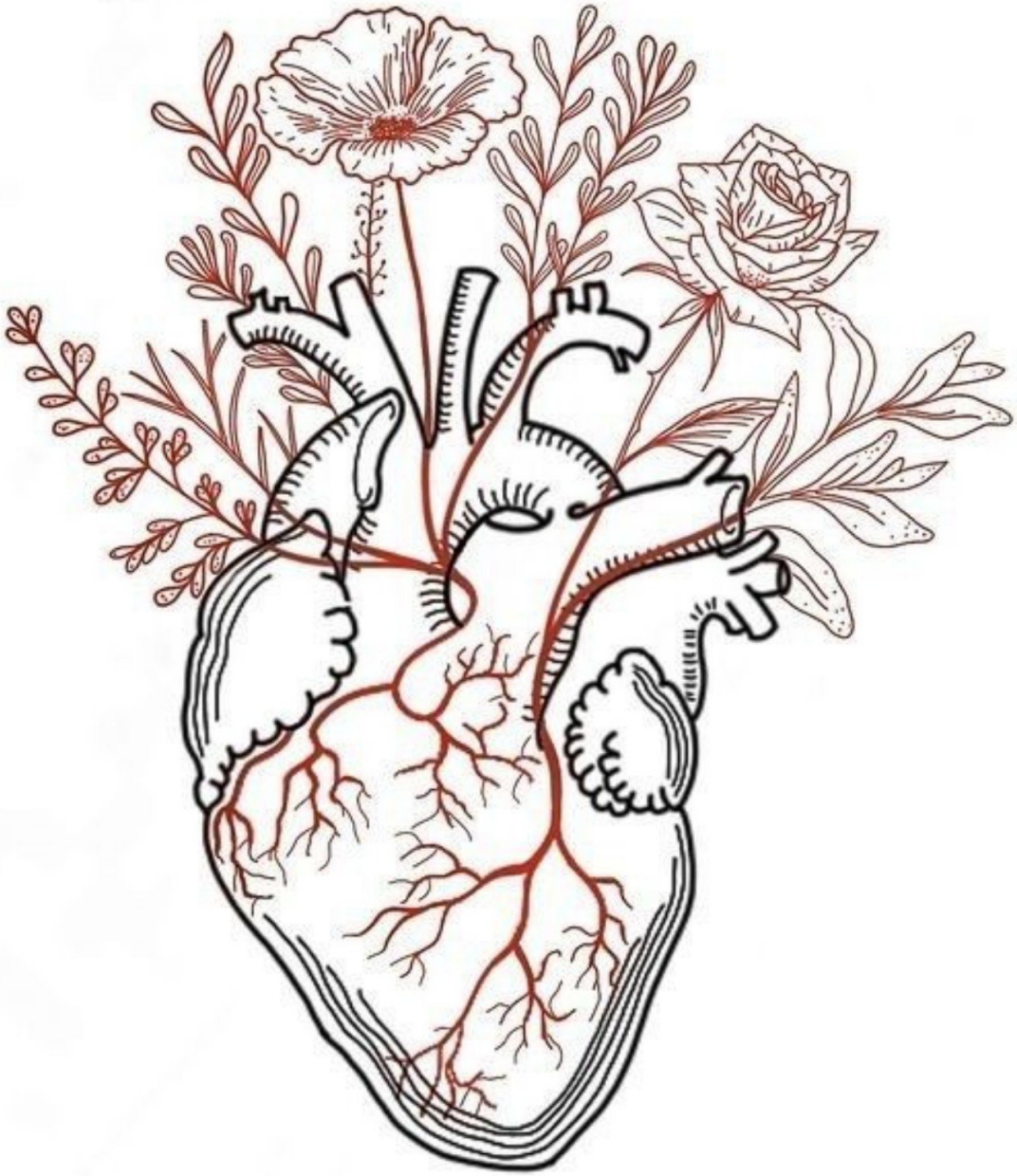
بسمه تعالی

Subject جلسه دوم Year 1401 – Month 4 – Date 29 –

عملگر in : برای مثال در یک متن دنبال شماره تلفن است.
به صورت خوشمند پیدا میکنه.
اگر بود True اگر نبود False

تابع split : میاد رشته رو میشکنه. هر وقت به . رسیدی
text.split (".")

جلسه سوم:
دستورات شرطی، تصمیم گیری ها و حلقه ها



بسمه تعالی

Subject: جلسه سوم Year: 1401 — Month: 4 — Date: 29 —

if , else : بر اساس اینا یکسری تصمیم گیری ها میکنه

```
number1, number2 = 4, 6

if number1 > number2 :
    print("number1")
else:
    print("number2")
```

اگر این شرایط برقرار بود
برو داخل این بخش کد و انجام بده
اگه نه، برو سراغ else
و این بخش کد و انجام بده

اگر شرایط به وجود اومده بیشتر از دو تا بود
از elif استفاده می کنیم.
else رو زمانی می زاریم که شرایطی باقی نمونده باشه.

این بخش تو رفتگی ها مهمه
مخصوصا در if و else ها
بنابراین if و print هم تراز باشند
استاندارد شون اینه که یه Tab بزنن
if و else باید هم تراز باشند

با استفاده از if و else ما می تونیم کدهامون رو توسعه بدیم.

با if و else ها شرط ها رو برقرار می کنیم.

کدهارو کنترل می کنیم.

Shabnam

بسمه تعالی

Subject: جلسه سوم Year: 1401 — Month: 4 — Date: 29 —

* بعضی وقت ها باید همه شرایط لحاظ بشه؛ پس همه شرایط اتفاق افتاده

* ما از if استفاده می‌کنیم. نه از elif و else.

while: تا زمانی که شرایط برقرار بود فلان کار رو انجام بده

while (True): حلقه بی نهایت

* ما می‌تونیم while و if رو ادغام کنیم *

import random as rand

کتابخانه جدید — تغییر اسم

break → خارج شو

round → تا یک رقم گرد میکنه

for → ترتیب رو توی برنامه اجرا میکنه

range → از ۰ تا یکی کمتر عدد تولید میکنه

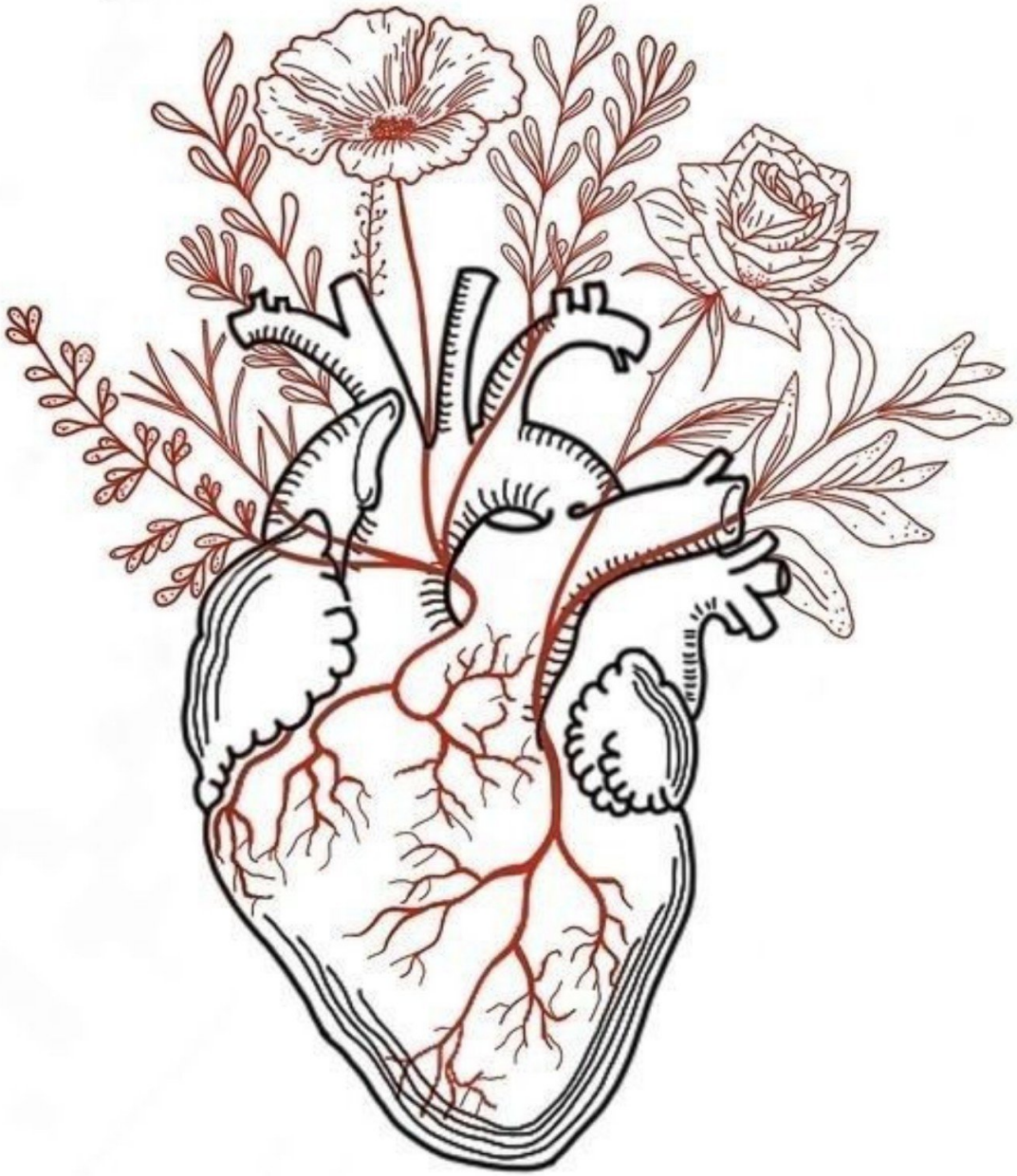
```
for i in range (10):
    print (i)
```

می‌تونیم ۳ تا عدد قرار بدیم

(0, 10, 2) ← دو تا دو تا برو جلو تا برسی

Shabnam

جلسه پنجم:
مجموعه ها (Set)



بسمه تعالی

Subject جلسه پنجم Year 1401 — Month 4 — Date 29 —

set: مجموعه → اعداد و کاراکتر تکراری ندارد.

