بسم الله الرحمن الرحيم

دانشگاه پیام نور استان تهران

مرکز / واحد پرند

گروه فنی و مهندسی

پروژه کارشناسی

رشته ی مهندسی کامپیوتر

گرایش نرم افزار

عنوان پروژه:

ماشین حساب دیجیتال پایتون

استاد راهنما:

سید علی رضوی ابراهیمی

تهیه کننده:

سارا ناصری

مرداد ماه 1400

کلیه حقوق مادی مترتب بر نتایج مطالعات، ابتکارات و
نوآوری های ناشی از این پروژه متعلق به :

"دانشگاه پیام نور استان تهران /مرکز پرند"

می باشد.

چکیده

همانطور که می دانید امروزه علم کامپیوتر از مهمترین و پر کاربرد ترین علم های دنیا محسوب می شود و کامپیوتر یک ابزار مهم و جدا نشدنی در زندگی امروزی ما حساب می شود .برنامه نویسی نیز یکی از مهم ترین رکن های کامپیوتر حساب می شود که باعث می شود کار های الکترونیکی خود بدون هیچ زحمتی انجام دهیم. از گوشی های همراه گرفته تا چراغ های راهنمایی و رانندگی همه نیازمند برنامه نویسی هستند تا بتواند کار خود را به درستی انجام دهد. کامپیوتر چیست ؟کامپیوتر یک وسیله چند منظوره است که دارای حافظه بوده و قابل برنامه ریزی است و می تواند انواع داده ها را به عنوان ورودی بپذیرد و به عنوان خروجی ارائه دهد. کامپیوتر ها بر اساس دو مدار صفر و یک کار می کنند و عملیات خود را بر اساس مدار الکترونیکی انجام می دهند. کامپیوتر دو بخش اساس سخت افزار و نرم افزار دارد. سخت افزار شامل قطعات فیزیکی کامپیوتر است که قابل لمس است. بخش نرم افزار نیز بخش غیر قابل لمس کامپیوتر است. نرم افزار می تواند به شکل داده ورودی به ماشین ، برنامه های موجود در سیستم و بسته های نرم افزاری کاربری باشد. یک سیستم کامپیوتری علاوه بر قطعات سخت افزاری نیازمند نوعی نرم افزار است که به آن جان بخشیده و فعالیت کند. پس در سیستم کامپیوتری نرم افزار و سخت افزار نیازمند یکدیگر هستند. نرم افزار در حقیقت روح و جان کامپیوتر است که به سخت افزار هویت می بخشد .

فهرست مطالب

فهرست جداول

فهرست تصاویر

پیشگفتار

در این مقاله سعی کردیم در طی چهار فصل با زبان برنامه نویسی پایتون آشنا شویم و برای راحت تر درک کردن این زبان برنامه نویسی یک برنامه ساده اما کاربردی نیز ساخته و پیاده سازی می کنیم. پایتون جزو زبان های ساده و شیرین و همچنین پرکاربرد و قدرتمند در دنیای برنامه نویسی است. پس اگر اول راه هستید و می خواهید یک زبان برنامه نویسی یاد بگیرید پایتون بهترین پیشنهاد است. چون هم سرعت یادگیری بالایی دارد هم طرفداران زیادی را به خود اختصاص داده است . در این مقاله با من همراه باشید تا با همدیگر نحوه ساخت یک برنامه به زبان پایتون را یاد بگیریم .

فصل اول

مقدمه

در این فصل یاد می گیریم پایتون چیست و چگونه به وجود آمده است. همچنین تابع ها ، رابط ها و حکم های دستوری پایتون را توضیح می دهیم. در مورد عملگر های مختلف پایتون صحبت می کنیم و یاد می گیریم کار کردن عملگر ها چگونه است. بعد از آن نحوه نصب برنامه رسمی پایتون را یاد میگیریم تا بتوانیم پروژه روی آن پیاده سازی کنیم

1-1 پایتون چیست

پایتون[1] یک زبان برنامه نویسی شی گرا ، تفسیری ، سطح بالا و همه منظوره است که توسط خیدو فان روسوم[2] مهندس هلندی در کشور هلند طراحی شد و اولین بار در سال 1991 منتشر شد. هدف از ساخت این زبان بهره وری بالا و خوانایی بیشتر بوده. ساختار زبانی و دیدگاه شی گرا در پایتون به گونه ای طراحی شده است که به برنامه نویس امکان نوشتن کد منطقی و واضح را برای پروژه های کوچک بزرگ می دهد.

در پایتون مدل های مختلف برنامه نویسی را پشتیبانی می کند و برای مشخص کردن نوع متغیر ها از یک سامانه پویا استفاده می شود.

این زبان از زبان های برنامه نویسی مفسر بوده و به صورت کامل یک زبان شی گرا است و شباهت بسیاری به زبان های پرل[3] ، روبی[4] ، تی سی ال [5]دارد.

ویژگی های پایتون :

- چندمنظوره
- شی گرا
- سطح بالا
- قابلیت خوانایی و بهره وری بالا
- سادگی در یادگیری
- متن باز بودن
- ایمن بودن

1 -2 تاریخچه

پایتون اواخر دهه 80 میلادی در موسسه ملی تحقیقات ریاضی و رایانه در کشور هلند توسعه یافت. هدف

---

[1] Python
[2] Guido van Rossum

[3] Perl
[4] Ruby
[5] Tool Command Language

خیدو از توسعه پایتون پیدا کردن جانشینی برای زبان برنامه نویسی ای بی سی [6]بود که بتواند استثنا ها را نیز پردازش کند.

پایتون در دو نسخه 2.0 و 3.0 منتشر شد که نسخه 2.0 شامل بازیافت حافظه با قابلیت شناسایی دور و پشتیبانی از یونیکد[7] بود و پایتون 3.0 بازنویسی نسخه قبل بود و شامل تغییرات عمده ای مثل تغییر پرینت بود.

زبان پایتون نسبت به زبان های دیگر سرعت کمتری دارد و این زبان سطح بالایی دارد و مانند زبان C رابطه خوبی با سخت افزار ندارد ، و برای کار هایی که نیاز به حافظه کوتاه مدت دارند پیشنهاد نمی شود.

چه کسانی از پایتون استفاده می کنند؟ کلیه افرادی که در حوزه شبکه و اسکریپت نویسی فعالیت می کنند از این زبان استفاده می کنند. همچنین به خاطر امنیت بالا و سازگاری خوبی که دارد هکر ها در حوزه امنیت اطلاعات نیز از پایتون استفاده می کنند. این زبان برنامه نویسی در موتورهای جستجوی گوگل و اسپایدار های وب و موتور گرافیکی یوتیوب استفاده شده است. از آنجایی که این زبان بسیار انعطاف پذیر و کاربردی در طراحی نرم افزارهای محاسباتی است ، سازمان فضایی ناسا نیز در ساخت برنامه های خود از آن بهره می برد. این زبان برنامه نویسی در توزیع های مختلف سیستم عامل لینوکس نیز وجود دارد.

1-2-1 واژه های ابداعی

پایتونیک ، یک واژه ابداعی در انجمن پایتون است که محدوده معنایی وسیعی را در رابطه با سبک برنامه نویسی[8] در بر می گیرد . به کدی پایتونیک می گویند که از اصطلاحات پایتون به خوبی استفاده کرده باشد و مطابق خوانایی بالا در پایتون باشد. در مقابل آن کد آنپایتونیک است که در واقع کدی است که فهم آن مشکل است یا مانند رونویسی از زبان دیگری است.

1-3 دستور زبان

پایتون برخلاف زبان های دیگر که از نقطه گذاری استفاده می کنند ، پایتون به دلیل خوانایی بالا اغلب از کلمات کلیدی انگلیسی استفاده می کند.

---

[6] زبان برنامه نویسی همه منظوره که در CWI کشور هلند تولید شده است.

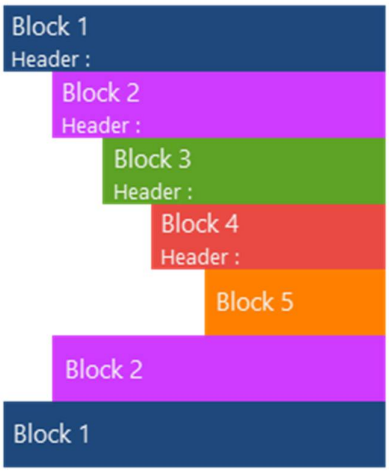
[7] Unicode

[8] مجموعه ای از قوانین و دستور العمل است که هنگام نوشتن کد منبع برای یک برنامه کامپیوتری مورد استفاده قرار می گیرد.

پایتون برای جدا کردن بلوک کد به جای استفاده کردن از آکولاد از تورفتگی فاصله خالی استفاده می‌کند. برای شروع بلوک کد یک پله تورفتگی را بیشتر می‌کنیم و برای اتمام آن یک پله بلوک کد را کمتر می‌کنیم. پس ساختار ظاهری برنامه نمایش دهنده ساختار معنایی آن است.

روش رایج تورفتگی در پایتون استفاده از کلیدspace و tab است . استفاده از تورفتگی اجباری نیست ولی برای خوانایی بیشتر مفید است.

