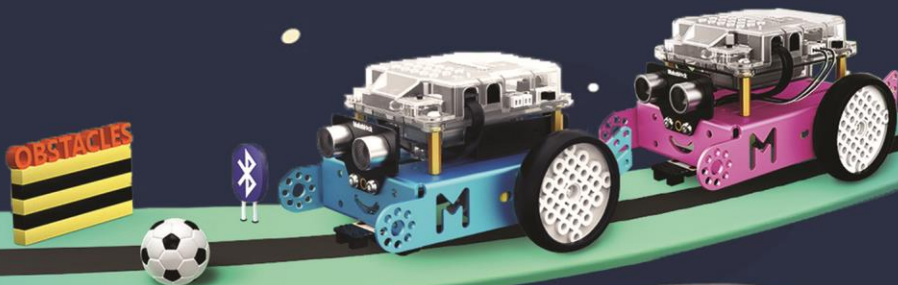
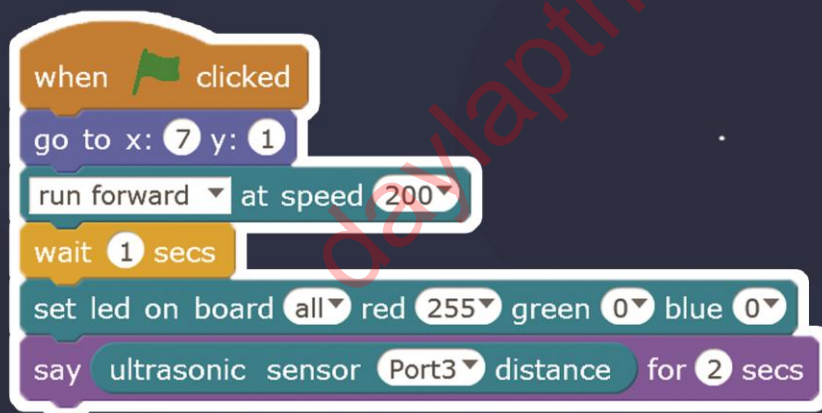


Dịch và biên soạn: Nguyễn Hữu Hưng - Dương Lực

# LẬP TRÌNH ĐIỀU KHIỂN ROBOT với SCRATCH (Hành trang cho tương lai)



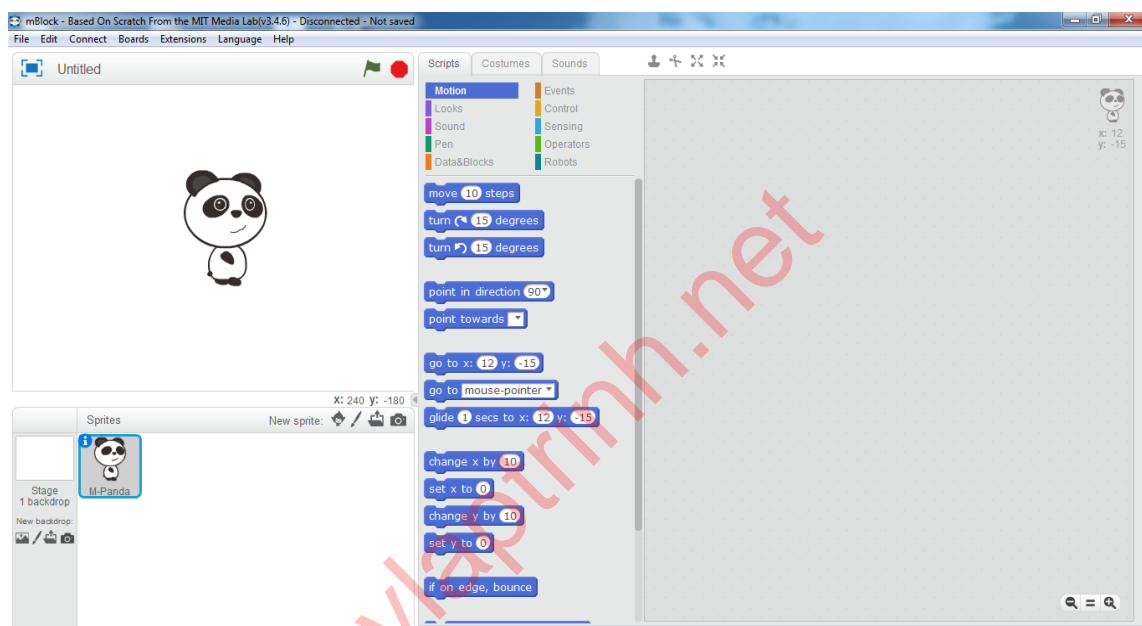
# Mục lục

<b>Lập trình robot với mBlock .....</b>	<b>1</b>
Giới thiệu .....	3
Bài 1. Chuột yêu táo .....	7
Bài 2. Hứng táo .....	11
Bài 3. Thách thức số học .....	16
Bài 4. Phòng đoán .....	23
Bài 5. Bảo vệ đảo .....	29
Bài 6. Quét mã vạch .....	37
Bài 7. Trò chơi nhịp điệu .....	43
Bài 8. Video Ball .....	47
Bài 9. Chạy nào ! Robot .....	53
Bài 10. Bậc thầy né tránh .....	59
Bài 11. Con đường thành công .....	65
Bài 12. Robot ngoan ngoãn .....	72
Bài 13. Robot linh hoạt .....	76
Bài 14. Nhà vô địch đua xe .....	79

## Giới thiệu

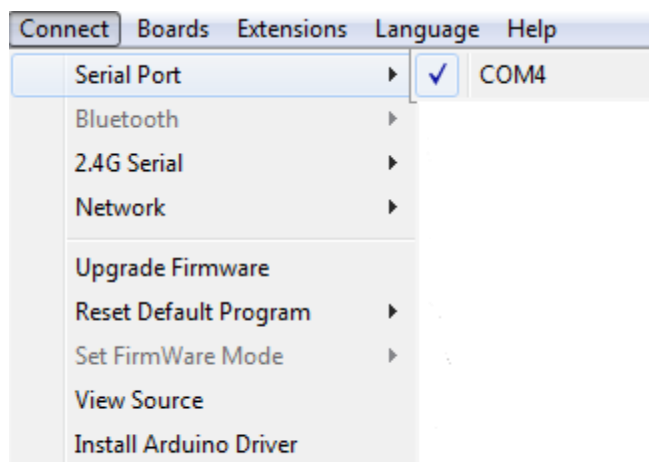
Các bạn sẽ được trải nghiệm sự tương tác giữa thế giới vật lý và phần mềm thông qua cuốn sách. Cuốn sách bao gồm 3 phần: robot, bảng mạch và **mBlock** (bạn có thể tải phần mềm **mBlock** tại địa chỉ <http://mblock.cc/download>). **mBlock** là một phần mềm được phát triển trên cơ sở Scratch 2.0. Nó có thể điều khiển bảng mạch của robot và thực hiện các chức năng tương ứng.

Dưới đây là giao diện của phần mềm **mBlock**:

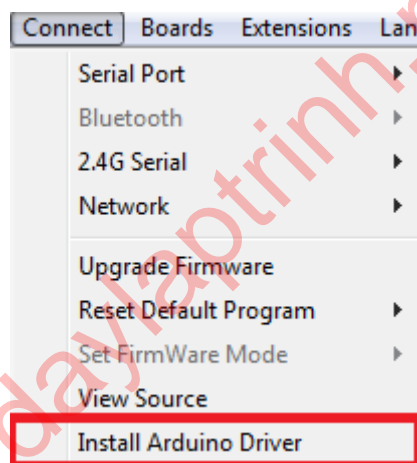


Khi **mBlock** và bảng mạch được kết nối với nhau, các bạn có thể điều khiển robot bằng phần mềm **mBlock**. Dưới đây là các bước để kết nối **mBlock** với robot.

1. Cắm dây nối USB tới máy tính và bảng mạch. mở phần mềm **mBlock** và chọn cổng COM.  
Sau khi nhấn chuột, mục **Connect** sẽ chuyển thành "COM Connected".



2. Cài đặt Arduino Driver: chọn **Connect** ⇒ **Install Arduino Driver** sau đó nhấn **Install** để cài đặt.



3. Tìm số cổng COM bằng cách nhấn chuột phải vào **MyComputer** ở ngoài Desktop, sau đó chọn **Manager** ⇒ **Device Manager** ⇒ **Port(COM & LPT)**.
4. Cập nhật (Update) Firmware: Bên danh mục **Board** chọn **mBot** ; Bên danh mục **Connect** chọn **Update Firmware**.

Tuy nhiên khi kết nối máy tính và **mBot** bằng dây cáp USB, thì sẽ hạn chế khoảng cách di chuyển của robot, do đó để robot di chuyển được khoảng cách xa hơn, các bạn có thể kết nối bằng module Bluetooth hoặc module 2.4G.

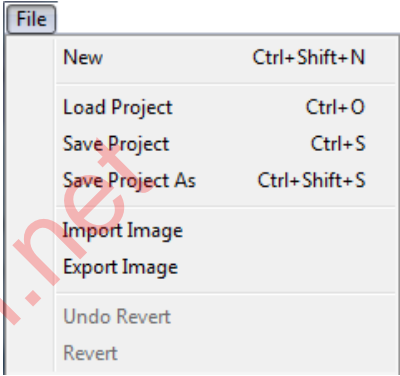
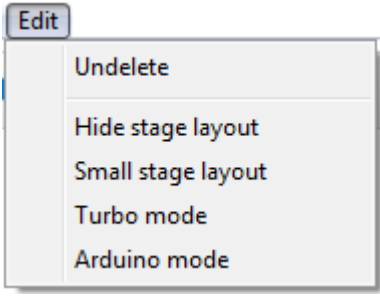
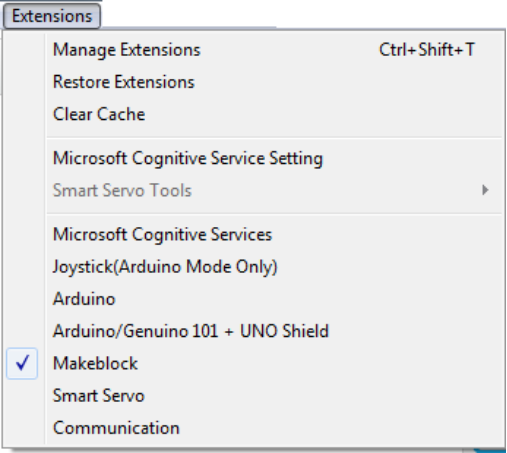
## Bluetooth

1. Mở Bluetooth trên máy tính (trong trường hợp cần trợ giúp, truy cập trang web [www.daylaptrinh.net](http://www.daylaptrinh.net) để nhận được hỗ trợ trực tuyến).
2. Ngắt kết nối **mBlock** với **mBot**: nhấn lên COM6
3. Chọn **Connect** ⇒ **Bluetooth** ⇒ **Discover**. Khi danh sách Bluetooth hiện lên, chọn Bluetooth tương ứng

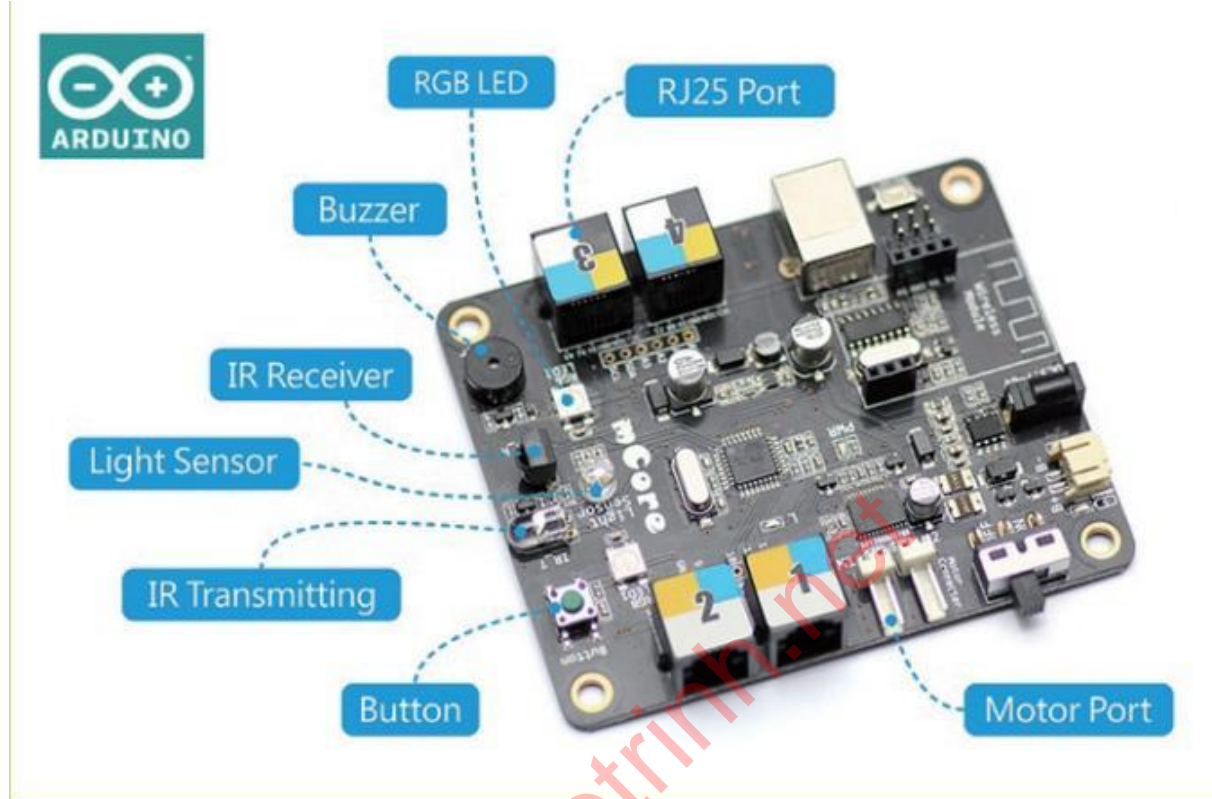
## 2.4G

1. Cắm module 2.4G trên máy tính.
2. Nhấn lên mục **Connect** -> **2.4G** -> **Connect** để kết nối với robot.

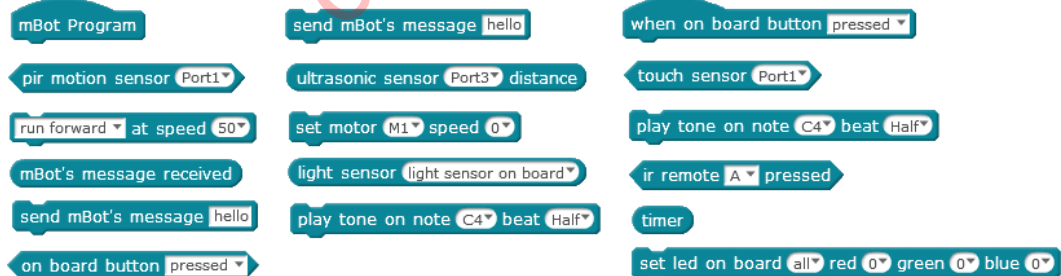
Một số chức năng hay được sử dụng trên thanh Danh mục:

<p><b>New:</b> Tạo một dự án mới</p> <p><b>Load Project:</b> Mở 1 dự án trên máy tính</p> <p><b>Save Project:</b> Lưu dự án hiện tại</p> <p><b>Save Project as:</b> Lưu dự án dạng bản sao</p> <p><b>Import Image:</b> thêm hình ảnh vào thư viện</p> <p>Lưu ý: Dự án sẽ được lưu dưới tệp tin có định dạng <b>.sb2</b></p>	
<p><b>Undelete:</b> lấy lại đoạn khối lệnh mới xóa</p> <p><b>Hide stage layout:</b> ẩn sân khấu</p> <p><b>Small stage layout:</b> sân khấu nhỏ</p> <p><b>Turbo mode:</b> chế độ chạy cực nhanh các khối lệnh)</p> <p><b>Arduino mode:</b> chuyển đổi Scripts của <b>mBlock</b> vào chương trình Arduino và tải lên bảng mạch của Arduino để có thể hoạt động offline.</p>	
<p>Lựa chọn trong mục <b>Extensions</b> sẽ liên quan tới các khối lệnh hiển thị trong nhóm lệnh lập trình cho Robots.</p> <p><b>Arduino:</b> sẽ hiển thị các khối lệnh tương thích với drduino</p> <p><b>Makeblock:</b> các khối lệnh tương thích với bảng mạch.</p> <p><b>Communication:</b> Cung cấp các chức năng kết nối mạng Lan</p>	

Những thành phần trên bảng mạch **mBot**:



Chọn **makeBlock** trong danh mục **extensions**, khi đó bên nhóm lệnh **Robots** các bạn sẽ thấy các khối lệnh có thể điều khiển được bằng mạch.





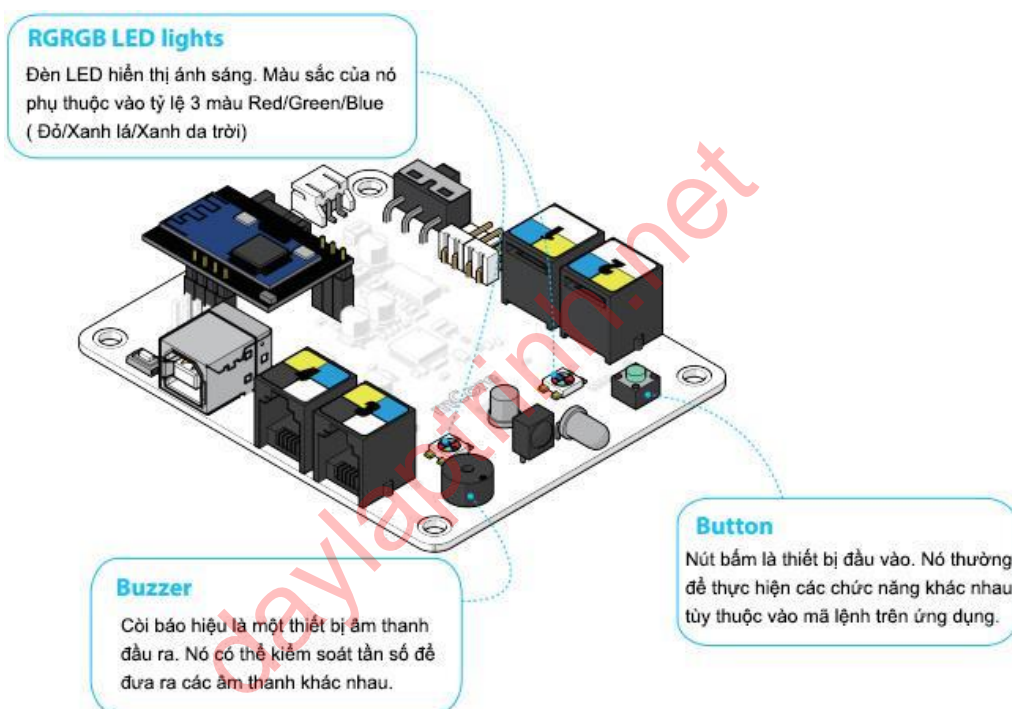
## Bài 1. Chuột yêu táo

Sử dụng bảng mạch để điều khiển chú chuột trong dự án trên máy tính tiến tới và ăn được táo.






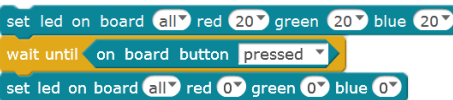




### Kiến thức lập trình:

1. Học cách điều khiển đèn LED.
2. Học cách điều khiển âm thanh của còi báo hiệu.

### Module điện tử



## Kiến thức bổ sung

Khối lệnh	Mô tả	Ví dụ
	Sự kiện bắt đầu và kích hoạt chương trình. Chức năng của khối lệnh là bắt đầu chương trình khi lá cờ xanh được nhấn.	
	Đợi 1 giây. Số giây có thể là 1 số nguyên hoặc số thập phân bất kỳ.	
	Chương trình sẽ đợi tới khi nút bấm <b>Button</b> trên robot được nhấn. Khối lệnh màu xanh có thể được thay thế bởi khối lệnh lục giác khác.	
	Thiết lập màu của đèn LED. Dãy màu sẽ nằm trong khoảng từ 0 - 255. Đèn sẽ tắt khi cả 3 màu = 0.	
	Còi báo hiệu có thể phát ra âm thanh từ C2 - D8 trong khoảng thời gian nhất định.	



## Cấu trúc lập trình

Cấu trúc lập trình	Lưu đồ
<p><b>Cấu trúc lập trình tuần tự:</b></p> <p>Đoạn khối lệnh bắt đầu chạy từ khối lệnh đầu tiên, tiếp theo tất cả các khối lệnh phía dưới được thực hiện theo thứ tự. Đó là cấu trúc tuần tự.</p> <p>Biểu đồ bên phải là một cấu trúc tuần tự cơ bản.</p> <p>Sau khi chương trình bắt đầu, nó thực hiện lần lượt 3 khối lệnh A, B, C, và cuối cùng kết thúc đoạn khối lệnh. Cấu trúc tuần tự là cách chạy cơ bản của một chương trình.</p>	<pre> graph TD     A([Bắt đầu]) --&gt; B[Khối lệnh A]     B --&gt; C[Khối lệnh B]     C --&gt; D[Khối lệnh C]     D --&gt; E([Kết thúc])         </pre>

## Lưu ý

Giá trị được thiết lập trong các khối lệnh sẽ thay đổi trạng thái của thiết bị điện tử.

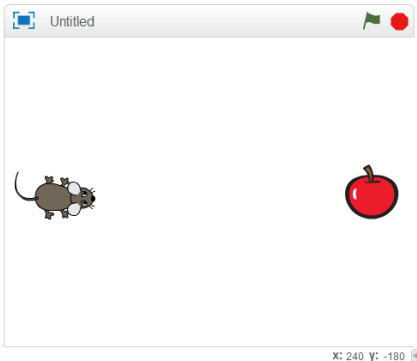
## Thực hành

<p>Viết thứ tự chạy của đoạn khối lệnh bên cạnh:</p> <ol style="list-style-type: none"> <li>Bắt đầu chương trình( Khi lá cờ xanh được nhấn).</li> <li>Đèn LED được bật lên 1 giây</li> </ol>	<pre> when green flag clicked   set led all red 20 green 0 blue 0   wait 1 secs   set led all red 0 green 0 blue 0   wait 1 secs   play tone on note C4   wait until button pressed   stop tone         </pre>
--	--

## Lập trình

Trong chương trình, chú chuột muốn ăn quả táo, do đó chương trình sẽ có 2 đối tượng: **Chuột** và **Táo**. Khi chương trình bắt đầu, bảng mạch sẽ phát ra tiếng còi và đợi

tới khi bạn nhấn nút trên bảng mạch. Khi nút đã được nhấn, chú chuột sẽ di chuyển tới quả táo. Khi chuột chạm vào quả táo, còi trên bảng mạch sẽ phát ra âm thanh.