می‌توانیم داخل set باهم دیگه هم عدد هم کاراکتر و هم رشته داشته باشیم.

به این شکل ← {...، ...، ...}

می‌توانیم len رو بگیریم. type رو هم می‌توانیم مشخص کنیم.

S5.add(10) → به ورودی بدیم که اضافه کنه به مجموعه.

S5.clear() → فقط عضوهای داخل set رو پاک میکنه.

S6=S5.copy() → کپی میکنه.

S7=set() → S7 از نوع set ولی داده‌ای وارد نکردیم.

```
type(S7)
set
```

S8.union(S9) → اجتماع میگیره.

S8.intersection(S9) → اشتراک میگیره.

Shabnam

بسمه تعالی

Subject جلسه پنجم Year 1401 — Month 4 — Date 29 —

S8.difference(S9) → S8_S9 ← روجداکن ← S9 ← تفاضل

S9.difference(S8) → S9_S8 ← روجداکن ← S8 ← تفاضل

S8.pop(2) → ورودی که بهش میدیم از مجموعه برمیداره.

S8.remove → پاک میکنه.

sub={۲،۴}
sub.issubset(S8) → زیرمجموعه.

S10={1,2,3,5}
S10.update{4,6,7} → با مقدار جدید آپدیت میشه.

S10.diseard(71) → میاد تو مجموعه ببینه اون عدد هست یا نه
اگه نبود که هیچ، اگه بود اون رو حذف میکنه.

discard
↓
اگه نباشه خطا نمیگیره

remove
↓
اگه نباشه خطا میگیره

Shabnam

بسمه تعالی

Subject جلسه پنجم Year 1401 Month 4 Date 29

help(s10.discard) ↓ میاد یه توضیحی درباره توابع یا کتابخونه‌ای که براش فرستادی، میده.

?s10.discard? ↓ یا این ولی چاپ نمیکنه

?s10.discard ↓ یا این

split تابع → به کلمه‌ها اشاره میکنه

replace → جایگزین کردن

برای بدست آوردن درصد شباهت دوستان از این فرمول استفاده میشه:

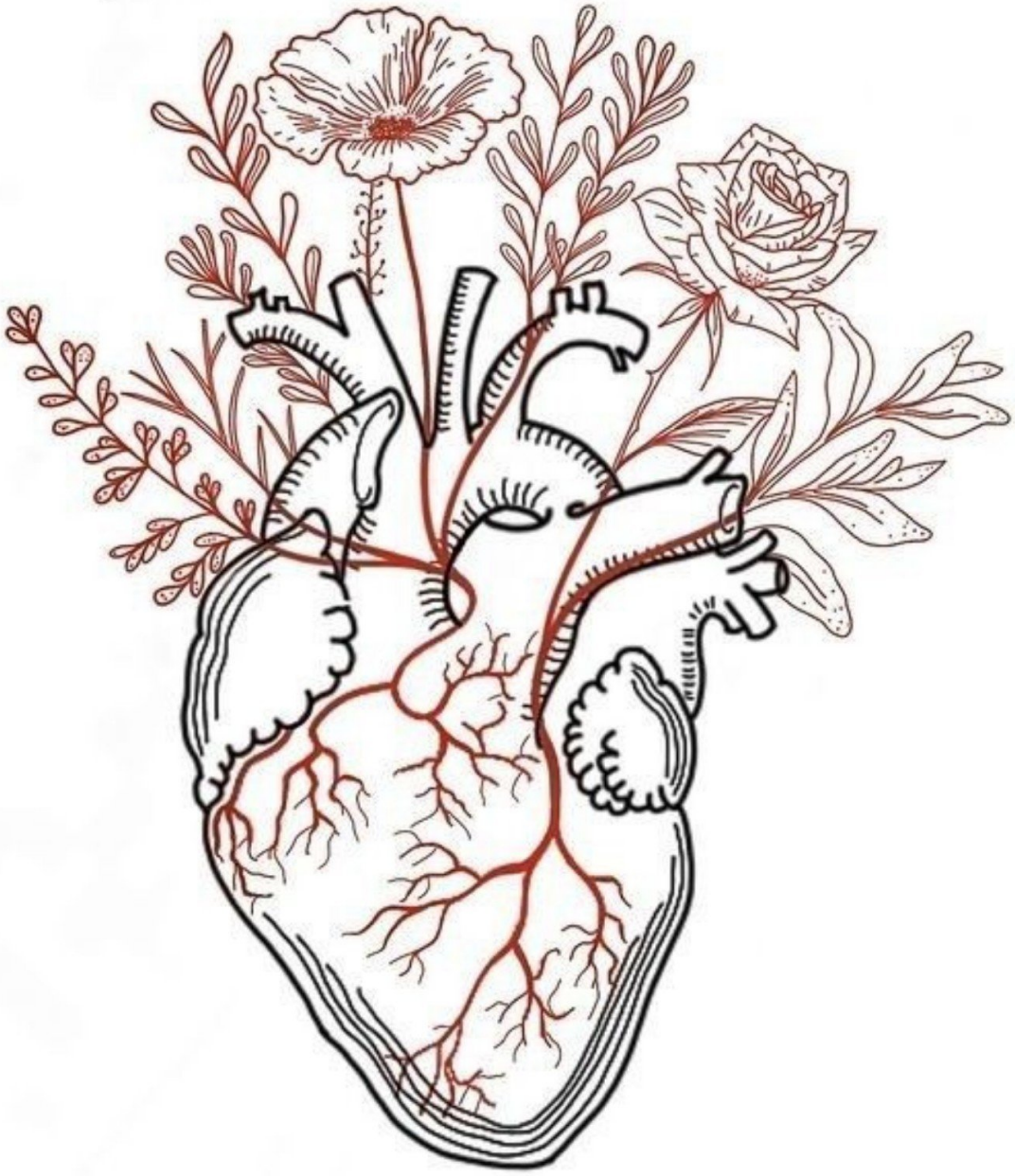
(len(set3) / len(set2) x 100

set3 ↓ اشتراک set2, set1

round(x, 2) → تا دو رقم رند میکنه

Shabnam

جلسه هفتم:
لیست ها (list)



بسمه تعالی

Subject جلسه هفتم Year. 1401 – Month. 4 – Date. 29 –

list: یک لیستی از داده‌هاست. پس میتونه چیزای تکراری هم داشته باشه.
کارکتر + عدد + رشته

```
L1 = [1,2,3,4,4]
L1.append(5)
L1
1,2,3,4,4,5
```

یک داده رو وارد لیست میکنه
ولی فقط یک مقدار روی تو نیم به ورودی بدیم.

```
L1.claer() →
```
عضوهاشو پاک میکنه.

```
L1.append([2]) →
```
میتونه لیست قرار بگیره.

```
L1.extend([5,3,6,1]) →
```
اگر بخوایم تعدادی عدد وارد کنیم.

```
a = [2,4,6]
L1.extend(a) → ([a]) →
```
به صورت یک لیست قرار میگیره.

```
L2 = list([8,9])

L3 = L1 + L2
```

Shabnam

بسمه تعالی

Subject جلسه هفتم Year 1401 – Month 4 – Date 29 –

L4 = ["_____"]

L5 = L4 → مقدار L4 کپی میشه در L5.

L6 = L4.copy() → کپی میکنه.

L4.count("___") → می خوام ببینم یک مقدار در لیست چند بار تکرار شده.

L4.index("___") → می خواد ببینه اون مقدار اندیس چندمه آیه اون ورودی چند بار تکرار شده باشه به اولین که میرسه همونو نشون میده.

L4.insert(1,"___") → کجا / چه چیزی

L4.pop() → آخرین مقدار رو بیرون میکشه. یا یه مقدار براش بفرستیم.

L4.remove("___") → اون مقدار رو حذف کن.

L4.reverse() → برعکس میکنه.

L4.sort() → مرتب میکنه.

Shabnam

بسمه تعالی

Subject جلسه هفتم Year 1401 – Month 4 – Date 29 –

* من می تونم مقدارها رو در یک متغیر قرار بدم.
ولیست جدید رو بسازم و بگم هر متغیر در کدام اندیس قرار بگیره. *

L6[1] → اندیس یک رو نشون میده

L6[1] = "___" → به جای مقدار اندیس 1، مقدار جدید بزار

L6[2:] → می خوام بگم از اندیس فلان تا آخر چاپ کن.

L6[1:4] → از 1 تا یکی کمتر از 4.

L6[-1] → آخرین مقدار

L6[:-1] → از اول تا یکی مونده به آخر

اگر بخوام بفهمم یه مقداری وجود داره یا نه:

"___" in L6

یا

x = "___"
x in L6

Shabnam

بسمه تعالی

Subject جلسه هفتم Year. 1401 – Month. 4 – Date. 29 –

* ما در لیست ها مقدارهای تو در تو داریم.
(که آنها برای مثال بخواهیم به اندیس اول (که خودش شامل مقداری است) دسترسی داشته باشیم و داخل اون به اندیس دوم دسترسی داشته باشیم :) *

L7[7][2]

می خوام که یک لیست رو به str تبدیل کنم از متد join استفاده می کنم باید به join دو تا مقدار بدیم:

```
text = ";" join (my_favorites)
```

برو از داده ساختار my_favorites با استفاده از ; اونارو جدا کن و در text بریز.

برعکس این هم امکان دارد:

```
TEXT = []
for i in text.split(','):
    TEXT.append(i)
```

ما با استفاده از حلقه for هم می تونیم لیست رو خروجی بگیریم.

