



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

صفحه	عنوان
ج	چکیده
د	پیشگفتار
1	فصل اول
2	1-1 پایتون
2	2-1 تاریخچه
3	1-2-1 واژه های ابداعی
3	3-1 دستور زبان
4	1-3-1 حکم های پایتون
5	4-1 عملگر ها
5	1-4-1 انواع عملگر
8	5-1 عملگر های خاص
8	1-5-1 عملگر های همانی

9	1-5-2 عملگر های عضویت
9	1-6 مازول ها
10	1-7 رابط گرافیکی پایتون
11	1-8 نصب پایتون نسخه 3.0
13	1-9 جمع بندی فصل
14	فصل دوم
15	2-1 تاریخچه ماشین حساب
15	2-2 انواع ماشین حساب
16	2-3 ماشین حساب سیستم
16	2-4 مقایسه زبان های برنامه نویسی برای ساخت ماشین حساب
17	2-5 چرا زبان پایتون؟
19	2-6 جمع بندی فصل
20	فصل سوم
21	3-1 کتابخانه Tkinter
22	3-2 توابع و دستور
22	3-2-1 def
22	3-2-2 frame
22	3-2-3 class
23	3-3 ابزار های پایتون

24	lambda 4-3
25	button 5-3
26	for حلقه 6-3
26	if...else دستور 7-3
27	try...except 8-3
28	9-3 جمع بندی فصل
29	فصل چهارم
30	1-4 تصویر نهایی ماشین حساب به زبان پایتون
31	2-4 کد های ماشین حساب به زبان پایتون
33	3-4 جمع بندی فصل
34	جمع بندی و ارائه پیشنهادات



### 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 به ساخته و پیاده سازی می کنیم. پایتون جزو زبان های ساده و شیرین و همچنین پر کاربرد 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<sup>1</sup> یک زبان برنامه نویسی شی گرا، تفسیری، سطح بالا و همه منظوره است که توسط خیدوفان روسوم<sup>2</sup> 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<sup>3</sup>، روبی<sup>4</sup>، تی سی ال<sup>5</sup> 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

---

<sup>1</sup> Python

<sup>2</sup> Guido van Rossum

<sup>3</sup> Perl

<sup>4</sup> Ruby

<sup>5</sup> Tool Command Language

خیدو از توسعه پایتون پیدا کردن جانشینی برای زبان برنامه نویسی ای بی سی<sup>6</sup> 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<sup>7</sup> 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 هکرها 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<sup>8</sup> 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 آپایتونیک است که در واقع کدی است که فهم آن مشکل است یا مانند رونویسی از زبان دیگری است.

## 1-3-3 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.

<sup>6</sup> زبان برنامه نویسی همه منظوره که در CWI کشور هلند تولید شده است.

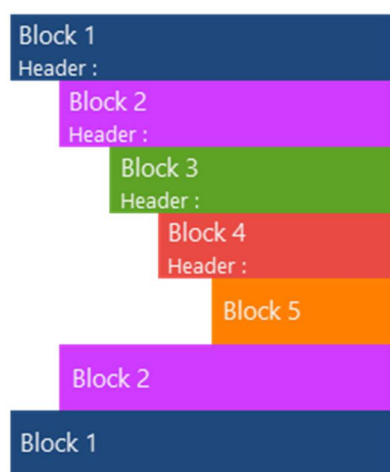
<sup>7</sup> Unicode

<sup>8</sup> 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 می کند. برای شروع بلوک کد یک پله تورفتگی را بیشتر می کنیم و برای اتمام آن یک پله بلوک کد را کمتر می کنیم. پس ساختار ظاهری برنامه نمایش دهنده ساختار معنایی آن است.