Sân khấu		<b>Mô tả:</b> <i>Chuột</i> di chuyển về phía quả táo, khi chạm vào quả táo thì robot sẽ phát ra âm thanh
<i>Chuột</i>	<pre> when space key pressed   go to x: -185 y: -5   play tone on note C4 beat Half   wait until on board button pressed   glide 1 secs to x: 128 y: -7         </pre>	Điều khiển chuột di chuyển tới vị trí quả táo <ul style="list-style-type: none"> <li>- Nhấn phím cách để bắt đầu chương trình</li> <li>- Chuột nhảy tới vị trí bắt đầu</li> <li>- Còi chạy âm thanh C4</li> <li>- Đợi 0.5 giây</li> <li>- Ngừng âm thanh</li> <li>- Đợi tới khi <b>Button</b> được nhấn</li> <li>- <i>Chuột</i> di chuyển tới vị trí quả táo</li> </ul>
<i>Táo</i>	<pre> when space key pressed   wait until touching Mouse1 ?   play tone on note C4 beat Half   play tone on note F2 beat Half         </pre>	Còi kêu khi <i>Chuột</i> chạm vào <i>Táo</i> <ul style="list-style-type: none"> <li>- Nhấn phím cách để bắt đầu chương trình</li> <li>- Đợi tới khi chạm vào chuột</li> <li>- Còi kêu âm thanh C4</li> <li>- Đợi 0.2 giây</li> <li>- Còi kêu âm thanh F2</li> <li>- Đợi 0.2 giây</li> <li>- Ngừng âm thanh</li> </ul>

## Bài tập

Sử dụng **Button** trên mCore để thay đổi màu sắc cho đèn LED để thêm hiệu ứng cho đèn gồm các màu: Tím, vàng và trắng.

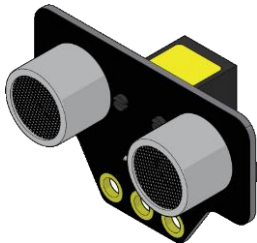
## Bài 2. Hứng táo

Chú chuột tìm thấy một cây táo. Bây giờ cũng là mùa táo chín. Táo sẽ rơi từ trên cây xuống khi có gió thổi. Chú chuột hy vọng sẽ thu được 1 thùng táo mang về.

### Kiến thức lập trình




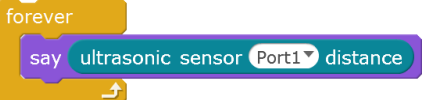
1. Học cách điều khiển và sử dụng cảm biến siêu âm.

### Modules điện tử

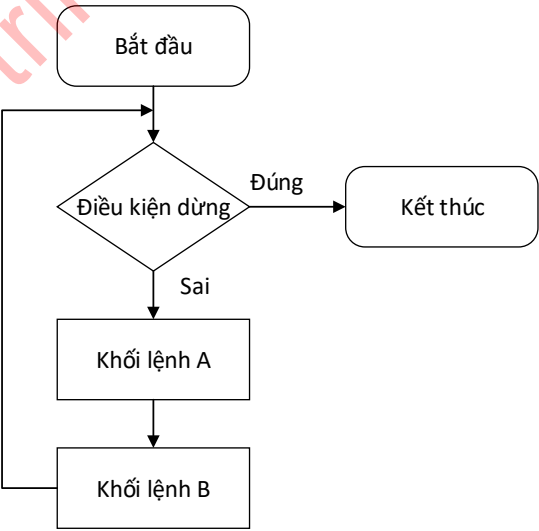
Tên	Chức năng	Chỉ dẫn
Cảm biến siêu âm (ultrasonic sensor) 	Cảm biến siêu âm là một thiết bị đầu vào để đo khoảng cách. Nó có 2 “mắt”, một để phát sóng siêu âm, mắt còn lại để thu sóng siêu âm bật trở lại khi gặp vật cản. Từ đó tính ra khoảng cách tới vật cản đó.  Phạm vi phát hiện: 3 – 400 cm. Góc phát hiện: 30 độ	Cảm biến siêu âm được đánh dấu với nhãn màu vàng, vì vậy cần được kết nối với logo màu vàng trên bảng mạch

### Kiến thức bổ sung

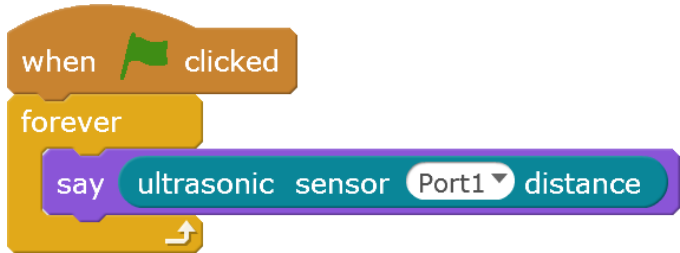
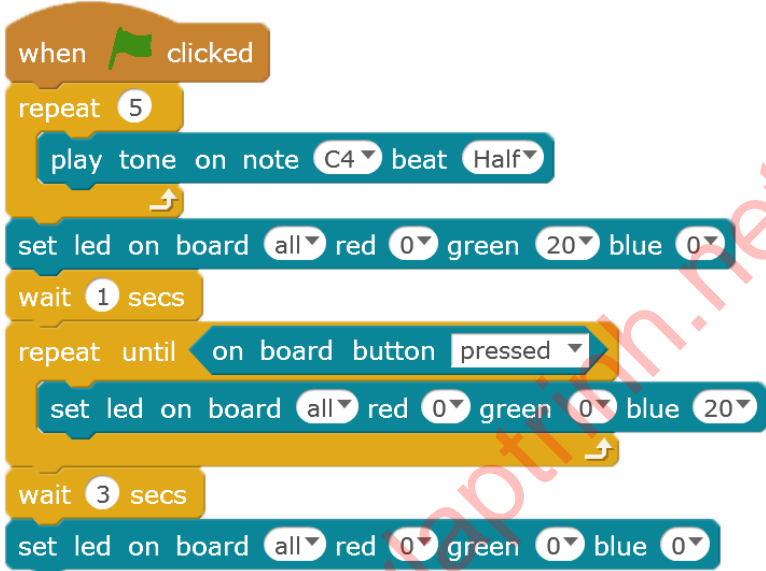
Khối lệnh	Chỉ dẫn	Ví dụ
	Lặp lại đoạn khối lệnh bên trong với số lần nhất định, và sau đó thực hiện khối lệnh bên dưới (nếu có).	
	Lặp lại cho tới khi điều kiện dừng đúng (điều kiện là ô lục giác đang trống).  Khi điều kiện sai, đoạn khối lệnh bên trong sẽ được lặp đi lặp lại, chỉ	

	khi điều kiện đúng mới dừng vòng lặp, sau đó thực hiện khối lệnh bên dưới (nếu có)	
	Lặp lại mãi mãi: Đoạn khối lệnh bên trong sẽ liên tục được lặp đi lặp lại, và hành động lặp chỉ dừng lại khi dừng chương trình.	
	Khối lệnh cảm biến siêu âm: để trả về khoảng cách giữa mắt cảm biến và vật cản phía trước nó.	

## Cấu trúc lập trình



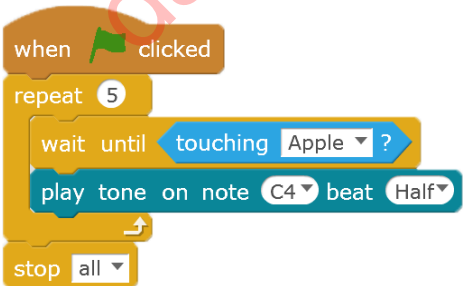
Cấu trúc lập trình	Lưu đồ
<p><b>Cấu trúc vòng lặp:</b></p> <p>Cấu trúc vòng lặp là một cấu trúc lặp lại đoạn khối lệnh bên trong nó.</p> <p>Như biểu đồ phía bên, khối lệnh A và khối lệnh B có bên trong vòng lặp. Nếu điều kiện dừng sai thì sẽ thực hiện tiếp vòng lặp, nếu điều kiện dừng đúng thì vòng lặp sẽ dừng lại. Trong lập trình, khi cần thực hiện lặp lại cùng 1 đoạn khối lệnh ta sẽ sử dụng vòng lặp.</p>	 <pre> graph TD     Start([Bắt đầu]) --&gt; Decision{Điều kiện dừng}     Decision -- Đúng --&gt; End([Kết thúc])     Decision -- Sai --&gt; A[Khối lệnh A]     A --&gt; B[Khối lệnh B]     B --&gt; Decision </pre>

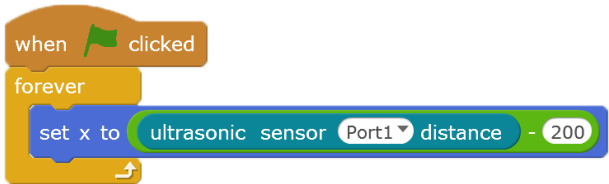
## Thực hành

Đoạn khối lệnh	Giải thuật
 <pre> when green flag clicked   forever loop     say ultrasonic sensor Port1 distance </pre>	Viết thứ tự khối lệnh theo cấu trúc giải thuật. Chỉ ra khối lệnh nào nằm bên trong vòng lặp.
 <pre> when green flag clicked   repeat 5     play tone on note C4 beat Half   set led on board all red 0 green 20 blue 0   wait 1 secs   repeat until on board button pressed     set led on board all red 0 green 0 blue 20   wait 3 secs   set led on board all red 0 green 0 blue 0 </pre>	Lập trình như đoạn khối lệnh bên trái và theo dõi sự thay đổi.

## Lập trình

Bằng cách sử dụng cấu trúc vòng lặp và kiến thức cấu trúc tuần tự đã học, hãy lập trình cho chuột thu được thật nhiều táo. Thông qua kiến thức về vòng lặp, lập trình để điều khiển sự di chuyển của chú chuột thông qua cảm biến siêu âm.

<p><b>Sân khấu</b></p>		<p>Khi chuột di chuyển về phía trước và chạm vào quả táo, bằng mạch sẽ phát tiếng còi</p>
<p>Nhân vật đối tượng <i>Táo</i></p>		<p><b>Táo rơi</b></p> <ul style="list-style-type: none"> <li>- Vòng lặp thực hiện các khối lệnh bên trong.</li> <li>- Quả táo hiện lên.</li> <li>- Nhảy tới vị trí ban đầu ở trên cây.</li> <li>- Lặp lại việc rơi xuống của quả táo tới khi chạm vào biên.</li> <li>- Sau khi chạm biên, ẩn Táo đi.</li> </ul>
<p>Đối tượng nhân vật <i>Chuột</i></p>		<p><b>Dừng chương trình sau khi chuột thu được 5 quả táo</b></p> <ul style="list-style-type: none"> <li>- Vòng lặp 5 lần các khối lệnh bên trong</li> <li>- Đợi tới khi <i>Chuột</i> chạm vào <i>Táo</i></li> <li>- Âm thanh C4 được phát trong 0.5 giây</li> <li>- Ngừng chương trình sau khi thu đủ 5 quả táo</li> </ul>

<p>Đối tượng nhân vật <i>Chuột</i></p>		<p>Sử dụng bảng mạch cùng với cảm biến siêu âm để điều khiển di chuyển của chuột</p> <p>- Bạn cần sử dụng vòng lặp để liên tục cập nhật vị trí của chuột thông qua cảm biến siêu âm</p>
--	---	---

## Bài tập

1. Sử dụng vòng lặp để điều khiển đèn RGB trên bảng mạch. Mỗi lần sáng trong 1s và chỉ ra đâu là đoạn lệnh bên trong vòng lặp.
2. Lập trình để còi lặp đi lặp lại công việc kêu âm thanh C4 10 lần, E5 20 lần, B6 30 lần.

daylaptrinh.net




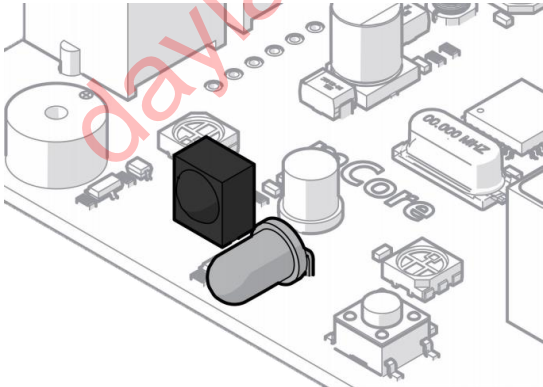
### Bài 3. Thách thức số học

Chú khi đưa ra một số bất kỳ, và yêu cầu dơi phải đạt được số điểm tương tự trong 30 giây.

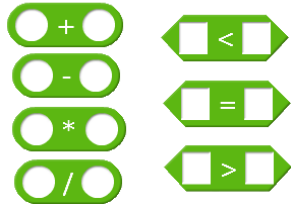





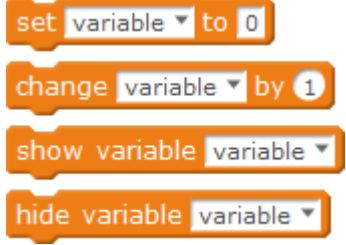

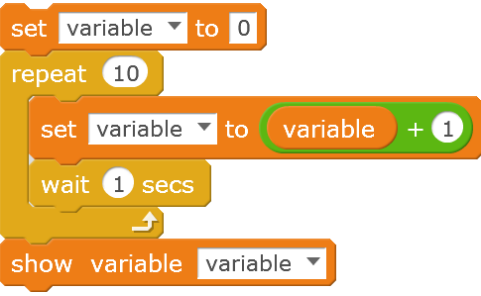
#### Kiến thức lập trình


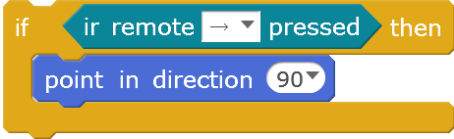
1. Sử dụng phép toán.
2. Sử dụng biến.

#### Module điện tử

Tên	Hình ảnh	Chỉ dẫn
Điều khiển từ xa hồng ngoại		Gửi thông tin qua mắt hồng ngoại tới module nhận hồng ngoại. Sau đó xử lý tín hiệu nhận được trong chương trình.
Module hồng ngoại		Module hồng ngoại trên bảng mạch của robot có thể nhận và gửi thông tin đi. Thông tin có thể là số và văn bản.

## Kiến thức bổ sung

Khối lệnh	Chỉ dẫn	Ví dụ
<p><b>Operators</b></p> 	<p>4 phép tính bao gồm cộng, trừ, nhân, chia. Bạn có thể điền giá trị hoặc kéo biến vào.</p> <p>Phép toán so sánh có thể được sử dụng để so sánh các giá trị với biến, so sánh biến với biến hay giá trị với giá trị.</p> <p>Đầu vào của cảm biến cũng có thể được sử dụng như 1 biến</p>	 <p>Giá trị của phím điều khiển chia cho 10.</p>  <p>Đợi tới khi khoảng cách mà cảm biến siêu âm đo được &lt; 30cm, sau đó thực hiện khối lệnh tiếp theo.</p>  <p>So sánh 2 biến có bằng nhau hay không. Nếu không bằng sẽ tiếp tục đợi.</p>
	<p>Khối lệnh lấy số ngẫu nhiên, có thể điền giá trị hoặc kéo biến ghép vào khối lệnh</p>	 <p>Xoay về phía ngẫu nhiên mỗi giây 1 lần.</p>
<p><b>Data&amp;Blocks</b></p> <p><b>Make a Variable</b></p> <p><input checked="" type="checkbox"/> variable</p> 	<p>Biến dùng để lưu trữ dữ liệu. Giá trị của biến có thể thay đổi được</p>	 <p>Dữ liệu cảm biến được lưu vào trong biến</p>  <p>Biến <b>variable</b> nhận giá trị ban đầu là 0, mỗi một giây biến <b>variable</b> sẽ tăng thêm 1</p>

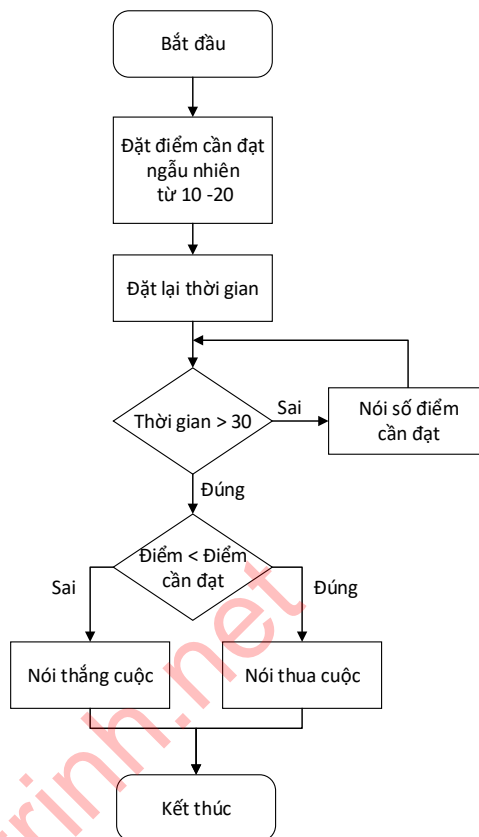
	<p>Nếu điều kiện đúng thì sẽ thực hiện các khối lệnh bên trong</p>	 <p>Nếu nhấn mũi tên sang phải trên điều khiển từ xa, đối tượng sẽ xoay sang bên phải.</p>
---	--	--

## Ý tưởng lập trình

Mô tả ý tưởng	Lưu đồ
<p><b>Đối tượng Bóng điểm</b> Xuất hiện ở vị trí ngẫu nhiên, nếu chạm vào <i>Dơi</i> thì tăng <b>Điểm</b> thêm 1.</p> <p><b>Đối tượng Bóng zero</b> Xuất hiện ở vị trí ngẫu nhiên, nếu chạm vào dơi thì đặt <b>Điểm</b> bằng 0.</p>	<div style="display: flex; justify-content: space-around;"> <div style="text-align: center;"> <p><b>Bóng điểm</b></p> <pre> graph TD     A([Bắt đầu]) --&gt; B[Đặt Điểm = 0]     B --&gt; C[Đặt vị trí ngẫu nhiên]     C --&gt; D[Hiện]     D --&gt; E{Chạm vào Dơi}     E -- Sai --&gt; D     E -- Đúng --&gt; F[Tăng điểm thêm 1]     F --&gt; G[Ẩn]     G --&gt; H[Đợi 1 giây]     H --&gt; C           </pre> </div> <div style="text-align: center;"> <p><b>Bóng zero</b></p> <pre> graph TD     A([Bắt đầu]) --&gt; B[Đặt vị trí ngẫu nhiên]     B --&gt; C[Hiện]     C --&gt; D{Chạm vào Dơi}     D -- Sai --&gt; C     D -- Đúng --&gt; E[Đặt Điểm = 0]     E --&gt; F[Ẩn]     F --&gt; G[Đợi ngẫu nhiên 3 - 10 giây]     G --&gt; B           </pre> </div> </div>

### Đối tượng Khi

Đưa ra số điểm cần đạt, khi hết thời gian thì so sánh xem đã đạt đủ điểm chưa.



### Thực hành

Đoạn khối lệnh	Giải thuật
<pre> wait until key space pressed? go to x: -157 y: -127 set y to -120 set position to -200 set pen color to blue set pen size to 10 pen down change x by position + 100 * -2 pen up           </pre>	<p>Viết giải thuật của đoạn khối lệnh bên trái.</p>

```

forever
  if on board button pressed then
    change LED by 1
  else
    change LED by -1
  set led on board all red LED green LED blue LED

```

Chạy thử đoạn khối lệnh bên trái và viết lại sự thay đổi của bảng mạch.

## Lập trình

Đối tượng nhân vật *Dơi*.



Sử dụng điều khiển hồng ngoại để điều khiển *Dơi*.

```

when green flag clicked
  forever
    if ir remote E pressed then
      point in direction 45
    if ir remote F pressed then
      point in direction 135
    if ir remote D pressed then
      point in direction -45
    if ir remote R0 pressed then
      point in direction -135
    if ir remote 1 pressed then
      point in direction 0
    if ir remote down pressed then
      point in direction 180
    if ir remote left pressed then
      point in direction -90

```

```

when green flag clicked
  forever
    move 15 steps

```


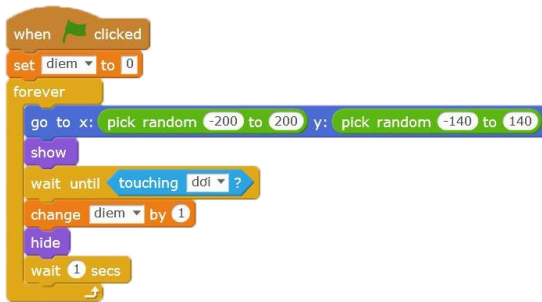
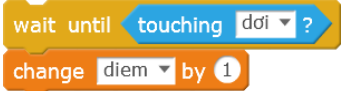




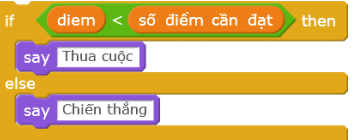



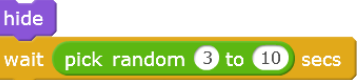
*Dơi* tự động di chuyển về phía trước

```

if ir remote - pressed then
  point in direction 90

```

Sử dụng khối lệnh **If**. sau đó kéo khối lệnh xoay hướng để thay đổi hướng bay

<p>Đối tượng nhân vật <b>Bóng điểm</b>.</p> <div data-bbox="215 295 339 419">  </div> <p>Xuất hiện ngẫu nhiên trên sân khấu và di chuyển. Nếu nó chạm vào con dơi, làm tăng giá trị điểm.</p>		 <p><i>Bóng điểm</i> khi chạm vào <i>Dơi</i> sẽ tăng thêm điểm.</p>  <p>Sau khi chạm vào <i>Dơi</i>, <i>Bóng điểm</i> sẽ biến mất và xuất hiện trở lại sau 1 giây.</p>
<p>Đối tượng nhân vật <b>Khỉ</b>.</p> <div data-bbox="191 897 325 1085">  </div> <p>So sánh 30 giây một lần</p>		 <p>Đặt giá trị <b>số điểm cần đạt</b> là một số ngẫu nhiên, và đặt lại thời gian về 0</p>  <p>Khi hết 30 giây, nếu số <b>điểm</b> lớn hơn hoặc bằng số <b>điểm</b> cần đạt thì sẽ thắng cuộc.</p>
<p>Đối tượng nhân vật <b>Bóng zero</b>.</p> <div data-bbox="215 1379 339 1503">  </div> <p>Xuất hiện ngẫu nhiên, nếu chạm vào <i>Dơi</i> sẽ đặt <b>điểm</b> bằng 0.</p>		 <p><i>Bóng zero</i> sẽ đặt <b>điểm</b> = 0 khi chạm vào <i>Dơi</i></p>  <p>Sau khi chạm vào <i>Dơi</i>, thời gian xuất hiện trở lại của <i>Bóng zero</i> lâu hơn <i>Bóng điểm</i>.</p>

## Lưu ý

Lưu ý: nếu chế độ xoay của nhân vật được đặt là `set rotation style left-right`, nó sẽ có sự thay đổi trên nhân vật. Hãy thử và kiểm tra sự thay đổi.