ذخیره سازی در list:

```
numbers =[]
for i in range(10):
    numbers.append(i)
```

```
numbers=[i for i in range(10)]
numbers
```

Shabnam

بسمه تعالی

Subject جلسه هفتم Year 1401 — Month 4 — Date 29 —

برای زمانی که ما یک شرط (if) هم داشتیم اون رو جلوی for قرار میدیم ← از بیرون به داخل.

اگه کار پیچیده‌تری داشتیم با (:) انجام میدیم.

```
math.factorial(i)
```

حالا اگه ما if و else ← ۳تا موجودیت داشتیم به این صورت نوشته میشه:

```
numbers=[i if i%2 ==0 else i*10 for i in range(10)]
numbers
```

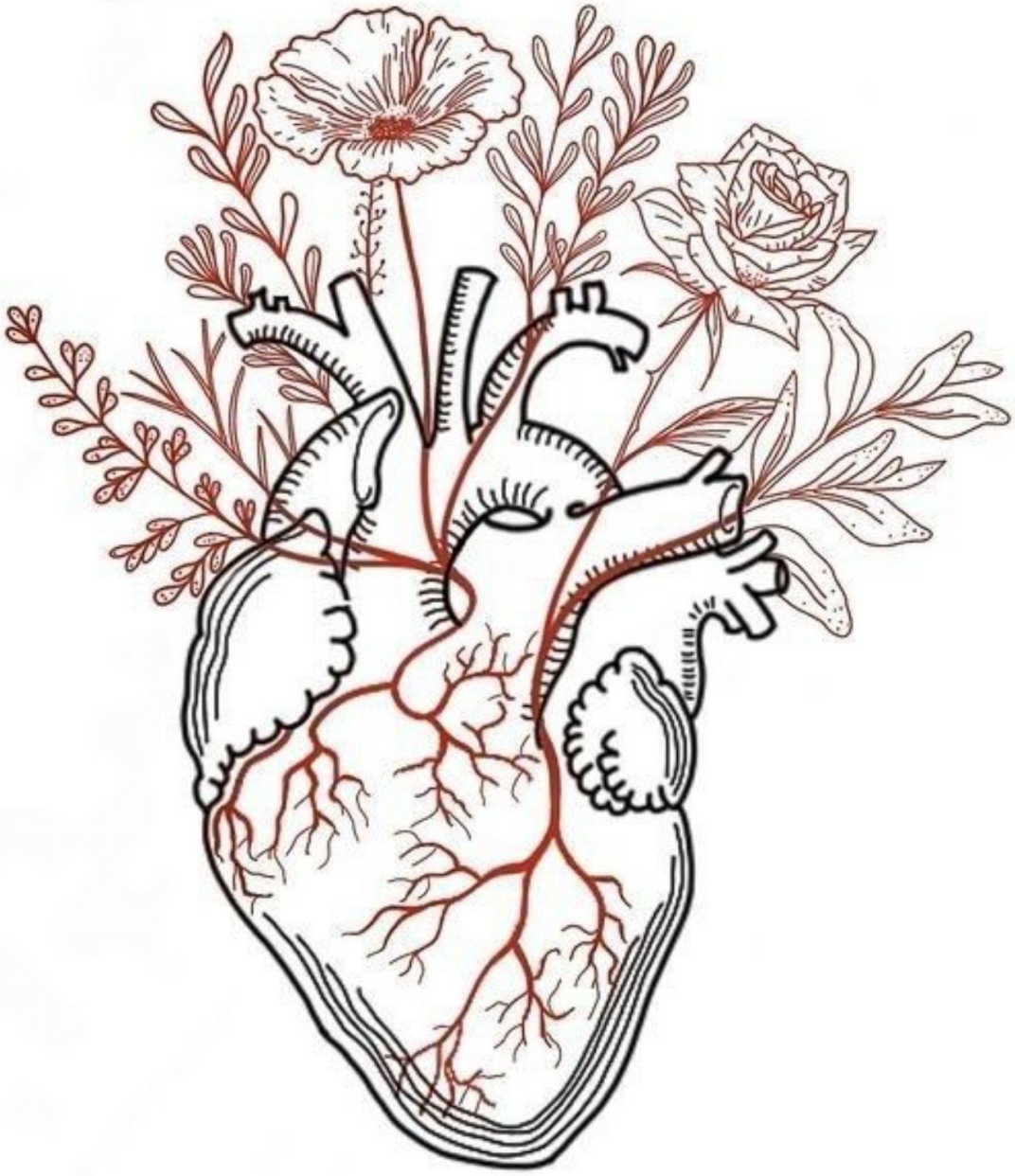
```
numbers = []
for i in range(10):
    if i%2 ==0:
        number.append(i)
    else:
        number.append (i*10)
```

نوشتن کد list تودرتو:

```
[[i*j for j in range(10)] for i in range(5)]
```

Shabnam

جلسه نهم:
تاپل ها (tuple)



بسمه تعالی

Subject جلسه نهم Year 1401 — Month 4 — Date 29 —

۳ نکته مهم درباره tuple :

۱- از پویایی و انعطاف پذیری خیلی پایین تری برخوردارند.
(یعنی هر مسئله ای رو نمی تونیم با این داده ساختار حل کنیم.)

۲- توابع بسیار کمی برای تایپ ها در نظر گرفته شده.
(یعنی فقط ۲ تا تابع دارند)

۳- زمانی که یک داده ساختار از نوع تایپ تعریف کردید و یکبری داده براش فرستادید، دیگه اون داده رو نمی تونی تغییر بدی.
(قابل تغییر نیستند.)

```
T = (2)
type (T)
int  → مثله از نوع int است

T = (2,)  ← کاما
type (T)
tuple
```

اگر و نباشه tuple نیست.

تابع count : چندبار تکرار شده.

تابع index : اندیس چندمه.

Shabnam

بسمه تعالی

Subject جلسه نهم Year 1401 — Month 4 — Date 30 —

* با استفاده از براکت می توانیم به داده ها دسترسی داشته باشیم.

T1[0]

برای ذخیره سازی tuple :

T2 = tuple([1,2])

برای اینکه ببینیم داده ای وجود دارد یا نه از in استفاده می کنیم:

2 in T2

del → این جوری یک تاپل رو حذف می کنیم.

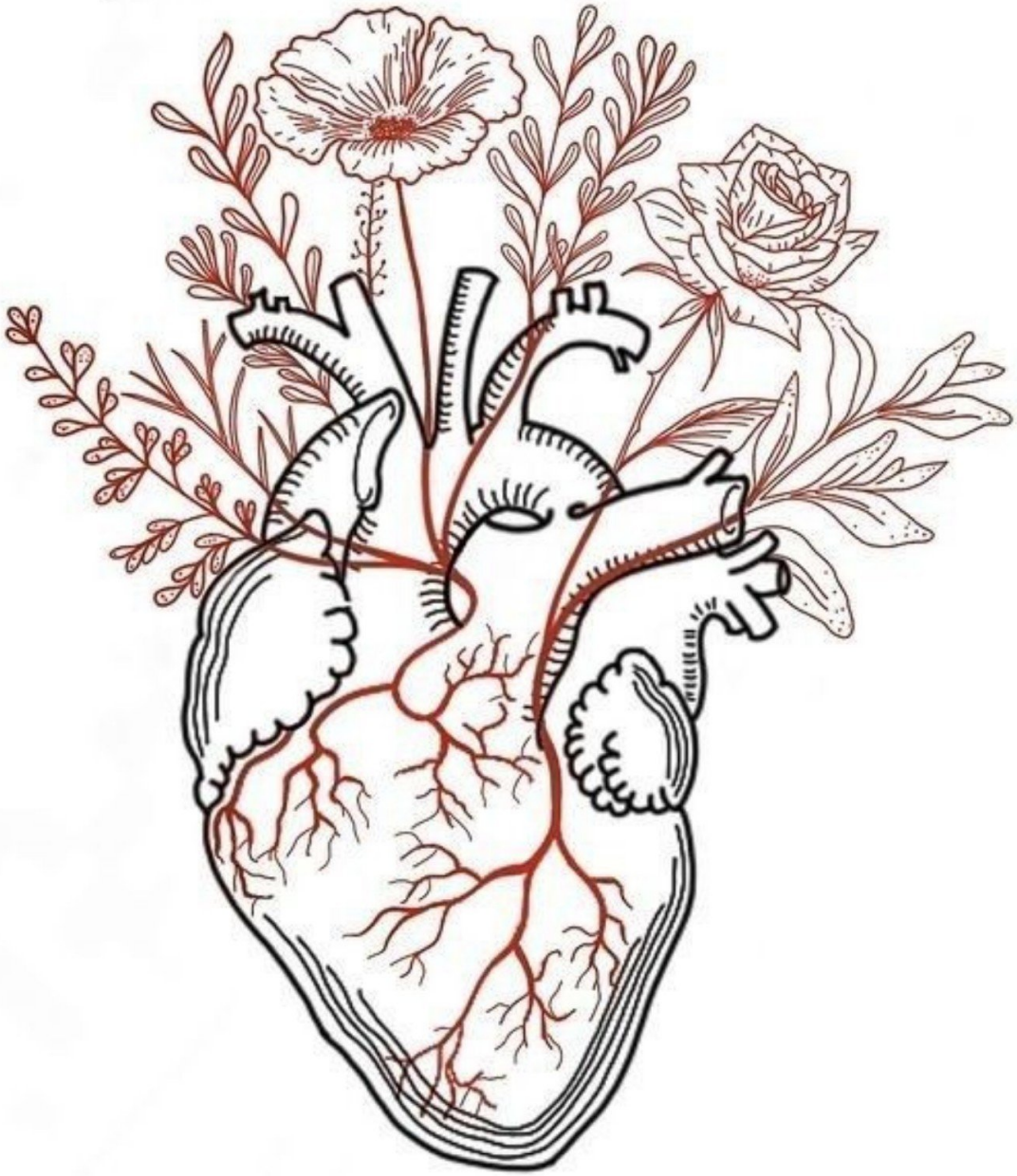
T4 += T5 → اضافه کردن داده، ولی نمی تونیم تغییر بدیم.

