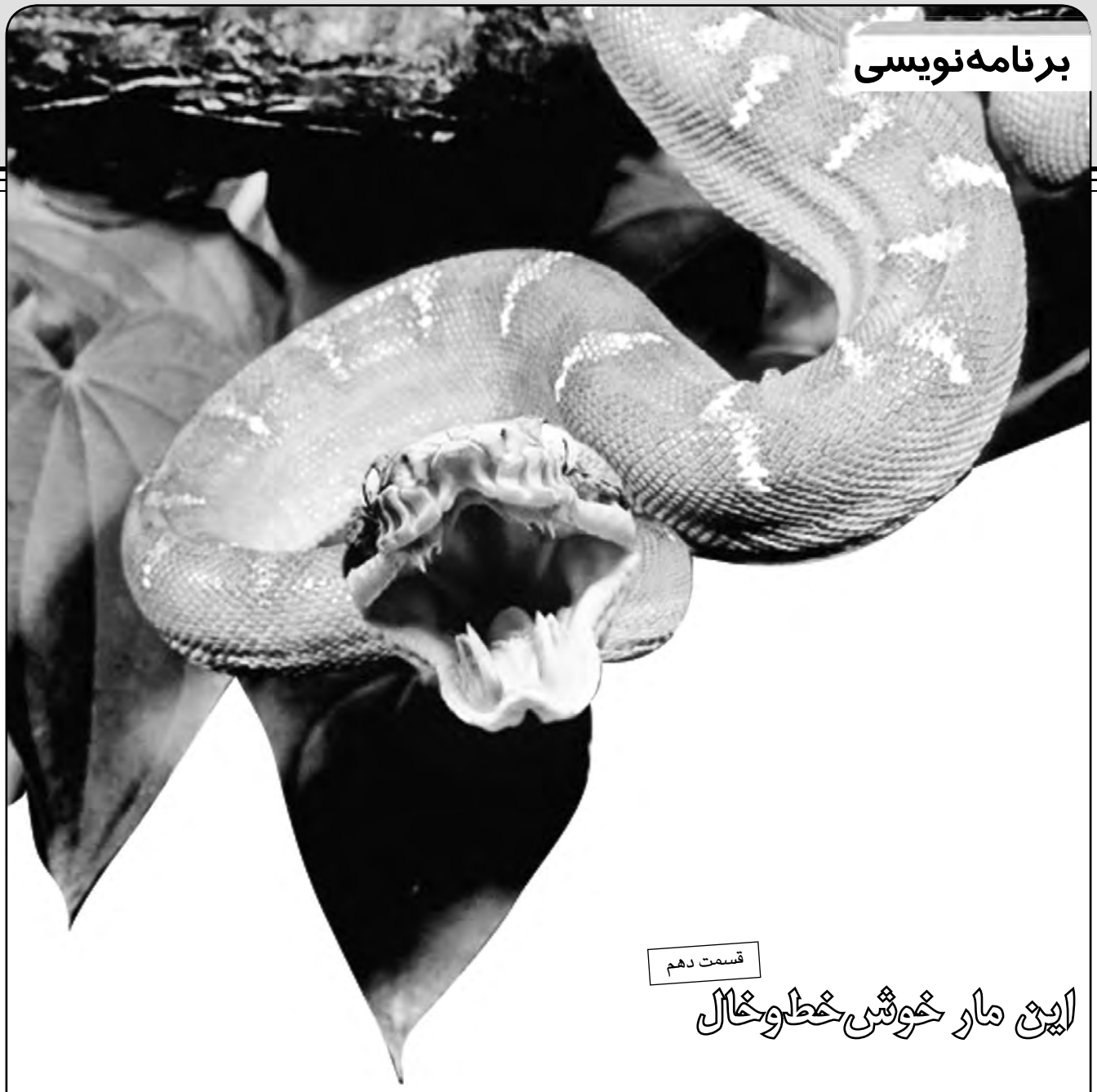


قسمت دهم

## این مار خوش‌خط‌وخال

# رنگ‌ولعاب در ترمینال‌های گنگ

» احمد شریف‌پور

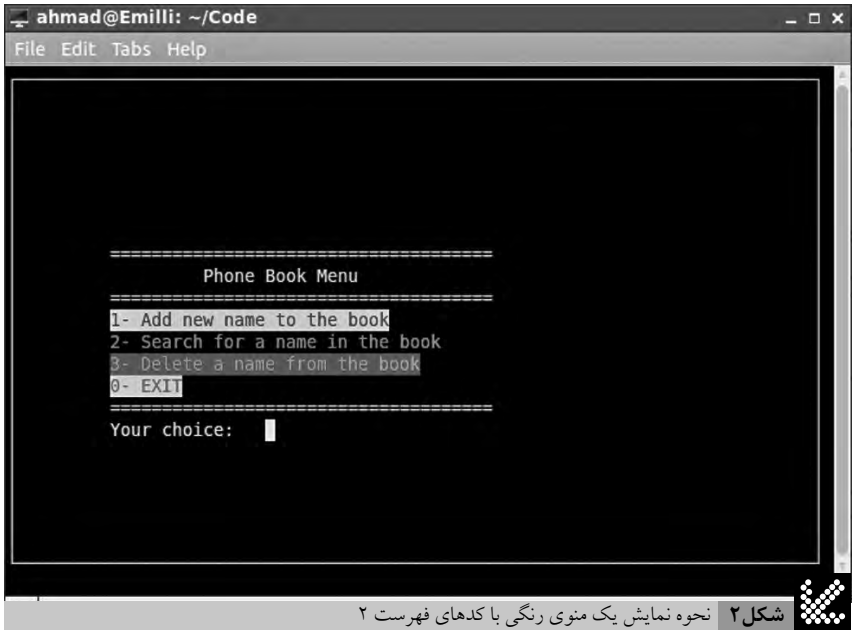
**اگر سن شما به اندازه‌ای باشد که نخستین روزهای کامپیوترها را به یاد بیاورید، به یاد خواهید داشت که در آن دوران کامپیوترها اغلب مین‌فریم بودند و «ترمینال‌های گنگ» به عنوان دستگاه‌های ورودی و خروجی به آن‌ها متصل می‌شدند. شما می‌توانستید به هر تعداد ترمینال که می‌خواهید به یک مین‌فریم متصل کنید. مشـــکل این‌جا بود که ترمینال‌ها واقعاً گنگ بودند! در آن‌ها خبری از پنجره‌ها، رنگ و هیچ چیز دیگری نبود. تنها در بهترین حالت صفحه‌ای با ۲۴ سطر ۸۰ کاراکتری در اختیار شما قرار می‌گرفت. در نخستین روزهای DOS و CPM نیز اوضاع به همین منوال بود. زمانی که برنامه‌نویسان آن زمان روی صفحات زیبا و تجملی به‌ویژه برای ورود و خروج داده‌ها کار می‌کردند، از کاغذهای شطرنجی برای طراحی صفحه استفاده می‌کردند. هر مربع نشانه یک کاراکتر بود. اکنون نیز زمانی که شـــما با برنامه‌های پایتون در ترمینال کار می‌کنید (که بیشـــتر اوقات چنین خواهد بود) اوضاع به همان منوال اســـت. اما هر مشکلی با پیش‌بینی و کسب آمادگی لازم قابل حل خواهد بود، پس کاغذهای شطرنجی‌تان را آماده کنید تا به سراغ نخستین برنامه Curses خودمان برویم.**