## Bài tập

1. Tạo một quả bóng nhỏ có thể làm giảm số điểm.
2. Lập trình cho bóng 0 di chuyển và nó có chức năng so sánh số điểm.

daylaptrinh.net




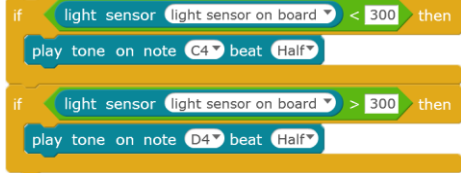

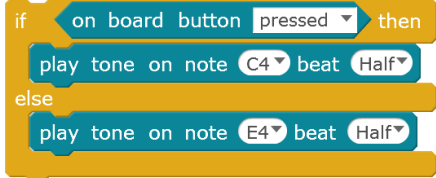

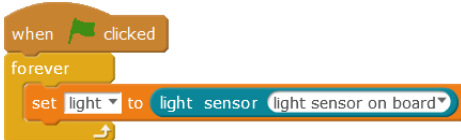
## Bài 4. Phỏng đoán

Chúng ta thường chơi trò chơi đoán tiền với bạn bè. Và chúng ta cũng có thể sử dụng bảng mạch robot để chơi trò chơi này. Và tìm kiếm ai là người đoán giỏi nhất.

### Kiến thức lập trình

1. Cấu trúc rẽ nhánh.
2. Sử dụng và so sánh biến.

### Kiến thức bổ sung

Khối lệnh	Mô tả	Ví dụ
	<p>Khối lệnh <b>If - then</b> là một cấu trúc điều kiện đơn. Khối lệnh cần được ghép với khối lệnh hình lục giác. Các khối lệnh bên trong sẽ được thực hiện nếu điều kiện đúng. Còn khi điều kiện sai thì sẽ bỏ qua.</p> <p>Nếu cần thực hiện một số hành động trong điều kiện nào đó, chúng ta sẽ cần sử dụng khối lệnh <b>If - then</b></p>	
	<p>Khối lệnh <b>If - then - else</b> là một khối lệnh điều kiện kép. Đoạn khối lệnh được ghép bên trên sẽ được thực hiện khi điều kiện đúng, còn khi điều kiện sai sẽ thực hiện đoạn khối lệnh bên dưới.</p>	
	<p>Thông thường, giá trị của cảm biến sẽ được lưu vào biến để liên tục cập nhật giá trị của cảm biến.</p>	

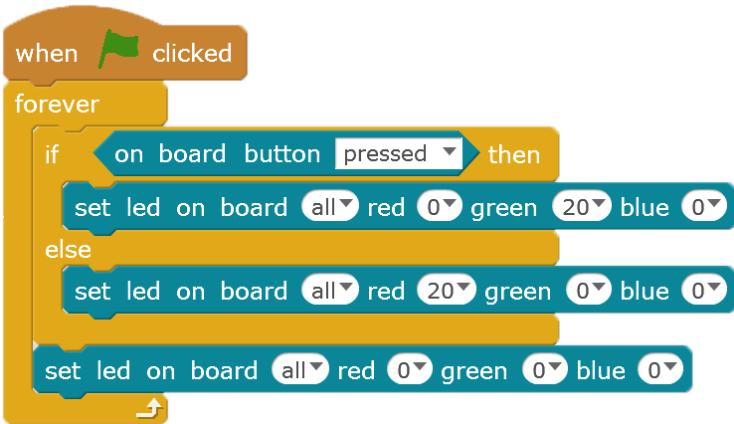
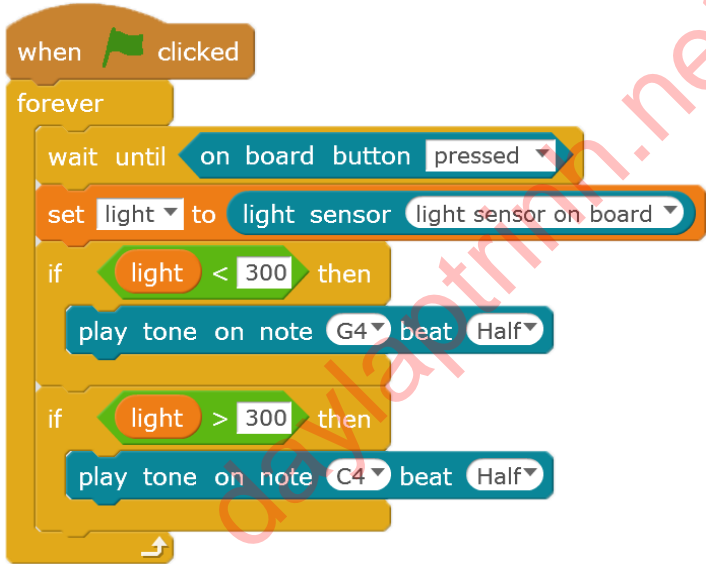
## Lưu ý

Vòng lặp và cấu trúc rẽ nhánh thường được sử dụng cùng nhau để điều kiện liên tục được kiểm tra và đánh giá.

## Ý tưởng lập trình

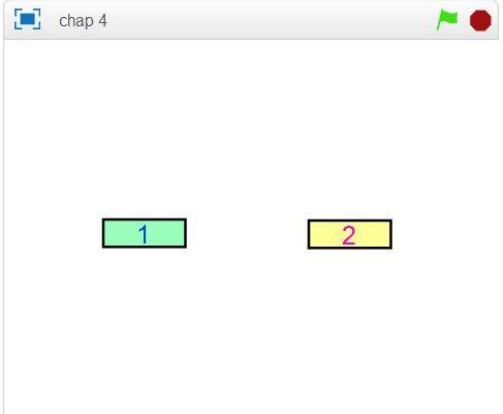


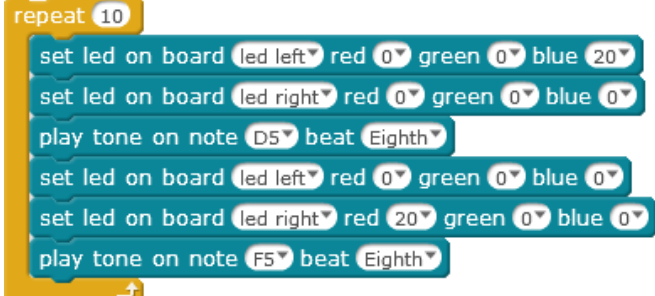
Mô tả ý tưởng	Lưu đồ
<p><b>Cấu trúc rẽ nhánh</b></p> <p>Cấu trúc rẽ nhánh dùng để kiểm tra một điều kiện nhất định và điều khiển luồng thực hiện của chương trình tùy thuộc vào kết quả kiểm tra</p>	<pre> graph TD     Start([Bắt đầu]) --&gt; SetState[Đặt Trạng thái = 0]     SetState --&gt; PlayMusic[Chơi nhạc]     PlayMusic --&gt; Decision1{Trạng thái = 1}     Decision1 -- Sai --&gt; PlayMusic     Decision1 -- Đúng --&gt; PlayLight[Chơi nhạc và nháy đèn]     PlayLight --&gt; SetAnswer[Đặt Đáp án ngẫu nhiên từ 1 - 2]     SetAnswer --&gt; Decision2{Đáp án = 1}     Decision2 -- Đúng --&gt; LightLeft[Sáng đèn bên trái]     Decision2 -- Sai --&gt; LightRight[Sáng đèn bên phải]     LightLeft --&gt; Decision3{Đáp án = Lựa chọn}     LightRight --&gt; Decision3     Decision3 -- Đúng --&gt; PlayCorrect[Chơi nhạc đúng]     Decision3 -- Sai --&gt; PlayWrong[Chơi nhạc sai và tắt đèn]     PlayCorrect --&gt; End([Kết thúc])     PlayWrong --&gt; End     </pre>

## Thực hành

Khối lệnh	Giải thuật
	<p>Tự mình viết lưu đồ của đoạn khối lệnh bên trái.</p>
	<p>Chạy đoạn khối lệnh bên trái, ghi lại sự thay đổi.</p>

## Lập trình

Lập trình trò chơi: Có 2 nút bấm trên sân khấu, hai đèn RGB sẽ nhấp nháy qua lại. Đoán xem đèn nào sẽ là đèn sáng cuối cùng.

		<p><b>Mô tả:</b></p> <p>Đầu tiên, chọn nút 1 hoặc 2 trên màn hình. Sau một thời gian, máy tính sẽ so sánh kết quả đúng với lựa chọn của người chơi nghĩa là đoán đúng, thì 1 đoạn âm thanh sẽ được chạy còn nếu sai, thì 1 đoạn âm thanh ngắn khác sẽ được chạy.</p>
<p>Sân khấu</p>		<p>Biến <b>Trạng thái</b> đặt bằng 0, có nghĩa người chơi chưa thực hiện chọn.</p> <p>Âm thanh được chơi tức là chương trình đã sẵn sàng, người chơi có thể chọn.</p>
		<p>Đợi đến khi <b>Trạng thái</b> bằng 1, tức là một trong hai nút đã được nhấn.</p>
		<p>Đèn LED sẽ sáng 10 lần và còi sẽ phát ra âm thanh</p>

	      	<p>Biến <b>Đáp án</b> sẽ lưu một số ngẫu nhiên là 1 hoặc 2 tương ứng với đèn trái hoặc phải sẽ sáng.</p>
	         	<p>Chương trình sẽ so sánh kết quả <b>Đáp án</b> với biến <b>Lựa chọn</b> lưu sự lựa chọn của người chơi. Nếu đúng thì sẽ chơi âm thanh đúng, còn nếu sai thì đèn sẽ tắt hết và chơi âm thanh sai.</p>
1	  	<p>Khi nút 1 được nhấn, đặt <b>Trạng thái</b> bằng 1 (đã thực hiện chọn), đặt <b>Lựa chọn</b> bằng 1.</p>
2	  	<p>Khi nút 1 được nhấn, đặt <b>Trạng thái</b> bằng 1 (đã thực hiện chọn), đặt <b>Lựa chọn</b> bằng 2.</p>

## Lưu ý

Khối lệnh sẽ thực hiện đoạn khối lệnh bên dưới khi đối tượng được nhấn, kể cả khi lá cờ xanh chưa được nhấn (chương trình chưa bắt đầu). Trong một số trường hợp thì cách lập trình như vậy không thích hợp. Vấn đề này sẽ được giải quyết sau khi chúng ta học các phép toán logic.

## Bài tập

1. Sử dụng cấu trúc rẽ nhánh để lập trình chương trình đếm số lần nút trên bảng mạch được nhấn trong 5 giây.
2. Sử dụng cấu trúc rẽ nhánh và giá trị cảm biến cường độ ánh sáng để lập trình thay đổi hình nền.

daylaptrinh.net

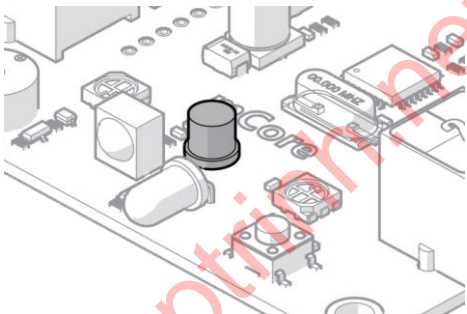
## Bài 5. Bảo vệ đảo

Kho báu còn lại của cướp biển được giấu trên một hòn đảo nhỏ. Khi những cướp biển khác nghe được thông tin và muốn tới để cướp kho báu về. Nhiệm vụ người chơi là bảo vệ kho báu đó.


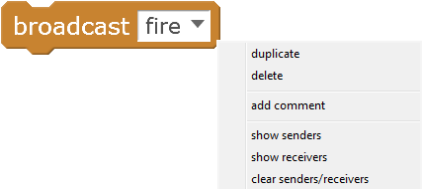
### Kiến thức lập trình

1. Thông báo và nhận thông báo.
2. Sử dụng bản sao.




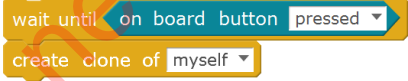

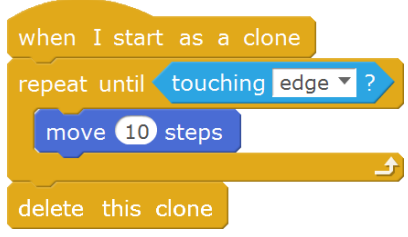

### Module điện tử

Tên	Hình ảnh	Chỉ dẫn
Cảm biến ánh sáng		Đưa ra giá trị độ sáng của ánh sáng xung quanh.

### Kiến thức bổ sung

Khối lệnh	Mô tả	Ví dụ
	<p>Khối lệnh thông báo để gửi thông báo tới toàn bộ nhân vật có trong chương trình (bao gồm cả chính nó). Sau khi nhận được thông báo, các nhân vật sẽ thực hiện các hành động nhất định.</p> <p>Hình bên phải là khi bạn nhấn chuột phải lên khối lệnh, trong các lựa chọn xuất hiện có <b>show senders</b> - hiển thị đối tượng gửi,</p>	



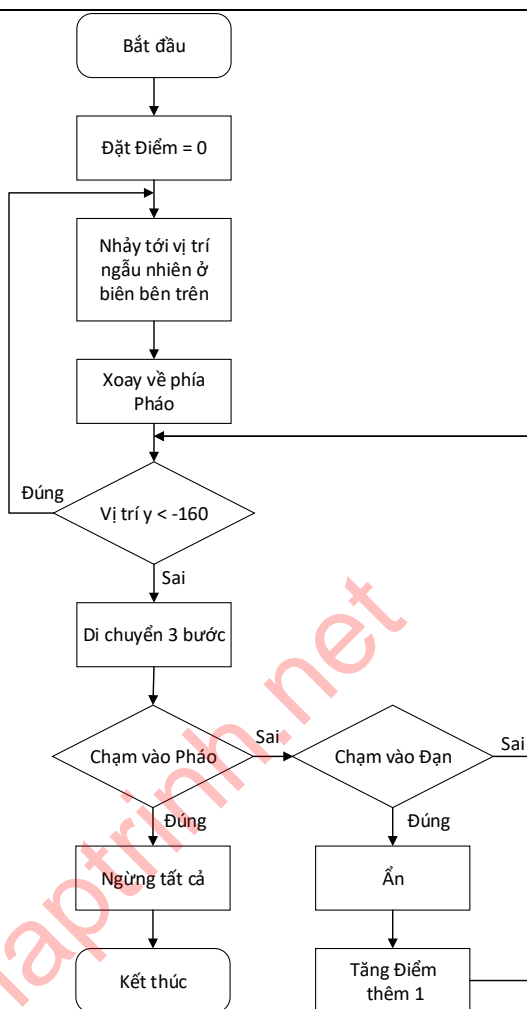
	<b>show recievers</b> - hiển thị đối tượng nhận, từ đó sẽ thấy được mối liên hệ giữa các đối tượng	
	Khối lệnh kích hoạt bên nhận vật nhận được thông báo. Khi nhận được thông báo, các khối lệnh được ghép phía dưới sẽ được thực hiện	
	Bản sao của đối tượng có hình dạng, kích thước tương tự bản chính. Bản sao và bản chính đều dung chung 1 khu lập trình, và tất cả các bản sao sẽ có hoạt động như nhau. Tính năng này để tránh việc chúng ta phải lập trình cho từng bản sao một.	
	Sau khi bản sao được tạo ra, bản sao sẽ hoạt động bằng các đoạn khối lệnh bắt đầu bởi khối lệnh này. Nó sẽ khác với khối lệnh bắt đầu của bản chính.	
	Xóa bản sao không cần thiết. Quá nhiều bản sao sẽ dẫn đến chương trình chạy chậm và ảnh hưởng tới tốc độ thực hiện các khối lệnh.	

## Ý tưởng lập trình

Mô tả ý tưởng	Lưu đồ
<p><b>Đối tượng Pháo</b></p> <p>Liên tục xoay sang trái, chỉ khi nào nút bấm trên bảng mạch được bấm thì quay sang phải.</p>	<pre> graph TD     Start([Bắt đầu]) --&gt; Decision1{Nút trên bảng mạch đang được nhấn}     Decision1 -- Đúng --&gt; RotateRight[Xoay sang phải 1 độ]     RotateRight --&gt; Decision2{Hướng của Pháo &gt; 45 độ}     Decision2 -- Đúng --&gt; RotateBackRight[Xoay về hướng 45 độ]     Decision2 -- Sai --&gt; RotateRight     Decision1 -- Sai --&gt; RotateLeft[Xoay sang trái 1 độ]     RotateLeft --&gt; Decision3{Hướng của Pháo &lt; -45 độ}     Decision3 -- Đúng --&gt; RotateBackLeft[Xoay về hướng -45 độ]     Decision3 -- Sai --&gt; RotateLeft     RotateBackRight --&gt; Start     RotateBackLeft --&gt; Start </pre>
<p><b>Đối tượng Đạn</b></p> <p>Khi giá trị cảm biến ánh sáng nhỏ hơn 500 thì sẽ thông báo “Bắn”.</p> <p>Khi nhận được thông báo “Bắn” sẽ nhảy tới <i>Pháo</i>, xoay về phía của <i>Pháo</i> và tạo bản sao.</p> <p>Bản sao được tạo ra sẽ di chuyển về phía trước.</p>	<pre> graph TD     Start([Nhận thông báo Bắn]) --&gt; Jump[Nhảy tới Pháo]     Jump --&gt; Rotate[Xoay về hướng của Pháo]     Rotate --&gt; Move[Di chuyển lên đầu nòng pháo]     Move --&gt; Create[Tạo bản sao]     Create --&gt; End1([Kết thúc])          Start2([Khi khởi đầu là một bản sao]) --&gt; Show[Hiện]     Show --&gt; Decision{Chạm vào biên}     Decision -- Sai --&gt; Move10[Di chuyển 10 bước]     Move10 --&gt; Show     Decision -- Đúng --&gt; Delete[Xóa bản sao]     Delete --&gt; End2([Kết thúc]) </pre>


### Đối tượng Tàu

Xuất hiện ngẫu nhiên và di chuyển về phía *Pháo*.