* روی تاپل ها همیشه پردازش انجام داد. *

* وقتی ما یه داده هایی رو نمی خوایم تغییر بدیم، باید توی داده ساختار تاپل ذخیره کنیم. *

Shabnam

جلسه دهم:
دیکشنری ها (dictionary)



بسمه تعالی

Subject: جلسه دهم Year: 1401 Month: 4 Date: 30

داده ساختار دیکشنری :

داخل یک پرانتز {} و با (,) کاما از هم جدا میشن.

داده ها را در این قالب ذخیره می کنیم: "key" : value ,

* داخل داده دیکشنری هم میتونه یه دیکشنری دیگه وجود داشته باشه. *

یک راه دیگه برای ذخیره سازی وجود داره:

D3 = dict (course = "....")

هرچیزه میتونه باشه

اگه یک متغییر تعریف کنیم برای مثال:

person1 = { } → خالی باشه، میشه dict

person2 = {1} → اگه یه داده بدیم بهش، میشه set

Shabnam

بسمه تعالی

Subject: جلسه دهم Year: 1401 — Month: 4 — Date: 30

توابع دیکشنری :

ما می توینم به Key ها دسترسی داشته باشیم با تابع keys .
برای مثال با استفاده از حلقه for :

```
for Key in D3.Keys():
    print(Key)
```

مدل اول ←

```
D3.Keys()
```

مدل دوم ←

ما به همون صورت می توینم به value ها هم دسترسی داشته باشیم:

```
for value in D3.values():
    print(value)
```

مدل اول ←

```
D3.value
```

مدل دوم ←

ما اگر بخوایم هردو رو استخراج کنیم باید از تابع items کمک بگیریم.

```
for key, value in D3.items():
    print(key, ":", value)
```

Shabnam

بسمه تعالی

Subject جلسه دهم Year 1401 – Month 4 – Date 30 –

```
x=[]
y=[]
for key,value in D3.items():
    x.append(key)
    y.append(value)

print(x)
print(y)
```

D4=D3

D5=D3.copy() → کپی میکنه.

D3.clear → پاک میکنه.

del D3 → کلا پاک میکنه.

D5.pop('...') → یه مقداری رو پاک میکنه.

D5.popitem() → از آخر یکی پاک میکنه.

با استفاده از in میایم ببینیم توی داده هست یا نه:

'...' in D5 → D5.values()

Shabnam

بسمه تعالی

Subject جلسه دهم Year. 1401 — Month. 4 — Date. 30 —

update : مثلا مقدار D6 با مقدار D1 آپدیت میشه :

```
D6 = {}
D6.update(D1)
D6
```

ممکنه خوده D6 یه داده داشته باشه و با داده های D1 آپدیت بشه.
(جمع بشه)

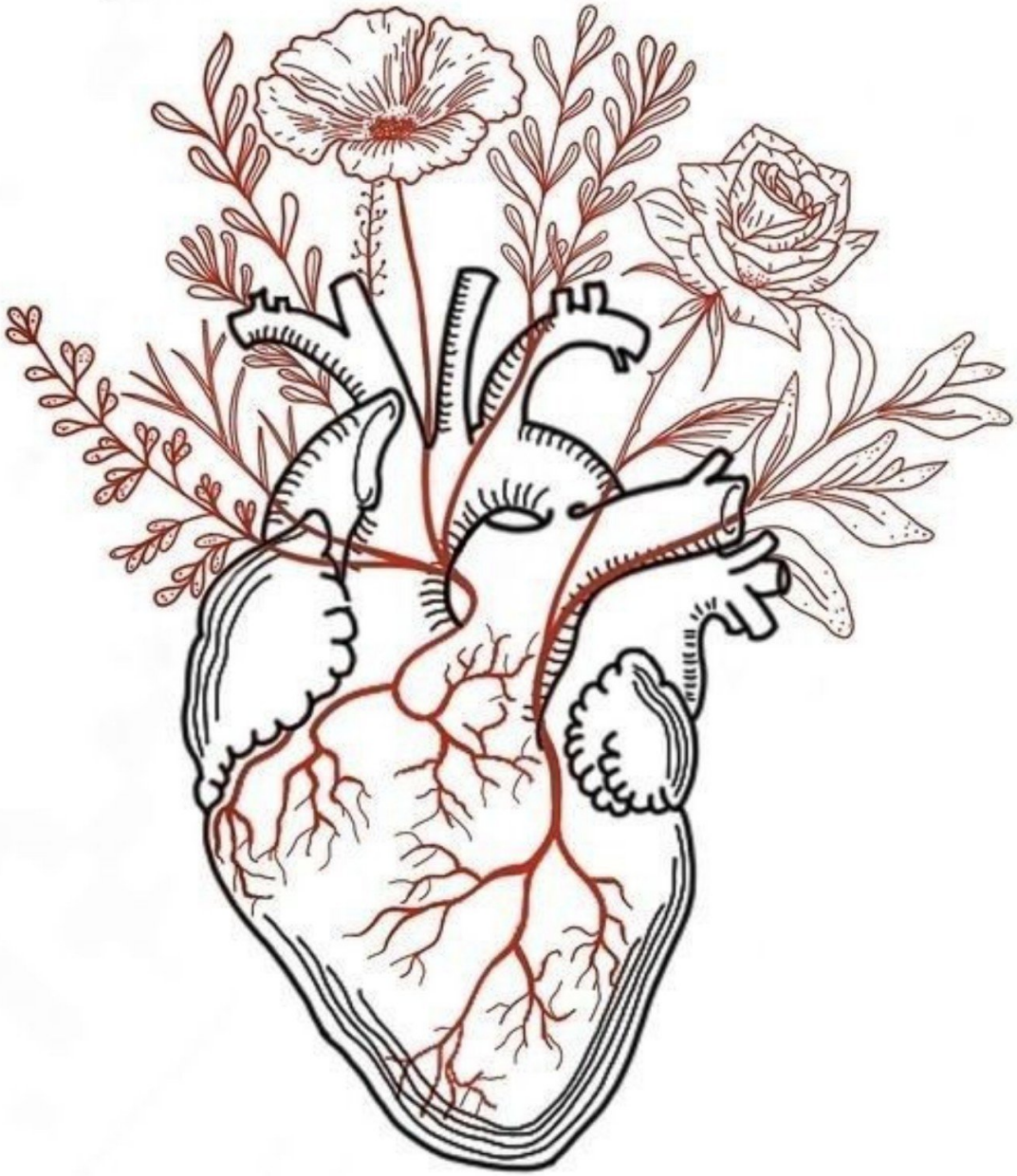
* اگر وقتی D6 دارای key باشد که با D1 برابر است.
اون مقدارها با مقدار جدید آپدیت می شوند. *

```
Key: value * 2  for key, value in D7.items()
```

Shabnam

جلسه دوازدهم:

تابع (function)



بسمه تعالی

Subject 12 جلسه Year 1401 — Month 5 — Date 6 —

تعریف تابع

$y = f(x)$

x = ورودی f = پردازش y = خروجی

جواب مسئله ⟸ پردازش‌هایی جهت رسیدن به جواب مسئله ⟸ ورودی مسئله

* بخش f یا پردازش خیلی مهمه ← هسته اون بخش است. الگوریتم

مثال:
ما یه متن داشتیم که می‌خواستیم ببینیم آیا شماره موبایل داخلش هست یا نه!

پیمایش تک تک کاراکترها: پردازش رشته یا str: ورودی
True یا False: خروجی

اگر بخواهیم یک مسئله رو به صورت تابع مدل کنیم:

فراخوانی تابع در برنامه ⟸ مدل نوشتن تابع ⟸ خروجی‌های مسئله ⟸ مشخص کردن عملیات پردازشی ⟸ داده‌ها یا ورودی مسئله

Shabnam

بسمه تعالی

Subject 12 جلسه Year 1401 — Month 5 — Date 6 —

* داده ها یا ورودی ها : میتونه متن ، عدد ، رشته و هرچیزی میتونن باشند.
(باید مشخص کنیم که اونا چی هستند.)

* پردازش : مشخص کنیم که چه پردازشی قراره روی x ها انجام بشه.
(الگوریتم رو مشخص کنیم.)

* خروجی : و در آخر باید مشخص کنیم خروجی که می خوایم چیه.

* مدل نوشتن تابع : بعد از سه مرحله ، باید مدل نوشتن که چهار مورد هست مشخص کنیم.

* فراخوانی : استفاده از تابع.

وقتی می خوایم تابع رو بنویسیم از کلمه کلیدی def استفاده می کنیم.

* نه ورودی داره نه خروجی

```
f1()

def f1():
    print("Python")
```

ورودی داخل پرانتز

کار خاصی میکنه و فقط Python رو چاپ میکنه.

f1() → "Python"

به این شکل استفاده میشه

Shabnam

بسمه تعالی

Subject 12 جلسه Year 1401 — Month 5 — Date 6 —

```
y=f2()
def f2():
return 2+2
```

* ورودی نداره ولی خروجی داره
* داخل یک متغییر ذخیره میشه
* y تغییر نمیکنه و کلا 4 است

یک کلمه کلیدی برای خروجی

y=f2() → y=4

```
f3(a)
def f3(a):
print(a**2)
```

هر اسمی میتونه باشه

* ورودی داره ولی خروجی نداره

نقطا گذاشتن میده

f3(8) → "64"

```
y=f4(a)
def f4(a):
return sqrt(a)
```

* هم ورودی داره هم خروجی
* تعداد ورودی و خروجی ها میتونه از یک الی آخر باشه
* این مدل انعطاف پذیری بالایی داره

y=f4(25) → y=5

```
def F1():
    print("Hi Friends")
```

در تابع ها هم تورفتگی هم است.

tab

Shabnam

بسمه تعالی

Subject. 12 جلسه Year. 1401 — Month. 5 — Date. 6 —

```
def F2():
    a = 2
    b = 3
    print(a**b)
```

} تغییر نمیکنه → الگوریتم → وورودی نیست

F2() → 8

```
def F4():
    a = [2,8,3,9,1]
    return max(a)
```

هیچوقتم لیست بفرستم → می تونم max بنویسم

```
def F7(a)
    return a
```

F7(5) → استاندارد → x = F7(5)

x

Shabnam

بسمه تعالی

Subject 12 جلسه Year 1401 — Month 5 — Date 6 —

تابع :

x = max (2,8,4) → بزرگترین

y = min (2,4,6) → کوچکترین

```
Print (x)
print (y)
 8
2
```

```
List = [6,7,8,2]
L = len (List)  → طول رو میده
L
```

* اینا هم تابع هستند . ما تا الان با خیلی از تابع ها آشنا شدیم.