کدهای فهرست ۱ را در ویرایشگر دلخواهتان وارد کرده، با نام Curse.py ذخیره و سپس اجرا کنید. خواهید دید که همانند شکل ۱ کادری در اطراف پنجره ترمینال ترسیم می‌شود و متن مورد نظر ما در سطر دوازدهم و از محل کاراکتر بیست و پنجم به نمایش در خواهد آمد. در خط ۱ ما کتابخانه curses را فراخوانده‌ایم. پس از آن و در خط ۲ شیء از نوع صفحات curses با نام myscreen ایجاد و برای استفاده آماده کرده‌ایم. این صفحه درواقع بوم نقاشی ما خواهد بود. پس از آن در خط ۳ و با عبارت .myscreen border(0) کادری را در اطراف صفحه نمایش ترسیم کرده‌ایم. این کار الزامی نیست، اما ظاهر برنامه ما را زیباتر می‌کند. پس از آن و در خط ۴ با تابع ()addstr و مشخص کردن مکان نقطه شروع متنی را به صفحه نمایش افزوده‌ایم. تابع ()addstr در واقع همانند دستور print عمل خواهد کرد. اما تا این لحظه هیچ‌کدام از اعمال انجام شده روی صفحه نمایش قابل مشاهده نخواهند بود. برای نمایش این تغییرات باید مانند خط ۶ از تابع ()Refresh استفاده کرد. پس از آن و در خط ۷ با تابع ()getch منتظر فشرده شدن یک کلید از سوی کاربر مانده‌ایم و در نهایت با تابع ()endwin صفحه نمایش را آزاد کرده‌ایم تا ترمینال ما به عملکرد عادی خود بازگردد. توجه داشته باشید که دستور endwin بسیار مهم است. اگر این تابع فراخوانده نشود، ترمینال شما دیگر قابل استفاده نخواهد بود، پس همواره مطمئن شوید که این تابع درست قبل از پایان برنامه شما فراخوانده شده است. نکته دیگری که به احتمال متوجه آن شده‌اید این است که کادر یا border ما حاشیه‌ای به عرض یک کاراکتر را اشغال خواهد کرد.

```
1 import curses
2 myscreen = curses.initscr()
3 myscreen.border(0)
4 myscreen.addstr(10,12,»This Is\
5 The Simple Test . . .»)
6 myscreen.refresh()
7 myscreen.getch()
8 curses.endwin()
```

**فهرست۱** نمونه ساده کار با کتابخانه curses


**شکل۱** نتیجه اجرای کدهای فهرست یک




**شکل۲** نحوه نمایش یک منوی رنگی با کدهای فهرست ۲

به یاد داشته باشید شمارش سطرها و ستون‌های صفحه نیز همواره از صفر آغاز خواهد شد. به عبارت دیگر، بالاترین نقطه در سمت چپ صفحه مختصاتی برابر 0,0 خواهد داشت.

حال که با اصول کلی کار با curses آشنا شدیم، بیایید از این کتابخانه برای ایجاد پوسته ارتباطی یک برنامه فرضی دفتر تلفن استفاده کنیم. در این مثال ما عملاً از کدهای مربوط به منطق درونی برنامه که برای افزودن نام و شماره به دفتر تلفن، جست‌وجوی نام‌ها، حذف رکوردها و نحوه نگه‌داری و ذخیره‌سازی داده‌ها صرف‌نظر کرده و تنها روش پیاده‌سازی یک پوسته یا رابط کاربری ساده را با curses توضیح خواهیم داد.

کدهای فهرست شماره ۲ را در ویرایشگر دلخواهتان وارد کرده و با نام .PhonebookUI py ذخیره و اجرا کنید. در این حالت با صفحه‌ای مشابه شکل ۲ مواجه خواهید شد که در واقع منوی اصلی برنامه دفتر تلفن را نمایش می‌دهد. در این مرحله با فشردن کلیدهای ۱، ۲ و ۳ می‌توانید به منوهای مربوط به هر بخش وارد شده و عملیات مورد نظر را انجام دهید یا با وارد کردن عدد ۰ از برنامه خارج شوید.

کدهای فهرست ۲ از چهار بخش اساسی تشکیل شده است. در بخش نخست ما تنها import برنامه را برای استفاده از کتابخانه curses انجام داده‌ایم.

