شماره یک

پایانی به کابوس‌های شبانه کامپیوتر

نویسنده و طراح: هومان سلیم زاده

با سیستم آموزشی جدید

hooman salimzadeh

سلام در این جزوه سعی شده با سرعتی بسیار سریع، نکات مهم درس کامپیوتر را بیاموزید، پس با ما همراه باشید |

متغیر ها

ما یک سری متغیر داریم که هر یک نوعی دارد، به طور مثال ما با اعداد، رشته ها و لیست ها آشنا شده ایم.
که هر یک از اینها خصوصیات خاص خودش را دارد و به ما امکانات متعددی را ارائه می کند

اعداد:

اعداد را کامپیوتر درک می کند، و می توان با آنها اعمال ریاضی انجام داد

نکته : // مقسوم علیه بدون اعشار را به ما خروجی می دهد

نکته : % باقیمانده تقسیم را به ما خروجی می دهد

نکته: پایتون الویت های ریاضیاتی را درک می کند

رشته ها:

رشته چیز هایی هستند که کامپیوتر کاری ندارد چه معنی دارند

نکته : می توان اعداد را هم رشته کرد: "1"

در این صورت کامپیوتر با معنی آن کاری نخواهد داشت، و م دیگر نمی توانیم روی آن اعمال ریاضیاتی انجام دهیم، در عوض به ما امکانات یک رشته را می دهد

امکانات یک رشته:

ما می توانیم با دستور len تعداد خانه های آن را به دست بیاوریم و ما می توانیم یک خانه آن را استخراج کنیم، به وسیله a[1]، و اینکه ما می توانیم به انتهای یک رشته یک رشته دیگر را اضافه کنیم.

نکته : کامپیوتر برای شمارش از صفر شروع می کند، ولی برای خواندن از یک!

لیست:

یک سوال! اگر قرار باشد شما یک برنامه بنویسید که درون آن قرار باشد اطلاعات کارمندان یک شرکت را ذخیره کنید، از چند متغییر استفاده می کردید؟

برنامه نویس ها در آغاز می شستند و متغییر تعریف می کردند برای تک تک اعضا! ولی بعد به این فکر رسیدند که یک نوع از متغییر را ایجاد کنند، که بتوان به آن بینهایت تا چیز دیگه اضافه کرد.

بدون اینکه کاری داشته باشه هر کدوم چیه!، و محدودیتی قرار بده

پس هر وقت ما نمی دونستیم توی برنامه هامون باید چند تا متغییر تعریف کنیم، می توانیم از لیست استفاده کنیم.

ما مانند رشته در لیست ها به تعداد خانه دسترسی داریم + اینکه می توانیم محتویات داخل آن را بخوانیم، لازم به ذکر است که نحوه شماردن در لیست مانند رشته ها است.

ساخت یک لیست:a=[1,1]

نکته : ما در لیست ها برای جدا کردن عضو ها از ویرگول انگلیسی استفاده می کنیم.

لیست ها امکانات دیگری هم به جز امکانات رشته ها به ما می دهند

به طور مثال اضافه کردن یک عضو:

a.append(1)

یا گرفتن بزرگ ترین عدد موجود در لیست:

Max(a)

یا گرفتن کوچکترین عدد:

Min(a)

یا مرتب کردن خانه ها از کوچک به بزرگ:

a.sort()

تبدیل شوندگان!

ما میتوانم یک رشته را به عدد، و یک عدد را به رشته تغییر دهیم، تا کارمان راحت تر شود! به طور مثال ما در دنیای اعداد نمی توانیم رقم دوم یک عدد را به راحتی بدست بیاوریم، یا یک عدد را به پایان عدد دیگر اضافه کنیم!(ما در اعداد می توانیم آنها را با هم جمع کنیم) و یا ...

نحوه تبدیل:

فرض می کنیم a یک متغیر اعددی است، و ما میخواهیم به رشته تبدیلش کنیم

S = str(a)

و برعکس(تبدیل رشته به عدد)

S2 = int(a)

نکته : ما می توانیم هر عددی را رشته کنیم، ولی نمی توانیم هر رشته ای را عدد کنیم، مثلا "salam" که یک رشته است را ما نمی توانیم عدد کنیم زیرا کامپیوتر نمی توانید آن را درک کند.