* ما اگر داده ها رو مستقیم ذخیره کنیم نیازی به ترتیب نیست .

مثال ← Python2_12 ← سورس کدها

* آیا از if و else ها می تونیم داخل تابع استفاده کنیم؟

```
if Adult (16) == True:
```

اول Adult (16) رو بررسی میکنه ببینه 0 یا 1 چه مقدار میگیره برابر False

5? True

مثال ← Python2_12 ← نمونه سورس کدها

Shabnam

بسمه تعالی

Subject: 12 جلسه Year: 1401 — Month: 5 — Date: 6 —

* گاهی اوقات ما در کدمون ها از return استفاده می کنیم.
و این چیز جالبی نیست.
به جای اون می تونیم برای هر بخش if یا else ها یک متغیر در نظر بگیریم
و در آخر return رو بیاریم.
مثال ← Python2_12 ← نمونه سورس کدها

* ما توی تابع های مختلف می تونیم از تابع های دیگه استفاده کنیم.
مثال ← Pyton2_12 ← نمونه سورس کدها

* ما به طور پیش فرض می تونیم داده هارو دخیو کنیم.

```
def Person(Name = "Reyhane", Family= "Mohammadi", Age=16):
    return Name, Family, Age

N, F, A = Person()
print(N, F, A)

N, F, A = Person(Name="Mahoora", Family="Akbari")
print(N, F, A)
```

وقتی خالیه به طور پیش فرض پر میشه

می تونیم مقدار هارو تغییر بدیم ↑

Shabnam

بسمه تعالی

Subject: 12 جلسه Year: 1401 – Month: 5 – Date: 6 –

توابع می توانند داده ساختارها هم دریافت کنند.
مثال ← Python2_12 ← نمونه سورس کد

* اگر ما به این صورت ← F13(2,4,6,8) داده وارد تابع کنیم به ما ارور میده.
چرا؟ چون طولش مشخص نیست.

یه مسئله برای حل این موضوع پیش میاد ← *args

```
def F14(*args):
    for i in args:
        print(i)

F14(4,7,"python",5,2,[10,20,"C#"])
```

دیگه ارور نمیده ✂

* زمانی که می خوایم روی تک تک داده های یک داده ساختار پردازش انجام بدیم و می خوایم اون داده ساختار رو فراخوانی کنیم باید قبلش * بیاریم.
ولی وقتی چاپ میکنیم دیگه لازم نیست.

Shabnam

بسمه تعالی

Subject 12 جلسه Year 1401 — Month 5 — Date 6 —

* برای زمانی که ما به تک تک داده های دیکشنری ها دسترسی داریم از ** استفاده می کنیم.

مثال ← Python2_72 ← نمونه سورس کدها

* من می تونم همه اون موارد رو باهم برای تابع بفرستم.

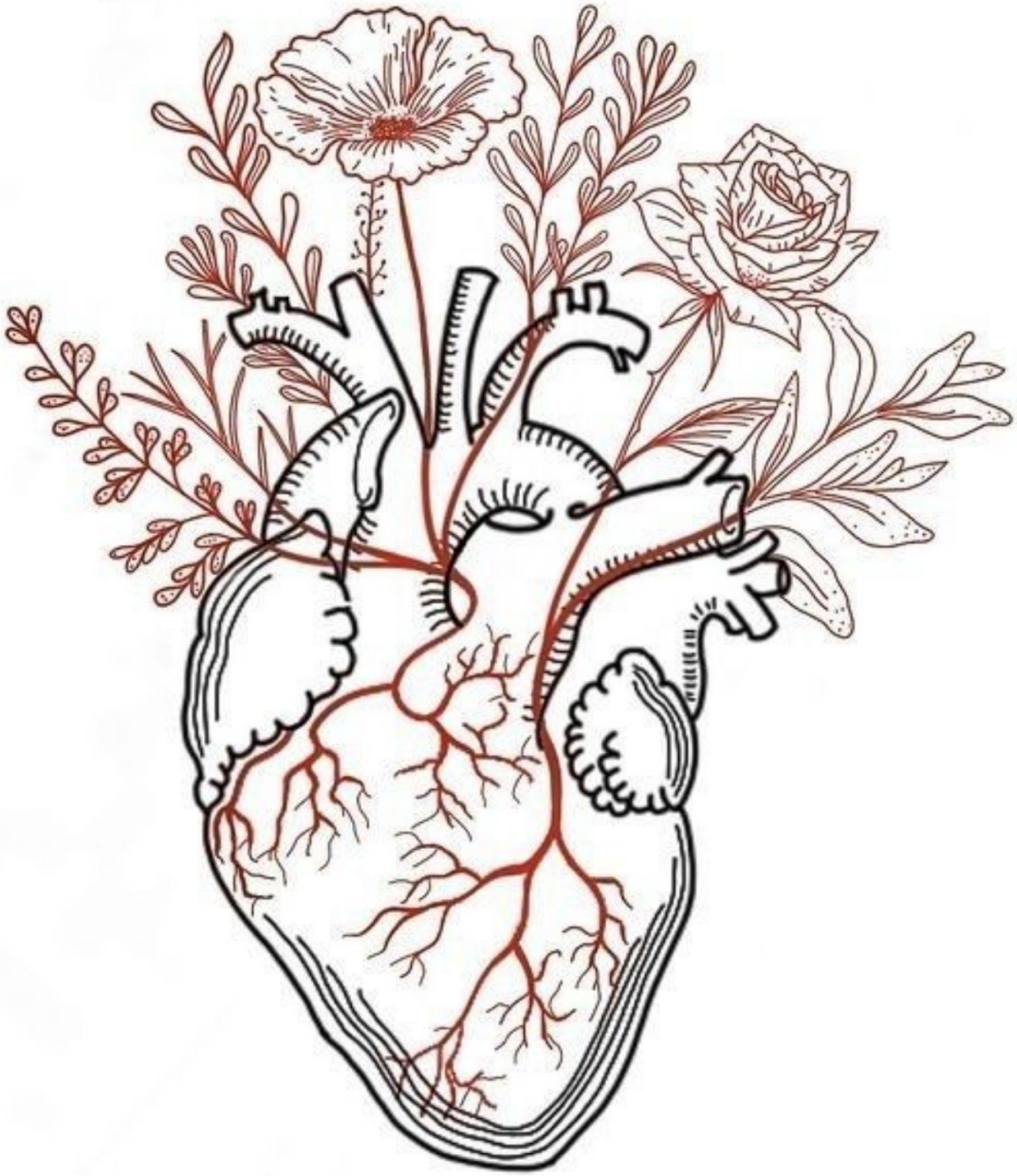
```
def F20(a, *args, default="Python", **kwargs):
    return a, args, default, kwargs

F20(2, [2,3,4], default="bigdataworld", **D2)
```

من می تونم هرکدوم اینا رو داخل متغیر قرار بدم.

Shabnam

جلسه چهاردهم:
توابع داخلی پایتون



بسمه تعالی

Subject: 14 جلسه Year: 1401 — Month: 5 — Date: 8 —

توابع داخلی پایتون:

lambda: تابعی که نام نداره، یک تابع بی نام که باید در یک خط کد تعریف بشه. و میتونه به عنوان ورودی در توابع دیگه مورد استفاده قرار بگیره.

```
def F1(a):
    return a**2

F2 = lambda a: a**2
```

کلمه کلیدی

```
F3 = lambda a,b: a*b
```

ورودی ها

کاربرد دارسی

```
print(F1(5))
print(F2(5))
print(F3(5,3))
```

F2 و F3 به عنوان اسم تابع است

ما می تونیم از lambda در لیست ها استفاده کنیم.

```
L1 = [F2(i) for i in range(10)]
L1

L2 = [(lambda x: x**2)(x) for x in range(10)]
L2
```

Shabnam

بسمه تعالی

Subject 14 جلسه Year 1401 — Month 5 — Date 8 —

lambda در دیکشنری هاهم کاربرد دارد.

```
Result4 = lamda x: x.values()

Result5 = lambda x: sum(x.values()) / len(D2)
```

تابع جدید

python2_14 ← نمونه کد

---

map : map(func, *iterales) دوتا ورودی میگیره.

func : تابع

*iterales : داده ساختار، خروجی که برمی گردونه ← map object.

map : Result = map(lambda x, x**2, L3)

← map میاد ورودی اول که یک تابع هست رو نگاشت میده به ورودی دوم که یک داده ساختار دیکشنری است.

نکته: ورودی اول lambda هست که مسئله رو به صورت lambda حل میکنه و بعد map نگاشت میده به L3.

x ← ورودی که میگیره به توان 2 میرسونه.

برای اینکه خروجی map رو ببینیم باید به لیست تبدیلش کنم list(Result)

Shabnam

بسمه تعالی

Subject: 14 جلسه Year: 1401 — Month: 5 — Date: 8 —

یه راه حل دیگه هم برای خروجی گرفتن هست، اینکه list رو مستقیم همون اول بیاریم:

```
Result3 = list(map(lambda x: x**2, L3))
Print(Result3)
```

* آخه من یکبار خروجی رو بینم و دوباره بخوام چاپش کنم داده ها پاک میشه.