در بخش بعدی تابعی با نام MainMenu را تعریف کرده‌ایم. این تابع ابتدا در خط ۶ و با تابع erase() صفحه نمایش (که قبلاً توسط curses تعریف شده است) را پاک می‌کند. حاشیه یا کادر در خط ۷ فعال می‌شود. پس از آن و با استفاده از چندین تابع ()addstr شکل منوی اصلی را روی صفحه نمایش ترسیم کرده‌ایم. در این بخش شما می‌توانید فرم کامل تابع ()addstr را مشاهده کنید. این تابع در حالت عادی ۴ آرگومان دریافت می‌کند. دو آرگومان نخست شماره سطر و ستون محل نوشته شدن متن هستند. آرگومان سوم، متن مورد نظر را تعریف می‌کند و در نهایت آرگومان چهارم رنگ متن را مشخص می‌کند. رنگ‌های مورد نظر برای استفاده در ()addstr به صورت جفت رنگ‌های پیش‌زمینه و پس‌زمینه هستند که در مورد نحوه تعریف آن‌ها کمی بعدتر توضیح خواهیم داد.

در بخش سوم تعریف تابع GetKey آورده شده است. ما از این تابع برای دریافت پاسخ کاربر و واکنش به آن استفاده می‌کنیم. در خط ۲۷ یک حلقه while تعریف شده است و تا زمانی که کاربر کلید ۰ را فشار ندهد، به کار خود ادامه خواهد داد. اگر کاربر کلید ۰ را بزند، این حلقه و در نتیجه اجرای تابع به پایان خواهد رسید و کنترل به بخش فراخواننده (در این مثال حلقه اصلی) بازگردانده می‌شود. همان‌طور که پیش‌تر هم گفتیم، در اینجا از پیاده‌سازی کامل برنامه صرف‌نظر کرده‌ایم و تنها محل‌هایی را که باید کدهای منطق برنامه آورده شوند (خط‌های ۳۴،۳۱ و ۳۷) مشخص کرده‌ایم. تنها نکته جدید در مورد تابع GetKey شاید استفاده از تابع ()ord باشد. تابع ()ord با دریافت یک رشته تک کاراکتری، عدد متناظر یونی‌کد آن را بازمی‌گرداند. استفاده از این تابع باعث می‌شود که برنامه بتواند فشرده شدن کلید ۱ را در همه آرایش‌های صفحه کلید (مثلاً فارسی یا

عربـی) درک کرده و به آن واکنش نشـان دهد. بدنـه اصلی بخش آخر کدهای ما اسـت. در این بخش همانند مثـال قبلی ابتدا در خط ۴۰ صفحه نمایش راه‌اندازی شـده است. خط ۴۱ به curses اعلام می‌کند، ما قصد استفاده از رنگ در نوشتن متن‌ها را داریم. در صورت عدم اسـتفاده از تابع start_color() هر تلاشـی بـرای نمایش متون رنگی روی صفحه نمایش بی نتیجه خواهد ماند. نکته مهم دیگر این اسـت که این دستور باید بعد از ایجاد و آماده‌سازی صفحه نمایش توسط تابع initscr() مورد استفاده قرار بگیرد.

همان‌گونه که پیش‌تر هم اشـاره شـد برای ایجـاد متـون رنگی مـا باید از جفـت رنگ‌های مخصوص curses استفاده کنیم. خطوط ۴۲ تا ۴۵ ایـن کار را انجام می‌دهد. برای تعریف جفت‌های رنگـی از تابع init_pair() اسـتفاده می‌کنیم که سـه آرگومان را قبول می‌کند. آرگومان نخست شماره جفت رنگی را مشخص می‌کند. آرگومان دوم رنـگ متن یا پیش‌زمینه را مشـخص‌کرده و آرگومـان سـوم رنـگ پس‌زمینـه را تعیین می‌کند. برای معرفـی این رنگ‌ها ما از ثابت‌های نام‌دار خود کتابخانه curses اسـتفاده کرده‌ایم. همان‌گونه که مشـاهده می‌شـود، این ثابت‌ها از ترکیب curses.COLOR_ بـا نام رنگ مورد نظر (با حروف بزرگ) به وجود آمده‌اند.

