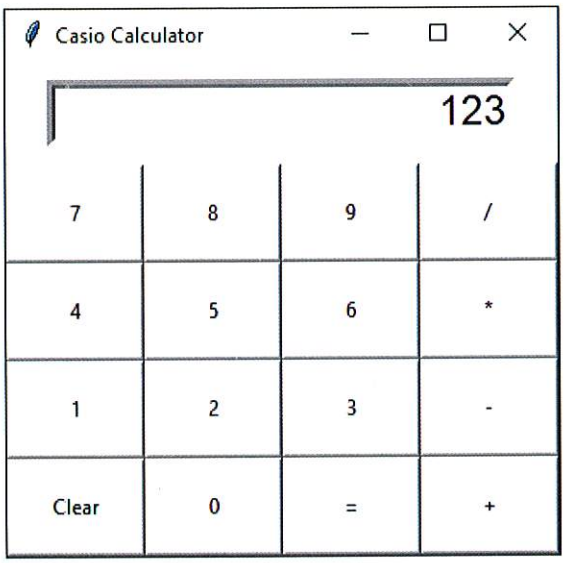
CHƯƠNG III

Máy tính bỏ túi Casio

Trong chương này, chúng ta sẽ tạo một chiếc máy tính bỏ túi giống như chiếc máy tính cầm tay Casio quen thuộc. Máy tính thực hiện được các phép tính cộng, trừ, nhân chia đơn giản. Qua sản phẩm này, các bạn sẽ biết cách lập trình một ứng dụng có các thành phần như nút bấm, ô nhập văn bản…

Chúng ta sẽ trải qua các bước sau:

- Tìm hiểu các đối tượng cửa sổ, nút bấm, ô văn bản… và lập trình thiết kế giao diện chương trình.
- Lập trình nhận sự kiện bấm nút.
- Lập trình tính toán và hiển thị kết quả.

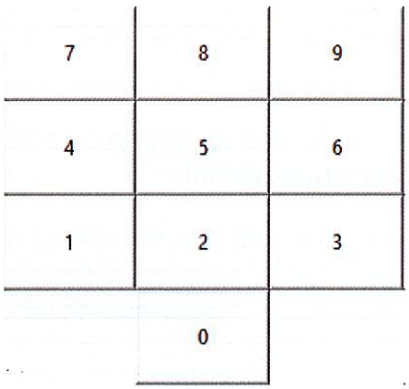


Tổng quan chương trình

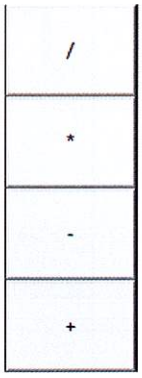
Chương trình gồm các thành phần chính:

1. Ô hiển thị văn bản (hộp văn bản – Textbox): Hiển thị các chữ số khi nút chữ số được nhấn.

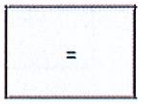
2. Các nút chữ số: Khi nhấn nút, chữ số sẽ được thêm vào hộp văn bản.



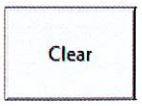
3. Các nút phép tính: Thực hiện các phép tính cộng, trừ, nhân chia tương ứng.



4. Nút dấu bằng: Hiển thị kết quả phép tính trên hộp văn bản.



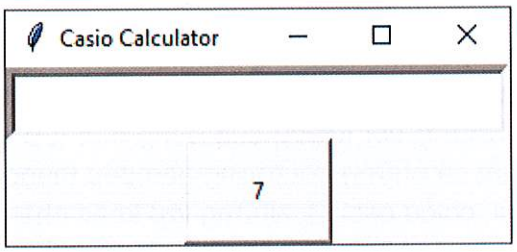
5. Nút xóa: Xóa toàn bộ hộp văn bản.