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

تصویر زیر نشان دهنده تورفتگی 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)
```

شکل 1-1

## 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 می شود.

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

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

حکم break برای خروج از نزدیک ترین حلقه استفاده می شود.

حکم continue که برای پریدن از دور جاری و ادامه دادن از دور بعدی در نزدیک ترین حلقه استفاده می شود.

حکم del که برای پاک کردن متغیرها از آن استفاده می شود. و حکم pass که به عنوان NOP<sup>9</sup> از آن استفاده می شود و از آن برای درست کردن بلوک کد خالی استفاده می شود.

حکم assert که هنگام دیباگ کردن استفاده می شود. شرطی که باید اتفاق بیافتد را بررسی می کند.

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

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

حکم return که برای برگرداندن مقداری در تابع استفاده می شود.

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

## 1-4 عملگرها

عملگر<sup>10</sup> به نمادی گفته می شود که عمل مشخصی را بر روی اشیاء به انجام می رساند، همچنین به اشیایی که عملگر بر روی آن ها عملی را به انجام می رساند عملوند<sup>11</sup> گفته می شود. عملگرها دارای انواع مختلفی هستند که در ادامه به آن ها می پردازیم.

## 1-4-1 انواع عملگر

عملگرهای حسابی<sup>12</sup> 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.	$3 - 2 = 1$

<sup>9</sup> دستور زبان برنامه نویسی که هیچ کاری انجام نمی دهد.

<sup>10</sup> Operator

<sup>11</sup> Operand

<sup>12</sup> Arithmetic Operators

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

جدول 1-1

جدول بالا نشان دهنده عملگرهای حسابی پایتون است.

عملگرهای مقایسه ای<sup>13</sup> 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.

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

جدول 2-1

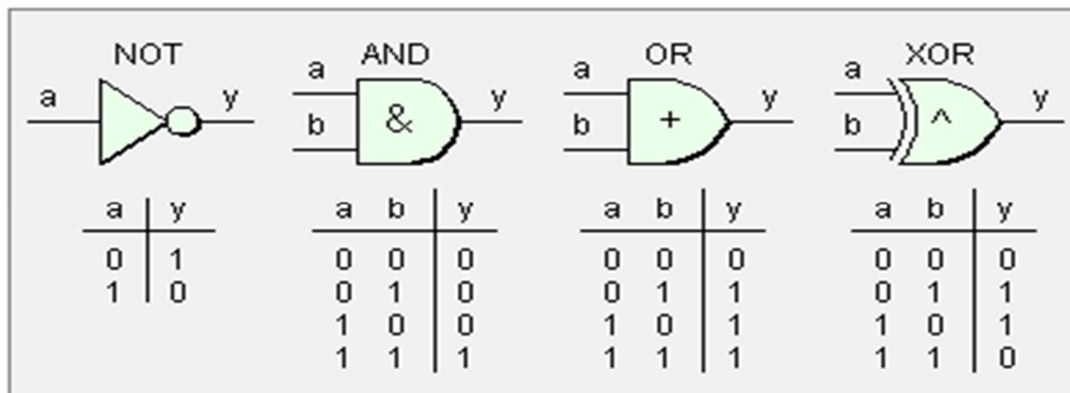
عملگرهای منطقی<sup>14</sup> در واقع and، or و not است و در پایتون با یک سری عبارت های شرطی که به کار می رود چک می شود و این عبارت ها معادل True و False هستند. جدول زیر توضیحاتی در مورد این عملگر می دهد:

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

جدول 3-1

<sup>13</sup> Comparison Operators  
<sup>14</sup> Logical Operators

عملگر های بیتی<sup>15</sup> 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 2-1 داریم :



شکل 2-1

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 2-1 فرمت باینری آن ها به صورت زیر خواهد بود:

$a = 0000\ 0100$ ,  $b = 0000\ 1010$

مثال	توضیح	عملگر