هرچه دل تنگت می خواهد بگو

ما یک سری چیز هایی داریم که مربوط می شوند به کاربر، و ما باید از کاربر بگیریم

نکته : ما چون نمی دانیم که ورودی چی هستش پایتون به صورت پیش فرض رشته ورودی می گیرد، و اگر ما بخواهیم باید آن را تبدیل کنیم

تا کنون آموختید : که یک سری اعداد و متون و ... را ذخیره، و در آن ها تغییر ایجاد کنیم، حالا می که کامل خیالمان از پایه راحت شد برویس سراغ کار اصلی خودمان!

کپی پیست نکنی ها!

برنامه نویس ها، وقتی اومدن یک سری برنامه طولانی بنویسند، دیدند که هی مجبور می شوند یک سری کد را تکرار کنند، که این موجب طولانی شدن برنامه می شد، و سر درآوردن از آن را سخت می کرد، ولی آنها خیلی مشکلی نداشتند، چون می شد کپی پیست کرد! ولی جمعی از دانش آموزان علامه حلی 3 تهران، به پیش بیلگیتس رفتند و گفتند ما در امتحاناتمان که نمی توانیم این همه کد را باز نویسی کنیم، و آن موقع بود که بیلگیتس نشت یه چیزیی به نام تابع رو آورد! این چیزی که بیلگیتس درست کرد کارش این بود که براش می شد یک سری اعمال رو تعریف کرد، و هر وقت که صداش می زدیم، اون کارها رو انجام می داد.

خب پس کار تابع رو فهمیدیم! اما نحوه استفاده:

```
Def a():
    Print(“salam”)
A()
```

در مثال بالا، ما گفتیم که هروقت گفتیم a یعنی برو و a رو پیدا کن، و دستورات توش رو اجرا کن

با اینکار بیلگیتس مشکل طولانی شدن برنامه ها تا حدودی کم شد، تا اینکه برنامه نویس ها دیدن که مثلا ما هی داریم دوتا عدد رو می گیم که از هم کم کنه و به توان سه برسونه، حالا برای این چطوری میشه تابع ساخت؟

برای همین به فکر ورودی گرفتن تابع افتادن!

اومدن گفتن که بیاییم بگیم که این متغییر را بتونند یک سری ورودی بگیرند، و با اونها کار کنند، اما چند تا مشکل بازم مونده بود!(بیچاره بیلگیتس که باید می شسته این همه چیز درست می کرده!)

قبل از اینکه به سراغ مشکلات برویم، ببینیم این ورودی ها چطوری کار می کنند:

```
Def a(b):
    Print(b)
A(“salam”)
```

در توابعی که ورودی می گیرند، ورودی جایگزین متغییر های ما می شود، و اعمال صورت می گیرد(مانند برنامه های عادی، فقط با یک تغییر که انگار ما دیگه متغییر تعریف نمی کنیم و ورودی می گیریم!)

اما حالا مشکل

اولا که ممکن بود دوتا متغییر هم نام باشند، یعنی من خودم یه جای برنامه از b استفاده کرده باشم، و توی تابع هم باشه! اینطوری که برنامه خراب می شد. اومدن گفتن که اون نام هایی که توی ورودی تابع هستن فقط توی تابع می مونند، یعنی با اون b بیرونی کاری نداریم، اون اسم یه اسمی هستش که قرار داد می کنیم برای اینکه بگوییم اون ورودیه!

مشکل اول که حل شد، دیدن ما بعضی جا ها نیاز داریم که توی متغییر های برنامه خودمون تغییر ایجاد کنیم

برای همین گفتن اگه توی برنامتون یه متغییری دارید که می خواهید همه جا یکسان باشه، اونو جهانی کنید

برای اینکار هم از دستور global بهره می بریم به طور مثال:

```
Global b
```

در مثال بالا، ما در برنامه خود هرجا بگوییم b با یک b کار داردا!. مثال:

```
Global b
B=10
Def c(a):
    Print(a-b)
C(2)
```

در مثال بالا همانطور که فهمیدید، برنامه به جای b از 10 استفاده می کند

خب تا الان کامل فهمیدیم که توابع چی هستن

حلقه ها وارد می شوند

یه روزی دانش آموزان حلی سه وقتی داشتن برنامه می نوشتن، دیدن که هی یک تیکه کد رو دارن تکرار می کنند، مثلا مجبور شده بودند ده بار پشت سر هم تابع a رو صا بزنند! درسته که با توابع کارشون راحت تر شده بود، ولی وقتی قرار بود یه کار ده بار نه! صد بار تکرار بشه چی؟

اول اومدن گفتن ته یه تابع یه اتفاق چند بار بیفته، بعد وسط برنامه وقتی می خواستند یه اتفاق بیفته به مشکل خوردن و مجبور شدن کلی تابع تعریف کنند، که پشیمون شدند! و رفتند به پیش فدرگی گفتن ما بدبخت شدیم(بیلگیتس خسته شده بود، برای همین رفتن توی اپل گفتن مسئول نرم افزارتون کیه؟) اونم اومد گفت من یه فکری دارم! یه دستور جدید درست می کنیم که بشه با اون یه قطعه کد رو چند بار اجرا کرد. و همین جا بود حلقه for ساخته شد!