BÀI 1 GIỚI THIỆU VÀ TÌM HIỂU THÀNH PHẦN GIAO DIỆN

Trong bài học này, chúng ta sẽ tìm hiểu về khái niệm lập trình hướng đối tượng và thực hành lập trình tạo cửa sổ, hộp văn bản (Textbox) và nút bấm bằng đoạn chương trình:

```
1  from tkinter import *

2  wd = Tk()
3  wd.title("Casio Calculator")
4  e = Entry(width=20, borderwidth=5, font="Arial 18")
5  e.grid()
6  bt_7 = Button(text="7", width=10, height=3)
7  bt_7.grid()
8  wd.mainloop()
```



Kết quả của bài học

1.1. Giới thiệu về lập trình hướng đối tượng

Trong chương II, các bạn đã lập trình sản phẩm Lái tàu vũ trụ theo từng bước:

- Khởi tạo chương trình.
- Thiết lập cửa sổ, tiêu đề.
- Thiết lập các hình ảnh và âm thanh.
- Thiết lập các thông số tọa độ, điểm, trạng thái.
- Liên tục kiểm tra sự kiện phím được nhấn.
- Liên tục vẽ hình ảnh lên một vị trí tọa độ nhất định.
- ...

Kỹ thuật lập trình như vậy được gọi là "lập trình hướng thủ tục" (Procedure Oriented Programming), chương trình được chia thành các hàm. Với kỹ thuật này, chúng ta đơn thuần quan tâm đến việc chia công việc thành các phần nhỏ hơn và sử dụng các hàm để thực hiện từng công việc cụ thể (hàm có thể tự tạo hoặc sử dụng từ thư viện có sẵn), dữ liệu được sử dụng chung trong cả chương trình và dùng chung giữa các hàm. Với những chương trình lớn và phức tạp, lập trình hướng thủ tục bộc lộ nhiều nhược điểm về tính bảo mật dữ liệu và tái sử dụng mã lệnh, mặc dù các hàm

có thể tái sử dụng nhưng nếu thay đổi cấu trúc dữ liệu thì có thể phải thay đổi thuật giải và cả hàm.

Thuật ngữ "lập trình hướng đối tượng" (Object-Oriented Programming) chỉ một kỹ thuật lập trình khác, trong đó lập trình viên sẽ coi chương trình hoàn toàn được cấu thành từ các đối tượng (hay còn gọi là các thành phần, các nhân vật...), bất kỳ công việc nào cũng đều gắn với một hoặc nhiều đối tượng nào đó. Cách tiếp cận này mô phỏng thực tế: Mọi thứ trong cuộc sống của chúng ta đều là các đối tượng và những đối tượng này đều thuộc một loại nào đó. Ví dụ những cái bàn trong phòng chúng ta được gọi chung là "bàn", tên gọi chung mà con người đặt này là loại của đối tượng (còn gọi là lớp (**class**) của đối tượng), còn cụ thể chính cái bàn trong phòng mà chúng ta nhìn thấy, cầm nắm, chạm tay vào được là một cái bàn cụ thể, được gọi là một đối tượng (**object**).

Trong lập trình hướng đối tượng, chúng ta thường gặp hai khái niệm quan trọng: **Thuộc tính** và **phương thức**:

- Thuộc tính (**attribute**) là những thông tin, đặc điểm của đối tượng (ví dụ cái bàn sẽ có những thuộc tính chiều dài, chiều cao, cân nặng, chất liệu, màu sắc, năm sản xuất...). Những đối tượng của cùng một lớp sẽ có các thuộc tính giống nhau (ví dụ tất cả những đối tượng bàn đều thuộc lớp "bàn" và đều có thuộc tính chiều dài, chiều cao...). Những lớp khác nhau có thể có các thuộc tính giống và khác nhau (ví dụ lớp "sách" cũng có thuộc tính chiều dài hay chiều rộng giống lớp "bàn" nhưng có các thuộc tính khác như số trang, nhà xuất bản...). Trong lập trình, thuộc tính của đối tượng được thể hiện là các biến nhớ lưu giá trị.
- Phương thức (**method**) là những hoạt động mà đối tượng đó có thể hoạt động được, ví dụ đối tượng con mèo có phương thức chạy, nhảy, nằm... Những đối tượng trong cùng một lớp sẽ có các phương thức giống nhau, lớp khác nhau sẽ có phương thức khác nhau. Trong lập trình, phương thức của đối tượng được thể hiện là các hàm.