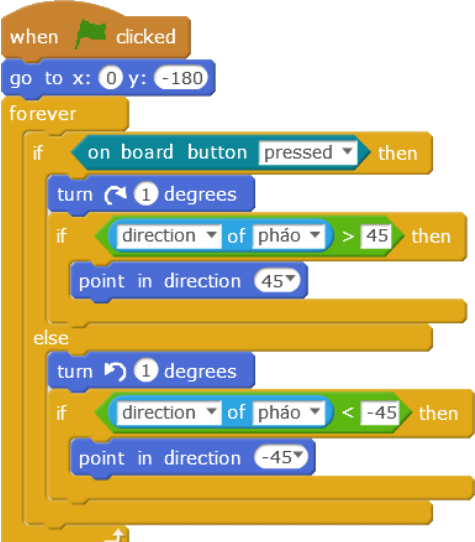
### Thực hành

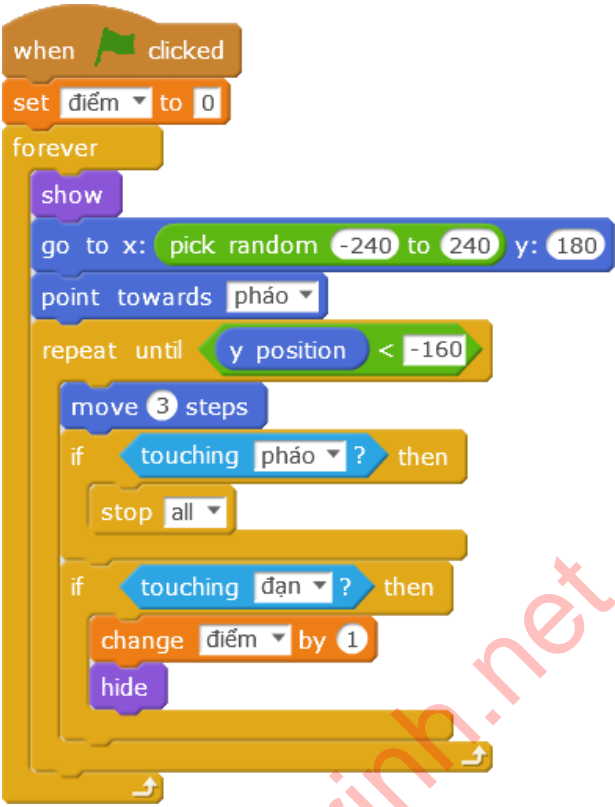
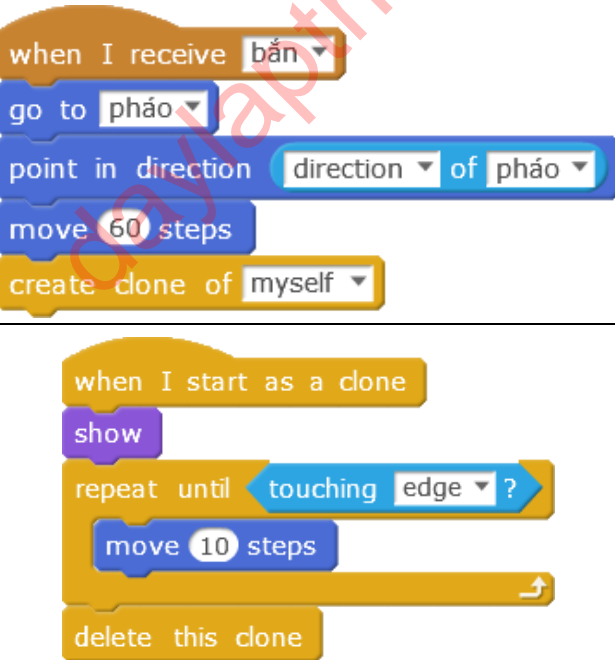
Khối lệnh	Giải thuật
	Viết lại sơ đồ thuật toán

	<p>Chạy và ghi lại sự thay đổi trên <b>mBlock</b>.</p>
---	--

## Lập trình

<p><b>Bảo vệ đảo</b></p>		<p><b>Hướng dẫn</b></p> <p>Sử dụng nút trên bảng mạch để điều chỉnh hướng của <i>Pháo</i>.</p> <p>Sử dụng cảm biến ánh sáng để điều khiển <i>Dạn</i>.</p> <p>Nếu <i>Tàu</i> đến được <i>Pháo</i>, trò chơi sẽ kết thúc.</p>
--------------------------	--	---

<p>Đối tượng <i>Pháo</i></p>		<p>Sử dụng nút trên bảng mạch để điều chỉnh góc của <i>Pháo</i>.</p> <p>Khi nút được nhấn, <i>Pháo</i> quay dần sang phải.</p> <p>Khi nút không được nhấn, <i>Pháo</i> quay dần sang trái.</p> <p>Giới hạn góc quay tối đa về bên phải và bên trái là 45 độ so với phương thẳng đứng.</p>
<p>Sân khấu</p>		<p>Điều khiển hành động bắn.</p> <p>Nếu ánh sáng yếu (tối) thì sẽ thông báo “Bắn”</p>

<p>Đối tượng Tàu</p>		<p>Xuất hiện ngẫu nhiên phía trên và di chuyển tới <i>Pháo</i></p> <p>Nếu bị bắn trúng (chạm vào <i>Đạn</i>) sẽ tăng điểm và ẩn đi</p> <p>Nếu chạm vào <i>Pháo</i> thì trò chơi kết thúc</p>
<p>Đối tượng Đạn</p>		<p>Khi nhận được thông báo “Bắn”, nhảy tới <i>Pháo</i>, xoay về phía của <i>Pháo</i> và tạo bản sao.</p> <p>Bản sao được tạo ra di chuyển liên tục về phía trước cho đến khi chạm vào biên của sân khấu thì bị xóa.</p>

## Lưu ý

1. Các trạng thái ban đầu của bản sao sẽ giống bản chính. Nếu bản chính được ẩn đi thì bản sao của nó cũng được ẩn đi và ngược lại.
2. Thông báo cũng là một sự kiện. Đoạn khối lệnh bắt đầu bằng “when receiving the message” sẽ được thực hiện khi nhận được 1 thông báo phù hợp.

## Bài tập

1. Sử dụng chức năng tạo bản sao để tạo hiệu ứng mưa cho chương trình.
2. Sử dụng cảm biến ánh sáng trên robot để thay đổi sân khấu.
3. Sửa đổi lại trò chơi và cho phép 3 đến 5 kẻ địch xuất hiện cùng lúc.

daylaptrinh.net



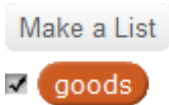
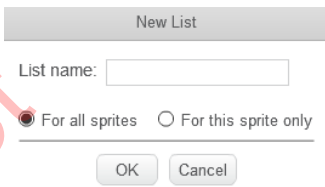

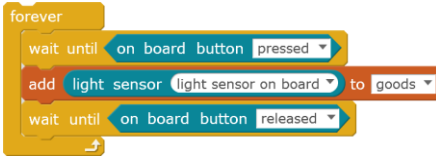


## Bài 6. Quét mã vạch

Trong siêu thị, mỗi sản phẩm đều có mã vạch. Mỗi mã vạch tương ứng với 1 loại hàng hóa cụ thể. Chương này các bạn sẽ tạo được một máy quét mã vạch.

### Kiến thức lập trình

1. Sử dụng danh sách.

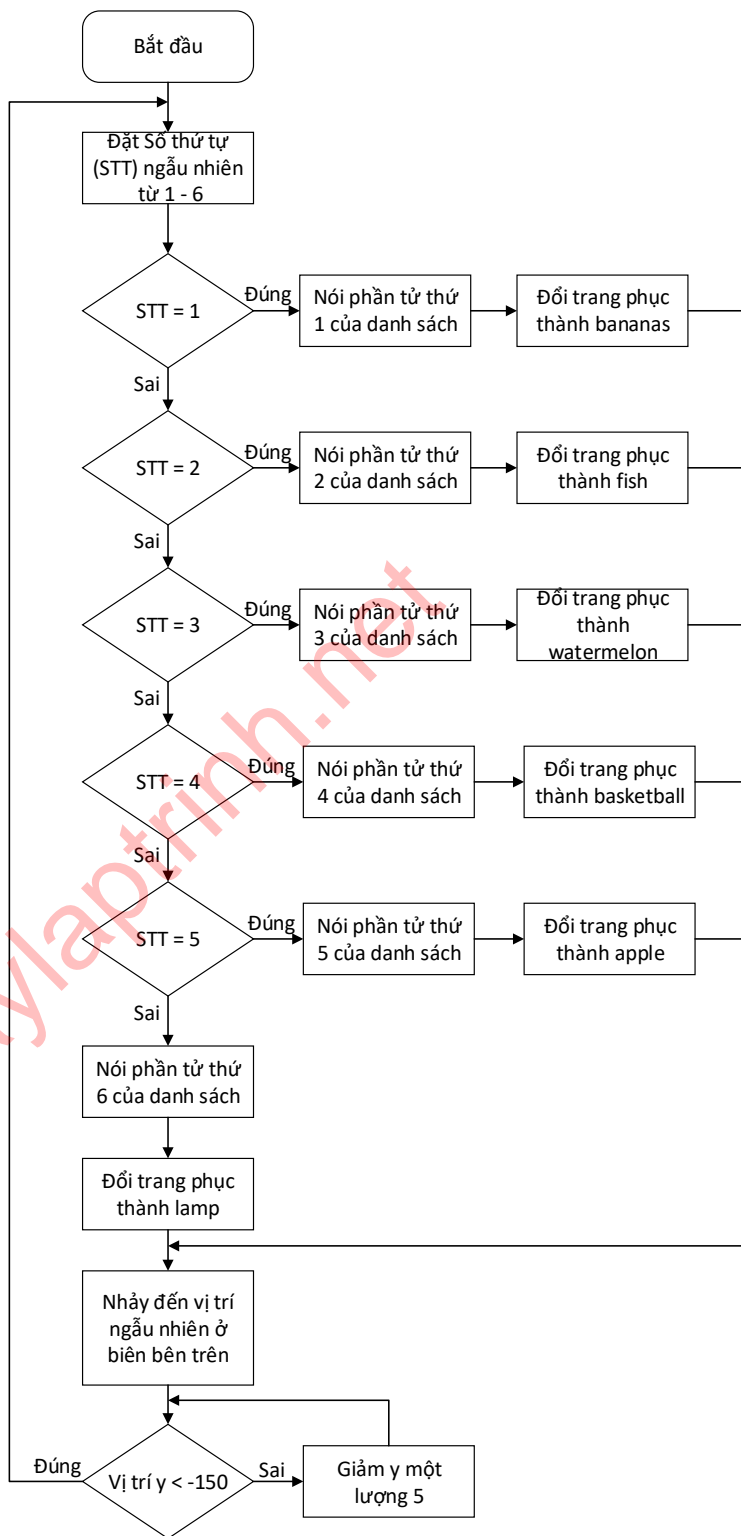
### Kiến thức bổ sung

Khởi lệnh	Mô tả	Ví dụ
	<p>Một danh sách có thể bao gồm 1 tập hợp các biến. Nó có thể lưu trữ được các biến và cũng như đưa ra một biến có trong danh sách.</p>	
	<p>Gõ để nhập vào giá trị tại vị trí “thing” hoặc ghép biến vào trong các khối lệnh bên trái. Dữ liệu điền vào sẽ tự động được đánh số thứ tự trong danh sách. Các khối lệnh có thể thao tác với danh sách:</p> <ol style="list-style-type: none"> <li>1. Lưu dữ liệu vào vị trí cuối cùng của danh sách</li> <li>2. Xóa dữ liệu tại một vị trí trong danh sách</li> <li>3. Chèn dữ liệu vào một vị trí trong danh sách</li> <li>4. Thay thế dữ liệu tại một vị trí trong danh sách</li> </ol>	
	<p>Chức năng các khối lệnh:</p> <ol style="list-style-type: none"> <li>1. Lấy dữ liệu tại vị trí nhất định trong danh sách.</li> <li>2. Độ dài của danh sách.</li> <li>3. Kiểm tra dữ liệu có trong danh sách hay</li> </ol>	

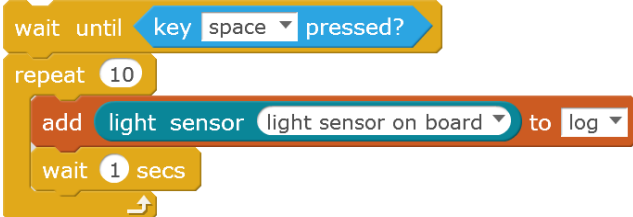
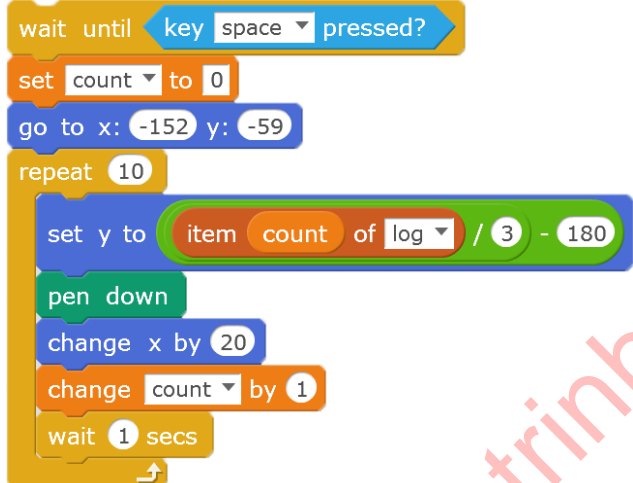
	không	
--	-------	--

# Ý tưởng lập trình

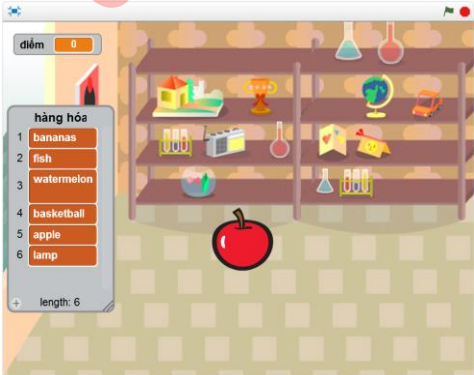
Mô tả ý tưởng	Lưu đồ
Sân khấu	<pre> graph TD     Start([Bắt đầu]) --&gt; SetBrightness[Đặt Độ sáng = số từ 0 – 9 sau khi chuyển từ Cường độ ánh sáng đo được]     SetBrightness --&gt; ResetTime[Đặt lại thời gian]     ResetTime --&gt; TimeGT60{Thời gian &gt; 60}     TimeGT60 -- Đúng --&gt; StopAll[Ngừng tất cả]     StopAll --&gt; End([Kết thúc])     TimeGT60 -- Sai --&gt; BrightnessEqSeq{Độ sáng = Số thứ tự}     BrightnessEqSeq -- Đúng --&gt; IncreaseScore[Tăng Điểm thêm 1]     IncreaseScore --&gt; SetSeq0[Đặt Số thứ tự = 0]     SetSeq0 --&gt; ResetTime     BrightnessEqSeq -- Sai --&gt; ResetTime </pre>
Đối tượng Hàng hóa	<pre> graph TD     Start([Bắt đầu]) --&gt; SetScore0[Đặt Điểm = 0]     SetScore0 --&gt; AddItems[Thêm bananas, fish, waterlemon, basket ball, apple, lamp vào danh sách]     AddItems --&gt; End([Kết thúc]) </pre>

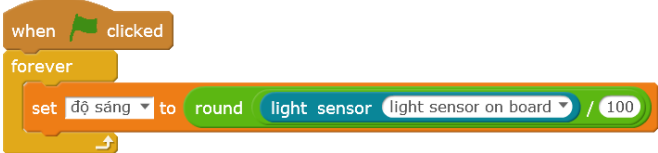
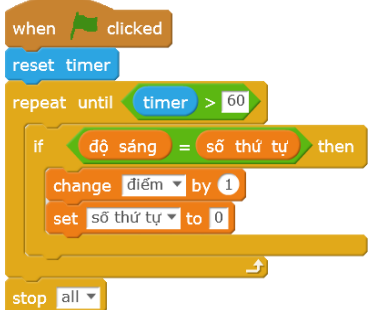

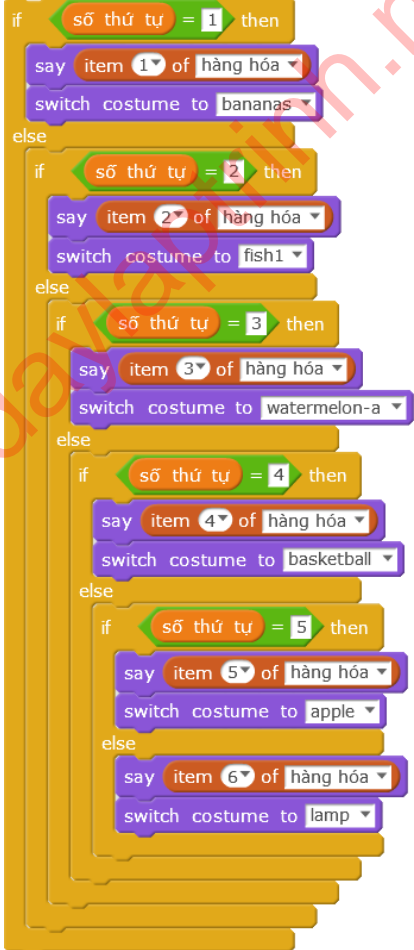


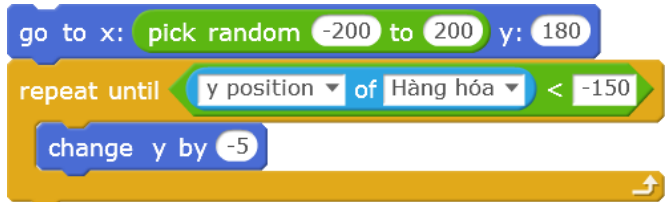
## Thực hành

Khối lệnh	Giải thuật
	Viết lưu đồ thuật toán của khối lệnh bên trái.
	Lập trình và chạy thử đoạn khối lệnh bên trái. Theo dõi sự thay đổi trên <b>mBlock</b> .

## Lập trình

<p><b>Quét mã vạch</b></p>		<p><b>Hướng dẫn</b></p> <p>Thay đổi độ sáng ứng với số thứ tự của hàng hóa để ghi điểm.</p>
----------------------------	---	---

<p><b>Sân khấu</b></p>		<p>Chuyển giá trị độ sáng đo được thành số từ 1 đến 9 và đặt vào biến <b>Độ sáng</b></p>
		<p>Trong 60 giây, nếu <b>Độ sáng</b> bằng <b>Số thứ tự</b> của hàng hóa thì sẽ được cộng điểm và đặt lại <b>Số thứ tự</b>.</p>
<p><b>Đối tượng Hàng hóa</b></p>		<p>Đặt số thứ tự hàng hóa ngẫu nhiên từ 1 đến 6</p>
		<p>Đặt trang phục và lời nói của hàng hóa tương ứng với số thứ tự.</p>

		<p>Nhảy đến biên bên trên và di chuyển xuống phía dưới.</p>
--	---	---

## . Lưu ý

Chúng ta có thể nhập dữ liệu của danh sách từ một tệp tin văn bản (định dạng **.txt**) bằng cách nhấn chuột phải vào danh sách hiện trên sân khấu và chọn **Import**. Để xuất dữ liệu, chọn **Export**.



## Bài tập

1. Sử dụng danh sách để ghi lại giá trị Cảm biến ánh sáng (20 lần trong vòng 20 giây) và xuất ra tệp tin.
2. Sử dụng danh sách để ghi lại thời gian và cảm biến.
3. Kết hợp cùng chức năng vẽ, thử vẽ đồ thị hoặc biểu đồ bằng dữ liệu của bài học trước.


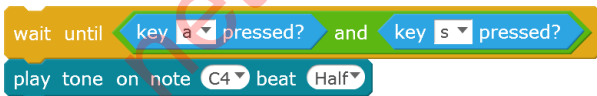



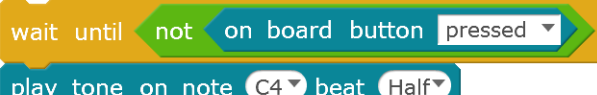
## Bài 7. Trò chơi nhịp điệu

Hãy cùng chơi một trò chơi và xem ai là người có cảm giác về nhịp điệu tốt hơn. Trò chơi này tập trung vào sự phối hợp giữa tay và mắt.

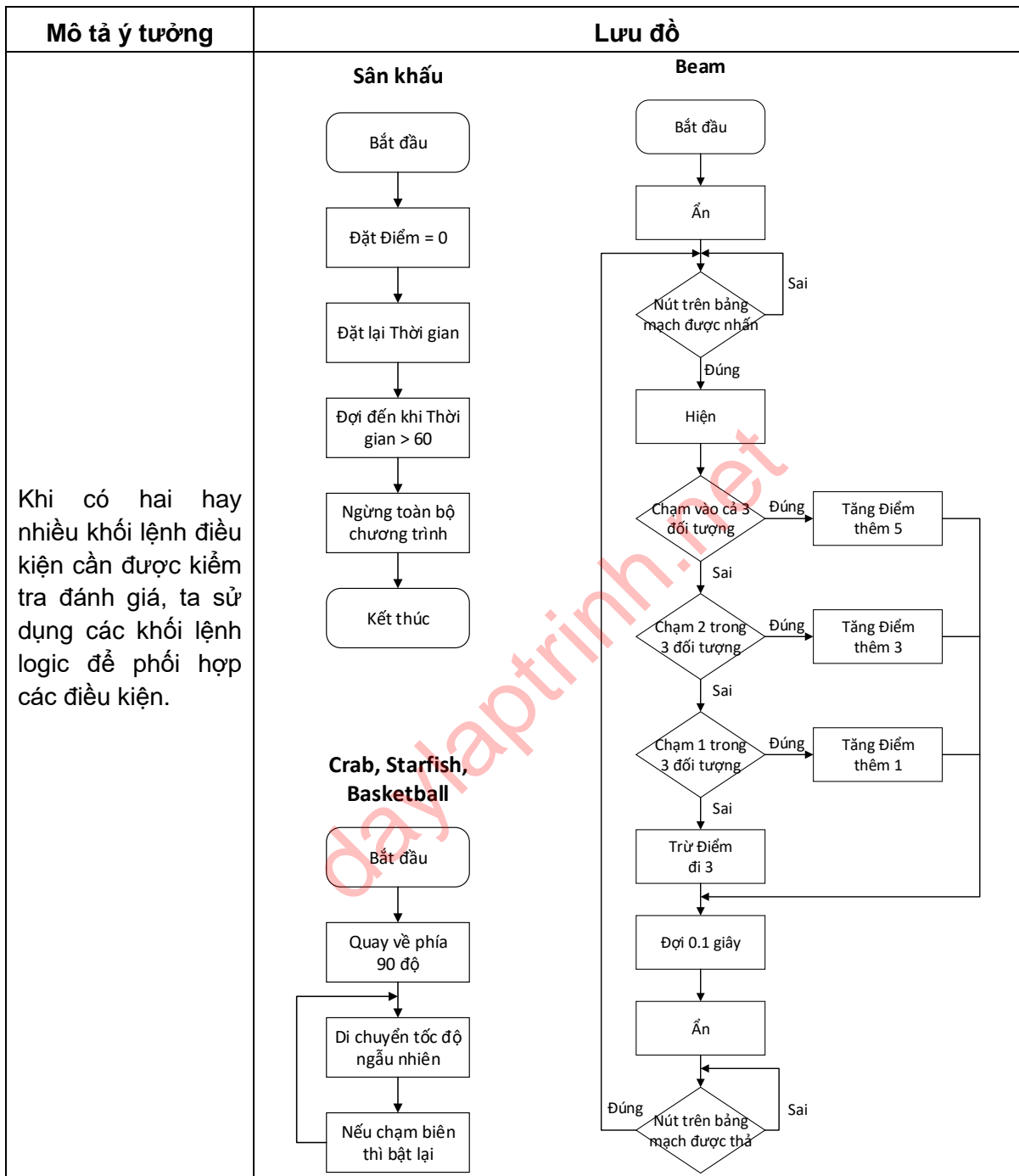
### Kiến thức lập trình

- Sử dụng khối lệnh logic.