تصویر زیر نشان دهنده تو رفتگی پایتون است:




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

def bubble_sort(seq):
    L = len(seq)
    for i in range(L):
        for n in range(1, L):
            if seq[n] < seq[n - 1]:
                seq[n - 1], seq[n] = seq[n], seq[n - 1]
    return seq

print bubble_sort(items)
```

شکل1-1

1-3-1 حکم های پایتون

حکم if که یک بلوک کد است تا رسیدن به else و elif را اجرا می کند.

حکم for روی یک شی تکرار شدنی تکرار می شود و هر عضو آن را برای استفاده توسط بلوک مربوط به متغیر محلی می دهد.

حکم while تا زمانی که شرط برقرار باشد بلوک کد را اجرا می کند.

حکم try حکمی است که برای استثنا ها بکار می رود و بلوک کد آن همراه با except و finally به کار می رود.

حکم raise برای ایجاد استثنا ها از آن استفاده می‌شود.

حکم class که یک بلوک کد را اجرا می‌کند و فضاهای محلی آن را به یک کلاس ملحق می‌کند، برای استفاده در برنامه‌نویسی شی گرا بکار می رود .

حکم def که تابع را تعریف می‌کند. و حکم with که یک بلوک کد را به یک مدیر محتوا ضمیمه می‌کند.

حکم break برای خروج از نزدیک‌ترین حلقه استفاده می‌شود.

حکم continue که برای پریدن از دور جاری و ادامه دادن از دور بعدی در نزدیک‌ترین حلقه استفاده می‌شود.

حکم del که برای پاک کردن متغیر ها از آن استفاده می‌شود. و حکم pass که به عنوان NOP[9] از آن استفاده می‌شود و از آن برای درست کردن بلوک کد خالی استفاده می‌شود.

حکم assert که هنگام دیباگ کردن استفاده می‌شود. شرطی که باید اتفاق بیافتد را بررسی می‌کند.

حکم yield که از آن در توابع مولد به جای return استفاده می‌شود.

حکم print که در پایتون 3 تبدیل به تابع ()print شد. و حکم import که برای وارد کردن ماژول استفاده می‌شود.

حکم return که برای برگرداندن مقداری در تابع استفاده می‌شود.

هر حکمی برای خودش قواعد معنایی خاصی دارد ،مثلا حکم def برخلاف دیگر حکم ها بلوک خود را فورا اجرا نمی کند.

1-4 عملگر ها

عملگر [10] به نمادی گفته می شود که عمل مشخصی را بر روی اشیا به انجام می رساند ، همچنین به اشیایی که عملگر بر روی آن ها عملی را به انجام می رساند عملوند[11] گفته می شود. عملگر ها دارای انواع مختلفی هستند که در ادامه به آن ها میپردازیم.

1-4-1 انواع عملگر

عملگر های حسابی[12] برای انجام پردازش های ریاضی مانند جمع ، تفریق ، ضرب و... استفاده می شود. جدول نشان دهنده عملگر حسابی است :

عملگر	توضیح	مثال
+	دو عملوند را جمع می کند	5 + 4= 9
-	عملوند دوم را از اولی کم می کند.	3-2 =1

[9] دستور زبان برنامه نویسی که هیچ کاری انجام نمی دهد.
[10] Operator
[11] Operand
[12] Arithmetic Operators

*	دو عملوند را ضرب می کند	2 * 6 = 12
/	دو عملوند را تقسیم می کند	40/8 = 5
%	باقی مانده تقسیم دو عدد را بر می گرداند	10 % 4 = 2
//	خارج قسمت صحیح	18 // 5 = 3
**	به توان رساندن	3 ** 5 = 243

جدول 1-1

جدول بالا نشان دهنده عملگر های حسابی پایتون است.

عملگر های مقایسه ای [13] برای انجام مقایسه مقادیر مورد استفاده قرار می گیرد. متناسب با شرط خروجی را true یا false می دهند. جدول زیر نشان دهنده عملگر مقایسه ای پایتون است.

عملگر	توضیح	مثال
>	بزرگ تر است	a > b
<	کوچک تر است	a < b
==	برابر است	a==b
!=	نامساوی	a ! = b
>=	بزرگتر مساوی	a>=b
<=	کوچکتر مساوی	a<=b

جدول 1-2

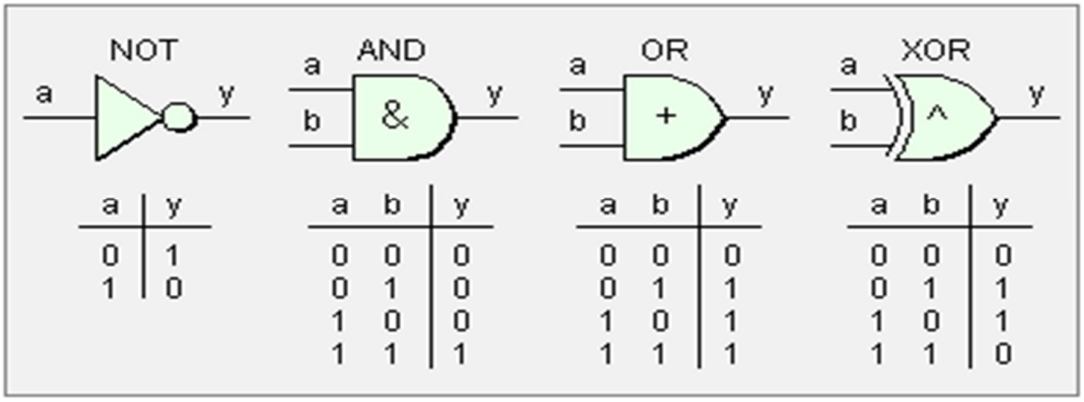
عملگر های منطقی[14] در واقع and ، or و not است و در پایتون با یک سری عبارت های شرطی که به کار می رود چک می شود و این عبارت ها معادل True و False هستند. جدول زیر توضیحاتی در مورد این عملگر می دهد:

عملگر	توضیح	مثال
And	در صورتی که هر دو عملوند true باشد درست است	a and b
Or	در صورتی درست است که یکی از عملوند ها true باشد	a or b
Not	در صورتی درست است که عملوند false باشد	Not a

جدول 1-3

[13] Comparison Operators
[14] Logical Operators

عملگر های بیتی[15] روی عملوند هایی از نوع رشته یا ارقام دودویی کار می کند. همانطور که مشخص است پردازش را بیت به بیت انجام می دهند. برای مثال 2 در حالت دودویی برابر 10 است. قبل از اینکه با نماد های عملگر بیتی آشنا شویم بهتر است در ابتدا با گیت های and ، or ، not و xor آشنا شویم. طبق شکل 1-2 داریم :



شکل 1-2

طبق تصویر بالا یاد میگیریم هر گیت در چه نقطه ای صفر و در چه نقطه ای یک می باشد . حال برای فهمیدن نماد های عملگر بیتی تصور کنید a=10 و b=4 باشد. طبق شکل 1-2 فرمت باینری آن ها به صورت زیر خواهد بود:

a = 0000 0100, b = 0000 1010

عملگر	توضیح	مثال
&	And(و) بیتی	a & b = 0000 0000
|	Or (یا) بیتی	a | b = 0000 1110
~	Not (نقیض) بیتی	~a = 1111 0101
^	Xor بیتی	a ^ b = 0000 1110
>>	جا به جایی به راست بیتی	a >> = 0000 0010
<<	جا به جایی به چپ بیتی	a << =0010 1000

جدول 1-4

[15] Bitwise Operators

عملگر بعدی عملگر تخصیص[16] است که در پایتون برای تخصیص مقدار به یک متغیر مورد استفاده قرار می گیرند.

مثلا a=5 یک عملگر تخصیص ساده است که مقدار 5 را در سمت راست به متغیر a در سمت چپ تخصیص می دهد.

عملگر های ترکیبی مانند a+=5 هم در پایتون وجود دارد که به متغیر اضافه می شود و بعد ها به طور مشابه تخصیص پیدا می کنند. مثلا تبدیل می شود به a= a+5 ، در جدول زیر باهم دیگر عملگر های تخصیص را میبینیم :

عملگر	مثال	برابر است با
=	a=5	a= 5
+=	a +=5	a= a+5
-=	a-=5	a= a-5
*=	a*= 5	a= a* 5
/=	a /= 5	a = a/5
%=	a%=5	a= a%5
//=	a//=5	a= a//5
**=	a**=5	a= a**5
&=	a& =5	a= a&5
|=	a|= 5	a=a|5
^=	a^= 5	a=a^ 5
>>=	a >>=5	a= a>>5
<<=	a<<=5	a= a<<5

جدول 1-5

1-5 عملگر های خاص

در زبان پایتون انواع خاصی از عملگر ها وجود دارد مانند عملگر همانی و عملگر عضویت که در ادامه این دو عملگر را تشریح می کنیم

1-5-1 عملگر های همانی

is و is not عملگرهای همانی در پایتون هستند. این عملگر ها برای بررسی اینکه آیا دو مقدار در بخش های مشابهی از حافظه قرار دارند یا نه مورد استفاده قرار می گیرند. البته مساوی بودن تو متغیر به معنای همانی بودن آن نیست

[16] Assignment Operators

مثال	توضیح	عملگر
a is true	در صورتی درست است که عملوند ها همانی باشند(به شی مشابهی ارجاع داشته باشند)	Is
a is not true	در صورتی درست است که عملوند ها همانی نباشند(به شی مشابهی ارجاع نداشته باشند)	is not

جدول 1-6

1-5-2 عملگر های عضویت

in و not in عملگرهای عضویت در پایتون هستند. این عملگرها برای بررسی اینکه یک مقدار یا متغیر در یک توالی پیدا می‌شود یا نه مورد استفاده قرار می‌گیرند. مانند رشته ، لیست ، مجموعه و دیکشنری

در یک دیکشنری ، فقط می توان وجود یا عدم وجود یک کلید را بررسی کرد نه یک مقدار. در جدول زیر عملگر های عضویت را میبینیم:

مثال	توضیح	عملگر
5 in a	در صورتی صحیح است که مقدار/ متغیر در توالی پیدا شود	In
5 not in a	در صورتی صحیح است که مقدار / متغیر در توالی پیدا نشود	not in

جدول 1-7

1-6 ماژول[17]

ماژول به فایلی گفته می شود که حاوی دستورات و تعاریف پایتون است. یک فایل حاوی کد های پایتون ، یک ماژول

[17] Module

نامیده می شود. ماژول ها برای شکستن برنامه‌ها به فایل‌های کوچک و قابل مدیریت استفاده می‌شود. ماژول ها قابلیت استفاده مجدد از کد را فراهم می‌کنند. کاربران می‌توانند به جای کپی کردن تعاریف در برنامه های گوناگون، توابع پر استفاده خود را تعریف و وارد کنند.

با دستورimport در پایتون ماژول را وارد می کنیم و برای دسترسی به اعضای ماژول از نقطه (.) استفاده می کنیم. همچنین ماژول ها می توانند متغیر ها را در خود نگه دارند؛ و با استفاده از کلید as می توانیم یک ماژول را نام گذاری کنیم. البته در پایتون ماژول های از پیش تعریف شده هم وجود دارد که می توانیم از آن ها استفاده کنیم.

تابع ()dir ، تابعی است که برای لیست کردن تمام محتویات ماژول اعم از توابع و متغیر ها ، از این تابع استفاده می کنیم. همچنین با استفاده از کلمه کلیدی from می توانید فقط بخشی از ماژول را وارد برنامه کنید. و هنگامی که از دستور from در پایتون استفاده می کنید، از نام ماژول برای دسترسی به اعضای دیگر استفاده نکنید.

1-7 رابط گرافیکی پایتون

زبان برنامه نویسی پایتون به گونه ای ساخته شده که می تواند برنامه های گرافیکی هم پیاده سازی کند. برای ساخت نرم افزار گرافیکی در پایتون از GUI Framework استفاده می کنند. در حال حاضر برای زبان برنامه نویسی پایتون رابط های گرافیکی زیادی وجود دارد که ما سه تا از بهترین رابط های گرافیکی پایتون را معرفی میکنیم:

1) رابط گرافیکی Tkinter:

Tkinter رابط گرافیکی استاندارد پایتون می باشد و یک gui framework متن باز می باشد. با استفاده از این کد می توانیم به راحتی برنامه های گرافیکی خود را بسازیم. برای استفاده از این رابط چهار مرحله داریم. مرحله اول اضافه کردن کتابخانه به پروژه، مرحله دو ساخت پنجره اصلی برنامه ، مرحله سه اضافه کردن ویجت های مورد نیاز ، مرحله چهار استفاده از تابع mainloop است.

2) رابط گرافیکی WxPython

WxPython رابط گرافیکی پرکاربرد در پایتون است و برای برنامه های گرافیکی کاربرد دارد و به صورت متن باز است. همچنین یک framework چند پلتفرمی است و توسط سیستم عامل های ویندوز و لینوکس پشتیبانی می شود. برای استفاده از این رابط پنج مرحله داریم. مرحله اول اضافه کردن کتابخانه به پروژه ، مرحله دوم ساختن

اپلیکیشن آبجکت ، مرحله سوم ساخت پنجره اصلی و نمایش دادن آن ، مرحله چهارم اضافه کردن ویجت های موردنیاز و مرحله پنجم استفاده از تابع mainloop است.

3) رابط گرافیکی PyQt

این رابط گرافیکی ترکیبی از زبان پایتون و QT می باشد. QT فریم ورکی است که توسط شرکت نوکیا برای ایجاد نرم افزارهای گرافیکی برای محصولاتش ارائه شده است. QT بیشتر در زبان c++ کاربرد دارد و با استفاده از برنامه PyQt تمام امکانات QT در پایتون استفاده می شود. Qt یک فریم ورک چند پلتفرمی است و قابلیت اجرا در هر سیستم عاملی را دارد.

حال که اصول پایتون را یاد گرفتیم باهم ساده ترین برنامه پایتون را می نویسیم. کد نویسی در پایتون 3 :