تنهـا نکته جدیـد دیگر در ایـن برنامه قالب try…except…finally اسـت. ایـن سـاختار کنترلی معمولا بـرای جلوگیری از بوجود آمدن خطا در برنامه‌ها و متوقف شدن برنامه استفاده می‌شـود. فرم ساده اسـتفاده از این دستورات بدین صورت اسـت که پس از کلمـه کلیدی try مجموعه دسـتوراتی که ممکن است باعث بروز خطـا شـوند آورده می‌شـود. مجموعه کدهای مدیریت کننده خطا نیز پس از کلمه کلیدی except نوشته می‌شـود. در نهایت هم دستوراتی که در هـر صـورت (چه با بروز خطـا و چه بدون آن) باید اجرا شوند پس از کلمه کلیدی finally آورده می‌شود. مثال زیر نمونه‌ای از کاربرد این دستور را نشان می‌دهد:

```
a=input()
b= input()
try:
    c=a/b
except:
    print "division by zero ."
    c="Nothing"
finally:
    print c
```

در بخش‌های بعدی این مجموعه درباره این قالب به‌صورت مفصل توضیح خواهیم داد.

```
1  #! /usr/bin/env python
2  # ======= curses ========
3  import curses
4  # ======== Main Menu ============
5  def MainMenu():
6         myscreen.erase()
7         myscreen.border(0)
8         myscreen.addstr(8,10, "=============================
9  ======")
10        myscreen.addstr(9,10, "         Phone Book Menu")
11        myscreen.addstr(10,10, "=============================
12 =======")
13        myscreen.addstr(11,10, "1- Add new name to the
14 book",curses.color_pair(1))
15        myscreen.addstr(12,10, "2- Search for a name in the
16 book",curses.color_pair(2))
17        myscreen.addstr(13,10, "3- Delete a name from the
18 book",curses.color_pair(3))
19        myscreen.addstr(14,10, "0- EXIT",curses.color_pair(4))
20        myscreen.addstr(15,10, "=============================
21 =======»)
22        myscreen.addstr(16,10, "Your choice:   ")
23        myscreen.refresh()
24 # ======== Get Key ============
25 def GetKey():
26        key="X"
27        while key != ord('0'):
28               key = myscreen.getch(16,25)
29               #بخش کنترل کلید فشرده شده
30               if key == ord('1'):
31                      ## روال افزودن نام و شماره آورده شود
32                      pass
33               if key == ord('2'):
34                      ## روال جستوجو آورده شود
35                      pass
36               if key == ord('3'):
37                      ## روال حذف آورده شود
38                      pass
39 # ======= Main Part ========
40 myscreen = curses.initscr()
41 curses.start_color()
42 curses.init_pair(1, curses.COLOR_BLUE,curses.COLOR_WHITE)
43 curses.init_pair(2, curses.COLOR_GREEN,curses.COLOR_BLACK)
44 curses.init_pair(3, curses.COLOR_CYAN,curses.COLOR_MAGENTA)
45 curses.init_pair(4, curses.COLOR_RED,curses.COLOR_WHITE)
46 myscreen.border(0)
47
48 try:
49        MainMenu()
50        GetKey()
51
52 finally:
53        curses.endwin()
```

**فهرست ۲** کدهایی برای نمایش منوی رنگی یک نرم‌افزار فرضی دفتر تلفن

هر چنـد مثال‌های این بخش بسـیار سـاده و ابتدایـی هسـتند، اما شـما را با اصـول اولیه آرایـش متن‌ها در صفحه و اسـتفاده از رنگ‌ها و... آشـنا می‌کنـد. شـما هم به احتمال بسـیار پیش‌تـر به ایـن نتیجه رسـیده‌اید که داشـتن توانایـی ایجاد چنین رابط‌هـای کاربری آن هم در زبان‌های اسکریپت‌نویسـی مانند پایتون که به‌صورت معمول از رابط‌های گرافیکی اسـتفاده نمی‌کنند بسـیار حیاتی و لازم اسـت و کتابخانه curses بـرای شـروع و حتـی در حالتـی که با برنامه‌های حرفه‌ای سروکار دارد گزینه مناسبی خواهد بود.