### Kiến thức bổ sung

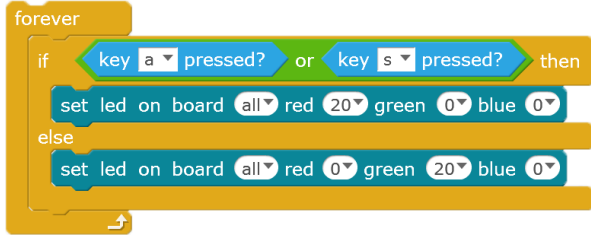
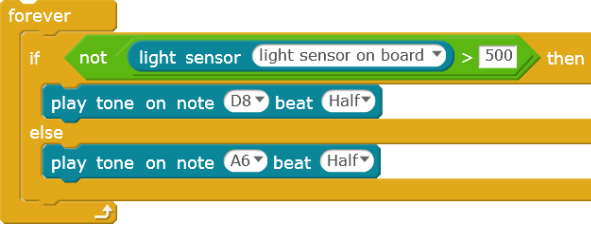
Khối lệnh	Mô tả	Ví dụ
	Khi cả 2 khối lệnh điều kiện ghép bên trong đều có giá trị đúng thì khối lệnh <b>and</b> sẽ có giá trị đúng, ngược lại thì có giá trị sai	
	Chỉ cần một trong hai khối lệnh điều kiện có giá trị đúng thì khối lệnh <b>or</b> có giá trị đúng, ngược lại thì có giá trị sai	
	Khi khối lệnh điều kiện được ghép bên trong có giá trị đúng thì khối lệnh <b>not</b> có giá trị sai và ngược lại	

## Ý tưởng lập trình






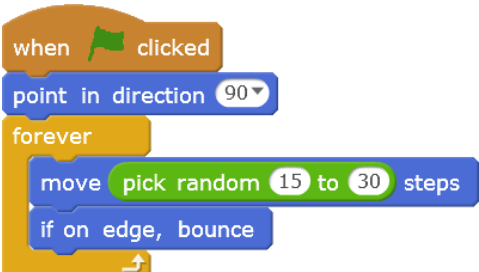



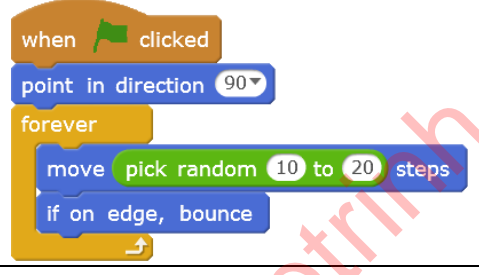

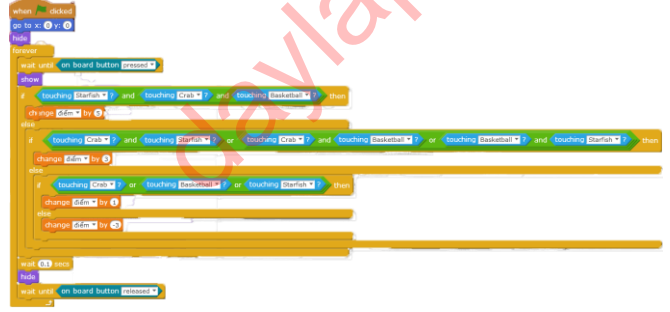
## Thực hành

Khối lệnh	Giải thuật
	Viết lưu đồ thuật toán của đoạn khối lệnh bên trái
	Chạy thử đoạn khối lệnh bên trái và theo dõi sự thay đổi.



## Lập trình

Chương trình có 3 đối tượng nhân vật di chuyển qua lại với tốc độ khác nhau. Sử dụng nút bấm trên bảng mạch để điều khiển cột sáng ở giữa màn hình ẩn hoặc hiện. Nếu cột sáng chạm vào một hay nhiều nhân vật, điểm sẽ được tăng. Ngược lại, điểm sẽ bị trừ đi.

Trò chơi nhịp điệu		Trò chơi kéo dài trong 60 giây. Nếu cột sáng chạm vào càng nhiều đối tượng thì điểm cộng thêm càng cao.
Sân khấu		Đặt lại <b>Điểm</b> và thời gian. Nếu thời gian lớn hơn 60 giây thì trò chơi kết thúc.

		<p>Các nhân vật di chuyển ngang trên sân khấu. Mỗi đối tượng sẽ có 1 tốc độ khác nhau.</p>
		<p>Các nhân vật di chuyển ngang trên sân khấu. Mỗi đối tượng sẽ có 1 tốc độ khác nhau.</p>
		<p>Các nhân vật di chuyển ngang trên sân khấu. Mỗi đối tượng sẽ có 1 tốc độ khác nhau.</p>
		<p>Mỗi khi nhấn nút trên bảng mạch thì cột sáng hiện ra trong khoảng thời gian ngắn rồi bị ẩn đi.</p> <p>Số điểm tăng hay giảm phụ thuộc vào số lượng đối tượng mà cột sáng bắt được.</p>

## Bài tập

1. Sử dụng khối lệnh  để lập trình tạo hiệu ứng: Khi chạm vào con trỏ chuột thì phóng to nhân vật, còn nếu không chạm vào thì thu nhỏ về kích thước cũ.
2. Sử dụng khối lệnh  để lập trình điều khiển đèn LED nhấp nháy.




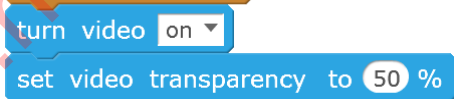

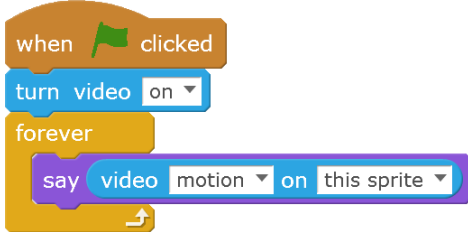
## Bài 8. Video Ball

Trong bài học này, chúng ta sẽ tạo ra một trò chơi tương tác. Trò chơi này sẽ kiểm tra sự phối hợp giữa tay và mắt của bạn.

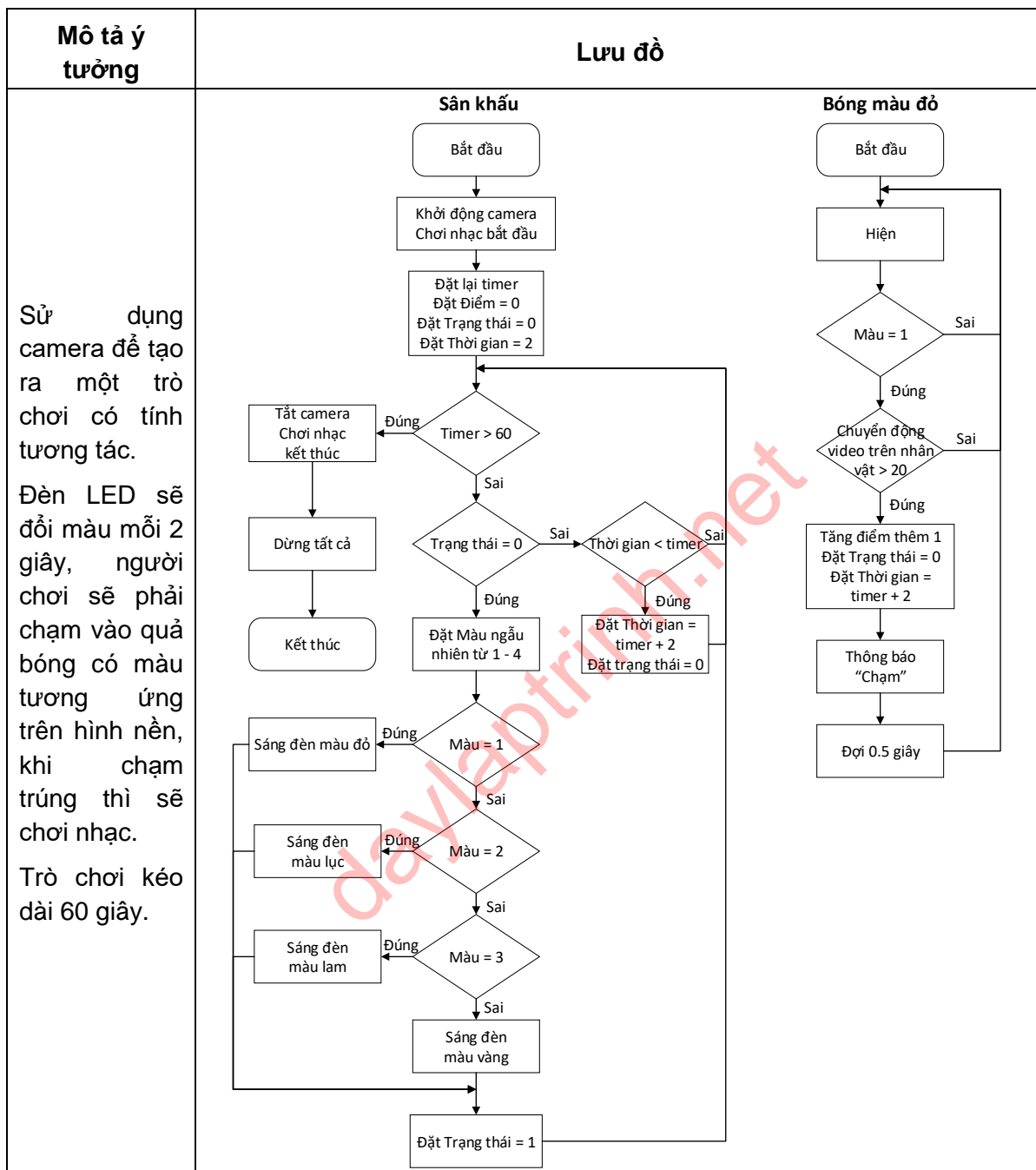
### Kiến thức lập trình

1. Sử dụng các khối lệnh máy quay
2. Sử dụng biến

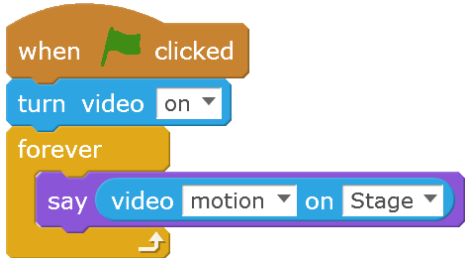
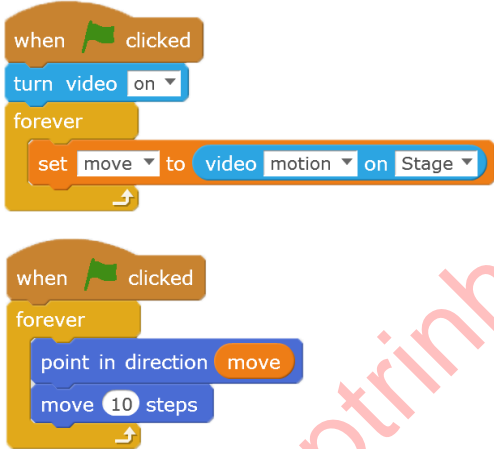
### Kiến thức bổ sung

Khối lệnh	Mô tả	Ví dụ
	Khởi động camera và kết nối với chương trình.	
	Điều chỉnh độ trong suốt từ 0 – 100%, trong đó 0% là rõ nét nhất còn 100% là mờ hoàn toàn.	
	Xác định sự chuyển động của video trên nhân vật đối tượng.	

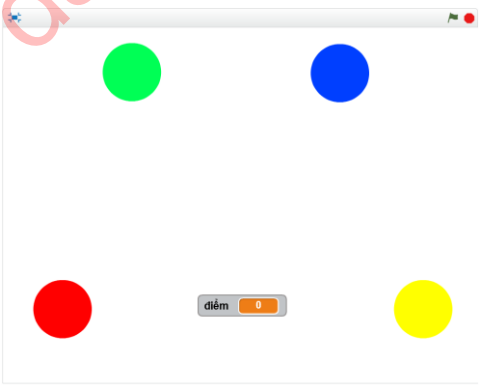
## Ý tưởng lập trình



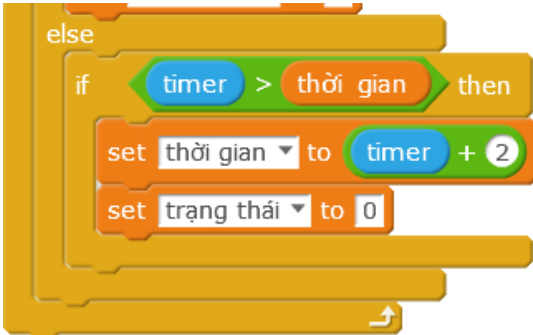
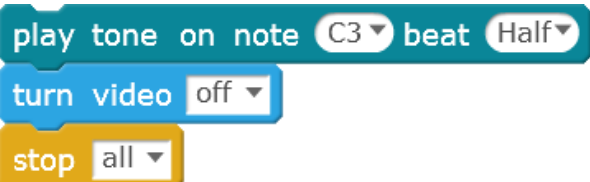
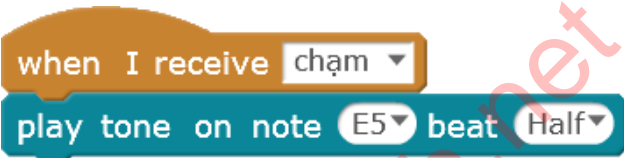

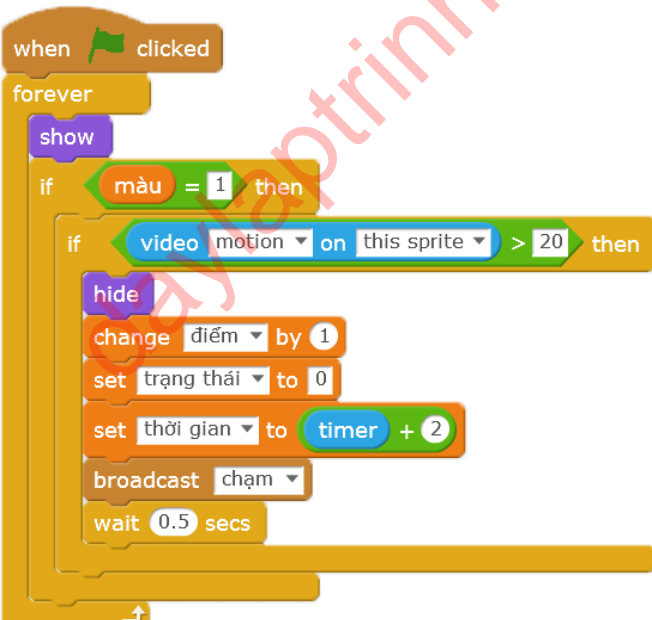
## Thực hành

Khối lệnh	Giải thuật
	Viết lưu đồ thuật toán của đoạn khối lệnh bên trái.
	Chạy thử đoạn khối lệnh bên trái và theo dõi sự thay đổi.

## Lập trình

Video Ball		<p>Đèn LED sẽ đổi màu mỗi 2 giây, người chơi sẽ phải chạm vào quả bóng có màu tương ứng trên hình nền, khi chạm trúng thì sẽ chơi nhạc.</p> <p>Trò chơi kéo dài 60 giây.</p>
------------	---	--

Sân khấu		Khởi động camera
		Chơi nhạc bắt đầu
		<p>Đặt lại <b>timer</b> và giá trị ban đầu cho các biến.</p> <p><b>Trạng thái:</b> nếu người chơi đã chạm vào bóng thì có giá trị 0, ngược lại là 1</p>
		<p>Trò chơi kéo dài 60 giây.</p> <p>Nếu người chơi đã chạm thì đặt lại Màu ngẫu nhiên và thay đổi màu cho đèn LED tương ứng, sau đó chuyển sang trạng thái bằng 1 (chưa chạm)</p>

		<p>Còn nếu người chơi chưa chạm thì đếm thời gian để nếu quá 2 giây mà vẫn chưa chạm đúng thì sẽ đổi sang trạng thái đã chạm (lượt chơi mới)</p>
		<p>Sau khi hết thời gian, chơi nhạc thua, tắt camera và dừng chương trình</p>
		<p>Nếu chạm đúng thì chơi nhạc</p>
		<p>Nếu <b>Màu</b> = 1 và đang bị chạm thì tăng điểm, đặt lại <b>Trạng thái</b>, <b>Thời gian</b> và thông báo "Chạm".</p> <p>Các nhân vật đối tượng bóng màu lục, lam và vàng cũng có hoạt động tương tự.</p>

## Lưu ý

Tại sao quả bóng cần được ẩn đi 0.5 giây sau khi bị chạm phải ? Bởi nếu hiện liên tục thì Điểm sẽ được tăng liên tục, vì thế quả bóng cần ẩn đi trong một khoảng thời gian ngắn.

## Bài tập

1. Kiểm tra mối liên quan giữa camera và độ trong suốt của video.

Ghi lại sự thay đổi thông qua khối lệnh

video motion on this sprite

2. So sánh sự khác biệt giữa hai khối lệnh

video motion on this sprite

và

video motion on Stage

daylaptrinh.net



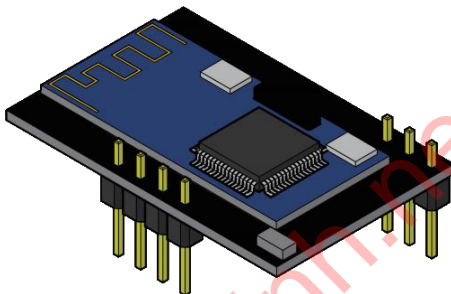
## Bài 9. Chạy nào ! Robot

Bài học này tập trung vào lập trình để điều khiển robot di chuyển theo ý muốn.


### Kiến thức lập trình

- Sử dụng **mBlock** để điều khiển robot di chuyển.

### Module điện tử

Tên	Hình ảnh	Chỉ dẫn
Bộ tín hiệu Bluetooth		Thay thế dây kết nối, hỗ trợ lập trình và truyền dữ liệu không dây.

### Kiến thức bổ sung

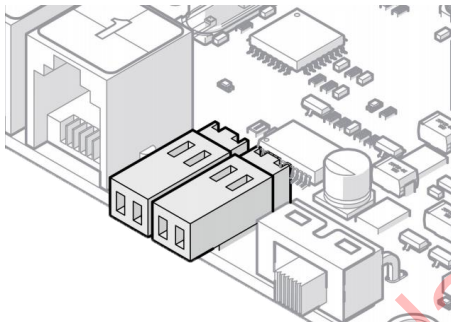
Nhóm khối lệnh	Khối lệnh	Mô tả
Robots		Thiết lập tốc độ quay và chiều quay của động cơ. Dải giá trị nằm trong khoảng từ -255 đến 255. Với giá trị 255, động cơ quay với tốc độ lớn nhất theo chiều tiến về phía trước. Với giá trị -255, động cơ quay với tốc độ lớn nhất theo chiều ngược lại.

## Ý tưởng lập trình

Mô tả ý tưởng	Lưu đồ
Sử dụng bộ tín hiệu Bluetooth để nhận lệnh từ <b>mBlock</b> , sau đó điều khiển cho động cơ chạy.	<div style="display: flex; justify-content: space-around; align-items: center;"> <div style="text-align: center;"> <p>Khi phím mũi tên được nhấn</p> <p>↓</p> <p>Chạy mô-tơ</p> </div> <div style="text-align: center;"> <p>Khi phím mũi tên được thả</p> <p>↓</p> <p>Dừng mô-tơ</p> </div> </div>

## Thực hành

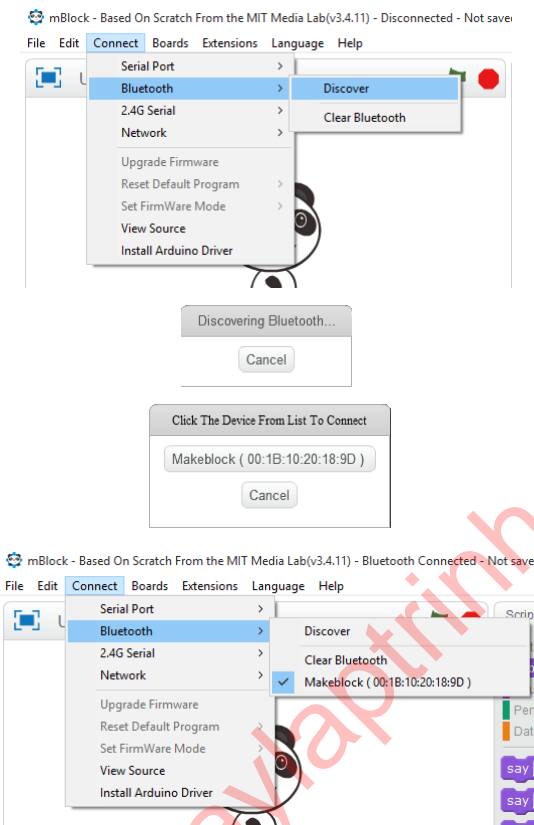
### 1. Kết nối động cơ

Hình ảnh	Chỉ dẫn
	Hai cổng ở bên trái bảng mạch dùng để kết nối tới hai động cơ.

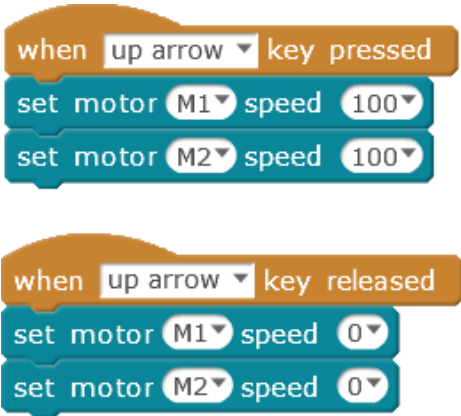
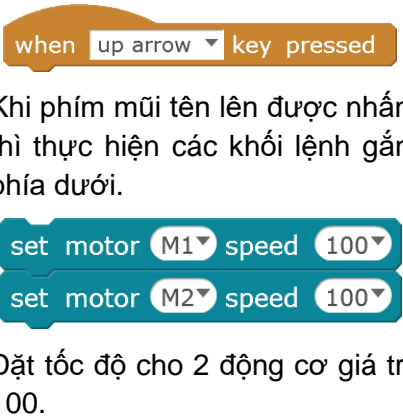
### 2. Cấp nguồn




Hình ảnh	Chỉ dẫn
	Trước khi sử dụng Bluetooth, hãy chắc chắn rằng bảng mạch đã được cấp nguồn bằng pin và được bật lên.