```
1 var1 = ('hello world')
2 print (var1)
```

خروجی که به ما می دهد :

```
1 Hello World
```

در این دستور از عبارت Var استفاده کردیم که برای تعریف متغیر استفاده می شود.

1-8 نصب پایتون نسخه 3.0

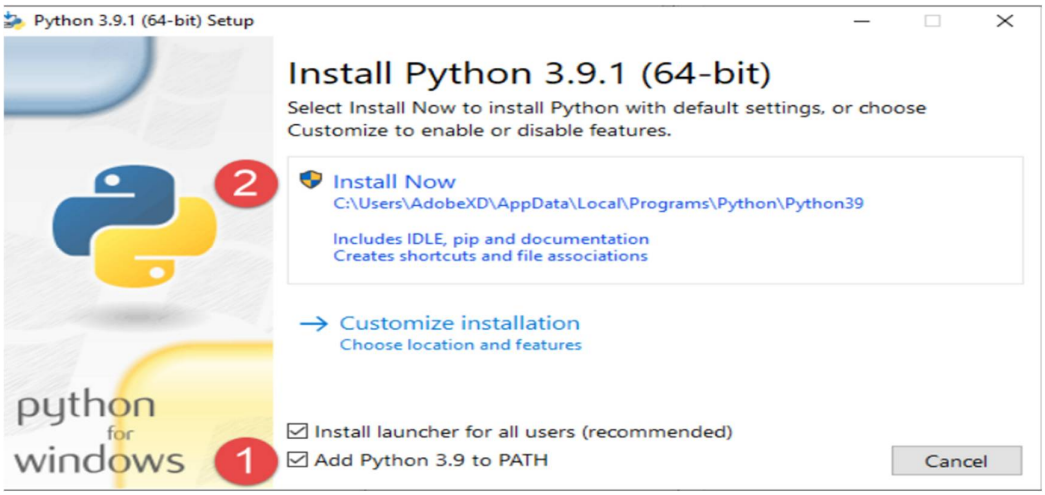
پایتون در دو نسخه 2 و 3 منتشر شده است هر کسی با توجه به سلیقه و نیاز های خودش هر کدام از این دو نسخه را که نیاز داشته باشد نصب می کند. در اینجا چون برای ساخت ماشین حساب از نسخه 3 استفاده کرده بودم طریقه نصب نسخه 3 پایتون را در اختیارتان قرار می دهم.

در ابتدا برای نصب پایتون به سایت رسمی پایتون می رویم : https://www.python.org/downloads/

وقتی وارد سایت می شویم اصولا خودش نسخه ای که روی کامپیوترمان نصب می شود را نشان می دهد.

شکل 1-3

نسخه مورد نظر را دانلود می کنیم که بر روی کامپیوتر نصب کنیم.

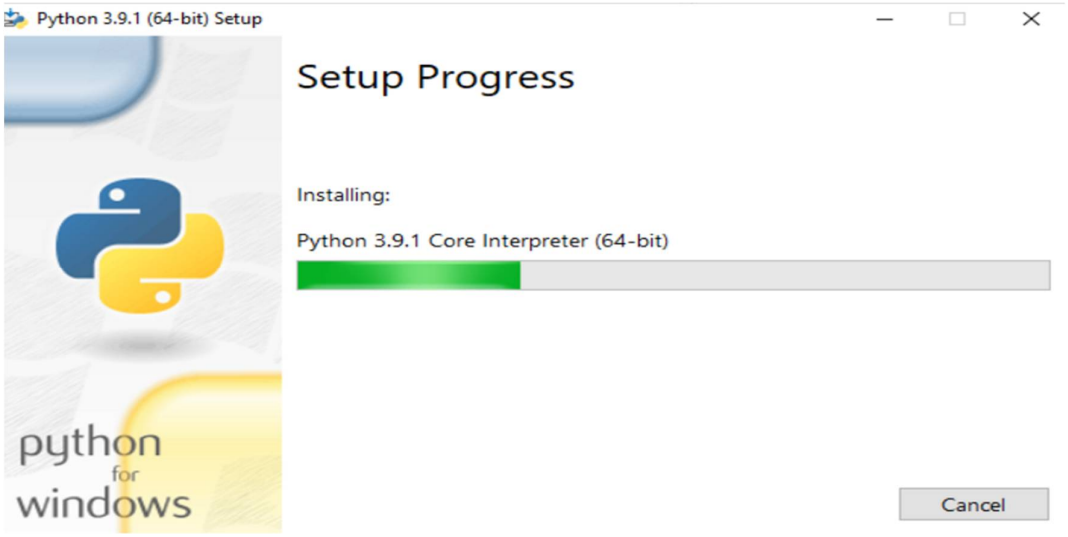


شکل 1-4

بعد از دانلود به محل ذخیره فایل رفته و آن را باز می کنیم و صفحه ای که در بالا مشاهده می کنید را نشان می دهد.

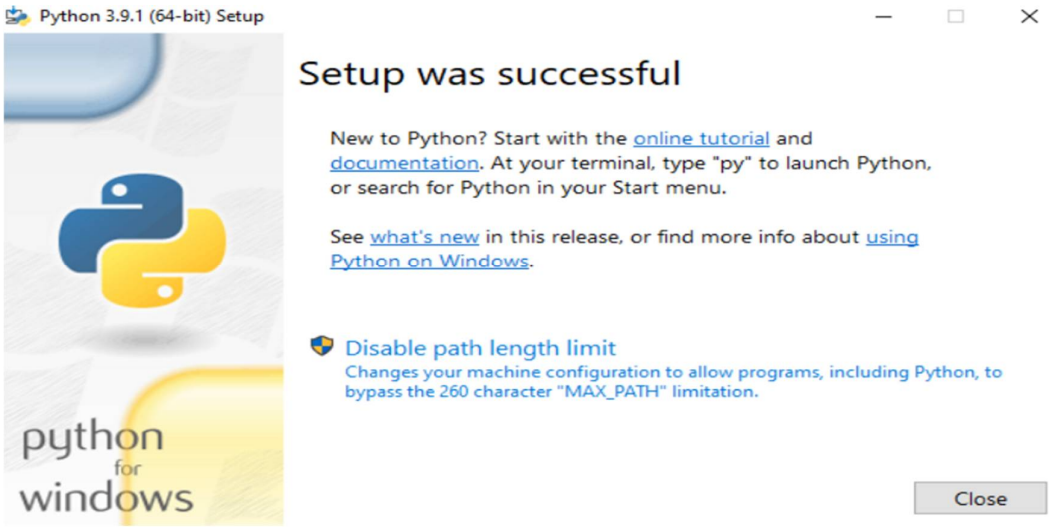
ابتدا تیک add python 3.9 to path را بزنید و سپس روی گزینه install now مطابق تصویر بالا کلیک کنید.

بعد از کلیک روی گزینه مورد نظر تصویر زیر را می بینید که نشان می دهد برنامه دارد نصب می شود.



شکل 1-5

در نهایت پس از نصب کامل برنامه پیام زیر را دریافت می کنید که نشان می دهد برنامه شما با موفقیت نصب شده است.



شکل 1-6

به همین راحتی برنامه پایتون نصب شد و می توانیم کد نویسی را شروع کنیم.

برای کد نویسی روی برنامه پایتون هم از آیکون IDEL(Python 3.9 32/64 -bit) استفاده می کنیم.

1-9 جمع بندی فصل

در این فصل فهمیدیم پایتون چیست و تا حدودی با تاریخچه آن آشنا شدیم . همچنین تابع ها ، رابط ها و حکم های دستوری مانند if و from و غیره پایتون را توضیح می دادیم. در مورد عملگر های مختلف پایتون اعم از حسابی و منطقی و ... صحبت کردیم و کار کرد عملگر ها را یاد گرفتیم . با سه رابط گرافیکی مهم Tkinter و WxPython و PyQt و نحوه کار کرد آن ها آشنا شدیم بعد از آن نحوه نصب برنامه رسمی پایتون را یاد گرفتیم و با هم برنامه را نصب کردیم.

فصل دوم

مقدمه

در فصل قبلی تا حدودی با مقدمه زبان پایتون و تا حدودی باهاش کار کردن آشنا شدیم تا بتوانیم یک برنامه ای مانند ماشین حساب تولید کنیم. در این فصل می خواهیم با انواع ماشین حساب و کاربرد آن آشنا بشویم. و ضرورت اینکه چرا باید تولید برنامه ماشین حساب را یاد بگیریم بفهمیم.

2-1 تاریخچه ماشین حساب

در زمان های قدیم بسیاری از افراد بخصوص ریاضی دان ها به علت نبود امکانات کافی برای محاسبه دقیق به مشکل بر می خوردند. امروزه با پیدایش ماشین حساب این مشکل حل شده است.

ماشین حساب چیست؟ دستگاهی است که بر روی اعداد عملیاتی انجام می دهد و پاسخ محاسبات را در اختیار ما قرار می دهد. ما دو نوع ماشین حساب ساده که عملیات اصلی را انجام می دهد و یک نوع ماشین حساب پیچیده تر که مهندسی هم می گویند که می تواند توابع مثلثاتی و لگاریتم و ... را محاسبه کند.

ماشین حساب در ابتدا در دهه 1960 اختراع شد و اولین شخصی که ماشین حساب واقعی را تولید کرد بلیز پاسکال در سال 1642 بود و همچنین در سال 1990 انسان ها توانستند به کمک تلفن های هوشمند از ماشین حساب بهره مند شوند. مدت ها قبل از ورود ماشین حساب های دیجیتالی و تلفن های هوشمند، چندین ماشین محاسباتی ساخته شده بود. به عنوان مثال چرتکه، مدتها قبل از تصویب سیستم اعداد مکتوب هندو-عربی، در خاور نزدیک باستان، اروپا، روسیه و چین مورد استفاده قرار گرفت.

2-2 انواع ماشین حساب

ابتدایی ترین نوع ماشین حساب چرتکه بود که طرح اولیه آن ها با چوب یا سنگ ساخته می شد و چهار عمل اصلی را انجام می داد. بعد ها ماشین حساب های جیبی به وجود آمد که به دلیل کوچک بودنش در هر محیطی قابل حمل بود. این ماشین حساب ها یک نمایشگر کوچک دارند که نتیجه محاسبه را می توان دید. قبل از اختراع کامپیوتر ماشین حساب های چاپی بسیار مورد توجه قرار گرفتند و برای محاسبات و ارقام بالا مورد استفاده قرار می گرفت. این نوع دستگاه محاسباتی به یک چاپگر داخلی مجهز بوده و محاسبات انجام شده را بر روی کاغذ چاپ می کند. امروزه حسابداران و مدیران مالی از این مدل استفاده می کنند.

ماشین حساب های مهندسی نوع دیگری از ماشین حساب هستند که یکی از پیشرفته ترین دستگاه های محاسباتی محسوب میشود که از یک نمایشگر دو خطی برای ارائه محاسبات و خروجی طراحی مجهز شده است. با این دستگاه علاوه بر چهار عمل اصلی محاسبات دیگر نظیر محاسبه اینورس اعداد، محاسبه فاکتوریل اعداد، تبدیل دکارتی به

قطبی و بالعکس، محاسبه اعداد توان دار، جذر، لگاریتم و نسبت های مثلثاتی انجام دهید. این مدل ماشین حساب را برای برنامه نویسی نیز پیشنهاد می دهند.

مدل بعدی ماشین حساب نوع گرافیکی است که می توان نمودار ها و توابع را طراحی کرد. بیشتر برای دانشجو و دانش آموزان مناسب است.

بنابراین هر شخصی بنا بر نیازش می تواند ماشین حساب مورد نیازش را تهیه کند. البته به لطف زبان های برنامه نویسی و علم کامپیوتر اگر شخصی به زبان برنامه نویسی مسلط باشد می تواند برای خودش یک ماشین حساب متناسب با نیازش طراحی کند.

2-3 ماشین حساب سیستم

امروزه وقتی یک گوشی هوشمند می خریم یا ویندوز بر روی لپ تاب خود نصب می کنیم متوجه می شویم به طور پیش فرض برنامه ماشین حساب دارند. هر سیستم عاملی با یک زبان برنامه نویسی مخصوص به خود ماشین حساب تولید می کند و در اختیار مشتری و طرفداران خود قرار می دهد تا بتوانند بدون دردسر اینکه دنبال برنامه باشند از برنامه پیش فرض تولید شده توسط شرکت استفاده کنند.

ما می توانیم با انواع زبان های برنامه نویسی مانند C++ ، C# ، اندروید ، پایتون و دیگر زبان های برنامه نویسی یک برنامه ماشین حساب بسازیم.

2-4 مقایسه زبان های برنامه نویسی برای ساخت ماشین حساب

برای مثال ما اگر بخواهیم یک ماشین حساب ساده به زبان C++ بنویسیم از دستور العمل if استفاده می کنیم و برای نشان دادن متن در داخل برنامه از دستور Char استفاده شده است و همچنین برای گرفتن دو عدد مورد نظر از Int استفاده می کنیم و با این دستور ماشین حساب با چهار عمل اصلی را می سازیم.
همچنین اگر بخواهیم با زبان سی شارپ برنامه ماشین حساب بسازیم پس از ساخت فرم از دو کد دستوری if یا switchاستفاده کنیم همچنین برای توابع از تابع result استفاده می کنیم.