$a \& b = 0000\ 0000$	And (و) بیتی	$\&$
$a   b = 0000\ 1110$	Or (یا) بیتی	$ $
$\sim a = 1111\ 0101$	Not (نقیض) بیتی	$\sim$
$a \wedge b = 0000\ 1110$	Xor بیتی	$\wedge$
$a \gg = 0000\ 0010$	جا به جایی به راست بیتی	$\gg$
$a \ll = 0010\ 1000$	جا به جایی به چپ بیتی	$\ll$

جدول 4-1



عملگر بعدی عملگر تخصیص<sup>16</sup> 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 5-1

## 5-1 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

### 5-1-1 عملگر های همانی

is و is not عملگر های 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

<sup>16</sup> 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 عملگرهای عضویت

in و not in عملگرهای عضویت در پایتون هستند. این عملگرها برای بررسی اینکه یک مقدار یا متغیر در یک توالی پیدا می شود یا نه مورد استفاده قرار می گیرند. 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 عملگرهای عضویت را می بینیم:

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

## 1-6-6 مازول<sup>17</sup>

مازول به فایلی گفته می شود که حاوی دستورات و تعاریف پایتون است. یک فایل حاوی کدهای پایتون ، یک مازول

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

با دستور `import` 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.

## 7-1 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 پر کاربرد 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 آجکت ، 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 نیازهای 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 اختیاران قرار می دهم.

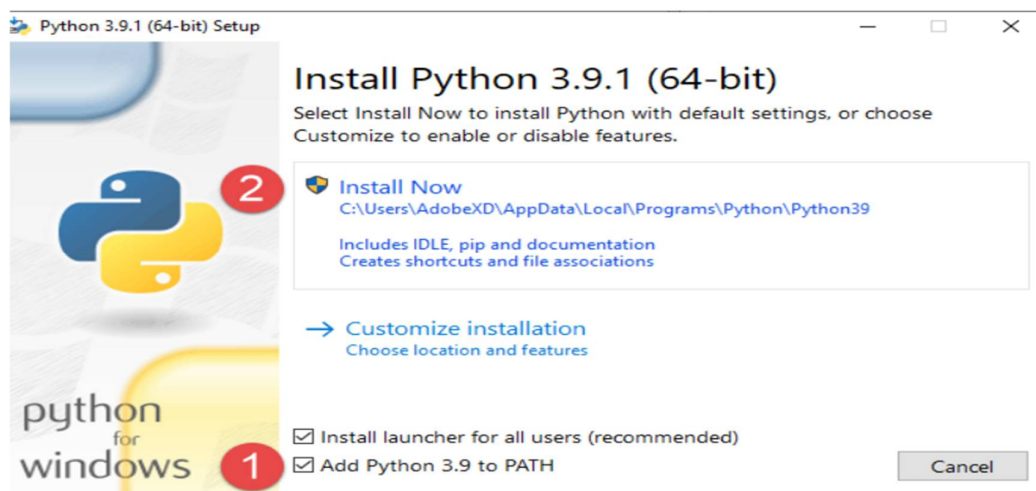
در ابتدا برای نصب پایتون به سایت رسمی پایتون می رویم : <https://www.python.org/downloads>

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



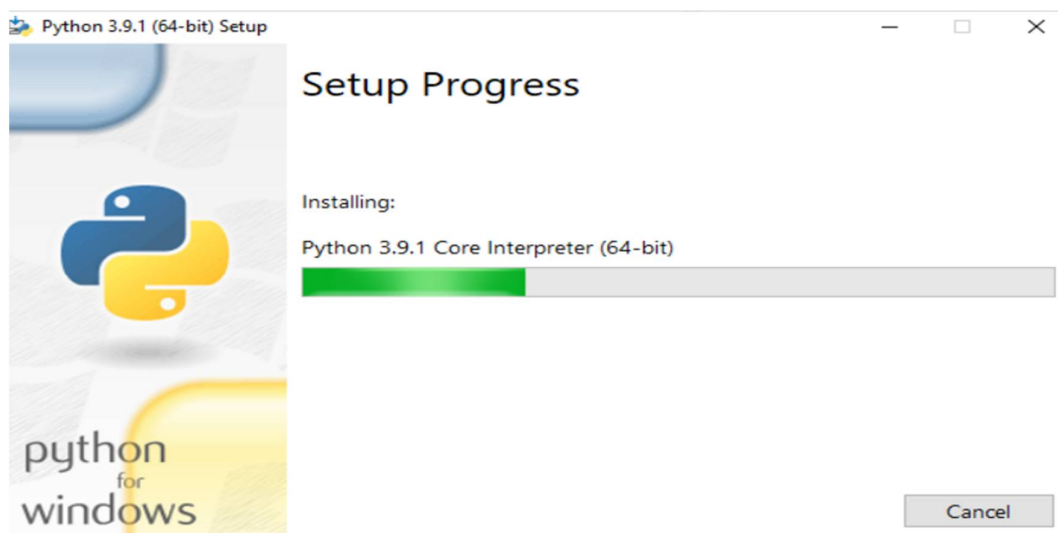
### شکل 3-1

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



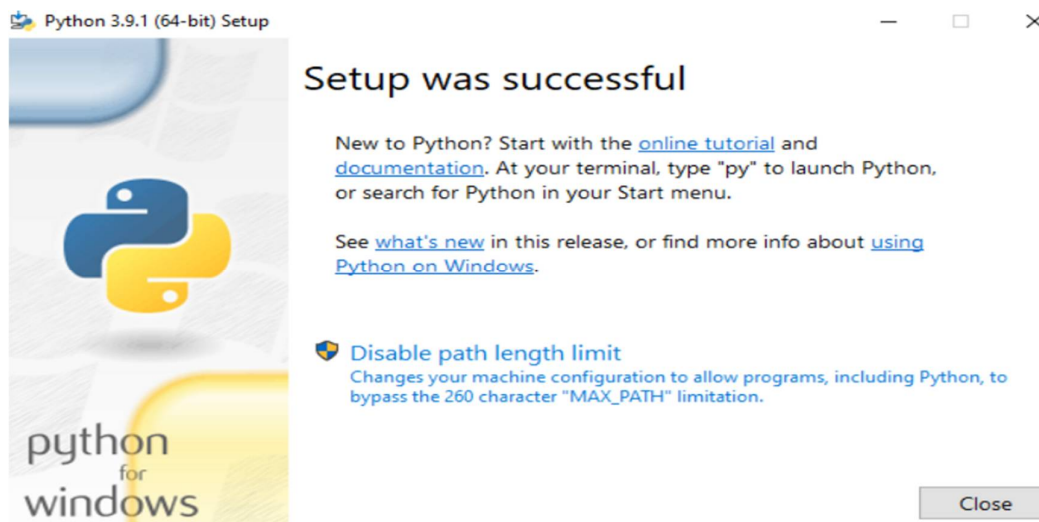
### شکل 4-1

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 5-1

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 6-1

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

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

## 9-1 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 switch استفاده 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 برنامه نویسی شی گرا معمولا با عنوان OOP شناخته می شود که این سه حرف مخفف کلمات Object-Oriented Programming می باشد. یک شیوه برنامه نویسی است که ساختار یا بلوک اصلی اجزای آن، شی ها می باشند.

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 برنامه نویسی چند الگویی و منظم شده مدل های تابعی، امری، عمومی، شی گرا و جز گرا و در بستر دات نت می باشد. سی شارپ یکی از 44 زبان برنامه نویسی است که توسط زمان اجرای زبان مشترک از چارچوب دات نت پشتیبانی می شود و در همه جا به وسیله مایکروسافت ویژوال استودیو شناخته می شود.