### 3. Thiết lập kết nối Bluetooth

Hình ảnh	Chỉ dẫn
	<p>Nhấn lựa chọn <b>Discover</b> và đợi kết quả tìm kiếm thiết bị Bluetooth. Tìm kết nối Bluetooth có tên "Makeblock".</p> <p>Sau khi kết nối thành công, phần trạng thái có sự thay đổi như hình bên.</p>

### 4. Thực hành viết các đoạn khối lệnh

Ý tưởng	Khối lệnh	Mô tả
Nhấn phím mũi tên lên để robot di chuyển về phía trước, thả phím để robot dừng lại.		<p>Khi phím mũi tên lên được nhấn thì thực hiện các khối lệnh gần phía dưới.</p> 







		 <p>Khi phím mũi tên xuống được nhấn thì thực hiện các khối lệnh gần phía dưới.</p>   <p>Đặt tốc độ cho 2 động cơ giá trị 0 (động cơ dừng lại).</p>
--	--	---

### Lưu ý



Tốc độ động cơ có giá trị 100 không phải là tốc độ thực, mà là dải giá trị để việc tính toán dễ dàng hơn. Dải giá trị này có độ lớn từ 0 – 255. Giá trị 0 ứng với tốc độ quay của động cơ bằng 0, giá trị 255 ứng với tốc độ quay của động cơ đạt giá trị lớn nhất.

Nếu muốn điều khiển robot, bạn sẽ lập trình như thế nào ?

Ý tưởng	Khối lệnh	Mô tả
Nếu một bánh xe quay về phía trước và bánh còn lại không quay, chuyện gì sẽ xảy ra ?	     	Khi mũi tên trái được nhấn, động cơ trái sẽ dừng và động cơ phải sẽ quay theo chiều tiến về phía trước, robot sẽ quay sang trái.

Nếu nghĩ robot quay quá chậm, bạn có thể lập trình như dưới đây:

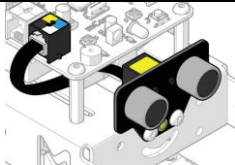
Ý tưởng	Khối lệnh	Mô tả
Nếu một bánh xe quay về phía trước và bánh còn quay về phía sau, chuyện gì sẽ xảy ra ?	<pre> when left arrow key pressed set motor M1 speed 100 set motor M2 speed -100  when left arrow key released set motor M1 speed 0 set motor M2 speed 0 </pre>	Nếu bánh trái quay về phía sau còn bánh phải quay về phía trước, robot sẽ nhanh chóng quay sang trái.

Tự lập trình theo kiến thức đã được học phía trên. Điều khiển robot tiến, lùi, quay trái, quay phải bằng các phím mũi tên trên bàn phím.

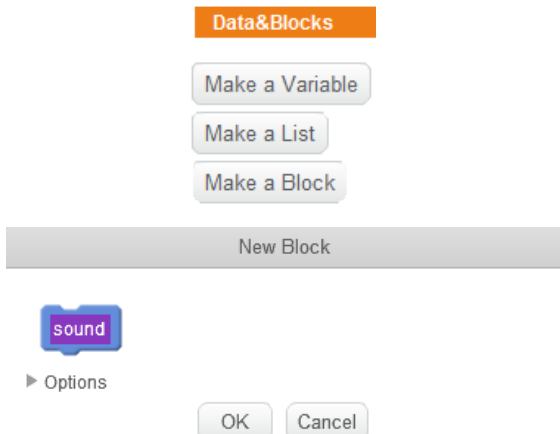
<pre> when up arrow key pressed set motor M1 speed 100 set motor M2 speed 100  when down arrow key released set motor M1 speed -100 set motor M2 speed -100  when left arrow key released set motor M1 speed 100 set motor M2 speed -100  when right arrow key released set motor M1 speed -100 set motor M2 speed 100 </pre>	<pre> when up arrow key released set motor M1 speed 0 set motor M2 speed 0  when down arrow key released set motor M1 speed 0 set motor M2 speed 0  when left arrow key released set motor M1 speed 0 set motor M2 speed 0  when right arrow key released set motor M1 speed 0 set motor M2 speed 0 </pre>	Điều khiển robot bằng các phím mũi tên trên bàn phím.
---	--	---

## Lập trình

### Cảm biến siêu âm

	Lắp đặt cảm biến siêu âm ở phía trước robot, kết nối dây dữ liệu vào cổng 3.
---	--

## 1. Định nghĩa khối lệnh thành phần

<p>Định nghĩa một khối lệnh có tên <b>"sound"</b></p>		<p>Chọn nhóm khối lệnh <b>Data&amp;Blocks</b></p> <p>Chọn <b>Make a Block</b></p> <p>Đặt tên khối lệnh được tạo là <b>"sound"</b></p>
---	---	---

## 2. Lập trình

<p>Khi robot di chuyển, khối lệnh <b>sound</b> sẽ được chạy. Nếu cảm biến siêu âm phát hiện vật cản ở phía trước, còi sẽ kêu để cảnh báo.</p>		<p>là khối lệnh để bắt đầu một định nghĩa.</p> <p>Khối lệnh <b>sound</b> được định nghĩa thực hiện công việc kiểm tra: Nếu cảm biến siêu âm (nối với cổng 3) đo được khoảng cách tới vật phía trước nhỏ hơn 10cm thì còi sẽ kêu.</p> <p>Khi robot di chuyển (bằng phím mũi tên), khối lệnh <b>sound</b> sẽ được thực hiện.</p>
---	--	--

### Bài tập

1. Sử dụng phím cách (space) trên bàn phím để điều khiển cảnh báo.

## Bài 10. Bậc thầy né tránh

Robot đã có thể di chuyển, nhưng thường xuyên bị đâm vào vật cản. Trong bài học này chúng ta sẽ tìm hiểu cách lập trình để robot tránh được vật cản phía trước.

### Kiến thức lập trình

1. Sử dụng cảm biến siêu âm
2. Lập trình để robot tránh vật cản

### Module điện tử

Tên	Hình ảnh	Chỉ dẫn
Cảm biến siêu âm		Cảm biến siêu âm có thể đo được khoảng cách tới vật cản phía trước

### Kiến thức bổ sung

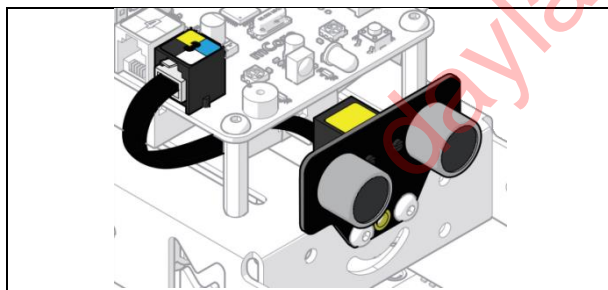
Nhóm khối lệnh	Khối lệnh	Ví dụ
<b>Robots</b>	ultrasonic sensor Port1 distance	Khoảng cách tới vật cản phía trước đo được bởi cảm biến siêu âm, tính bằng centimet (cm)

## Ý tưởng lập trình

Mô tả ý tưởng	Lưu đồ
<p>Nhấn giữ phím cách (space) trên bàn phím để robot di chuyển về phía trước. Nếu phía trước 50cm có vật cản thì còi kêu và âm thanh nhắc nhở. Nếu phía trước 10cm có vật cản thì robot sẽ rẽ sang hướng khác.</p>	<pre> graph TD     Start([Khi phím cách được nhấn]) --&gt; SetDis[Đặt Dis = khoảng cách đo được]     SetDis --&gt; Dis10{Dis &lt; 10 cm}     Dis10 -- Đúng --&gt; Turn[Rẽ hướng khác]     Turn --&gt; SetDis     Dis10 -- Sai --&gt; GoStraight[Đi thẳng]     GoStraight --&gt; SetDis     SetDis --&gt; Dis50{Dis &lt; 50 cm}     Dis50 -- Đúng --&gt; Sound[Chơi nhạc và âm thanh nhắc nhở]     Sound --&gt; Wait[Đợi]     Wait --&gt; Dis50     Dis50 -- Sai --&gt; SetDis     End([Khi phím cách được thả]) --&gt; Stop[Dừng động cơ]     </pre>

## Thực hành



### Cảm biến siêu âm

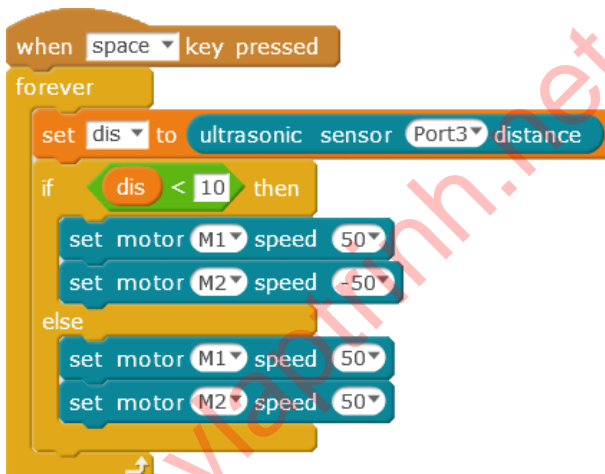

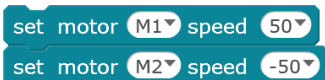



Lắp đặt cảm biến siêu âm ở phía trước robot, kết nối dây dữ liệu vào cổng 3.

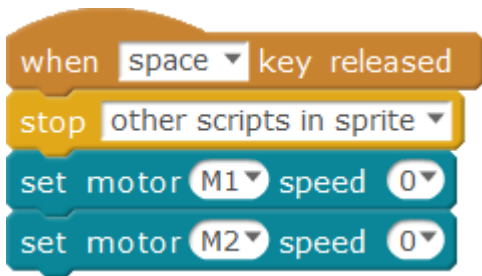

<p>Tạo biến để lưu giá trị khoảng cách đo được bằng cảm biến siêu âm</p>	<pre> when space key pressed   forever loop     set dis to ultrasonic sensor Port2 distance   </pre>	<p>Nhấn phím cách để bắt đầu</p> <pre> when space key pressed   forever loop   </pre> <p>Dữ liệu liên tục được đo đạc, vì</p>
--	--	---



		<p>vậy cần được lặp đi lặp lại.</p>  <p>Gán cho biến <b>Dis</b> giá trị là khoảng cách đo được.</p>  <p>Giá trị của biến được hiển thị trên sân khấu.</p>
--	--	--

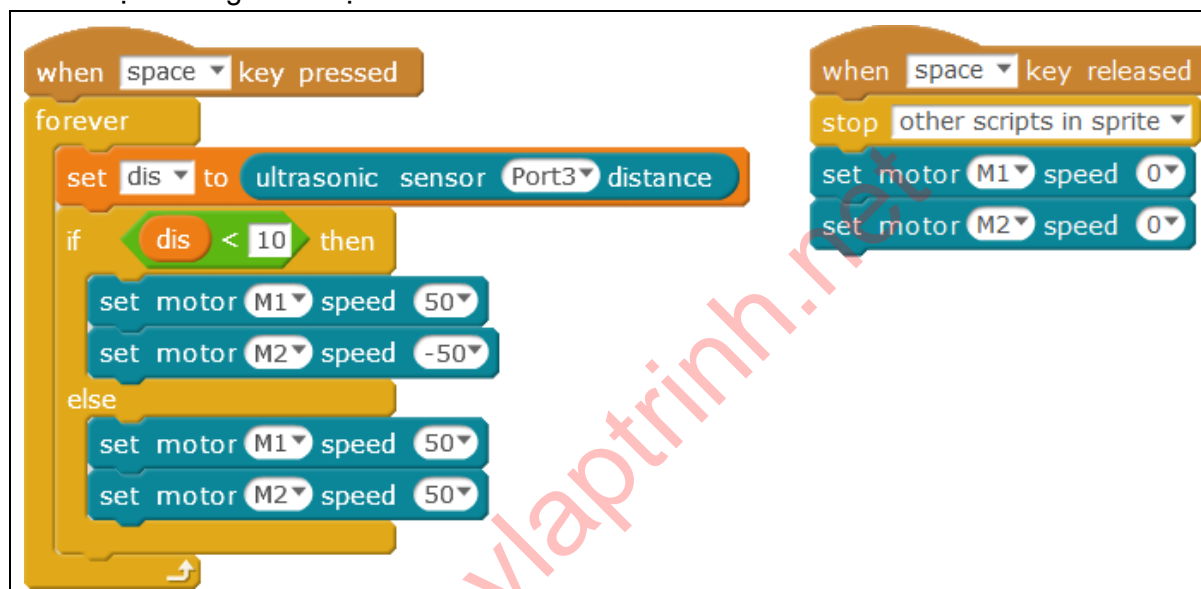
<p>Nếu khoảng cách đo được nhỏ hơn 10cm, 2 động cơ sẽ quay ngược chiều nhau để robot có thể quay phải hoặc quay trái. Ngược lại, 2 động cơ cùng quay về phía trước và robot sẽ tiến về phía trước</p>		 <p>Kiểm tra giá trị khoảng cách nhỏ hơn 10 hay không.</p>  <p>2 động cơ quay ngược nhau và robot quay</p>  <p>2 động cơ quay cùng chiều và robot tiến về phía trước</p>
---	--	---

### Lập trình sự kiện robot dừng lại

<p>Dừng các khối lệnh khác và dừng 2 động cơ</p>		<p>Chạy các khối lệnh khi thả phím cách</p>  <p>Dừng các khối lệnh khác trong chương trình (đoạn</p>
--	---	--

		<p>khởi lệnh kiểm tra giá trị khoảng cách)</p>  <p>Đặt giá trị tốc độ của 2 động cơ về 0 (động cơ dừng lại)</p>
--	--	---

Toàn bộ chương trình thực hành



```

when space key pressed
  forever
    set dis to ultrasonic sensor Port3 distance
    if dis < 10 then
      set motor M1 speed 50
      set motor M2 speed -50
    else
      set motor M1 speed 50
      set motor M2 speed 50
  end
  loop
end

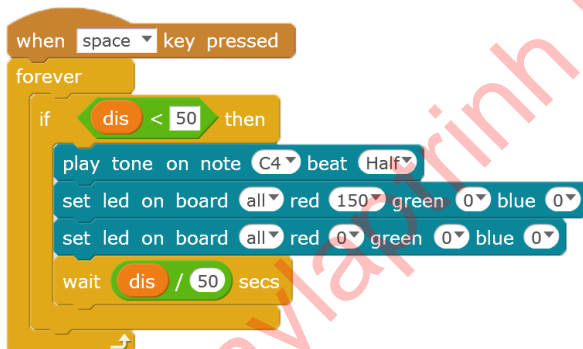
when space key released
  stop other scripts in sprite
  set motor M1 speed 0
  set motor M2 speed 0
  
```

## Lập trình

Lập trình bổ sung thêm vào chương trình phía trên: Khi robot di chuyển về phía trước, nếu phía trước (ví dụ 50cm) có vật cản, còi sẽ kêu và đèn sẽ nhấp để cảnh báo. Càng tiến gần đến vật cản, còi kêu và đèn nhấp càng nhanh.

### Lập trình sự kiện robot dừng lại

Thêm đoạn khối lệnh cảnh báo khi gặp vật cản



when space key pressed

Chạy các khối lệnh khi nhấn phím cách



Dữ liệu liên tục được đo đạc, vì vậy cần được lặp đi lặp lại.



Kiểm tra giá trị khoảng cách nhỏ hơn 50 hay không

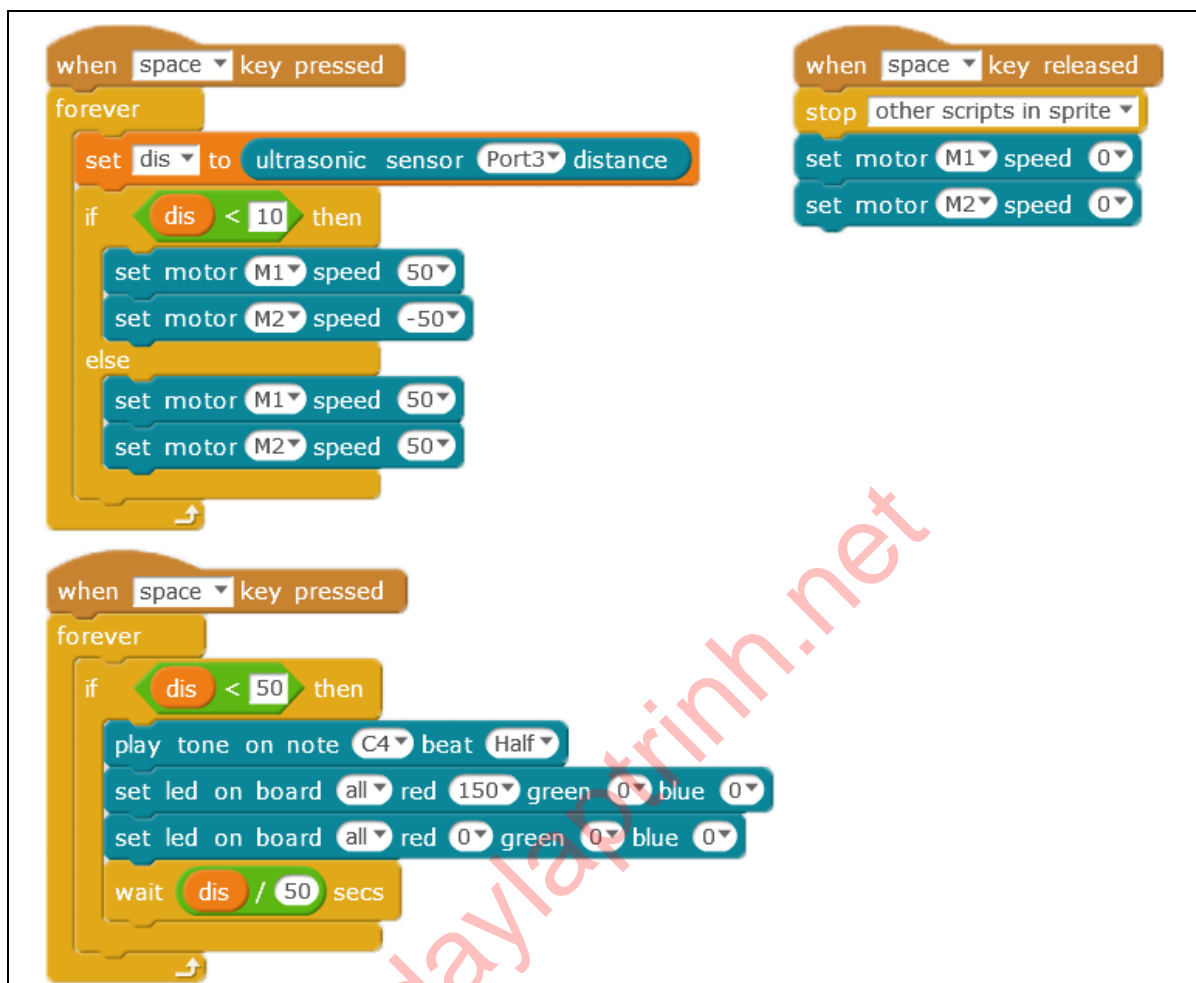


Chơi âm thanh và nhấp đèn LED để cảnh báo



Khi càng đến gần vật cản, thời gian đợi càng giảm

## Toàn bộ chương trình thực hành



### Bài tập

1. Thu âm và chơi âm thanh được thu âm để làm âm thanh cảnh báo.
2. Lập trình robot sẽ quay ngẫu nhiên khi đến gần vật cản.

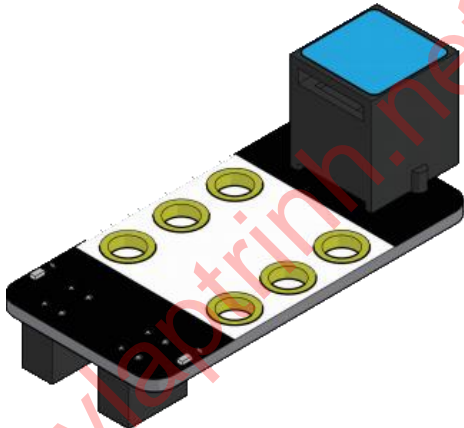
## Bài 11. Con đường thành công

Trong bài học này, chúng ta cùng tìm hiểu làm sao để robot có thể di chuyển theo một đường đi có sẵn.

### Kiến thức lập trình

1. Cảm biến dò đường line-patrolling.
2. Lập trình điều khiển robot dò đường.

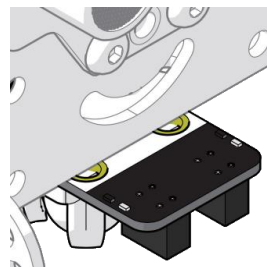
### Module điện tử

Tên	Hình ảnh	Chỉ dẫn
Cảm biến dò đường		Cảm biến dò đường giúp robot di chuyển theo đường đi màu đen trên mặt đất. Phía trước gồm 2 bộ dò. Thông qua sự phản xạ ánh sáng bởi đèn LED với mặt đất, bộ dò có thể xác định vị trí so với đường màu đen.

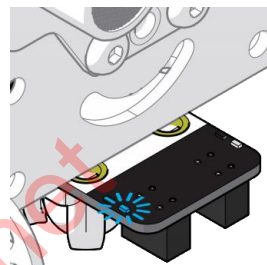
### Kiến thức bổ sung

Nhóm khối lệnh	Khối lệnh	Hướng dẫn
Robots	line follower Port1	Nhận giá trị vị trí của cảm biến dò đường. Có tổng cộng 4 giá trị: 0, 1, 2, 3. Trong đó các giá trị mang ý nghĩa: 0: Cả 2 bộ dò đều ở trong đường đen. 3: Cả 2 bộ dò đều nằm ngoài đường đen.

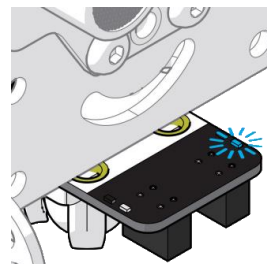
1,2: Một bộ dò nằm trong đường đen, còn bộ còn lại nằm ngoài.