برای زبان PHP نیز همانند برنامه سی شارپ از دستور if و switch استفاده می کنیم و از متغیر $result برای چاپ متغیر استفاده می کنیم.

اما در برنامه نویسی پایتون با استفاده از کتابخانه Tkinter و دستور def می توانیم به راحتی یک ماشین حساب

ساده اما گرافیکی تولید کنیم.

زبان های برنامه نویسی بالا همگی شی گرا هستند و برنامه‌نویسی شی گرا معمولا با عنوان OOP شناخته می شود که این سه حرف مخفف کلمات Object-Oriented Programming می باشد. یک شیوه برنامه‌نویسی است که ساختار یا بلوک اصلی اجزای آن، شی‌ها می‌باشند.

حال به طور خلاصه ویژگی این چهار برنامه را بررسی می کنیم تا دلیل اینکه چرا از زبان پایتون برای ساخت ماشین حساب استفاده کردیم را متوجه شویم.

اولین زبان قابل بررسی سی شارپ است، زبانی شی گرا و سطح بالا و قدرتمند از خانواده زبان های چارچوب دات نت شرکت مایکروسافت است. یک زبان برنامه‌نویسی چند الگویی و منظم شده مدل‌های تابعی، امری، عمومی، شی گرا و جز گرا و در بستر دات نت می باشد. سی شارپ یکی از 44 زبان برنامه‌نویسی است که توسط زمان اجرای زبان مشترک از چارچوب دات نت پشتیبانی می‌شود و در همه جا به وسیله مایکروسافت ویژوال استودیو شناخته می‌شود.

زبان بعدی سی پلاس پلاس (C++) است ، که از نظر دستوری شی گرا مشابه سی شارپ است. سی پلاس پلاس یک زبان برنامه نویسی همه منظوره، سطح میانی و شی گرا است. این زبان از نظر سطحی یک زبان سطح میانی محسوب می شود ولی دارای قابلیت زبان های سطح بالا و پایین به صورت هم زمان است.

زبان سوم php است. یک زبان برنامه نویسی شی گرا برای طراحی وب می باشد. PHP یک زبان برنامه نویسی اسکریپتی اپن سورس Open Source و شباهت زیادی به زبان برنامه نویسی سی دارد.

چهارمین زبان پایتون است زبانی که برنامه خود را باهاش نوشتیم. پایتون یک زبان برنامه نویسی چند منظوره و قدرتمند، پایتون در دنیا به شدت همه گیر شده است. زبان برنامه نویسی پایتون یک زبان شی گرا است و از ویژگی های پیشرفته شی گرایی مثل : وراثت، چند شکلی، سربار گزاری عملگر[18] و . . . پشتیبانی می کند. همچنین چون زبان برنامه نویسی پایتون با زبان پورتابل سی نوشته شده است میتواند به صورت مجازی بر روی هر سیستم و پلتفرمی کامپایل و اجرا شود.

[18] Operator overloading

2-5 چرا زبان پایتون؟

چهار مدل برنامه نویسی را به طور خلاصه توضیح دادیم و تا حدودی آشنا شدیم ، حال میخواهیم مزایای زبان برنامه نویسی پایتون را با هم بررسی کنیم تا دلیل اینکه از این برنامه برای ساخت ماشین حساب استفاده کردیم را بفهمیم.

اگر بخواهیم زبان پایتون را با زبان های دیگر مقایسه کنیم از نظر نوشتاری پایتون راحت تر از سی پلاس پلاس است. در زبان پایتون، علاوه بر برنامه‌نویسی شی گرا، از برنامه‌نویسی تابعی[19] و ساختاری[20] نیز پشتیبانی می‌شود.

تعریف و استفاده از کلاس‌ها[21] و اشیا[22] در زبان پایتون، به دلیل وجود ویژگی‌های شی گرایی، بسیار ساده است.

قابلیت خوانایی و اشکال زدایی کدها در زبان پایتون بسیار بالاست. به همین خاطر، یادگیری آن به برنامه‌نویسان مبتدی توصیه می‌شود.

همچنین کتابخانه استاندارد زبان پایتون بسیار غنی است و با تمامی پلتفرم‌های موجود نظیر ویندوز، لینوکس و مک مطابقت دارد.

زبان پایتون، انتخاب ایده آلی برای توسعه های برنامه‌های کاربردی شبکه[23] است؛ به ویژه برنامه‌های که باید از پروتکل‌های مختلف و متعددی استفاده کنند. همچنین به دلیل چرخه توسعه بسیار سریع برنامه‌های کاربردی در زبان پایتون، این زبان برای نمونه سازی[24] و تست کردن [25] برنامه‌های کاربردی بسیار مناسب است. و در نهایت زبان پایتون منبع باز است و به شکل بسیار مناسبی، توسط تیم‌های توسعه و جامعه برنامه‌نویسی پشتیبانی می‌شود.

یکی دیگر از ویژگی های پایتون این است که شما یک برنامه را مثلا در مدت دو الی سه ماه تمام می کنید ولی با زبانی مانند سی پلاس پلاس حدود دو الی سه سال شاید هم بیشتر طول بکشد.

در مورد برنامه پایتون و php هم می توان گفت هر دو برنامه های سطح بالا ، چند منظوره و تفسیری هستند و پایتون تحت لایسنس PSFL و PHP تحت مجوز PHP Licence منتشر شده است.

از نظر کد نویسی بین این دو برنامه نیز پایتون راحت تر و بهتر است چون به شیوه ای است که انگار با کامپیوتر

---

[19] Functional
[20] Structural
[21] Classes
[22] Objects
[23] Network Applications
[24] Prototyping
[25] Testing

صحبت می کنید.

در نهایت مطابق جدول زیر با رتبه بندی نشان می دهیم کدام برنامه از چه نظر بهتر است:

برنامه	محبوبیت	یادگیری		امنیت/هک	سرعت
python	1	1		1	4
php	3	2		4	3
C#	4	4		3	2
C++	2	3		2	1

جدول 2-1

در این جدول عدد 1 نشان بیشترین امتیاز در آن موقعیت و عدد 4 دارای کمترین امتیاز است.

طبق جدول بالا متوجه می شویم زبان پایتون از نظر محبوبیت و مدت زمان یادگیری و همچنین در حوزه امنیت نسبت به سه برنامه دیگر عملکرد بهتری دارد. البته نمیتوان گفت پایتون بهترین زبان برنامه نویسی است اما در کل می توان گفت جزء برنامه هایی است که برای شروع و برای افراد مبتدی بسیار مناسب است.

6-2 جمع بندی فصل

در این فصل با هم دیگه به طور خلاصه چهار زبان مختلف برنامه نویسی اعم از سی شارپ ، پایتون ، php و پایتون را بررسی کردیم و فهمیدیم پایتون از نظر سرعت و امنیت و مدت زمان یادگیری سریع تر از چهار برنامه دیگر است همچنین اگر کسی اول راه باشد و بخواهد برنامه نویسی یاد بگیرد پایتون گزینه مناسبی است. همینطور با تاریخچه ماشین حساب آشنا شدیم و فهمیدیم اولین ماشین حسابی که در گوشی ها پیدا شد در سال 1990 میلادی بوده است. همچنین با انواع ماشین حساب از اولین نوع آن که چرتکه بود تا ماشین حساب های مهندسی به طور مختصر آشنا شدیم

فصل سوم

مقدمه

در فصل قبلی با انواع ماشین حساب و ساخت آن با برنامه های مختلف آشنا شدیم و با هم بررسی کوتاهی کردیم که چرا از زبان پایتون استفاده کردیم. در این فصل باهم اولین ماشین حساب گرافیکی و ساده به زبان پایتون را پیاده سازی می کنیم.

1-3 کتابخانه Tkinter

اول از همه با کتابخانه Tkinter [26] آشنا می شویم. کتابخانه گرافیکی که باهاش میتوانید یک برنامه گرافیکی بنویسید. کتابخانه Tkinter برای طراحی رابط کاربری اپلیکیشن ها است و به طور پیش فرض بر روی پایتون نصب می باشد.

اول از همه باید کتابخانه tkinter را به روش زیر صدا بزنیم:

```
from tkinter import*
```

بعد از اینکه کتابخانه را به روش بالا صدا کردیم شروع به ساخت پنجره و عناصر داخل پنجره می کنیم.

الان برای مثال کد های زیر را وارد می کنیم تا با کتابخانه Tkinter و عملکرد آن آشنا شویم.

```
from tkinter import*

def fCalc (source, side):
      appObj = Frame (source, borderwidth=5, bd=5, bg = "light blue")
       appObj. Pack (side=side, expand=YES, fill=BOTH)
   return appObj

def button (source, side, text, command=None):
       appObj = Button (source, text=text, command=command)
       appObj. Pack (side=side, expand=YES, fill=BOTH)
   return appObj

class app (Frame):
   def __init__(self):
       Frame. __init__(self)
       self. option add ("*Font", 'arial 20 bold')
       self. pack (expand=YES, fill=BOTH)
       self. master. Title("Calculator")


if __name__ == '__main:'__
   app (). mainloop ()
```

بعد از اینکه کد بالا را در برنامه پایتون نصب کنید احتمالا با تصویر زیر روبرو می شوید.

[26] کتابخانه Tkinter در واقع از زبان برنامه نویسی TK یا TCL ساخته شده و مخفف Tk Interface می باشد.

شکل 3-1

مثال بالا شامل بخش ابتدایی و اولیه کد ما بود که برای نشان دادن عملکرد کتابخانه Tkinter انجام دادیم حال در مبحث بعدی با تابع و دستور هایی که در برنامه استفاده کردیم آشنا می شویم

3-2 توابع و دستور

تابع در پایتون به گروهی از عبارت های مرتبط می گویند که یک کار مشخص انجام می دهند. کار تابع در برنامه چیست؟ تابع ها کمک می کنند تا برنامه ای که داریم می سازیم به بخش های کوچکتر و دانه بندی شده ای شکسته شوند. همچنین تابع مانع از تکرار برنامه نویسی برای کار می شود و باعث می شود کد مجدد قابل استفاده باشد.

دستورات مرکب چیست؟ در زبان پایتون برای تعریف یک فانکشن[27] از دستورات مرکب استفاده می کنیم و همان طور که می دانید هر دستور مرکب با یک کلمه کلیدی آغاز می شود و کلمه کلیدی ما نیز def نام دارد.

3-2-1 def

به این مثال توجه کنید:

```
def fCalc (source, side):
    appObj = Frame (source, borderwidth=4, bd=4, bg = "light blue")
    appObj. Pack (side=side, expand=YES, fill=BOTH)
    return appObj
```

در عبارت بالا کلمه def را دیدیم که مخفف define و به معنای تعریف کردن است. همانطور که از معنی آن پیداست برای تعریف کردن تابع استفاده می شود. در این مثال می بینیم که def برای تعریف فانکشن fcalc استفاده میشود

3-2-2 frame

فریم ها حکم استخوان بندی را برای برنامه ای که می سازیم دارند. به این صورت که برای سازماندهی ویجت های دیگر برنامه استفاده می شود. در واقع محلی است برای نگهداری و سازماندهی ابزارک ها. برای مثال تصویر3-1 که در صفحه 21 مشاهده کردید به دلیل وجود ابزار frame بود. در واقع بدون frameپنجره ای که در تصویر دیدید باز نمی شود.

[27] Function به معنای تابع

البته باید بدانیم که frame خالی قابل رویت نیست و به درد ما نمی خورد. در این شرایط ما می توانیم با استفاده از ویجت master دیگر ویجت ها را در پیکر frame قرار دهیم.

class 3-2-3

همانطور که برای تعریف توابع از def استفاده می کنیم یک کلاس را نیز با کلید واژه class می سازیم. کلاس یک فضای نام محلی[28] جدید می سازد که درآن، تمام خصوصیت ها تعریف می شود.خصوصیت های

ما می تواند داده یا تابع باشد. همچنین، خصوصیت های خاصی در آن وجود دارند که با دو زیر خط (_) شروع می‌شود. مانند __ init __ که مخفف initialize است و برای مقدار دهی اولیه و کار های اولیه بعد از شی بکار می رود.

در class از self نیز استفاده می کنیم که به این صورت است که اگر بخواهیم از کلاس ها و متد ها تعریف کنیم برای دسترسی به آن ها از self استفاده می کنیم.

3-3 ابزار های پایتون

در زیر لیستی را ارائه می دهیم که توضیح مختصر و مفیدی از ابزار های مورد استفاده از کتابخانه Tkinter است که از آن ها پشتیبانی می کند:

Label: برای نمایش متن یا تصویر بر روی صفحه استفاده می‌شود.

Button: برای افزودن دکمه ها به برنامه ی شما استفاده می‌شود.

Entry: برای وارد کردن متن تک خطی کاربر استفاده می‌شود.

و یک متد pack() داریم که ابزارک ها را در سطر ها یا ستون ها دسته بندی می کند.

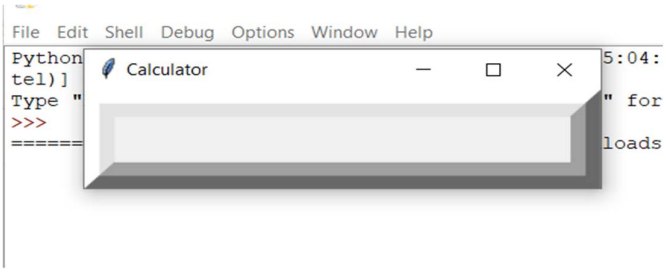
الان با توجه به توضیحات داده شده به کدی که در صفحه 21 نوشتیم عبارت زیر را اضافه می کنیم و

نتیجه ای که بدست می آید را می بینیم.

```
display = StringVar()
Entry (self, relief= RAISED ,
        textvariable=display, state=DISABLED, justify='right', bd=20,
bg="silver"). pack (side=TOP, expand=YES,
                fill=BOTH(
```

[28] Local Namespace

این کد را به ادامه برنامه اضافه کردیم نتیجه بدست آمده شد تصویر پایین:



شکل3-2

برای ایجاد نمایشگر ماشین حساب باید یک شی از کلاس Entry بسازیم. در اینجا شی ما display است که در اولین پارامتر خود Self را می گیرد و در ادامه ویژگی هایی مانند اندازه فونت، جایگاه، رنگ و غیره را به سبک دلخواه شما در می آورد.

lambda 4-3

لامبدا یک روش ساده برای تعریف تابع در پایتون است. قبل از ایجاد دکمه در ماشین حساب خود اول از همه با کلمه Lambdaآشنا می شویم. کسانی که از پایتون استفاده می کنند از تابع def استفاده می کنند. اما سوال این است که چرا برخی مواقع از تابع lambda استفاده می شود. چون که تابع های لامبدا ناشناس هستند. بدین معنی که این‌ها توابعی هستند که لازم نیست نامی برایشان تعیین کنید. این روش برای تعریف تابع های کوچک یک‌بار مصرف در مواردی که تابع اصلی بسیار بزرگ و حجیم است، استفاده می‌شود. لامبداها یک شی تابع بازمی گردانند که می‌توانند به یک متغیر انتساب یابد. لامبداها می‌توانند هر تعداد آرگومان که لازم باشد داشته باشند؛ اما تنها یک عبارت دارند. نمی‌توان توابع دیگر را درون یک لامبدا فراخوانی کرد. بیشترین استفاده از تابع های لامبدا در کدهایی است که نیازمند توابع یک خطی ساده‌ای هستند و نوشتن یک تابع معمولی کامل، زیاده کاری محسوب می شود.

همچنین هنگام استفاده از این تابع نیاز به کلید واژه return نداریم چون که لامبدا به طور خودکار این کار را برای شما انجام می‌دهد. الان که تابع lambda را فهمیدید کد زیر را به کد اولیه اضافه می کنیم تا دکمه C ماشین حساب را به برنامه اضافه کنیم.