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 این 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 گذاری عملگر<sup>18</sup> 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.

---

<sup>18</sup> 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 برنامه نویسی شی گرا، از برنامه نویسی تابعی<sup>19</sup> و ساختاری<sup>20</sup> نیز پشتیبانی می شود. تعریف و استفاده از کلاس ها<sup>21</sup> و اشیا<sup>22</sup> در زبان پایتون، به دلیل وجود ویژگی های شی گرایی، بسیار ساده است. قابلیت خوانایی و اشکال زدایی کدها در زبان پایتون بسیار بالاست. به همین خاطر، یادگیری آن به برنامه نویسان مبتدی توصیه می شود. همچنین کتابخانه استاندارد زبان پایتون بسیار غنی است و با تمامی پلتفرم های موجود نظیر ویندوز، لینوکس و مک مطابقت دارد. زبان پایتون، انتخاب ایده آلی برای توسعه های برنامه های کاربردی شبکه<sup>23</sup> است؛ به ویژه برنامه های که باید از پروتکل های مختلف و متعددی استفاده کنند. همچنین به دلیل چرخه توسعه بسیار سریع برنامه های کاربردی در زبان پایتون، این زبان برای نمونه سازی<sup>24</sup> و تست کردن<sup>25</sup> برنامه های کاربردی بسیار مناسب است. و در نهایت زبان پایتون منبع باز است و به شکل بسیار مناسبی، توسط تیم های توسعه و جامعه برنامه نویسی پشتیبانی می شود. 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

---

<sup>19</sup> Functional

<sup>20</sup> Structural

<sup>21</sup> Classes

<sup>22</sup> Objects

<sup>23</sup> Network Applications

<sup>24</sup> Prototyping

<sup>25</sup> 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 1-2

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.

## 2-6 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.

## 3-1 کتابخانه Tkinter

اول از همه با کتابخانه Tkinter<sup>26</sup> آشنا می شویم. کتابخانه گرافیکی که باهاش میتونید 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 احتمالاً با تصویر زیر روبرو می شوید.

---

<sup>26</sup> کتابخانه Tkinter در واقع از زبان برنامه نویسی TK یا TCL ساخته شده و مخفف Tk Interface می باشد.



شکل 1-3

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

## 2-3 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<sup>27</sup> 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.

### def 1-2-3

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