فونه کدی Python2_14

---

filter(function or None, iterable) :filter

filter object

فیلتر: filter(lambda x: math.Pow(x, 2), L4)

برای خروجی گرفتن هم مثل map باید به list تبدیل بشه.
و همینطوری تو نیم list رو مستقیم بیاریم در یک خط کد.

---

min: هم روی داده ساختار اعداد هم رشته کاربرد داره.
کوچکترین اعداد یا کاراکتر رو نشون میده.

```
List2 = [2, 8, 4, 3, 9]
min(List2, key = lambda x: x/2)
```

یه مدل دیگه هم هست که min ورودی میگیره
یکی داده ساختار یکی هم تابع

shift + ~~enter~~ tab

توضیح میده

max: این هم به همین صورت ولی بزرگتر رو نشون میده

Shabnam

بسمه تعالی

Subject 14 جلسه Year 1401 — Month 5 — Date 8 —

sum : داده های داخل داده ساختار رو جمع میکنه.

```
sum(set, 11)
```

جمع اینا ← باین ← بالاین مدل هم میشه کار کرد

sorted : از کوچک به بزرگ مرتب میکنه.

```
sorted (list1, reverse= True)
```

از بزرگ به کوچک

False از کوچک به بزرگ

```
sorted (list1, reverse= True, key= lambda x: x/2)
```

* می تونیم به لیست داشته باشیم که داخلش مقداری داریم این sorted میاد براساس اسم، نمره و... مرتب میکنه.

نمونه کدی Python2_14

reversed : داده هارو برعکس چاپ میکنه.

all : می آد شرایط رو بررسی میکنه اگه همه True بود ← True

اگه حتی یکیش False بود ← False

باید قبل از خروجی all بیاریم.

```
all ([i>= 1 for i in numbers])
```

آیا همه مقادیر داخل numbers بزرگتر مساوی 1 است.

Shabnam

بسمه تعالی

Subject: 14 جلسه Year: 1401 — Month: 5 — Date: 8 —

any : میاد میبینه حتی اگر یکی از شرایط True بود ← True

```
any([i >= 14 for i in numbers])
```

آیا یکی هست که بزرگتر مساوی 14 باشه

zip : داده های دوتا داده ساختار رو باهم قرار میده.

```
L1=[0,2,4,6,8]
L2=[1,3,5,7,9]
```

```
list(zip(L1,L2)) →  اول با داده های L1
```

یا

```
list(zip(L2,L1)) →  اول با داده های L2
```

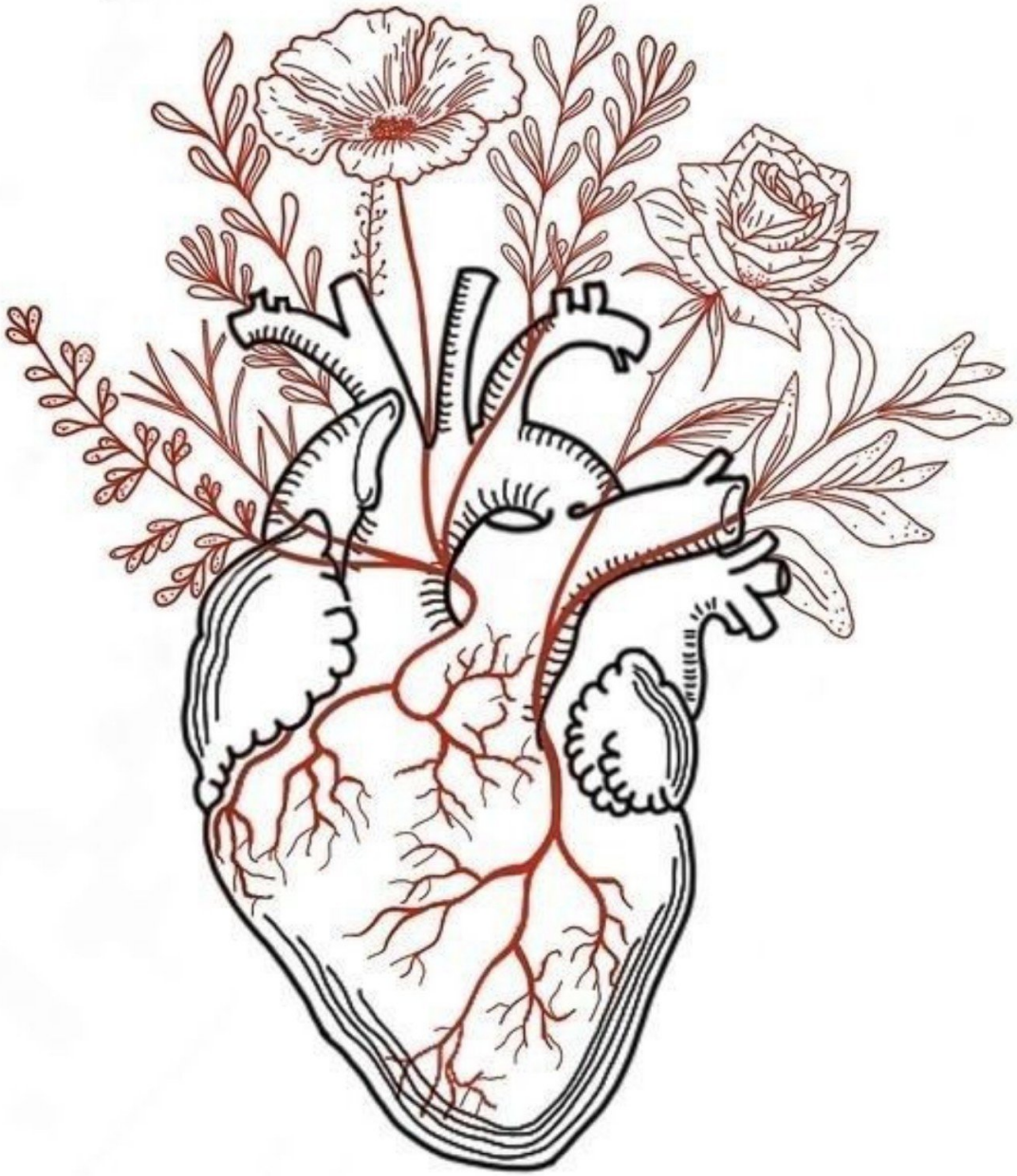
برای برعکس عمل کردن:

یعنی zip رو به داده ساختار اصلی تبدیل کنه.

```
list(zip(*L4))
```

Shabnam

جلسه پانزدهم:
خطاها و مدیریت آن ها



بسمه تعالی

Subject 15 جلسه Year 1401 – Month 5 – Date 8 –

Errors:

Indentation Error
syntax Error
name Error
type Error
index Error
value Error
key Error
attribute Error
ZeroDivision Error

Indentation Error:

* برای زمانی است که هم ترازی ها رعایت نشده.

syntax Eror:

* استاندارد های پایتون رعایت نشده.

name Error:

* متغیرها یا کتابخانه ها تعریف و فراخوانی نشده.

Shabnam

بسمه تعالی

Subject 15 جلسه Year 1401 — Month 5 — Date 8 —

Type Eror:

* برای مثال وقتی که روی دو تایپ متفاوت پردازش انجام میشه.
یا وقتی ورودی رو اشتباه بفرستیم.

index Error:

* برای وقتی که ترتیب اندیس ها رعایت نشده.
برای مثال ما اندیس چهارم داده ساختاری روی خوایم درحالی که اندیس چهارمی وجود نداره.

value Error:

* زمانی که یه رشته رو برای float یا int بفرستیم.
value اشتباه قرار بدیم.

key Error:

* زمانی که کلیدی که وجود نداره روی خوایم.

attribute Error:

* زمانی که داده ساختاری تابعی ندارد ولی ما ازاون استفاده میکنیم.

ZeroDivision Error:

* زمانی که عددی تقسیم بر صفر شود.

Shabnam

بسمه تعالی

Subject: 15 جلسه Year: 1401 – Month: 5 – Date: 8 –

raise:

میاد بررسی میکنه، خطاهایی که ممکنه پیش بیاد رو تشخیص میده.
و اگه نوع خطا رو هم بدونه بهتره ← در آخر یه پیغام چاپ میکنه.

fun تابع:

۳ تا ورودی داره ← یه عددی رو داخل یه داده ساختاری جای یه اندیس قرار میده.

try except fially:

زمانی که نمیدونی چه خطایی ممکنه بیوفته.

try:

اون لگدی که می خوام انجام بشه داخل try می نویسم.

except:

جمع میکنه که خطا رخ داده ← ولی نمیدونه چه خطایی.
بعضی وقتها هم ممکنه بدونی چه خطایی میوفته ← اونو بنویس.