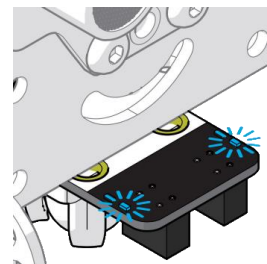
Giá trị 0



Giá trị 1



Giá trị 2



Giá trị 3

## Ý tưởng lập trình

Mô tả ý tưởng	Lưu đồ
<p>Nhấn giữ phím cách (space) để chạy chương trình. Kiểm tra giá trị vị trí cảm biến:</p> <p>Nếu 2 bộ cảm biến đều nằm trên đường đen: Robot tiến về phía trước.</p> <p>Nếu lệch sang bên trái hoặc phải: Quay robot hướng ngược lại để quay trở lại đường đi màu đen.</p> <p>Nếu chệch hẳn khỏi đường đen: Robot lùi lại.</p>	<pre> graph TD     Start([Khi phím cách được nhấn]) --&gt; GetSensors[Lấy giá trị cảm biến dò đường]     GetSensors --&gt; IsOnLine{Robot ở trên đường màu đen?}     IsOnLine -- Đúng --&gt; GoStraight[Đi thẳng]     IsOnLine -- Sai --&gt; DeviatedLeft{Robot lệch sang trái?}     DeviatedLeft -- Đúng --&gt; TurnRight[Quay phải]     DeviatedLeft -- Sai --&gt; DeviatedRight{Robot lệch sang phải?}     DeviatedRight -- Đúng --&gt; TurnLeft[Quay trái]     DeviatedRight -- Sai --&gt; OffLine{Robot ra khỏi đường đen?}     OffLine -- Đúng --&gt; GoBack[Lùi lại]     OffLine -- Sai --&gt; IsOnLine     GoStraight --&gt; IsOnLine     TurnRight --&gt; IsOnLine     TurnLeft --&gt; IsOnLine     GoBack --&gt; IsOnLine     Release([Khi phím cách được thả]) --&gt; StopMotor[Dừng động cơ]     </pre>

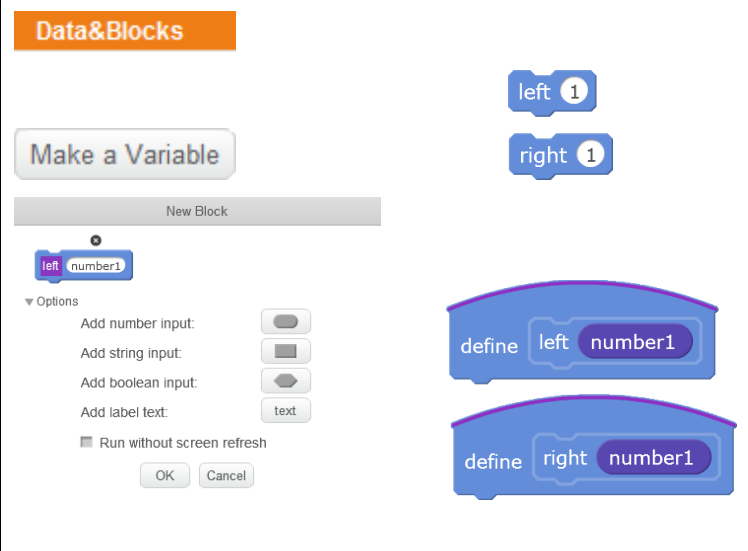
## Thực hành

### 1. Lắp đặt cảm biến dò đường

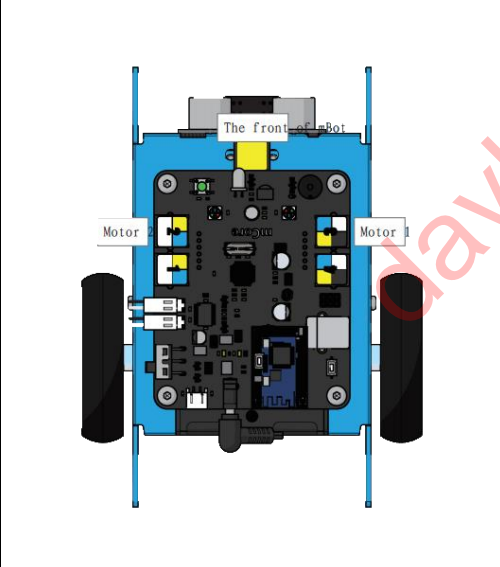
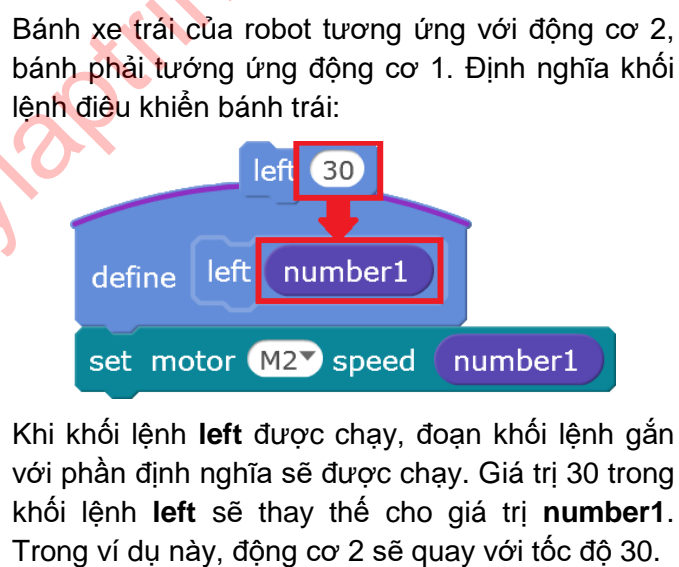
	<p>Cảm biến dò đường được gắn ở phía dưới robot. Hãy chắc chắn rằng hai bộ dò được quay xuống phía dưới.</p>
--	--

## 2. Định nghĩa khối lệnh điều khiển động cơ

Để dễ dàng cho việc học tập, chúng ta tạo và định nghĩa các khối lệnh điều khiển động cơ.




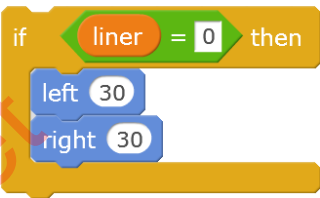
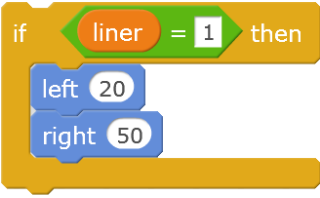
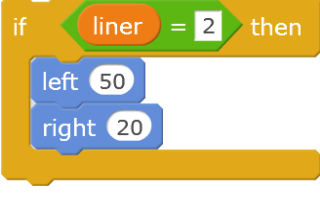
	<p>Trong nhóm <b>Data &amp; Blocks</b> nhấn chọn <b>Make a Block</b></p> <p>Cửa sổ hiện ra, nhấn <b>Add number input</b> (tham số này là tốc độ của động cơ) và đặt lại tên của khối lệnh là <b>left</b>, có nghĩa là khối lệnh này điều khiển động cơ bên trái của robot.</p> <p>Tương tự, tạo ra một khối lệnh tên là <b>right</b> để điều khiển động cơ bên phải.</p>
--	--

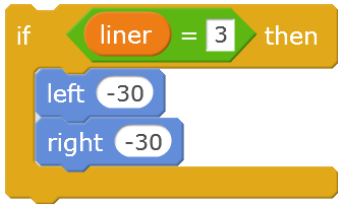
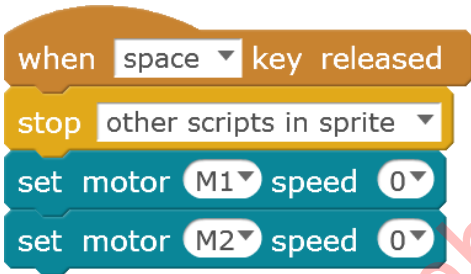


## 3. Robot di chuyển

	<p>Bánh xe trái của robot tương ứng với động cơ 2, bánh phải tương ứng động cơ 1. Định nghĩa khối lệnh điều khiển bánh trái:</p> 
---	--



#### 4. Ghép khối lệnh

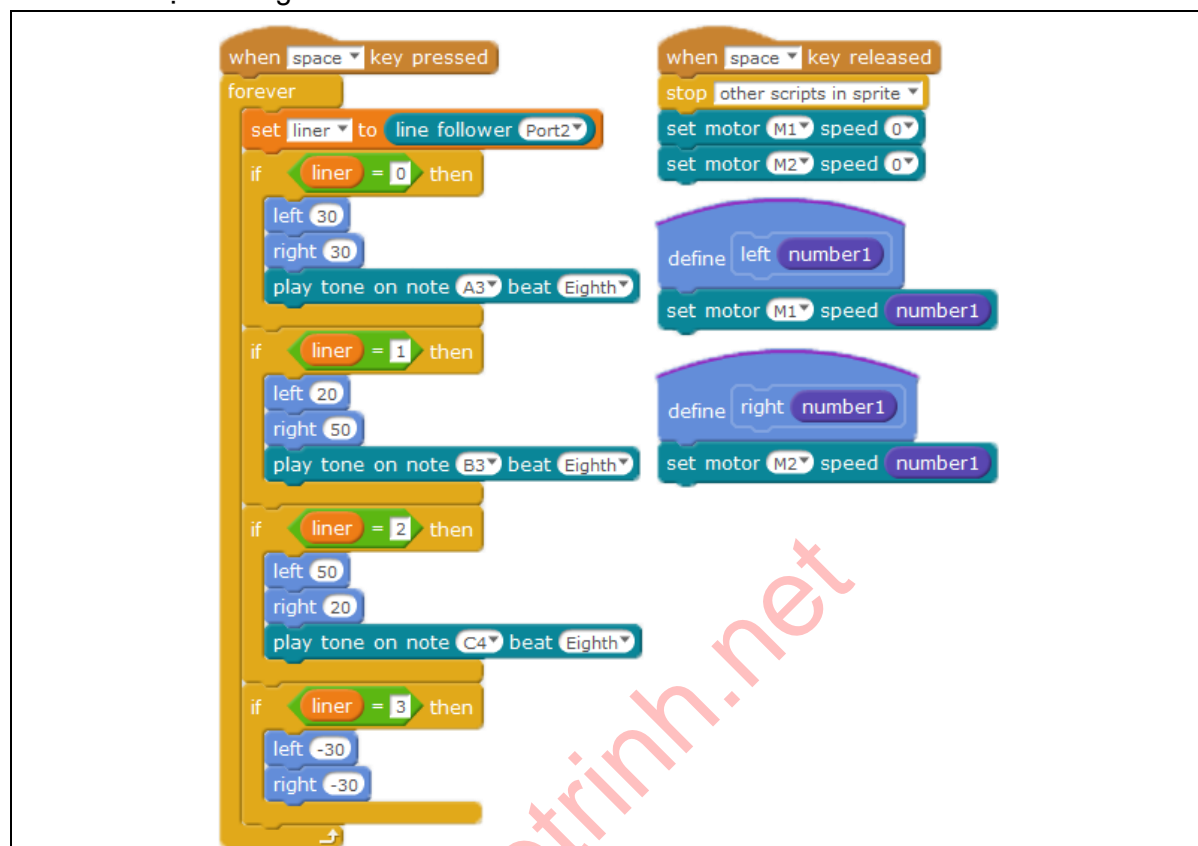
Ý tưởng	Khối lệnh	Mô tả
<p>Khi nhấn giữ phím cách, robot bắt đầu kiểm tra trạng thái của cảm biến dò đường. Tốc độ quay và chiều quay của động cơ phụ thuộc vào vị trí robot so với vạch đen.</p>		<p></p> <p>Nhấn giữ phím cách để kích hoạt.</p> <p></p> <p>Giá trị vị trí xác định bởi cảm biến được gán vào biến <b>liner</b>.</p> <p></p> <p>Nếu <b>liner</b> = 0, robot vẫn đang trên vạch đen, 2 động cơ tiến về phía trước.</p> <p></p> <p>Nếu <b>liner</b> = 1, robot đang bị lệch sang bên phải của vạch đen. Điều khiển robot di chuyển quay trái bằng cách giảm tốc độ động cơ trái, tăng tốc độ động cơ phải</p> <p></p> <p>Nếu <b>liner</b> = 2 thì ngược lại, điều khiển robot quay sang phải.</p>

		 <p>Nếu <b>liner</b> = 3, robot đã hoàn toàn chệch khỏi vạch đen, điều khiển robot lùi lại.</p>
Thả phím cách để động cơ dừng lại.		 <p>Thả phím cách để kích hoạt</p> <p>Ngừng các khối lệnh khác (nếu không thì các khối lệnh vẫn tiếp tục kiểm tra trạng thái)</p>  <p>Dừng động cơ</p>

### Lưu ý

Hãy chắc chắn khi bắt đầu chạy chương trình thì robot xuất phát từ trên vạch đen. Nếu việc dò đường được thực hiện không tốt, ví dụ như thường xuyên bị chệch khỏi đường đi màu đen, thì các bạn có thể tăng độ rộng của đường đen đó lên.

## Toàn bộ chương trình



## Lập trình

Lập trình bổ sung thêm chức năng chơi nốt nhạc ứng với mỗi giá trị xác định bởi cảm biến dò đường.

Ý tưởng	Khối lệnh	Mô tả
Chơi những âm thanh khác nhau ứng với những giá trị khác nhau của cảm biến dò đường	<pre> if liner = 0 then   left 30   right 30   play tone on note C2 beat Half </pre>	<pre> play tone on note C2 beat Half </pre> <p>Robot sẽ chơi những giai điệu khi đi theo đường đi. Các bạn có thể thay đổi những nốt nhạc khác nhau.</p>

## Bài tập

1. Lập trình điều khiển đèn LED để khi dò đường robot có hiệu ứng đẹp mắt hơn.

## Bài 12. Robot ngoan ngoãn

Robot rất thông minh và biết vâng lời. Nó có một bộ thu tín hiệu hồng ngoại, và chúng ta có thể điều khiển robot di chuyển từ xa thông qua bộ thu đó. Trong bài học này chúng ta sẽ tìm hiểu cách để điều khiển robot thông qua bộ thu tín hiệu hồng ngoại.


### Kiến thức lập trình

1. Điều khiển robot bằng bộ điều khiển từ xa hồng ngoại.

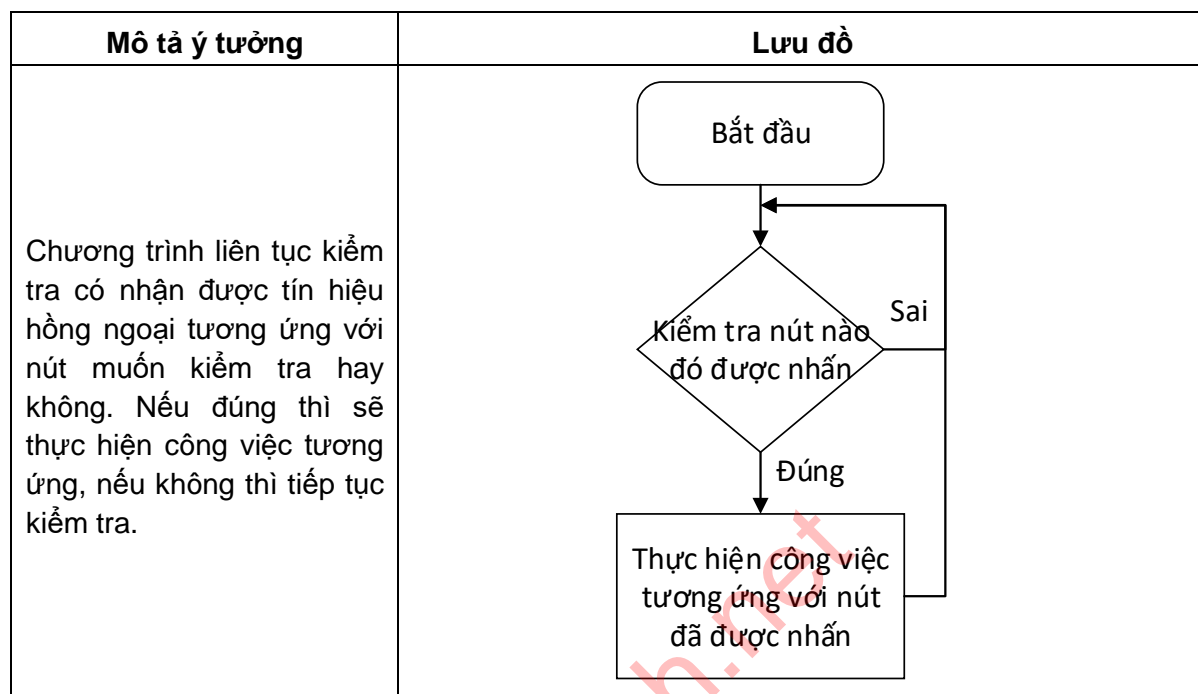
### Module điện tử

Tên	Hình ảnh	Chỉ dẫn
Điều khiển từ xa hồng ngoại		Truyền tín hiệu hồng ngoại đã được mã hóa. Tín hiệu này được nhận và xử lý bởi bộ thu hồng ngoại trên robot.

### Kiến thức bổ sung

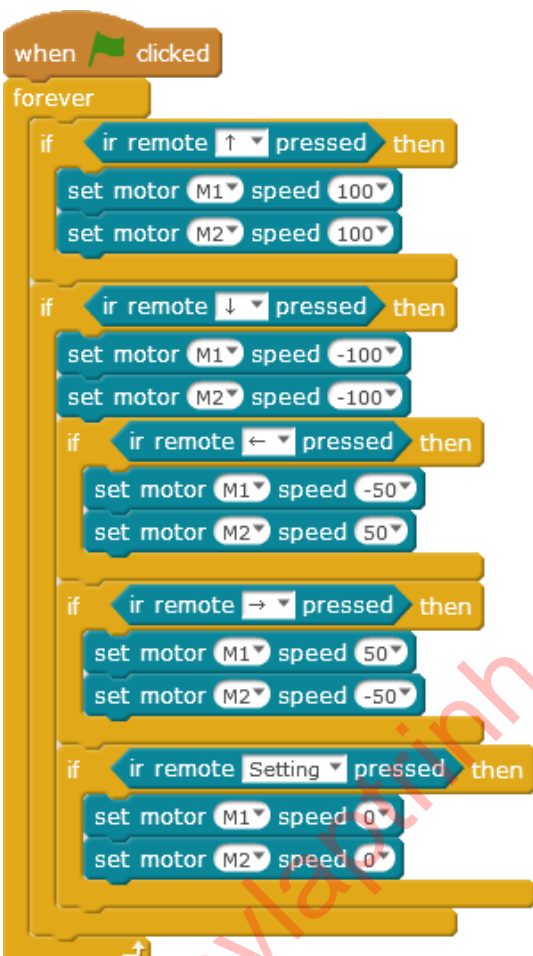

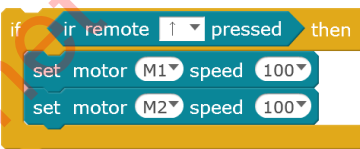
Khối lệnh	Mô tả	Ví dụ
<b>Robots</b>		Nhận thông tin về nút được bấm trên điều khiển từ xa hồng ngoại

## Ý tưởng lập trình



## Thực hành

Sử dụng các nút mũi tên để điều khiển robot di chuyển

	Khối lệnh	Mô tả
Chương trình liên tục kiểm tra tín hiệu nhận được. Nếu nhận được tín hiệu là nút mũi tên thì sẽ điều khiển rô-tơ di chuyển tương ứng.		<p>Nhấn lá cờ xanh để chạy chương trình</p>  <p>Dùng vòng lặp để liên tục kiểm tra tín hiệu.</p>  <p>Nhận biết thông tin phím trên điều khiển được nhấn và thực hiện công việc tương ứng.</p>

## Lưu ý

Bạn có thể lập trình các phím khác để thực hiện công việc mong muốn, ví dụ như nhấn phím A để chơi nhạc.

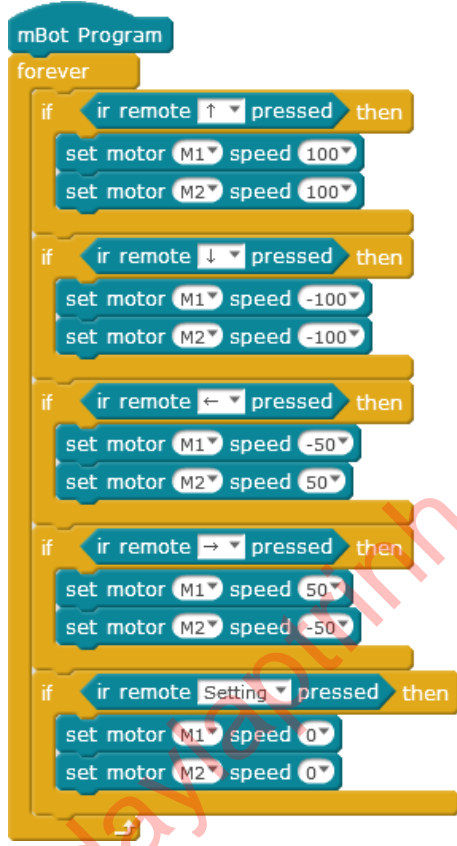
## Lập trình

Chúng ta có thể sử dụng các phím mũi tên trên điều khiển hồng ngoại để điều khiển robot di chuyển. Tuy nhiên công việc này được thực hiện khá phức tạp: Khi nhấn phím điều khiển, tín hiệu sẽ được thu nhận bởi bộ thu hồng ngoại trên robot, sau đó được truyền đến phần mềm **mBlock** trên máy vi tính thông qua Bluetooth. Sau khi xử lý tín hiệu, **mBlock** lại truyền tín hiệu điều khiển động cơ thông qua Bluetooth đến robot.

Chúng ta sẽ rút ngắn quá trình này bằng cách nạp chương trình trực tiếp lên robot, robot sẽ không cần kết nối với máy tính.

Sử dụng khối lệnh **mBot Program** trong nhóm **Robots** để thay thế khối lệnh **Khi nhấn vào lá cờ xanh**.

Nạp mã lệnh lên robot.



The Scratch code for the mBot Program is as follows:

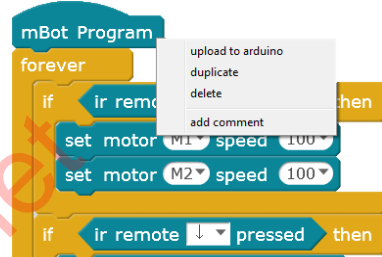
```

mBot Program
forever
  if ir remote ↑ pressed then
    set motor M1 speed 100
    set motor M2 speed 100
  if ir remote ↓ pressed then
    set motor M1 speed -100
    set motor M2 speed -100
  if ir remote ← pressed then
    set motor M1 speed -50
    set motor M2 speed 50
  if ir remote → pressed then
    set motor M1 speed 50
    set motor M2 speed -50
  if ir remote Setting pressed then
    set motor M1 speed 0
    set motor M2 speed 0
        
```

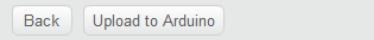
**mBot Program**

Cho phép mã lệnh của chương trình được nạp lên bộ xử lý của robot.