اومد گفت من یه متغییر از شما می گیریم که با اون بتونم چک کنم که اون اون اتفاق به تعداد لازم رخ بده، برای این هم که به درد شما بخوره این متغییره شما خودتون نام بزارید براش، و براش نقطه شروع و پایان قرار بدید، متغییره از عدد شروع شما هر بار که حلقه اجرا میشه یکی بیشتر میشه تا به یکی مونده نقطه آغاز برسه، اینطوری تعداد اجرا هم درست میشه. فقط گفت که من الان خیلی خستم! حوصله ندارم براتون منفیش رو بنویسم، پس حتما مراقب باشید که عدد دومی از اولی بزرگ تر باشه.

خب دانش آموزان حلی سه کامل نکات مربوط به حلقه for رو آموختند! مثال:

```
For I in range(10):
    Print(i)
```

همانطور که می دانید، مثال بالا اعداد 0 تا 9 را چاپ می کند(I همان شمارنده است که ما می توانیم هر نامی برایش قرار دهیم، و در پرانتز هم عدد آغاز و پایان را قرار می دهیم که به صورت پیش فرض صفر است).

شرط و شروط

دیگه لازم نیست برای اینا داستان درست کنم، ما هر وقت توی برنامه خواستیم شرط قرار بدیم از if استفاده می کنیم، مثال:

```
A=20
B=20
If a==b:
    Print(1)
```

نکته : ما در شروط از علامت های == برای چک کردن برابر بودن، =< و => برای بزرگ تر و کوچک تر مساوی بهره می بریم. اما وقتی دانش آموزان حلی سه اومدن از این استفاده کنند، دیدن بعضی وقتا نیاز میشه که چند تا شرط درست باشه، یا اینکه فقط کافیه که یکی از این دوتا شرط درست باشه، پس به پیش خودو فان روسوم(نویسنده اصلی پایتون) و بهش گفتن بیا و کار ما رو راحت کن!

اونم اومد and و or رو درست کرد، که میاد چند تا شرط رو چک میکنه.(الویت اول با or است و بعد and)

حلقه ها باز می گردند

دانش آموزان حلی سه که داشتن برنامه های مختلف می نوشتن، احساس نیاز به یک چیز خاص و جدید کردند، اینکه یه حلقه تا یک جایی اجرا بشه، مثلا تا وقتی که a دو باشه، برای همین این دفعه خودشون دست به کار شدن و یک چیز جدید درست کردند! حلقه while

مثال:

```
A=5
While a!=10:
    A=a+1
```

این حلقه تا زمانی ادامه پیدا می کند که شرط ما درست باشد، این حلقه به علاوه زمانی که ما می خواهیم یک کار تا یک زمان خاصی اجرا بشود، بیشتر به درد زمانی می خورد که ما می خواهیم یک چیز تا ابد ادامه پیدا کند، برای این کار می توانیم درون آن یک شرط درست قرار دهیم(مثلا 1==1)

خب حالا بچه ها کلی از مشکلاتشون حل شده بود، تا زمانی که رسیدن به جایی که یک سری از دانش آموزان گیر داده بودند که ما فقط for رو می خواهیم، برای همین اومدن یه دستور درست کردن به نام break که این دستور رو اگه توی حلقه for یا while قرار بدید، میای حلقه رو نگه می داره و خارج میشه، با اینکار هوشمندانه دانش آموزان، یک امکان جدید به برنامه نویس ها اضافه شد! اینکه بدون زحمت یک برنامه بنویسند که بیاد یک کاری رئ انجام تا اینکه یک شرطی درست بشه یا ده بار اجرا شده باشه،(الان اصلا انتظار ندارم فهمیده باشید چی گفتم! به مثال پایین نگاه کنید، اینطوری متوجه میشید!)