```
clrChar = "C"
    button (self, TOP, clrChar, lambda appObj=display, i=clrChar: appObj.set (''))
```

بعد از اضافه کردن این کد تصویر زیر ظاهر می شود:

شکل 3-3

3-5 button

برای اینکه برای برنامه خود کلید بسازیم باید از تابع button استفاده کنیم. دکمه ها می توانند متن یا عکس را به نمایش بگذارند. با این روش شما می توانید یک تابع را به دکمه متصل کنید، به صورتی که هنگام کلیک کردن دکمه این تابع به صورت خودکار فراخوانی می شود. این تابع دو آرگومان ورودی را شامل می شود که اولی پنجره ای است که کلید می خواهد در آن تشکیل شود و آرگومان دوم شامل تنظیماتی خواهد بود که برای این کلید در نظر می گیرید که شامل نام تغییرات ظاهری و عملکرد هایی است که انتظار می رود پس از تغییر وضعیت کلید صورت بگیرد.

چند تا از آرگومان هایی که می توانید برای صفت های هر کدام از کلید ها برای آن تخصیص دهید:

bd: (borderwidth) پهنای خط حاشیه دکمه ها را مشخص می کند و اصولا مقدار پیش فرض آن 2 است.

bg: (background) رنگ پس زمینه را در حالت عادی مشخص می کند.

Font: فونت متنی که برای دکمه استفاده کردیم را مشخص می کند.

height: ارتفاع دکمه، به صورت خطوط برای دکمه متنی و به صورت پیکسل برای دکمه تصویری را مشخص می کند.

width: عرض دکمه، به صورت تعداد حروف برای نمایش متنی یا به صورت پیکسل برای نمایش تصویر را مشخص

میکند

relif: نوع لبه کناری را مشخص می کند. این خاصیت دارای مقادیر GROOVE،RAISED، SUNKENو RIDGE است. این عبارت های بالا همون حالتی است که انتخاب کردیم صفحه نمایشگر ماشین حساب ما چگونه باشد.

State: برای ایجاد وضعیت کلید یا ویجت مورد نظر که شامل 3 پارامتر active و normal و disabled است که اصولا به صورت پیش فرض به حالت normal است.

text: نوشته ای که می خواهد بر روی آن المان قرار بگیرد

textvariable: برای نگه داری وضعیت کلید و دیگر المان ها استفاده می شود که می تواند ماهیت رشته و یا عدد را داشته باشد که می توان با استفاده از آن به وضعیت فعلی کلید در جای جای کد پی برد.

justify : بیانگر این است که چطور می خواهید نوشته های روی کلید قرار بگیرند. در واقع نمایش متن چند خطی هستند. Leftبرای تراز کردن هر خط به سمت چپ، center تراز خطوط به وسط است و right برای تراز هر خط به سمت راست است.

6-3 حلقه for

حلقه for همانند حلقه while برای تکرار اجرای یک قطعه کد مورد استفاده قرار می گیرد، با این تفاوت که در حلقه for دقیقا می دانیم که تسک موردنظر را چند مرتبه تکرار کنیم. نمونه حلقه for طبق کد برنامه ما عبارت است از:

```
for fEquals in btnNum:
        button (FunctionNum, LEFT, fEquals,
            lambda appObj=display, i=fEquals: appObj.set(appObj.get() + i)(
        EqualsButton = fCalc (self, TOP)
```

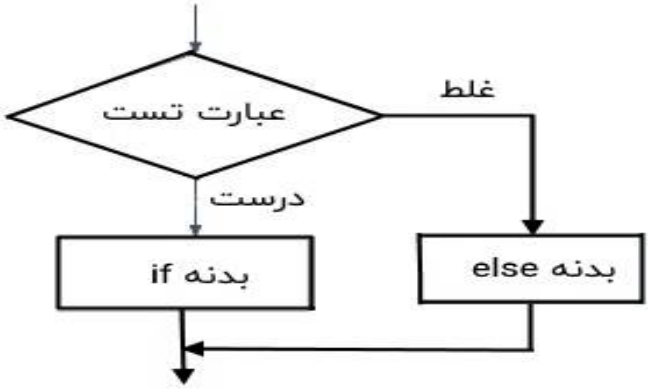
بر اساس این الگو هر حلقه به تعداد عضو های دنبالهbtnNum تکرار می شود به این صورت که هر یک از اعضای دنباله به ترتیب و با شروع از عضو اول به متغیر fEquals منتسب شده دستورات داخل بدنه حلقه به ازای هر یک از انتساب ها اجرا می‌شود و این پروسه تا زمان اتمام اعضای دنباله btnNum ادامه پیدا کند.

7-3 دستورif...else

دستور if...else در کد زیر به این صورت است:

```
if fEquals:"=" ==
        btnEquals = button (EqualsButton, LEFT, fEquals)
        btnEquals.bind('<ButtonRelease-1>')
                lambda e, s=self, appObj=display: s. result(appObj), "+")
      else:
        btnEquals = button (EqualsButton, LEFT, fEquals,
              lambda appObj=display, s=" %s "%fEquals: appObj.set (appObj.get ()
+s))
```

این دستور که برای دکمه مساوی (=) است و کد های آن مربوط به نحوه جمع کردن عدد ها است به شما میگوید اگر شرط اولی اتفاق افتاد انجام بده اگر نه از شرط دومی استفاده کن. یعنی اگر قسمت if اجرا شود دیگر نیاز به قسمت else نداریم.



شکل 3-4

نمونه تصویر نشان دهنده دستور if...else که به راحتی متوجه کار این دستور می شوید.

3-8 try...except

ما با استفاده از دستور try...except می توانیم خطا ها را اداره کنیم. برای مثال کدی را که احتمال می دهیم خطا ایجاد کند در داخل بلوک try قرار می دهیم. بلوک except هم شامل کدهایی است که وقتی اجرا می شود که برنامه با خطا مواجه شود. اگر بخواهیم ساده تر تعریف کنیم به این صورت است که بلوک try سعی می کند که دستورات را اجرا کند و اگر در بین دستوراتی که اجرا می شود خطایی وجود داشته باشد برنامه دستورات مربوط به

بخش except را اجرا می کند . در برنامه بخش استفاده از دستور try...except به صورت زیر است.

```
def result (self, display):
    try:
      display.set (eval (display.get ()))
    except:
      display.set("UNDEFINED")
```

این بخش از کد برای این است که ما وقتی خواستیم عملیات ریاضی را در ماشین حساب انجام دهیم اگر عبارت هایی که وارد کردیم درست بود نهایی را به ما نشان دهد ولی اگر عبارت ما غلط بود یا مشکل داشت UNDEFINED به معنای نامفهوم را به ما نشان می دهد.

9-3 جمع بندی فصل

در این به طور خلاصه با کد های ماشین حسابی که ساختیم آشنا شدیم و سپس با دو تابع def و lambda و کاربرد آن ها آشنا شدیم. کمی جلوتر رفتیم با ابزار های مورد استفاده آشنا شدیم مثل lable و entry . بعد هم که توضیح دادیم button چیست و ابزار هایی که در button استفاده می شود مانند font,text,relif و غیره را توضیح دادیم و کار کردن هر کدام را فهمیدیم. در هایت به سراغ حلقه for رفتیم و فهمیدیم کاربرد آن مانند حلقه while است. بعد از آن با دستور های if...else و try...except آشنا شدیم و فهمیدیم if...else به ما می گوید هر موقع شرط اولی انجام نشد از شرط دومی استفاده کن و دستور try...except برای این کاربرد دارد که بتوانیم خطا ها را در برنامه اداره کنیم و اگر خطایی داشتیم با دستور except نشان دهیم.

فصل چهارم

مقدمه

در این فصل بعد از نصب برنامه رسمی زبان پایتون بر روی سیستم و کامپیوتر خود سعی داریم با توضیح دادن در مورد کد ها نحوه ساخت ماشین حساب به زبان پایتون را یاد بگیریم و ماشین حساب خودمان را تولید کنیم.

4-1 مشخصات نرم افزاری پروژه

پروژه ما در محیط نرم افزاری پایتون 3.9.1 است و با استفاده از کتابخانه طراحی واسط کاربری tkinter تولید شده است و از تابع def مخفف define به معنی تعریف کردن است استفاده کردیم. همچنین از توابع Lambdaو eval نیز استفاده کردیم برای خود برنامه از چهار عملگر اصلی جمع ، منفی ، ضرب ، تقسیم و یک عملگر اعشار و همینطور مساوی و دکمه c برای پاک کردن کلی اعداد استفاده کردیم.

4-2 دلایل ساخت ماشین حساب

به این دلیل ماشین حساب به شیوه ای که در بخش بعدی توضیح دادم را ساختم چون نسبت به نوع های دیگر ماشین حساب به زبان پایتون ساده تر بود و بدون استفاده از عکس و فتوشاپ تنها با استفاده کتابخانه tkinter میتوانیم یک برنامه گرافیکی کاربردی و ساده بسازیم . همچنین حتی اگر در زبان پایتون مبتدی باشید می توانید این برنامه را تولید کنید.

4-3 پیاده سازی ماشین حساب به زبان پایتون

در این بخش سعی می کنیم با بررسی کد های برنامه با هم دیگر پروژه ماشین حساب خود را تولید کنیم.

در ابتدای کار از کتابخانه واسطه کاربری با اسم tkinter استفاده میکنیم و بعد از آن از دستور import استفاده می کنیم که بتوانیم کد ها را به پروژه خود اضافه کنیم. بعد از import نیز از (*) استفاده میکنیم و کاربرد (*) به این صورت است که باعث می شود همه اجزا توابع را وارد کنیم. و به صورت زیر نوشته می شود:

```
from tkinter import*
```

بعد از وارد کردن این کد دیگر می توانیم توابع و دستورات خود را وارد کنیم. پس با وارد کردن تابع def یا همان define و ساخت فانکشن fCalc ، f را به صورت مخفف برای فانکشن گذاشتیم یعنی می توانستیم به صورت function Calc هم بنویسیم فرقی ندارد به هر صورت بنویسید کار شما انجام می شود. سپس با

استفاده از ابزارک frame محل نمایش تصویر اعم از چارچوب و قاب نمایشگر و رنگ نمایشگر تنظیم می شود و سپس برای تکمیل کردن کار از متد pack() استفاده کردم تا بتوانم المان ها و تنظیمات را سرجای خودش قرار بدهم. در متد pack() نیز از سه گزینه side ، expand و fill استفاده کردم که از side برای به این دلیل استفاده کردم که تعیین کنم ویجت در کدام طرف قرار بگیرد. بعد از آن از expand برای گسترش ویجت با عنوان کلمه yes استفاده کردم و در نهایت از fill استفاده کردم که تعیین می کند ویجت فضای اضافی اختصاصی خودش را پر می کند و یا ابعاد خودش را حفظ می کند و fill را برابر both قرار دادم تا هم به صورت افقی و هم عمودی باشد. بعد از این از دستور return استفاده کردم که یک خروجی است و از function می آید ویک مقدار را به فانکشن بر می گرداند و همچنین درستی کارکرد فانکشن را به ما نشان می دهد. در نهایت شکل این بخش از کدی که نوشتیم به صورت زیر می شود:

```
def fCalc (source, side):
    appObj = Frame (source, borderwidth=5, bd=5, bg="light blue")
    appObj. Pack (side=side, expand=YES, fill=BOTH)
    return appObj
```

سپس برای تکمیل تر کردن کارم دوباره از تابع def استفاده کردم اینبار در فانکشن button را قرار دادم تا تنظیمات و مشخصات دکمه ها را تعیین کنیم سپس دوباره به وسیله pack() المان و تنظیمات را در جای خود قرار دادم به صورت زیر:

```
def button (source, side, text, command=None):
    appObj = Button (source, text=text, command=command)
    appObj. Pack (side=side, expand=YES, fill=BOTH)
    return appObj
```

در ادامه برای پیشرفت برنامه از class یا همان کلاس استفاده می کنیم و نام app را در نظر گرفتم. سپس برای تعریف رفتار برنامه از تابع def و __init__ استفاده کردیم و همچنین با استفاده از self ویجت root=tk را بهش توضیح می دهیم و طول و عرض ماشین حساب را تعیین می کنیم. سپس با فراخوانی آن فونت برنامه و

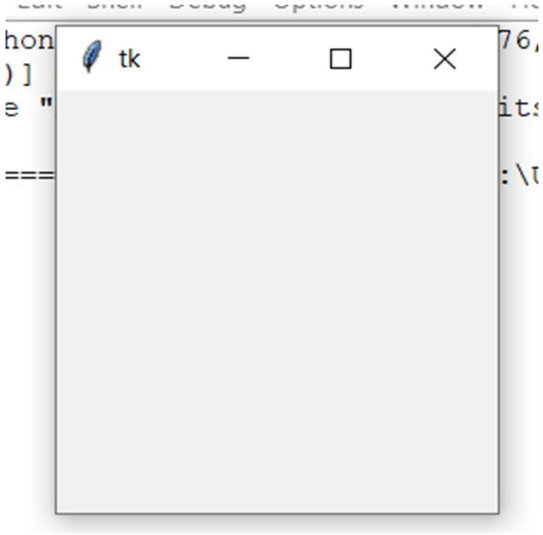
عنوان برنامه و طول و عرض کلی برنامه را نوشتم به صورت زیر:

```
class app (Frame):
   def __init__ (self, root=Tk (), width=420, height=540):
      Frame. __init__(self)
      self. Option_add ("*Font", 'arial 20 bold')
      self. pack (expand=YES, fill=BOTH)
      self. master. Title("Calculator")
      screen_width = root. winfo_screenwidth()
      screen_height = root. winfo_screenheight()
```

خب تا اینجا برنامه نوشتم و میخواستم تست کنم که متوجه شدم برای اینکه بتوانم خروجی بگیرم باید از متد Mainloop () استفاده کنم پس اومدم با استفاده از کلاس app و متد mainloop را صدا زدم تا برنامه را اجرا کنم.

```
if __name__ == '__main__':
   app (). mainloop()
```

سپس با استفاده از f5 برنامه را اجرا کردم و تصویر زیر نمایان شد:

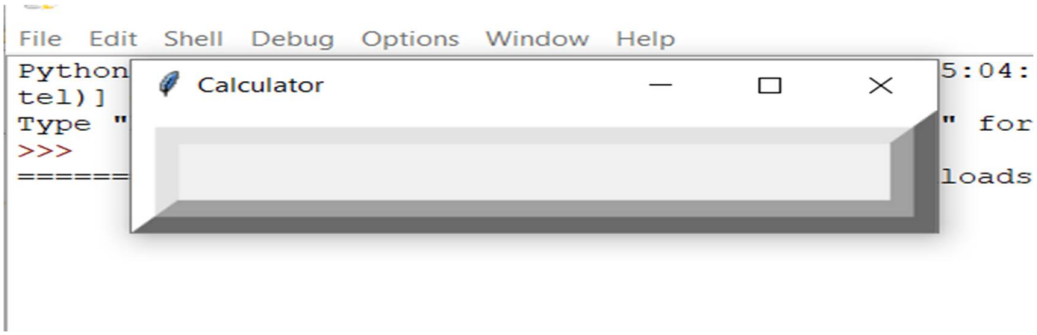


شکل 4-1

خب برنامه بدون مشکل اجرا شد حالا ادامه کد نویسی را انجام می دهیم تا شکل برنامه به مرور تکمیل شود. از دکمه ورودی display استفاده می کنیم و آن را مساوی StringVar() قرار می دهیم تا بتوانیم ویجت هایی مانند Entry را به طور موثر مدیریت کنیم. حال از ویجت یا همان ابزارک Entry استفاده می کنیم و با استفاده relief شکل صفحه نمایش ماشین حساب به صورت سه بعدی را در می آوریم و من برای پروژه خودم از حالت RAISED استفاده کردم تا صفحه نمایش ماشین حساب شکل برآمده داشته باشد. سپس با استفاده از textvariable استفاده کردم تا متن را از ویجت ورودی بازیابی کند و به عنوان نمونه ای از StringVar() باشد. با استفاده از justify تعیین می کنیم ماشین حساب ما راست چین باشد . با استفاده از State و قرار دادن آن در حالت DISABLED کاری کردم که دکمه ها فقط در حالتی که ماوس بر روی دکمه ها قرار بگیرد فعال شود. Bd که پهنای حاشیه را نشان می دهد را روی 20 تنظیم کردم، شما می توانید هر مقداری که دوست دارید تعیین کنید و در نهایت bg یا همان رنگ پس زمینه را آبی روشن تعیین کردم که شما می توانید هر رنگی که دوست دارید قرار دهید و به صورت زیر نوشتیم:

```
display = StringVar ()
    Entry (self, relief=RAISED,
        textvariable=display, state=DISABLED, justify='right', bd=20,
bg="light blue"). pack (side=TOP, expand=YES, fill=BOTH)
```

بعد از نوشتن این بخش کد برنامه را اجرا می کنیم تا ببینیم خطایی وجود دارد یا نه که برنامه ما بدون خطا انجام شد:



شکل 4-2

خب حالا که کمی پیشرفت کردیم ، دکمه ابتدایی ماشین حساب خودمان که دکمه C است را می نویسیم.

به وسیله دکمه ورودی clrChar که آن را برابر c قرار دادیم و تعیین کردیم دکمه c در بالای صفحه نمایش قرار بگیرد و همچنین از تابع lambda استفاده کردم که حدوداً جایگزین تابع def و دستور return می شود و به طور خودکار کار آن ها را برای ما انجام می دهد. به صورت زیر :

```
clrChar = "C"
button (self, TOP, clrChar, lambda appObj=display,
    i=clrChar: appObj.set(' ') )
```

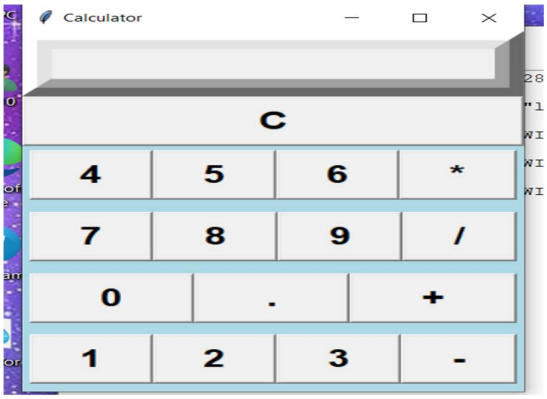
دوباره برنامه را اجرا کردم تا از درست بودن برنامه مطمئن باشم و برنامه به شکل زیر نمایش داده شد:

شکل 4-3

همانطور که دیدید با نوشتن این خط از کد دکمه C ماشین حساب ما نمایان شد و خطایی هم وجود نداشت.

حالا با خیال راحت ادامه کد نویسی را انجام می دهیم تا برنامه ما تکمیل شود.

در ادامه برنامه باید دکمه اعداد را روی ماشین حساب مشخص می کنیم که متوجه موضوعی شدم.



شکل 4-4

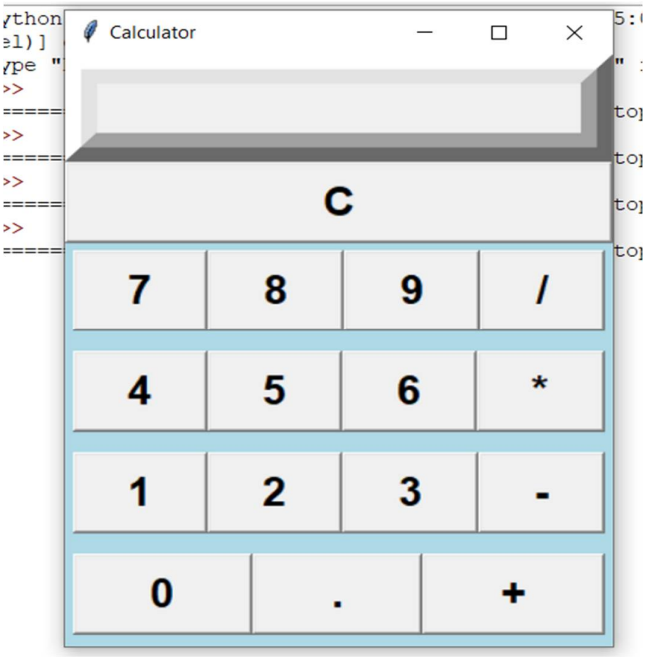
مطابق تصویر (شکل 4-4) متوجه شدم اگر بخواهم اعداد ماشین حساب کاملا مرتب و اصولی باشد باید اعداد حساب را به ترتیب از چپ به راست بنویسم مثلا اعداد 7، 8 و 9 را باید در ابتدا و سپس اعداد دیگر را بنویسم. پس با استفاده از حلقه for به صورت زیر اعداد را در کدنویسی برنامه اضافه کردم. به صورت زیر:

```
for btnNum in ("789/", "456*", "123-", "0. +"):
```

سپس در FunctionNum فانکشن fCalc را فراخوانی می کنیم و عملیات ها را در آن می نویسیم تا ماشین حساب ما شکل بگیرد. به صورت زیر :