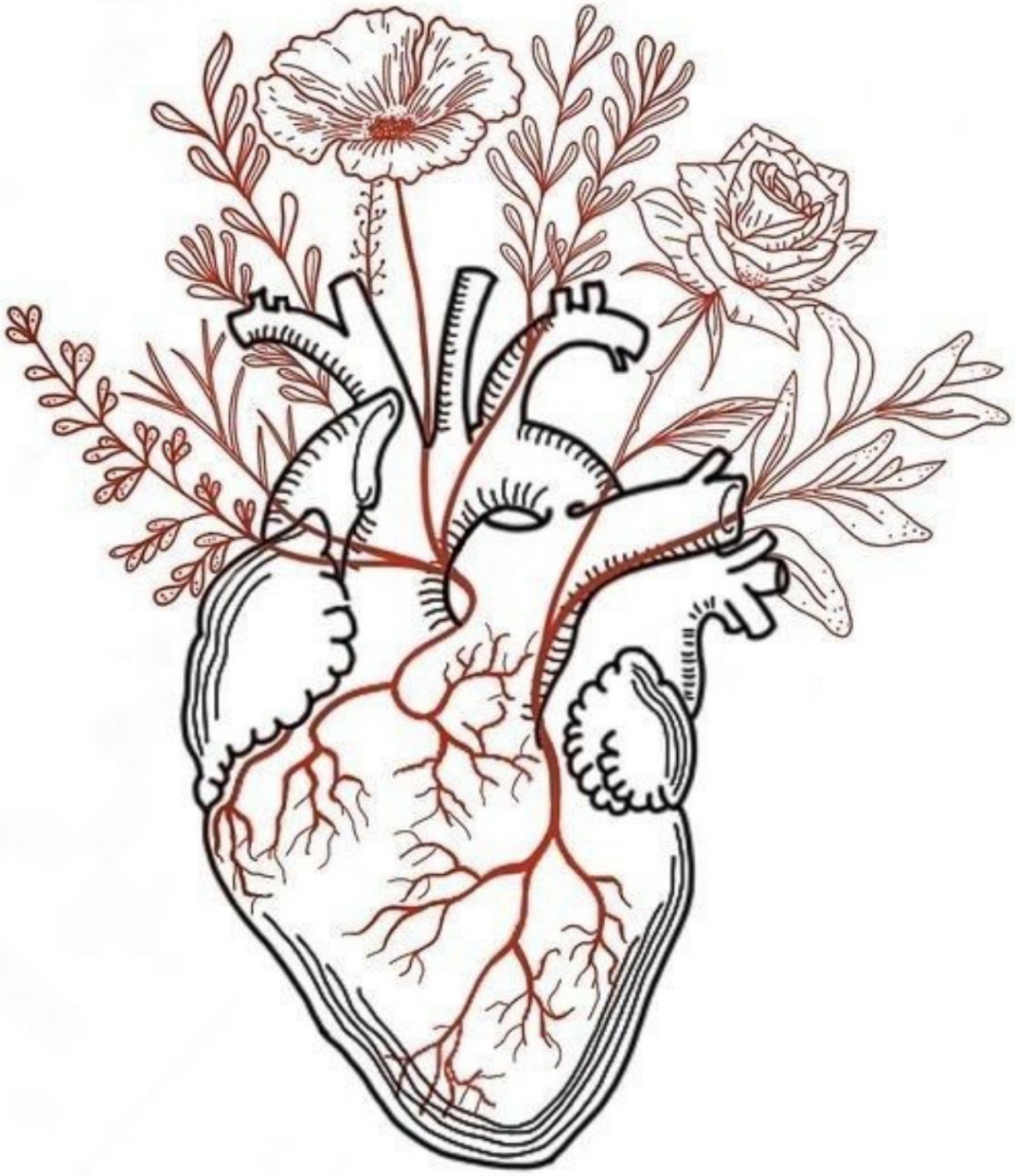
fially:

تحت هر شرایطی اجرا میشه.

Shabnam

جلسه شانزدهم:

فایل ها



بسمه تعالی

Subject 16 جلسه Year 1401 – Month 5 – Date 10 –

فایل ها: یکسری اطلاعات با پسوند خاص موجود در کامپیوتر.

txt: ← ساده ترین فایل متنی و عددی مرسوم.

استفاده از این فایل:

```
file = open("E:\Python\Python_Text.txt", 'r')
```

open متد — آدرس — پسوند — read

* برای اینکه این فایل رو وارد محیط پایتون کنیم:
باید با open فراخوانی کنیم، آدرس رو بدیم و داخل یک متغیر ذخیره کنیم.
حالا اون متغیر که حاوی آدرس و اسم فایل هست رو با یک متدی، داده هاش رو استخراج کنیم.

با tab + shift به توضیحات open دسترسی خواهیم داشت.

```
text = file.read()
```
با متد read می تونیم فایل رو بخونیم.

```
file.close()
```
برای اینکه جریان رو ببندیم؛ با close.

```
type(text) → str
```
رشته

```
text
```
فایل رو می تونیم ببینیم.

```
print(text)
```
به صورت مرتب ♡

Shabnam

بسمه تعالی

Subject 16 جلسه Year 1401 — Month 5 — Date 10 —

فایل قبلی متن انگلیسی بود، ولی اگه فایل ما فارسی بود چی؟

```
file = open("E:\Python\Python_Text2.txt", 'r', encoding
= "Utf-8")
```

بخش هایلایت شده رو به کد اضافه می‌کنیم.

* من می‌تونم اون فایل رو تغییر بدم و بجای یک کلمه مقدار جدید بفرستم:
با متد Replace

* من می‌تونم انتخاب کنم که چندتا کاراکتر اول برام چاپ بشه:
برای Read یه مقدار می‌فرستم. file.read(2)

* زمانی که بخوام فقط خط اول رو بخونه:
از متد Readline استفاده می‌کنم ← file.readline()

* من حق دیگه‌ای هم فایل رو وارد محیط پایتون کنم:

```
for f in file:
    Print(f)
```

* می‌تونیم داده‌های فایل رو داخل یک داده ساختار list بریزیم:
نمونه کد ← Ply

Shabnam

بسمه تعالی

Subject 16 جلسه Year 1401 – Month 5 – Date 10 –

من برعکس این عمل (read) رو می‌خوایم انجام بدم:
یعنی من یه متنی رو داخل متغیری ذخیره می‌کنم، و می‌خوام اون متن رو داخل یه فایل جدید بنویسه.
مثل read از تابع open استفاده می‌کنم و باز آدرس میدم که کجا و با چه نامی ذخیره بشه. اون فایل وجود نداره، خودش میسازه.
در آخر از متد writelines استفاده میکنه تا بنویسه.
نمونه کد ← Python2_16

برای مثال من می‌خوام یه متنی رو به آخر فایل اضافه کنم:
داخل open یای mode 'a' قرار میدم.

انواع مثال ها برای txt در ← Python2_16

* مدل سوم برای خوندن فایل ها:

```
with open ("E:\Python\Python_Text5.txt, 'a',
                                encoding="Utf-8")
```

Shabnam

بسمه تعالی

Subject 16 جلسه Year 1401 — Month 5 — Date 10 —

json: یک قالب شبیه به دیکشنری داره.

دیتا تونم داده‌ام رو در قالب دیکشنری بنویسم و بفرستم برای json و مستقیم بفرستم برای mongo db .

پایگاه داده Mongo db : یکی از محبوب‌ترین و معروف‌ترین پایگاه داده‌ای که در Database وجود داره.

باید حتما کتابخانه‌ی json را import کنیم.

برای خوندن فایل json :

```
with open('E:\Python\Python_json.json', 'r')
                                    as openfile:
    json_object = json.load(openfile)
```

تقریبا مثل txt هست ولی اینجا متد load به کار رفته.

* ما می‌تونیم به key و value ها هم دسترسی داشته باشیم.

* حالا برای اینکه خودم یه فایل بسازم:

```
with open('E:\Python\Python_json2.json', 'w',
                    encoding="utf-8") as json_file:
    json.dump(Python, json_file)
json_file.close()
```

Shabnam

بسمه تعالی

Subject 16 جلسه Year 1401 — Month 5 — Date 10 —

word:

من برای خوندن ورد باید از کتابخانه docx استفاده کنم.

pip install ← پس اون رو نصب می کنم

و بعد مراحوانی می کنم.

```
word = open('E:\Python\word_file.docx', 'rb')

word2 = docx.Document(word)

text2 = ""
for i in word2.paragraphs:
    text2 += i.text

text2
```

برای ذخیره کردن یک فایل ورد:

```
text = """ .... """

my_doc = docx_Document()
my_doc.add_paragraph(text)
my_doc.save("E:\Python\word_file2.docx")
```

Shabnam

بسمه تعالی

Subject 16 جلسه Year 1401 — Month 5 — Date 10

PDF: برای خوندن فایل Pdf هم باید از کتابخانه PyPDF2 استفاده کنیم ← بعد فراخوانی می کنیم.

```
pdf = open("E:\Python\Pdf_file.pdf", 'rb')
```
پسوند — mode

```
Pdf_reader = PyPDF2.PdfFileReader(pdf)
```
Pdf روی خونه

```
pdf_reader.numpages
```
→ تعداد صفحه