Trong thực tế, mục đích chính của lập trình hướng đối tượng hướng đến sự bảo mật dữ liệu và tái sử dụng mã nguồn. Các lớp khác nhau sẽ có quy định các mức bảo mật cho từng loại thuộc tính hay phương thức của lớp, điều này rất phức tạp đối với mức độ hiện tại của các bạn. Mục tiêu của cuốn sách là giúp chúng ta sử dụng những lớp có sẵn trong thư viện để tạo ra các đối tượng, đồng thời nắm được cách tư duy chia chương trình thành các đối tượng và lập trình các chức năng xoay quanh các đối tượng đó mà chưa cần tự mình tạo ra một lớp hoàn chỉnh.

Python là ngôn ngữ hỗ trợ cả lập trình thủ tục và lập trình hướng đối tượng. Vì vậy, chúng ta có thể sử dụng các biến và hàm tự do (biến và hàm không thuộc về một đối tượng nhất định nào) đan xen với các đối tượng cụ thể khác. Trong sản phẩm máy tính bỏ túi Casio, chúng ta sẽ sử dụng cấu trúc lập trình tuần tự kết hợp với các đối tượng đồ họa có sẵn trong thư viện.

1.2. Tạo cửa sổ, hộp văn bản và nút bấm

Với sản phẩm Máy tính bỏ túi Casio, chúng ta sẽ sử dụng thư viện **tkinter**. Đây là bộ công cụ hỗ trợ tạo giao diện đồ họa người dùng (user interface), ví dụ như cửa sổ phần mềm, nút bấm, hộp nhập văn bản, đoạn văn bản, menu... Đây là thư viện chuẩn đã được cài đặt mặc định cùng Python và PyCharm nên chúng ta không cần cài đặt thêm bằng lệnh lệnh **pip install** mà có thể trực tiếp thêm vào chương trình.

Đầu tiên, các bạn tạo một dự án mới. Ở tệp mã nguồn dự án, các bạn thêm thư viện vào chương trình bằng lệnh **import tkinter** và khởi tạo một đối tượng là cửa sổ giao diện với tên tùy ý (ví dụ: **wd**) bằng hàm **Tk()**.

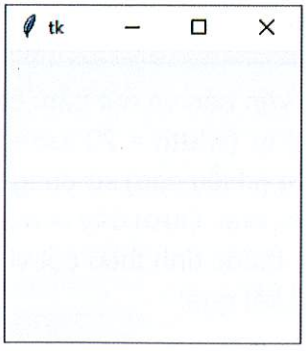
```
1  import tkinter
2  wd = tkinter.Tk()
```

Khi thêm thư viện như trên, để sử dụng hàm ta cần gọi theo cú pháp **<module>.<tên hàm>.** Tuy nhiên, khi cần sử dụng nhiều hàm khác nhau, điều này sẽ gây ra bất tiện, chúng ta có thể gọi trực tiếp tên hàm bằng cách sử dụng lệnh **from <module> import <tên hàm>**. Khi muốn gọi trực tiếp toàn bộ các hàm, chúng ta sử dụng dấu * thay cho tên hàm cụ thể. Khi đó, trong đoạn chương trình chúng ta có thể gọi trực tiếp tên hàm mà không cần gõ module phía trước nữa.

```
1  from tkinter import *

2  wd = Tk()  # không cần gọi module wd = tkinter.Tk()
```

Sau khi khởi tạo cửa sổ, chúng ta sử dụng phương thức **mainloop()** của đối tượng để có thể hiển thị giao diện lên trên màn hình. Các bạn lập trình và chạy thử chương trình sẽ thấy xuất hiện một cửa sổ. Cửa sổ này đã có sẵn một số tính năng cơ bản như các nút thu nhỏ, phóng to, đóng cửa sổ và có thể thay đổi kích thước.

```
1  from tkinter import *

2  wd = Tk()
3  wd.mainloop()
```