```
FunctionNum = fCalc (self, TOP)
    for fEquals in btnNum:
        button (FunctionNum, LEFT, fEquals,
            lambda appObj=display, i=fEquals: appObj.set (appObj.get () +i))
        EqualsButton = fCalc (self, TOP)
```

بعد از نوشتن این چند خط کد دوباره برنامه را اجرا می کنیم تا اگه خطایی بود برطرف کنیم.



شکل 4-5

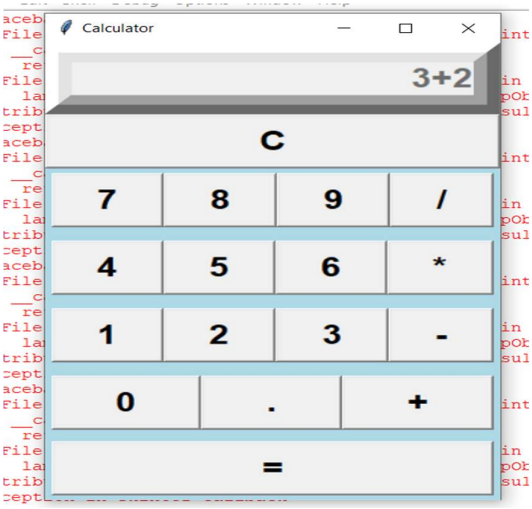
برنامه ما بدون خطا اجرا شد. حال می ریم سراغ دکمه مساوی برنامه که بدون این دکمه استفاده از این

ماشین حساب کاملا بیهوده است. پس ما با استفاده از حلقه for عملیات دکمه مساوی و عملیات هایی که نتیجه آن با مساوی نمایش داده خواهد شد را می نویسیم:

```
for fEquals in:"="
        if fEquals:== "="
           btnEquals = button (EqualsButton, LEFT, fEquals)
           btnEquals.bind('<ButtonRelease-1>')
                   lambda e, s=self, appObj=display: s. result(appObj),"+")
```

در کد بالا اگر دقت کنید از تابع bind() استفاده کردم که کاربرد این تابع در tkinter این است که بتوانیم رویداد ها را در پایتون کنترل کنیم. در داخل تابع bind نیز از رویداد ButtonRelease-1 استفاده کردم به این صورت که برای کار با ماشین حساب میتوانیم از دکمه چپ ماوس کلیک کنیم تا کار ما انجام شود. البته شما می توانید رویداد های دیگه ای نیز تنظیم کنید مثلا اگر جای عدد 1، 2 قرار دهید دکمه وسط ماوس کلیک می کند و اگر عدد 3 را قرار دهید دکمه راست ماوس کلیک می کند که انتخاب من دکمه چپ ماوس بود. بعد از این یک دستور else به کد نویسی برنامه اضافه کردم که خاصیت آن نشان دادن باقی مانده است. در این دستور دوباره از تابع lambda استفاده کردم و بعد از آن s را برابر s٪ قرار دادم که s همان self است و کاربرد s٪ این است محتوای رشته ما را چاپ کند.

خب تا این قسمت کد را نوشتیم و با هم برنامه را اجرا می کنیم:



شکل 4-6

مطابق تصویر دکمه مساوی برنامه بدون مشکلی به برنامه اضافه شد فقط یک مشکلی که پیش آمد

وقتی مقداری را وارد می کنیم با خطا مواجه می شویم و عدد نهایی را به ما نشان نمی دهد.

```
Exception in Tkinter callback
Traceback (most recent call last):
  File "C:\Program Files (x86)\Python39-32\lib\tkinter\__init__.py", line 1892,
in __call__
    return self.func(*args)
  File "C:\Users\WIN10\Desktop\سسس.py", line 44, in <lambda>
    lambda e, s=self, appObj=display: s.result(appObj), "+")
AttributeError: 'app' object has no attribute 'result'
```

خطایی که هنگام اجرا برنامه به ما نشان داده شد. برای برطرف کردن این خطا بود که برای برطرف کردن آن و اجرای درست برنامه تابع def را می نویسیم و با استفاده از فانکشن result و همینطور با استفاده از try except برای مدیریت خطا های موجود در برنامه که در فصل قبل در موردش توضیح دادیم شروع به نوشتن کد می کنیم و سپس به کمک متد های get و set مقدار ها را می گیریم و در نمایشگر نشان می دهیم.

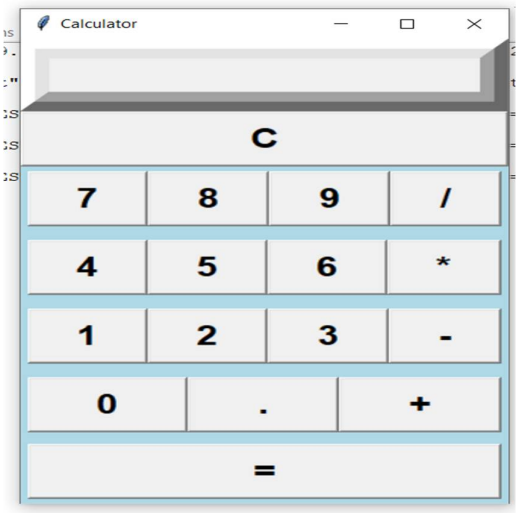
همچنین این دستور را نیز صادر می کنیم اگر بدون عدد زدن مساوی را بزنیم عبارت UNDEFINED به معنای نا مشخص و نا معلوم نمایش داده شود.

```
def result (self, display):
    try:
        display.set (eval (display.get ()))
    except:
        display.set("UNDEFINED")
```

در کد بالا از تابع eval نیز استفاده کردم که خاصیت آن برگرداندن عبارت عددی یا معادل عبارت است.

اگر به طور کلی بخواهم توضیح دهم ما کد را داخل یک try گذاشتیم که اگر هر مشکلی پیش آمد خطا را نشان دهد ولی در عین حال نیز بهش گفتیم از کاربر رشته ای را بگیر که قابل محاسبه باشد در غیر این صورت

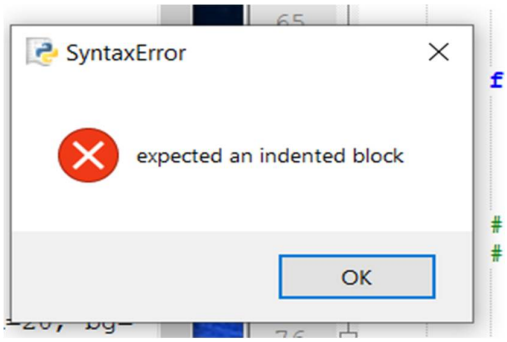
عبارت UNDEFINED را به کاربر نشان دهد. خب حالا دوباره برنامه را اجرا می کنیم:



شکل 4-7

می بینیم که برنامه ما بدون هیچ مشکلی اجرا شد و عملکرد برنامه نیز به صورت ویدئو نشان داده می شود.

حین کد نویسی این برنامه خطای جدی ای نداشتم فقط اینکه زمان برنامه نویسی پایتون متوجه شدم هر عبارت حتما باید در جایگاه خودش باشد ، به این صورت که حتی اگر میلی متری عبارت ما جابه جا شود برنامه ما با خطای زیر مواجه می شود:



شکل 4-8

خطای زیر برای تورفتگی است که نشان می دهد کد و تابع ما خطای تو رفتگی دارد و ما باید حد تو رفتگی ها را درست کنیم و گرنه برنامه ما اجرا نمی شود. پس رعایت تو رفتگی در پایتون بسیار مهم است.

4-4 ساخت برنامه بدون برنامه نویسی

مسلما اگر زبان برنامه نویسی بلد نبودم و مهندسی نرم افزار نخوانده بودم نوشتن این برنامه خیلی سخت و دشوار می شد، مجبور بودم برای هر دکمه کد جداگانه ای بنویسم، برای دکمه های عملیاتی نیز مجبور بودم کد جداگانه بنویسم و همین موضوع باعث می شد حجم برنامه بسیار زیاد شود و سرعت برنامه نیز کند شود.

5-4 جمع بندی فصل

در این فصل فهمیدیم برای ساخت برنامه ماشین حساب گرافیکی پایتون از کتابخانه tkinter و توابع مانند def استفاده کردیم و متوجه شدیم ساده ولی در عین حال کاربردی بودن این برنامه باعث شد آن را بسازیم تا افرادی که در برنامه نویسی مبتدی هستند نیز بتواند به راحتی کد نویسی کنند. سپس باهم دیگه روش ساخت برنامه را شرح دادیم و تابع و کد های موجود در برنامه را توضیح دادیم تا هنگام ساخت برنامه راحت تر عمل کنید. در حین ساخت برنامه متوجه شدم تو رفتگی بسیار در پایتون ضروری است و باید رعایت کنیم و گرنه برنامه ما اجرا نمی شود. سپس به این نتیجه رسیدیم اگر برنامه نویسی بلد نبود و نرم افزار را نمی شناختیم آنقدر ساخت برنامه دشوار می شد که باعث می شد سرعت کند و حجم بالایی داشته باشد.

جمع بندی و ارائه پیشنهادات

در طی چهار فصل باهم یاد گرفتیم زبان پایتون چیست و با تاریخچه آن آشنا شدیم. سپس در مورد عملگر ها و ماژول های پایتون صحبت کردیم و در نهایت نصب برنامه رسمی پایتون نسخه 3.9.1 را یاد گرفتیم. در فصل دوم نیز در مورد انواع ماشین حساب از چرتکه تا ماشین حساب مهندسی صحبت کردیم و سپس درباره ماشین حساب نرم افزاری صحبت کردیم و چهار برنامه پایتون ، سی شارپ ، سی پلاس پلاس و پی اچ پی را مورد بررسی قرار دادیم و فهمیدیم پایتون در سرعت یادگیری ، هک و امنیت و محبوبیت از برنامه های دیگر عملکرد بهتری دارد و همچنین چون برنامه بسیار کاربردی و راحتی بود از این برنامه استفاده کردیم تا افراد مبتدی نیز بتوانند به راحتی با این برنامه کار کنند. در فصل سوم نیز تا حدودی توابع و دستور هایی که در برنامه نویسی ماشین حساب استفاده کردیم را توضیح دادیم و با کتابخانه tkinter آشنا شدیم و در نهایت در فصل آخر نیز برنامه را تشریح کردیم و مرحله به مرحله توضیح دادیم تا افراد بتوانند به راحتی از آن استفاده کنند. در این چهار فصل سعی کردیم به افرادی که تازه دارند این زبان برنامه نویسی را یاد می گیرند تا حدودی هم برنامه را توضیح دهیم هم نحوه ساخت یک برنامه کاربردی را به آن ها یاد دهیم تا بتوانند درآینده با مشکلات کمتری برنامه خود را تولید کنند.

تقدیر

با تشکر از استاد محترم جناب آقای دکتر سیدعلی رضوی ابراهیمی که این جانب را در تهیه این پروژه یاری نمودند از خداوند متعال پیروزی و موفقیت روز افزون ایشان را خواستاریم.

Faculty of payam noor parand
Department of Computer engineering

B.Sc. Final Project Report

Title of the Report:

Python Digital Calculator

Under Supervision of:
d.r Ali Razavi

By:
Sara Naseri

<Date>
August of 2021