Nhấn chuột phải và chọn “Upload Arduino program”



Nhấn nút “Upload to Arduino”



```

1 #include <Arduino.h>
2 #include <Wire.h>
3 #include <SoftwareSerial.h>
4
5 #include <MeMCore.h>
6
7 MeDCMotor motor_9(9);
8 MeDCMotor motor_10(10);
9 void move(int direction, int speed)
        
```

Chú ý: Khi nạp mã lệnh Arduino program, robot cần được kết nối với máy tính qua cáp USB.

## Lưu ý

Khi chương trình được nạp thành công, chúng ta có thể điều khiển robot bằng điều khiển hồng ngoại. Nếu sử dụng một số khối lệnh không phù hợp (ví dụ như vẽ, in hình...) thì một cửa sổ thông báo sẽ hiện ra và chỉ ra đó là khối lệnh nào.

## Bài tập

1. Lập trình điều khiển còi và đèn LED trên robot, sau đó nạp vào robot.

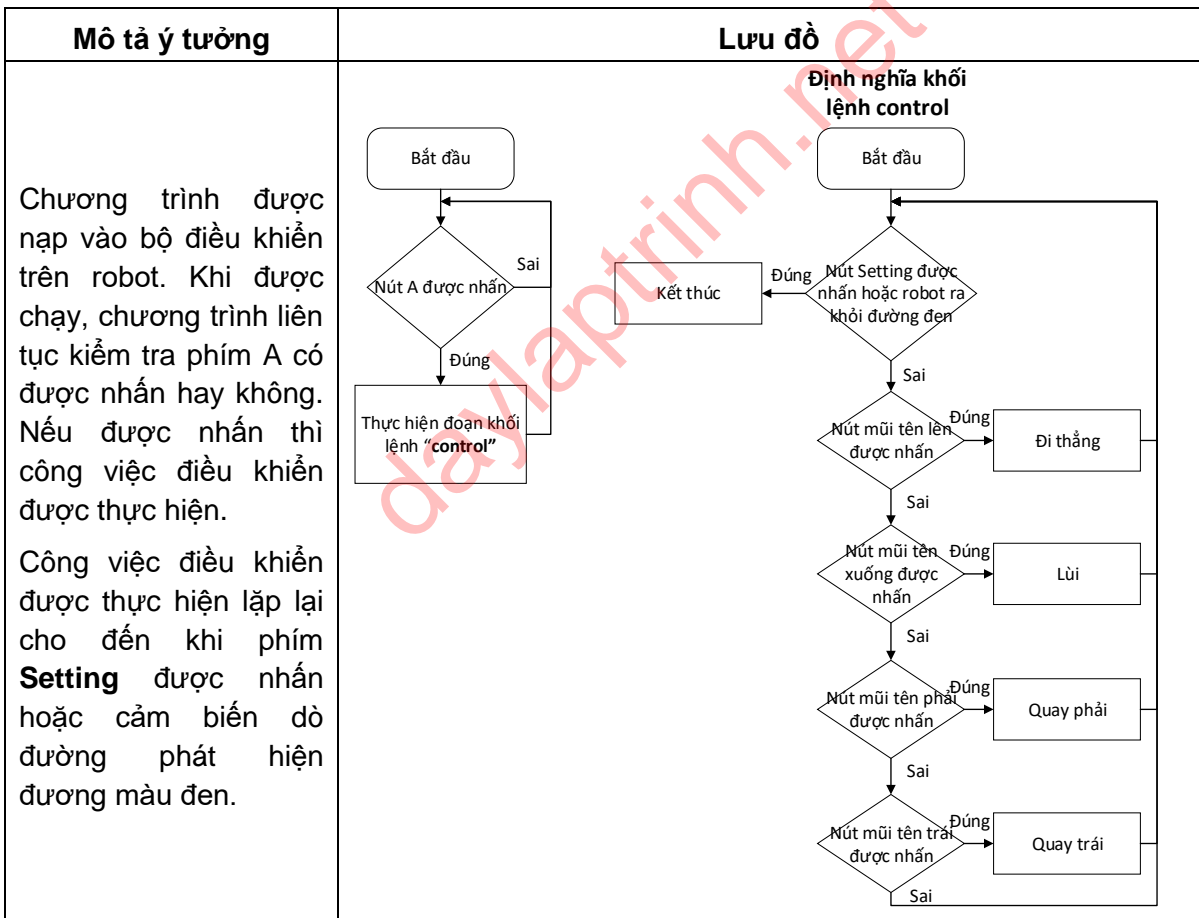
## Bài 13. Robot linh hoạt

Robot đã được học rất nhiều kỹ năng như tránh vật cản, dò đường, xử lý tín hiệu hồng ngoại. Trong bài học này, robot sẽ được lập trình để có tất cả các kỹ năng đó.

### Kiến thức lập trình

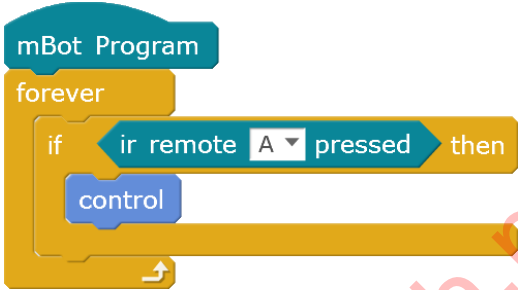

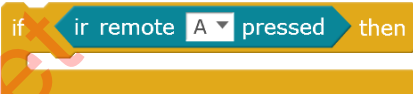


1. Nhấn phím A để chuyển sang chế độ điều khiển từ xa, sử dụng phím mũi tên để điều khiển robot tiến, lùi, quay. Nếu phía trước 10cm có vật cản thì robot sẽ tự động quay.
2. Nếu nhấn nút điều khiển trên điều khiển hồng ngoại hoặc cảm biến dò đường chạm vạch đen, chuyển sang chế độ dò đường.

### Ý tưởng lập trình

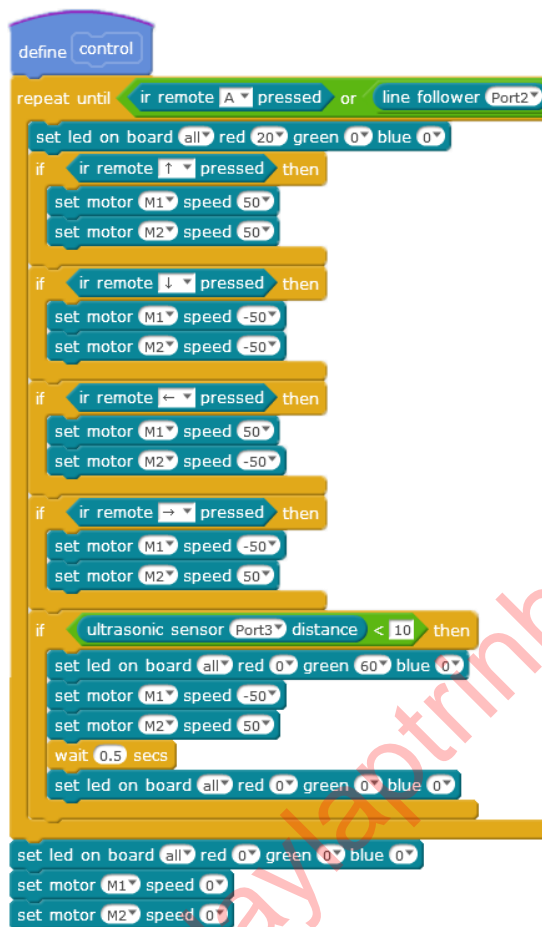




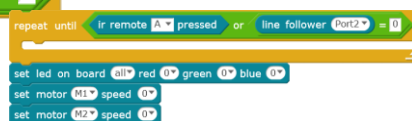
## Thực hành

<p>Trong vòng lặp, chương trình kiểm tra nếu nút A trên điều khiển hồng ngoại được nhấn thì thực hiện đoạn công việc điều khiển</p>		<p><b>mBot Program</b></p> <p>Chương trình có thể được nạp vào robot để robot có thể chạy mà không cần kết nối với máy tính.</p>  <p>Chương trình cần phải được chạy bên trong vòng lặp.</p>  <p>Kiểm tra nút A được nhấn</p>  <p>Định nghĩa khối lệnh điều khiển thực hiện công việc điều khiển.</p> <div data-bbox="915 1068 1336 1230"> <p>New Block</p>  <p>Options</p> <p>OK Cancel</p> </div>
---	---	---

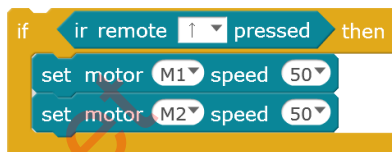
Công việc điều khiển được lặp lại cho đến khi nút **Setting** được nhấn hoặc cảm biến dò đường tìm thấy đường. Trong vòng lặp chương trình sẽ xác định nút nào được nhấn và kiểm tra 10cm phía trước có vật cản hay không.



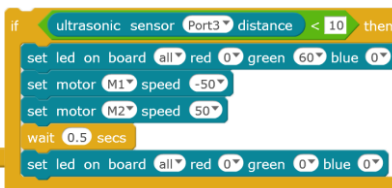
Định nghĩa khối lệnh “control”



Lặp lại cho đến khi nút “Setting” được nhấn hoặc cảm biến dò đường gặp vạch đen.



Mỗi phím trên điều khiển hồng ngoại tương ứng với hướng di chuyển khác nhau.



Nếu phía trước 10cm có vật cản, đèn LED chuyển màu xanh và robot đổi hướng.

## Bài tập

1. Lập trình để khi nhấn phím B trên điều khiển hồng ngoại, robot chuyển sang chế độ dò đường.



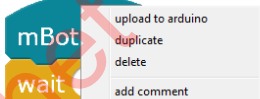


## Bài 14. Nhà vô địch đua xe

Cùng tổ chức một cuộc thi, mỗi bạn sẽ sử dụng một robot để tham gia thi đấu. Hãy xem robot ai có thể lập trình cho robot đi xa nhất.

### Kiến thức lập trình

1. Tải chương trình điều khiển lên robot và hoàn thành cuộc đua.

### Kiến thức bổ sung

Khối lệnh	Mô tả	Ví dụ
 	Tải chương trình vào robot	<p>Nhấn chuột phải vào khối lệnh.</p>  <p>play tone on note C4 beat Half</p> <p>set count to 0</p> <p>reset timer</p> <p>Chọn “Upload Arduino Program”.</p>
 	Sử dụng hàm thời gian	<p><b>Timer</b> là một hàm có sẵn trong mạch Arduino, tương đương với biến <b>timer</b> của <b>mBlock</b>.</p>
	Điều chỉnh hướng và tốc độ	Điều khiển hướng và tốc độ quay của 2 động cơ.

## Ý tưởng lập trình

Mô tả ý tưởng	Lưu đồ
<p>Quãng đường robot di chuyển tỉ lệ thuận với số lần nút trên bảng mạch được nhấn.</p> <p>Các bước:</p> <ol style="list-style-type: none"> <li>1. Khởi tạo các trạng thái và giá trị ban đầu.</li> <li>2. Ghi lại số lần nút được nhấn trong 10 giây.</li> <li>3. Khi đèn xanh được sáng, đưa robot đến vạch xuất phát. Nhấn nút thêm một lần nữa để robot bắt đầu di chuyển cho đến khi kết thúc.</li> </ol>	<pre> graph TD     Start([Bắt đầu]) --&gt; Init[Đặt count = 0 Đặt lại thời gian]     Init --&gt; Time{Thời gian &gt; 60}     Time -- Sai --&gt; Button{Nút trên bảng mạch được nhấn}     Button -- Đúng --&gt; Count[Tăng count thêm 1]     Count --&gt; Time     Button -- Sai --&gt; Wait[Đợi đến khi nút trên bảng mạch được nhấn]     Wait --&gt; Move[Tiến về phía trước]     Move --&gt; Delay[Đợi với số giây bằng giá trị biến count]     Delay --&gt; Stop[Dừng động cơ]     Stop --&gt; End([Kết thúc])     </pre>

## Thực hành

	Khối lệnh	Mô tả
<p>Khởi tạo giá trị cho biến.</p>	<pre> mBot Program wait 3 secs play tone on note C6 beat Half set count to 0 reset timer         </pre>	<p><b>set count to 0</b></p> <p>Biến <b>count</b> ghi lại số lần nút được nhấn. Nếu không đặt lại giá trị 0 thì biến sẽ được nhấn giá trị từ lần chạy trước.</p> <p><b>reset timer</b></p> <p>Đặt lại thời gian</p>

<p>Đếm số lần nút được nhấn trong 10 giây.</p>		<p>repeat until timer &gt; 10</p> <p>Vòng lặp kết thúc khi thời gian chạy được 10 giây.</p>  <p>Ghi lại số lần nút được nhấn.</p>
<p>Hết 10 giây, đèn xanh được bật. Nhấn nút một lần nữa để robot bắt đầu di chuyển</p>		 <p>Đợi cho đến khi nút được nhấn một lần nữa.</p> <p>Số lần nút được nhấn càng nhiều, thời gian di chuyển càng lâu.</p>

### Đoạn khối lệnh và mã lệnh hoàn chỉnh

	<pre>#include &lt;Arduino.h&gt; #include &lt;Wire.h&gt; #include &lt;SoftwareSerial.h&gt;  #include &lt;MeMCore.h&gt;  MeDCMotor motor_9(9); MeDCMotor motor_10(10); void move(int direction, int speed) {     int leftSpeed = 0;     int rightSpeed = 0;     if(direction == 1){         leftSpeed = speed;         rightSpeed = speed;     }else if(direction == 2){         leftSpeed = -speed;         rightSpeed = - speed;</pre>
--	--

## mBot Program

wait 3 secs

play tone on note C6 beat Half

set count to 0

reset timer

repeat until timer &gt; 10

if on board button pressed then

change count by 1

wait until on board button released

play tone on note C4 beat Half

set led on board all red 0 green 60 blue 0

wait until on board button pressed

run forward at speed 100

wait count secs

run forward at speed 0

set led on board all red 0 green 0 blue 0

```

    }else if(direction == 3){
        leftSpeed = -speed;
        rightSpeed = speed;
    }else if(direction == 4){
        leftSpeed = speed;
        rightSpeed = -

```

speed;

}

motor\_9.run((9)==M1?-

(leftSpeed):(leftSpeed));

motor\_10.run((10)==M1?-

(rightSpeed):(rightSpeed));

}

double angle\_rad = PI/180.0;

double angle\_deg = 180.0/PI;

double count;

MeBuzzer buzzer;

double currentTime = 0;

double lastTime = 0;

double getLastTime(){

return currentTime =

millis()/1000.0 - lastTime;

}

MeRGBLed rgbled\_7(7, 7==7?2:4);

void setup(){

delay(3);

buzzer.tone(1047, 500);

delay(20);

count = 0;

lastTime = millis()/1000.0;

pinMode(A7, INPUT);

```

while(!((getLastTime()) >
(10)))

```

{

\_loop();

```

if((0^(analogRead(A7)>10?0:1))){
    count += 1;

```

```

while(!((1^(analogRead(A7)>10?0:1)
)))

```

{

\_loop();

}

}

buzzer.tone(262, 500);

delay(20);

rgbled\_7.setColor(0,0,60,0);

rgbled\_7.show();

```

while(!((0^(analogRead(A7)>10?0:1)
)))

```

```
{
    _loop();
}
move(1,100);
_delay(count);
move(1,0);
rgblcd_7.setColor(0,0,0,0);
rgblcd_7.show();
}

void loop(){
    _loop();
}

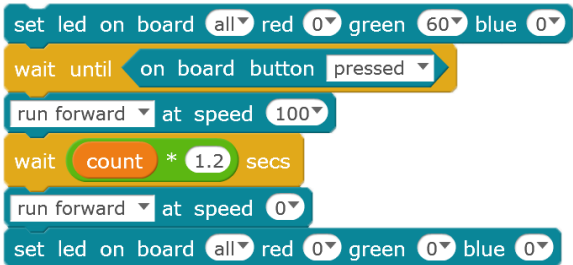
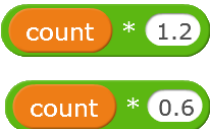
void _delay(float seconds){
    long endTime = millis() +
seconds * 1000;
    while(millis() <
endTime) _loop();
}

void _loop(){
}
```

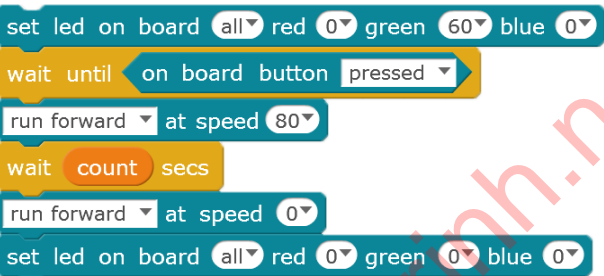

Nhấn nút “Upload to Adruino” để nạp mã lệnh vào robot

The screenshot shows the mBlock software interface. On the left, the 'Scripts' panel displays a Scratch script for an mBot robot. The script includes a 'wait 2 secs' block, a 'play tone on note C6 beat Half' block, a 'set count to 0' block, a 'reset timer' block, a 'repeat until timer > 10' loop, an 'if on board button pressed then' block with 'change count by 1' and 'wait until on board button released' blocks, a 'play tone on note C4 beat Half' block, a 'set led on board all red 0 green 60 blue 0' block, a 'wait until on board button pressed' block, a 'run forward at speed 100' block, a 'wait count secs' block, a 'run forward at speed 0' block, and a 'set led on board all red 0 green 0 blue 0' block. On the right, the 'Arduino' panel shows the C++ code for the mBot robot. The code includes headers for Arduino, Wire, SoftwareSerial, and MeRCore, defines two MeRCoreMotor objects (motor\_9 and motor\_10), and implements a move function that takes direction and speed as arguments. The main loop calls the move function with direction 1 and speed 100. The 'Upload to Arduino' button is highlighted with a red arrow.

## Thay đổi khoảng cách di chuyển mỗi giây

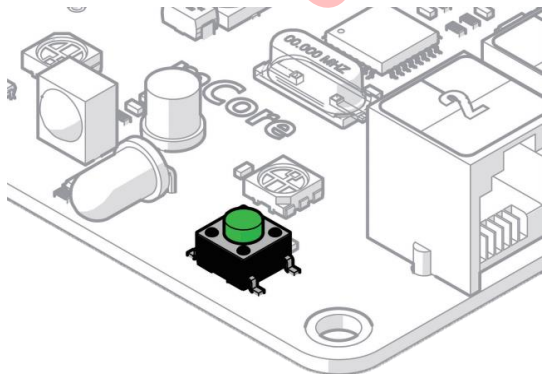
Thay đổi thời gian đợi trong chương trình		 <p>Thay đổi thời gian đợi và theo dõi sự thay đổi.</p>
---	---	---

## Thay đổi tốc độ robot

Thay đổi tốc độ động cơ		 <p>Thay đổi tốc độ bằng cách thay đổi số trong khối lệnh.</p>
-------------------------	---	--

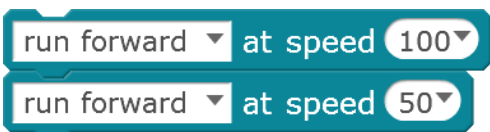
## Lập trình

### 1. Cài đặt

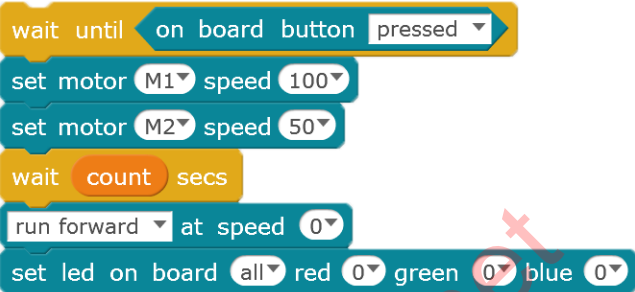
	<p>Xác định số lần nút trên bảng mạch được nhấn.</p>
---	--



## 2. Khởi lệnh điều chỉnh tốc độ

Thay đổi tốc độ.		Thay đổi tốc độ động cơ của robot.
------------------	---	------------------------------------

## 3. Lập trình

Di chuyển theo đường cong.		Thay đổi tốc độ động cơ để robot di chuyển theo đường cong.
----------------------------	--	---

## Bài tập

1. Lập trình xử lý trường hợp trên đường di chuyển có vật cản.

# TẠI SAO TRẺ EM NÊN HỌC LẬP TRÌNH ?



## Giải pháp kết hợp học và chơi

Thay vì để trẻ em ngồi ở nhà chơi điện tử, hãy cho các em học lập trình để tạo ra những sản phẩm cho riêng mình.

## Hội chứng tăng động, giảm chú ý

Góp phần giảm thiểu bệnh lý tăng động giảm chú ý ở trẻ em. Giúp các em tập trung học tập đạt kết quả cao.

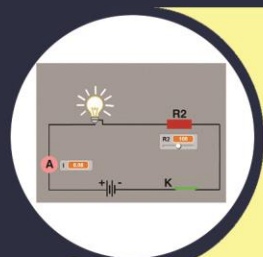


## Tư duy logic - Sáng tạo

Việc suy nghĩ theo giải thuật của khoa học máy tính giúp những đứa trẻ biết tư duy, giải quyết vấn đề mang tính logic và sáng tạo.

## Làm việc nhóm

Học lập trình thúc đẩy các em chia sẻ, phối hợp với nhau để hoàn thành công việc. Quá trình làm việc đòi hỏi các em giao tiếp với nhau, trao đổi ý tưởng.



## Công cụ mới hỗ trợ quá trình học tập

Lập trình là một công cụ đa năng, hỗ trợ hầu hết các lĩnh vực trong cuộc sống. Ta hoàn toàn có thể mô phỏng, giải các bài toán trong các môn học phổ thông như toán học, vật lý, sinh học, hóa học,...