### frame 2-2-3

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 تصویر 1-3 که در صفحه 21 مشاهده کردید به دلیل وجود ابزار frame بود. در واقع بدون frame پنجره ای که در تصویر دیدید باز نمی شود.

<sup>27</sup> 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<sup>28</sup> جدید می سازد که در آن، 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 می شود. 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 می شود.

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

Entry: برای وارد کردن متن تک خطی کاربر استفاده می شود.

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

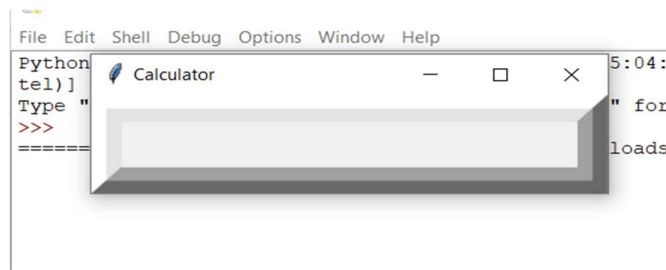
نتیجه ای که بدست می آید را می بینیم.

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

<sup>28</sup> Local Namespace



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



شکل 2-3

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.

### 4-3 lambda

لامبدا یک روش ساده برای تعریف تابع در پایتون است. قبل از ایجاد دکمه در ماشین حساب خود اول از همه با کلمه Lambda آشنا 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 این ها توابعی هستند که لازم نیست نامی برایشان تعیین کنید. این روش برای تعریف تابع های کوچک یک بار مصرف در مواردی که تابع اصلی بسیار بزرگ و حجیم است، استفاده می شود. لامبداها یک شی تابع باز می گردانند که می توانند به یک متغیر انتساب یابد. لامبداها می توانند هر تعداد آرگومان که لازم باشد داشته باشند؛ اما تنها یک عبارت دارند. نمی توان توابع دیگر را درون یک لامبدا فراخوانی کرد. بیشترین استفاده از تابع های لامبدا در کدهایی است که نیازمند توابع یک خطی ساده ای 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 خود کار این کار را برای شما انجام می دهد. 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

## button 5-3

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 اصولاً 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 RIDGE, SUNKEN, RAISED, GROOVE و

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 اصولاً 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. Left برای تراز کردن هر خط به سمت چپ، center تراز خطوط به وسط است و right برای تراز هر خط به

سمت راست است.

### 3-6 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 دقیقاً می دانیم که تسک مورد نظر 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 دنباله btnNum 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 می شود و این پروسه تا زمان اتمام اعضای دنباله btnNum ادامه پیدا کند.

### 3-7 دستور if...else

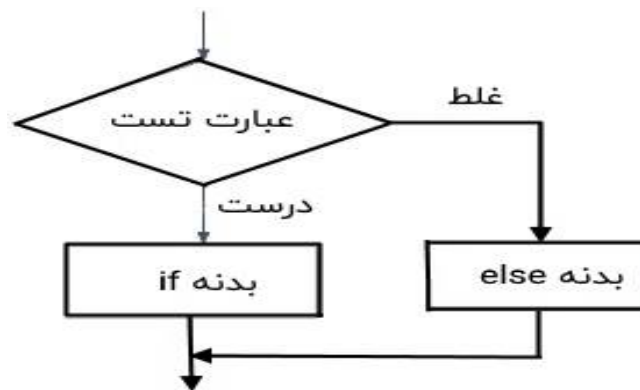
دستور if...else در کد زیر به این صورت است:

```

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 کدهای آن مربوط به نحوه جمع کردن اعداد 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 4-3

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

### try...except 8-3

ما با استفاده از دستور try...except می توانیم خطاها 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.

### 3-9 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, relief 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.

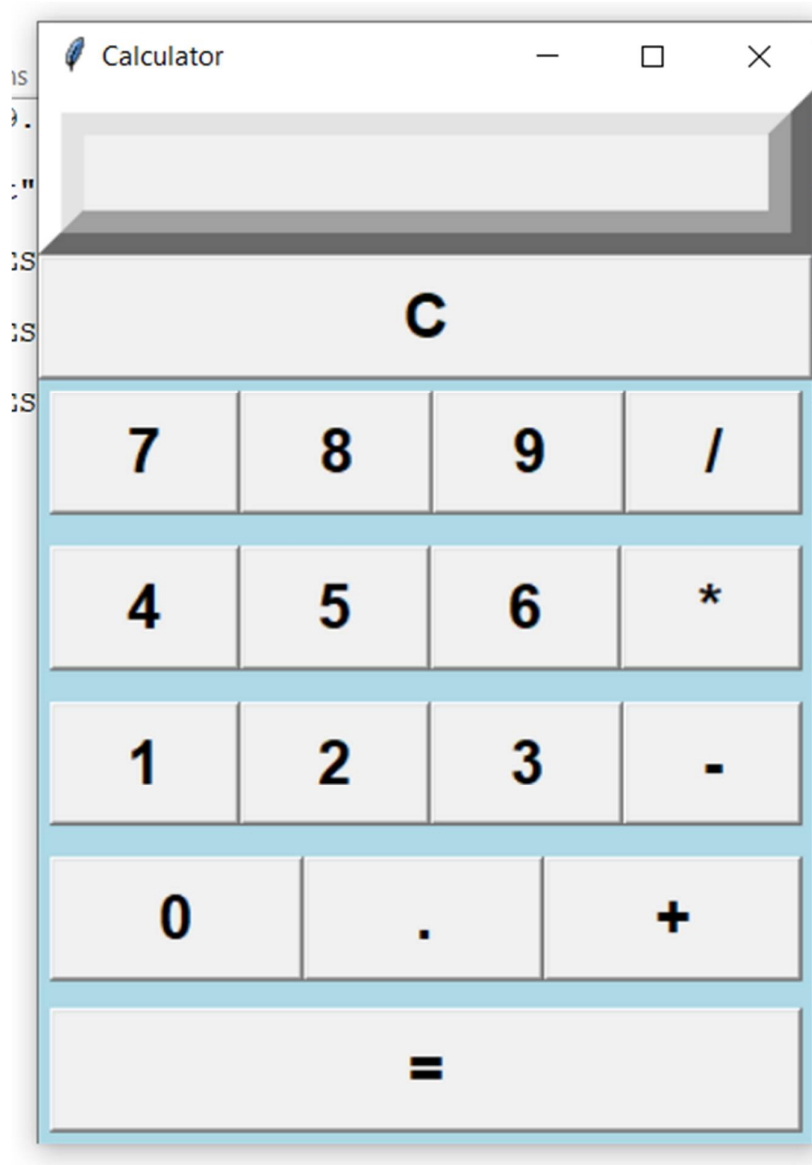
## فصل چهارم

### مقدمه

در این فصل کل نهایی برنامه ای که ساختیم را نشان می دهیم و کدهای آن را به نیز نشان می دهیم تا با نحوه ساخت این ماشین حساب ساده و کاربردی به طور کامل آشنا شوید.

#### 1-4 تصویر نهایی ماشین حساب به زبان پایتون

بعد از سه فصل توضیحات در مورد پایتون و کدهای مربوط به ماشین حساب بالاخره ماشین حساب خود را ساختیم که در تصویر زیر قابل مشاهده است.



شکل 1-4

اگر کدها را به درستی وارد کنیم میتوانیم یک ماشین حساب ساده و کاربردی مانند تصویر بالا داشته باشیم.

حالا که تصویر برنامه را دیدیم در ادامه می رویم سراغ کدهای برنامه تا با نحوه ساخت آن آشنا شویم.