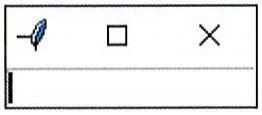


Các bạn thêm lệnh bổ sung tiêu đề cho cửa sổ, tự mình chạy thử và quan sát kết quả. Chú ý: Lệnh **wd.mainloop()** luôn được đặt ở vị trí cuối cùng chương trình.

```
1  from tkinter import *

2  wd = Tk()
   # thiết lập tiêu đề
3  wd.title("Casio Calculator")
4  wd.mainloop()
```

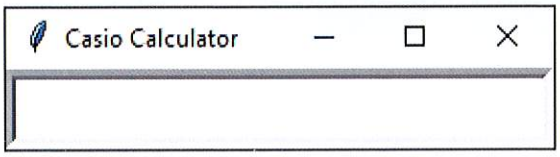
Với **tkinter**, khi chúng ta muốn thêm một thành phần nào đó vào giao diện, chúng ta cần 2 bước: Tạo đối tượng và "gắn" đối tượng đó lên cửa sổ. Hình dưới đây mô tả lệnh tạo đối tượng hộp văn bản **e = Entry()** và lệnh gắn đối tượng đó lên cửa sổ **e.grid()**.

```
1  from tkinter import *

2  wd = Tk()
3  wd.title("Casio Calculator")

4  e = Entry()  # tạo đối tượng hộp nhập văn bản
5  e.grid()  # gắn đối tượng lên cửa sổ

6  wd.mainloop()
```



Chúng ta có thể thiết lập các thuộc tính của hộp như **width** (chiều rộng), **height** (chiều cao), **font** (phông chữ), **borderwidth** (đường viền)... Các bạn thay đổi các thuộc tính bằng cách thêm tham số cho hàm khởi tạo.

```
4  e = Entry(width=20, borderwidth=10, font="Arial 18")
5  e.grid()
```



Các bạn lưu ý, đối tượng hộp văn bản và nút bấm có thuộc tính **width** (chiều dài) sử dụng đơn vị là số lượng các ký tự (**width** = 20 tương đương kích thước chiều dài bằng 20 ký tự) và thuộc tính **height** (chiều cao) sử dụng đơn vị là số dòng ký tự, ngoài ra các thuộc tính khác có đơn vị là pixel. Dưới đây là một số thuộc tính của lớp Entry, các bạn có thể thay đổi giá trị các thuộc tính theo cột ví dụ trong bảng bên dưới, sau đó chạy chương trình và quan sát kết quả:

Thuộc tính	Ý nghĩa	Ví dụ

background hoặc **bg**	Màu nền	`background='red'`
		`background='blue'`
		`bg='#00ffff'`

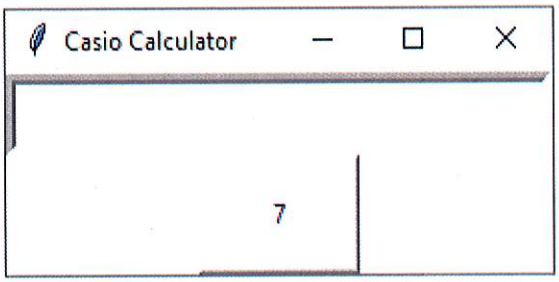
borderwidth hoặc **bd**	Độ dày đường viền	`borderwidth=10`
width	Chiều dài	`width=20`

font	Phông chữ	`font="Arial 18"`
		`font="Times 15 bold"`
		`font="Helvetica"`

foreground hoặc **fg**	Màu chữ	`fg="white"`
justify	Căn chỉnh chữ ở vị trí bên trái (LEFT), giữa (CENTER) hoặc bên phải (RIGHT)	`justify=RIGHT`