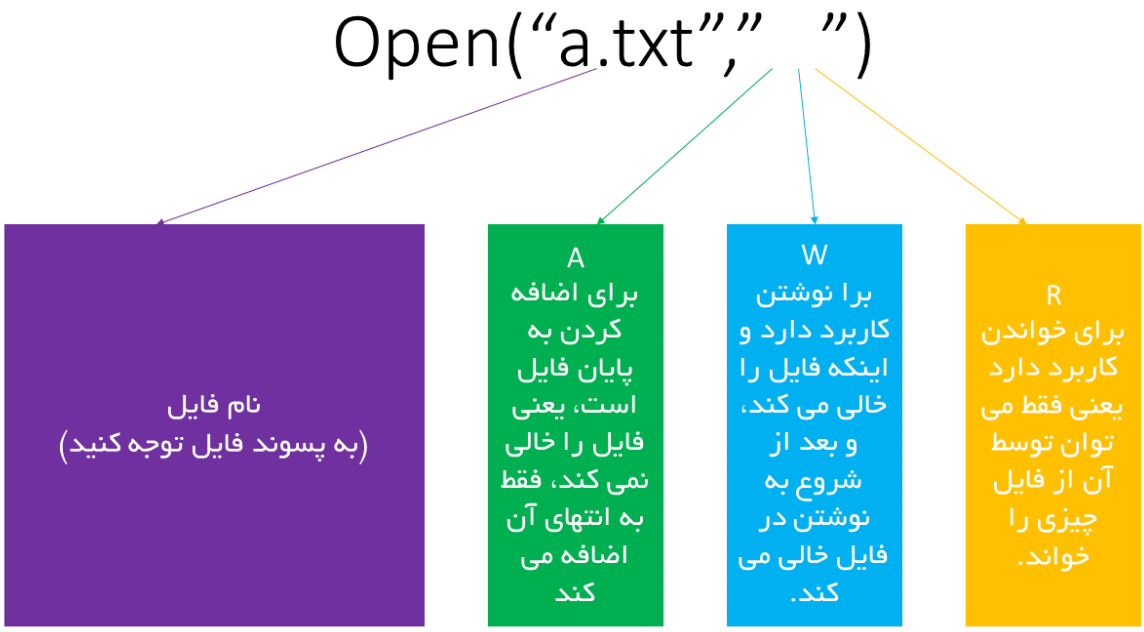
```
A=0
For I in range(10):
    A=a+5
    If a>=10:
        break
```

خب حالا مشکلمون با حلقه ها هم حل شد!، الان خیلی راحت می تونیم یک سری کار رو چند بار، نا محدود یا محدود به شرط یا محدود به تعداد بار بسازیم

ذخیره اطلاعات!

دانش آموزان حلی سه که با حلقه و لیست و تابع اینا یادگرفتن کار کنند، اومدن یه سامانه درست کنند برای یک شرکت، و به یک مشکل بزرگ برخورد کردند، اینکه چطوری اطلاعات کاربران رو ذخیره کنند، تا بار بعدی که برنامه داره اجرا میشه لازم نباشه دوباره وارد بشوند.

خب اومدن دیدند که چه چیزی هستش که پاک نمیشه؟، دیدن فایله! برای همین رفتن کار با فایل ها رو یادگرفتن! مثال:




```
f = open(“hello.txt”,”a”)
f.write(“salam”)
f.close()
```

همانطور که می دانید، حتما باید فایل را ببندید(حتما)

+ ما دو تا دستور داریم، یکی write و دیگری readline که write برای نوع های w و a کاربرد دارد و readline برای r

+ اینکه ما فقط در w اجازه داریم که فایلی با آن نام نساخته باشیم، وگرنه با ارور روبرو خواهیم شد

+ حتما باید رشته وارد فایل بشود

مشکلات ما! اینکه وقتی یه چیزی می رود توی فایل نوعش رو از دست میده، پس ما نیاز داریم که از توی فایل همون اول برنامه اطلاعاتمون رو برداریم، روش برداشتن اطلاعات در فایل:

```
F = open(“salam.txt”,”r”)
A=[]
G=f.readline ()
While g!=””:
        a.append(g)
        g=f.readline()
f.close()
print(a)
```

اما الان بازم یه مشکل داریم، اینکه اطلاعاتی که می روند توی فایل آخرشون یه "/n" اضافه می شود، و ما باید اونها رو برداریم، برای برداشتن اونها دانش آموزان علامه حلی سه تصمسم گرفتند خودشون مشکل رو حل کنند!، (خودتون این بخش رو بنویسید، و تست کنید)

خب حالا می روین سراغ نوشتن در فایل، که نکته خاصی نداریم به جز اینکه حواستان به اضافه کردن ”/n“ باشد، و به w و a هم دقت کنید.

پایان

جزوه شماره دو با ماجراهایی جذاب تر در راه است!