## 2-4 کد های ماشین حساب به زبان پایتون

در این بخش کد های استفاده شده در ماشین حساب را به اشتراک می گذاریم:

```
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, 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()

        display = StringVar()
        Entry (self, relief= RAISED ,
               textvariable=display, state=DISABLED, justify='right', bd=20, bg="light
blue"). pack (side=TOP, expand=YES,
               fill=BOTH(
        clrChar = "C"
        button (self, TOP, clrChar, lambda appObj=display, i=clrChar: appObj.set (""))

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

            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)

for fEquals in:"="
    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
0 +s))
def result (self, display):
    try:
        display.set (eval (display.get ()))
    except:
        display.set("UNDEFINED")

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

```

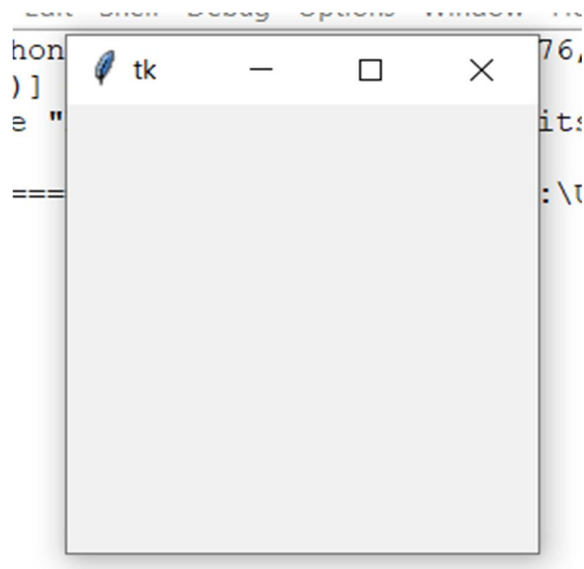
با این کد می توانید به سادگی یک ماشین حساب کاربردی به زبان پایتون برای خود بسازید.

همانطور که از اول توضیح دادیم در این برنامه از کتابخانه tkinter استفاده می کنیم و برای توابع از تابع def و

Lambda استفاده شد. همینطور از ابزارهای مختلفی مثل font و bg و دیگر ابزارهای مورد نیاز استفاده کردیم

و برنامه را مطابق میل خود ساختیم. در انتهای برنامه هم از mainloop() استفاده کردیم که وجود آن برای هر برنامه ای در پایتون کاملاً ضروری است، به دلیل اینکه اگر mainloop() نباشد برنامه خروجی ندارد و نشان داده نمی شود

و برنامه ما را به حالت زیر نمایش می دهد:



شکل 2-4

با دیدن این شکل به راحتی متوجه می شویم اگر برنامه ما یا هر برنامه به زبان پایتون `mainloop()` در آن نباشد یا برنامه به شکل ناقص نشان داده می شود یا در بعضی مواقع برنامه ای نشان داده نمی شود.

## 3-4 جمع بندی فصل

در این فصل نیز باهم دیگر شکل نهایی ماشین حساب و کدهایی که برای ساخت ماشین حساب استفاده کردیم را دیدیم و یاد گرفتیم اگر از `mainloop()` در انتهای کد نویسی استفاده نکنیم چه اتفاقی برای برنامه ما می افتد.

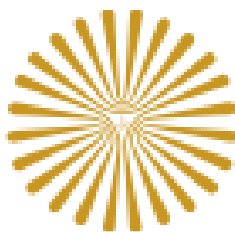
## جمع بندی و ارائه پیشنهادات:

چهار فصل باهم دیگه در مورد پایتون و ساخت یک برنامه به زبان پایتون صحبت کردیم. در فصل اول فهمیدیم پایتون در کشور هلند طراحی شد و اولین بار در سال 1991 توسط خیدو فان روسوم منتشر شد. سپس با ویژگی های پایتون آشنا شدیم که شی گرا، سطح بالا، چند منظوره و متن باز بودن از ویژگی های آن بود. سپس با پایتونیک و آنپایتونیک آشنا شدیم و فهمیدیم 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. یاد گرفتیم هنگام استفاده پایتون از تورفتگی به جای آکولاد استفاده می کنیم. با انواع حکم های پایتون مانند if و else و غیره آشنا شدیم و فهمیدیم عملگر به نمادی گفته می شود که عمل مشخصی را بر روی اشیاء به انجام می رساند، همچنین به اشیایی که عملگر بر روی آن ها عملی را به انجام می رساند عملوند گفته می شود. و با انواع عملگر ها مقایسه ای و حسابی و منطقی و تخصیص و غیره در پایتون و کار کرد آن ها در پایتون آشنا شدیم بعد از عملگر ها با سه رابط گرافیکی Tkinter, wxpython, pyqt و کاربرد آن ها آشنا شدیم و در برنامه خودمان از tkinter استفاده کردیم. و در نهایت یاد گرفتیم چگونه برنامه پایتون نسخه 3.0 را بر روی سیستم خود نصب کنیم

در فصل دوم با تاریخچه پایتون آشنا شدیم و فهمیدی 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. سپس با انواع ماشین حساب از از چرتکه گرفته تا ماشین حساب مهندسی آشنا شدیم بعد از آن پایتون را با سه زبان برنامه نویسی سی شارپ، سی پلاس پلاس و php مقایسه کردیم و فهمیدی پایتون از نظر یادگیری، امنیت و هک و محبوبیت بهتر از سه برنامه دیگر است و همچنین متوجه شدیم برای کسانی که تازه کار با برنامه نویسی را شروع کردند مناسب تر است. در فصل سوم به طور خلاصه کد های برنامه را بررسی کردیم و تابع و دستور ها و ابزار های مورد استفاده در پایتون را معرفی کردیم و فهمیدیم کتابخانه tkinter کتابخانه گرافیکی که باهاش میتونید یک برنامه گرافیکی بنویسید. کتابخانه Tkinter برای طراحی رابط کاربری اپلیکیشن ها است و به طور پیش فرض بر روی پایتون نصب می باشد. در مورد تابع ها فهمیدیم که تابع ها کمک می کنند تا برنامه ای که می سازیم به بخش های کوچتر 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. با ابزار هایی مانند Label، برای نمایش متن یا تصویر بر روی صفحه و Button، برای افزودن دکمه ها به برنامه و Entry، برای وارد کردن متن تک خطی کاربر استفاده می شود. و همینطور یک متد pack() داریم که ابزارک ها را در سطر ها یا ستون ها دسته بندی می کند. با حلقه for آشنا شدیم که حلقه for همانند حلقه while برای تکرار اجرای یک قطعه کد مورد استفاده قرار می گیرد، با این تفاوت که در حلقه for دقیقاً می دانیم که تسک موردنظر را چند مرتبه تکرار کنیم. بعد از آن با دستور if...else آشنا شدیم که فهمیدیم می گوید اگر شرط اولی انجام نشد شرط دومی را انجام بده و بعد از آن دستور try...except بود که وظیفه اش این بود اگر هنگام اجرا و کار برنامه مشکلی پیش آمد مثلاً اگر به ماشین حساب دستور اشتباهی دادیم به ما کلمه نا مفهوم یا UNDEFINED را نشان دهد. در فصل چهارم نیز شکل نهایی ماشین حساب را نشان دادیم و کد های آن را قرار دادیم و بررسی کردیم اگر از تابع mainloop استفاده نکنیم برنامه ما یا اجرا نمی شود یا به صورت نا درست اجرا می شود.

در نهایت شاید بگویید چه نیازی به ساخت ماشین حساب داریم یا چرا زبان برنامه نویسی پایتون که در این صورت می توانیم بگوییم ماشین حساب جزو یکی از ابزار های کاربردی در زندگی روزمره ما است علاوه بر اون اگر شخصی تازه شروع به یادگیری برنامه نویسی کرده باشد ساخت این برنامه مفید و زیبا برای یادگیری زبان پایتون خالی از لطف نیست. حتی اگر روزی سطح سوادتان نسبت به برنامه نویسی پایتون بالا رفت می توانید خیلی راحت همین برنامه را گسترش داده و به یک ماشین حساب حرفه ای و مهندسی تبدیل اش کنید یا اینکه می توانید از کد های این برنامه در برنامه های دیگر استفاده کنید.



**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