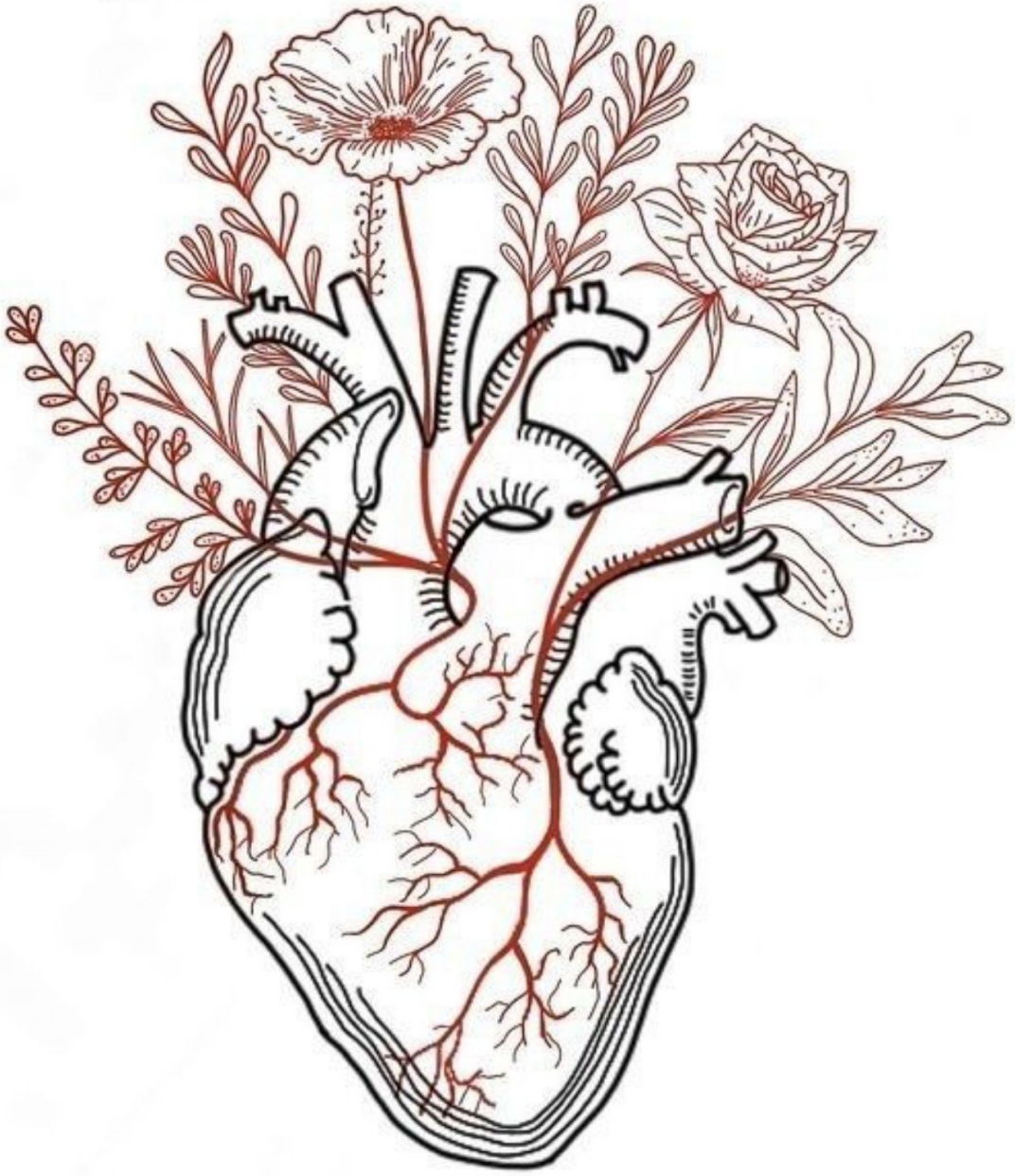
```
Page = Pdf_reader.getPage(0)
```
این صفحه رو نشون میده — اندیس صفحه

```
x=page.extractText()
```

```
x
```

Shabnam

جلسه بیستم:
کلاس خاموشی گرایی



بسمه تعالی

Subject 20 جلسه Year 1401 — Month 5 — Date 11

کلاس: یک قالب یا یک ساختار، که مسائل روی توسیم به این قالب نگاشت بدیم.

توصیف کننده یک موجودیت: تابع: رفتار / داده‌ساختار: ویژگی — شی یا نمونه

کلاس کاربر در لینکدین

ویژگی‌ها: نام، نام خانوادگی، آدرس پروفایل، عنوان، تخصص‌ها، ...

رفتارها: مشاهده پست، لایک کردن، کامنت گذاشتن، افزودن مهارت، ویرایش اطلاعات، ...

ریحانه، فاطمه، مهنا، هانیه

ساختن یه انسان با کلاس:

از کلمه کلیدی class استفاده می‌کنم. ویژگی رو داخل یک متغییر ذخیره می‌کنم.

برای ساختن رفتارها از def استفاده می‌کنم.

```
Reyhane = Human1()   → ذخیره شد
Reyhane.tab          → دسترسی به توابع
Reyhane.Name = "Mahoora"   → تغییر ویژگی‌ها
```

Shabnam

بسمه تعالی

جلسه 20 | 1407 | 5 | 11

کلاس های پایتون :

```
str1 = "Python Programming"
List1= list()
set1= set()
TuP1= tuple()
Dict1= dict()
```

برای ورودی توابع باید از self استفاده کنیم.

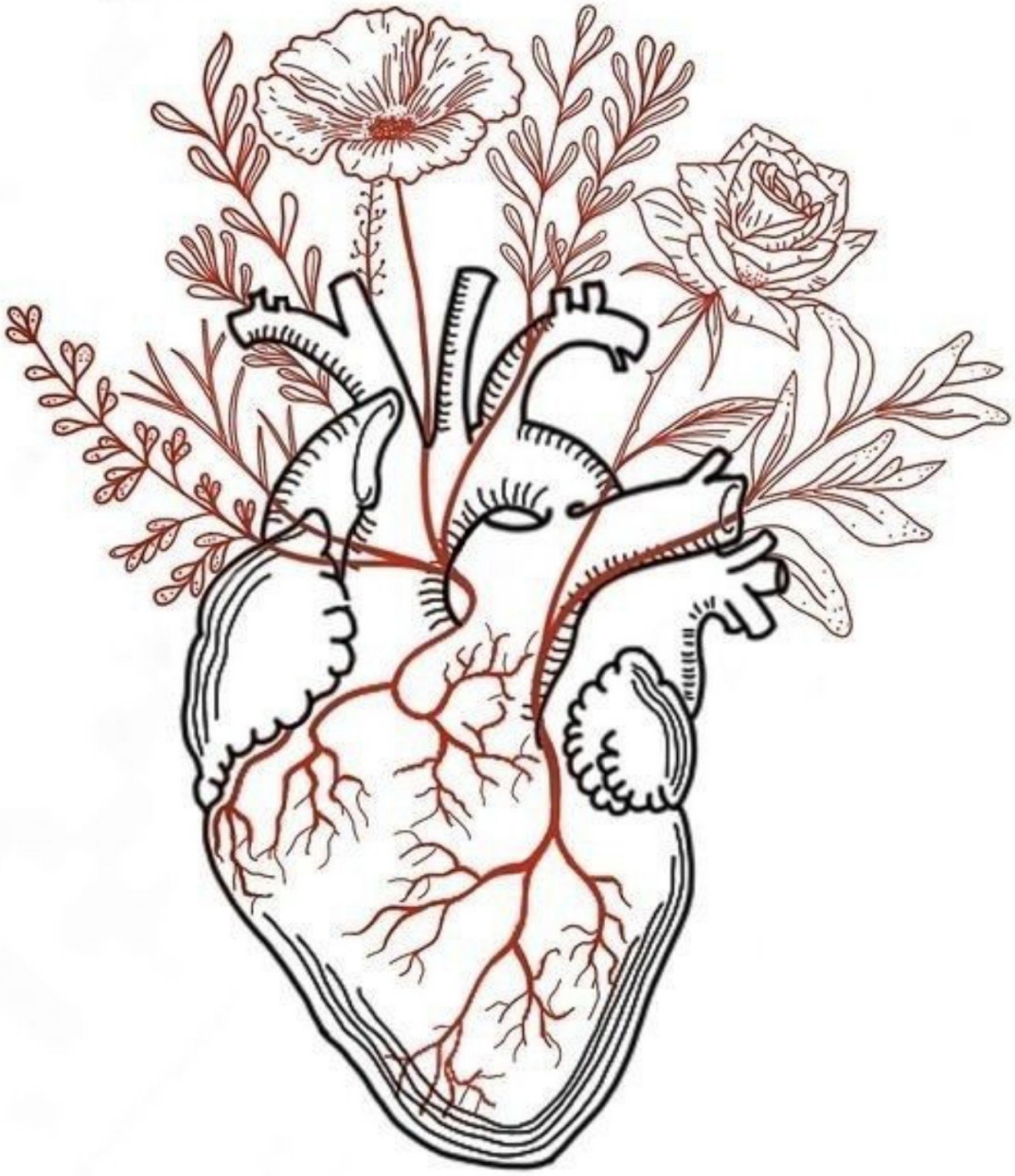
تابع __init__ : به طور پیش فرض داده ها رو ذخیره میکنه.
و می تونم اون رو تغییر بدم اسم جدیدی می سازم و با . +tab به توابع دسترسی پیدا کرد.
من می تونم بیام متغیر ها رو داخل تابع ذخیره کنم.
یک سازنده است.

* ورودی رو میشه از یک فایل دیگه هم خوند.

Shabnam

جلسه بیست و یکم:

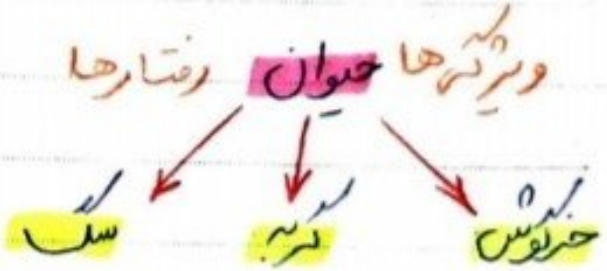
ارث بری



بسمه تعالی

Subject جلسه 21 Year 1401 – Month 5 – Date 11 –

ارث بری:



حیوان: کلاس پدر گربه: کلاس فرزند

* من یه موجودیت انسان می سازم با داده ساختار و توابع.
و حالا می خوام یه موجودیت جدید از موجودیت قبلیه ارث ببره، از Pass استفاده می کنم.
می تونم متدهاشو فراخوانی کنم و همینطور تغییرشون بدم.

موجودیت دوم ممکنه خودش داده هایی داشته باشه و با موجودیت اول یا پدر (ارث بیره) جمع بشه.

Shabnam

بسمه تعالی

Subject 27 جلسه Year 1401 — Month 5 — Date 11 —

* مثل کلاس ها مادی تونیم داخل ارث بری از متد __init__ استفاده کنیم.
و به صورت پیش فرض داده ها رو ذخیره کنیم.

* اگر کلاس پدر و فرزند هردو متد __init__ رو داشته باشه
و ما بخوایم هردو رو داشته باشیم باید:
هر دو متد رو با هم فراخوانی کنیم.
و یا از super استفاده کنیم.

برای ورودی ها هم باید مثل کلاس self رو بیاریم.

امیدوارم مفید باشه براتون:)