Tiếp theo, các bạn khởi tạo và thêm đối tượng nút bấm vào cửa sổ bằng lệnh **Button()**. Trên máy tính có tổng cộng 16 nút, vì vậy các bạn chú ý đặt tên đối tượng cho phù hợp, tránh nhầm lẫn. Ví dụ hình dưới đây khởi tạo và thêm đối tượng nút "7" vào cửa sổ với tên **bt_7**, nút này có chiều dài tương đương 10 ký tự và chiều cao tương đương 3 dòng. Các bạn lưu ý, thuộc tính cỡ chữ của Entry đã thay đổi, vì vậy kích thước ký tự của Entry sẽ khác so với kích thước ký tự của Button, đồng thời mặc định chiều dài đối tượng Button được cộng thêm một phần nhỏ khoảng cách từ văn bản hiển thị đến biên bên ngoài nút, vì vậy tỉ lệ chiều dài các đối tượng ta quan sát được không giống với tỉ lệ giá trị thuộc tính **width** của các đối tượng mà ta thiết lập.

```
6  bt_7 = Button(text="7", width=10, height=3)
7  bt_7.grid()

8  wd.mainloop()
```



Dưới đây là một số thuộc tính của lớp Button, các bạn tự mình thử nghiệm bằng cách thêm và thay đổi giá trị các thuộc tính sau, sau đó chạy chương trình và quan sát kết quả:

Thuộc tính	Ý nghĩa và một số giá trị	Ví dụ
background hoặc **bg**	Màu nền	`background='red'` `bg='#00ffff'`

borderwidth hoặc **bd**	Độ dày đường viền	`borderwidth=10`
width	Chiều dài	`width=20`
height	Chiều cao	`height=3`

font	Phông chữ	`font="Arial 18"`
		`font="Times 15 bold"`
		`font="Helvetica"`

foreground hoặc **fg**	Màu chữ, ví dụ "white", "black", "red", "green", "blue", "cyan", "yellow", "magenta", "gray"...	`fg="white"`
command	Hàm sẽ thực hiện khi nút được nhấn, trong ví dụ bên thì **addition** là tên một hàm.	`command=addition`
padx	Khoảng trống được đệm thêm vào bên trái và bên phải của chữ hiển thị trên nút, đơn vị pixel. Trong ví dụ bên, dòng trên thiết lập khoảng trống của bên trái và phải bằng nhau và bằng 20 pixel, dòng dưới thiết lập khoảng trống bên trái bằng 15 pixel, bên phải bằng 5 pixel.	`padx=20` `padx=(15, 5)`
paxy	Khoảng trống được đệm thêm vào bên trên và bên dưới của chữ hiển thị trên nút, đơn vị pixel. Trong ví dụ bên, dòng trên thiết lập khoảng trống của bên trên và dưới bằng nhau và bằng 20 pixel, dòng dưới thiết lập khoảng trống bên trên bằng 15 pixel, bên dưới bằng 5 pixel.	`pady=10` `pady=(15, 5)`

Vậy là chúng ta đã tìm hiểu về thành phần giao diện gồm hộp văn bản và nút bấm. Trong bài học sau, chúng ta sẽ tiếp tục thêm các nút và sắp xếp vị trí các nút trên cửa sổ phần mềm.

Tóm tắt lý thuyết và bài tập thực hành

Trong bài học này, chúng ta đã cùng nhau tìm hiểu về khái niệm lập trình hướng đối tượng. Bằng cách sử dụng thư viện **tkinter**, các bạn đã lập trình tạo hộp văn bản bằng lệnh **Entry()**, tạo nút bấm bằng lệnh **Button()** và gắn các đối tượng này lên cửa sổ bằng lệnh **grid()**.

Bài tập 1. Thử nghiệm thêm các nút và các hộp văn bản khác vào cửa sổ. Nhận xét vị trí và cách sắp xếp của các đối tượng sau khi thêm.

Bài tập 2. Thay đổi phông chữ, màu chữ, màu nền của nút theo ý thích. Tham khảo giá trị các thuộc tính đã được mô tả trong bài học.