MOONHACK 2020

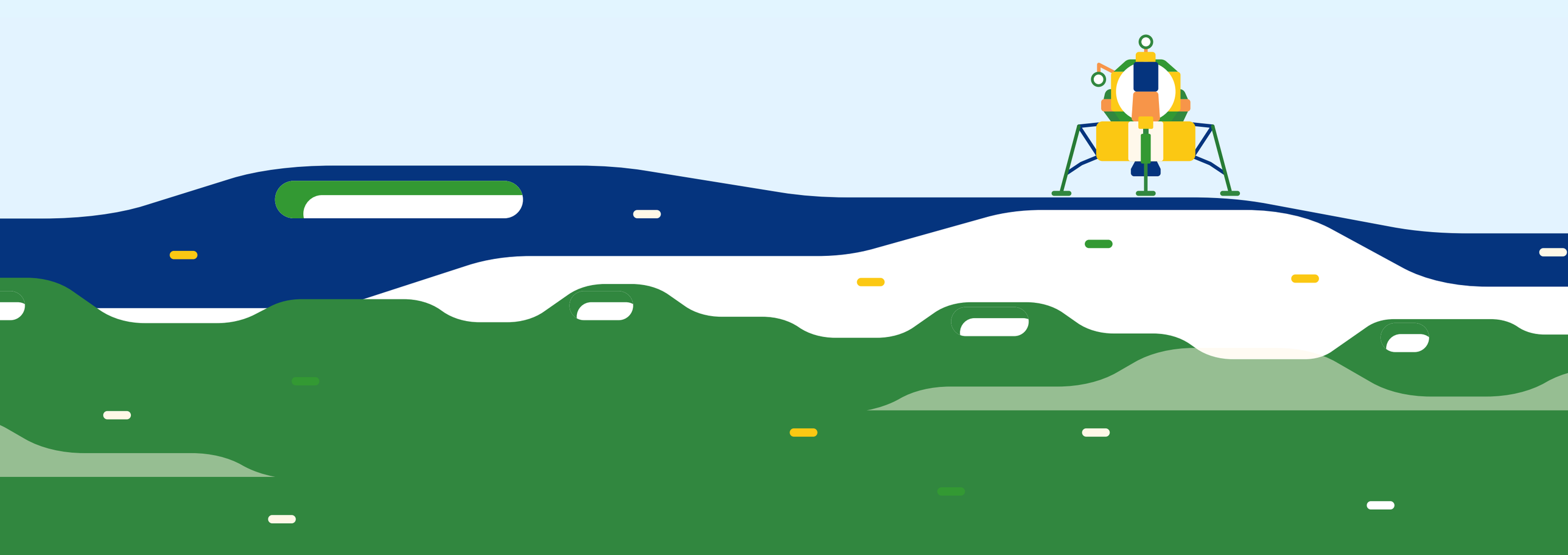
PYTHON WATER ON MARS

VIETNAMESE

BROUGHT TO YOU BY CODE CLUB AUSTRALIA
POWERED BY TELSTRA FOUNDATION

code club / AUSTRALIA | POWERED BY TELSTRA FOUNDATION

SUBMIT AND BE COUNTED AT
MOONHACK.COM



Nước trên sao Hỏa - Dự án Python Moonhack

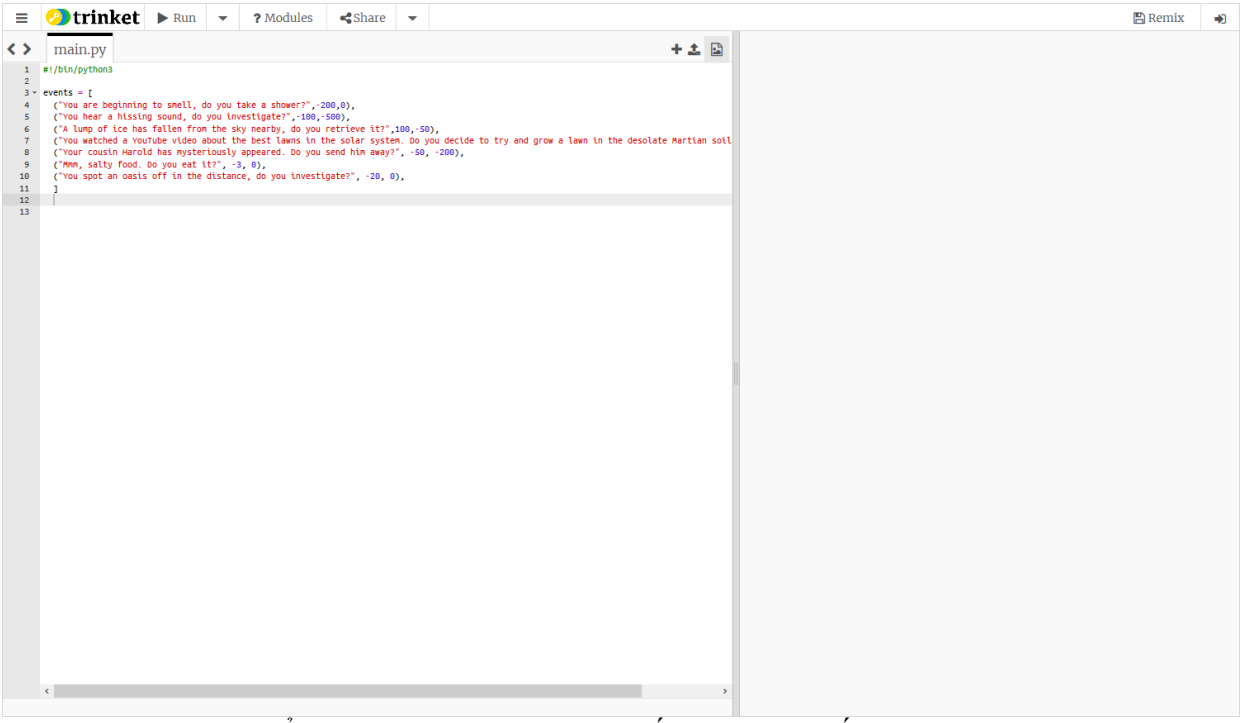
Giới thiệu

Nước là một nguồn tài nguyên vô cùng quý giá rất quan trọng cho sự sống. Việc bảo tồn nước là rất quan trọng, và sẽ càng quan trọng hơn khi loài người du hành vào không gian. Hôm nay chúng ta sẽ thực hiện một trò chơi sinh tồn, trong đó người chơi phải đưa ra quyết định để tồn tại càng lâu càng tốt mà không bị cạn nước trên sao Hỏa.

Bước 1: Theo dõi nước

Trong bước đầu tiên này, chúng ta sẽ bắt đầu trò chơi và thiết lập một biến mà chúng ta có thể sử dụng để theo dõi lượng nước chúng ta có.

- Mở dự án Scratch tại bit.ly/pythonwater . Cửa sổ của bạn sẽ trông như thế này:



Bên trái là cửa sổ mã của bạn; bạn sẽ nhận thấy đã có một số mã ở đây (chúng ta sẽ quay lại sau). Bên phải là cửa sổ thực hiện; đây là nơi kết quả của trò chơi của bạn sẽ hiển thị. Trên cùng là nút Run; bạn sẽ sử dụng nút này để chạy mã của bạn.

- Chúng ta cần phải cung cấp cho người chơi một lượng nước để bắt đầu. Hãy bắt đầu với số 1000 đẹp mắt. Thêm mã được tô sáng sau vào cuối chương trình của bạn:

```
10    ("You spot an oasis
11    ]
12
13  water = 1000
```

- Chúng ta muốn người dùng của chúng ta tiếp tục chơi trò chơi miễn là họ có nước. Khi họ hết nước, trò chơi sẽ kết thúc. Hãy tạo vòng lặp chính của trò chơi, sao cho nó sẽ tiếp tục miễn là người dùng có nước:

```
13  water = 1000
14
15  while water > 0:
```

Lưu ý dấu hai chấm (:) ở cuối dòng. Ký tự nhỏ bé này có vẻ không đáng kể, nhưng nếu chúng ta quên nó, Python sẽ phàn nàn!

- Tiếp theo, hãy thêm một số đại diện cho nước đã sử dụng vào vòng lặp. Đây sẽ là một số được trừ mỗi ngày, bất kể lựa chọn của người chơi, bởi vì người chơi cần phải uống. Thêm

mã sau vào vòng lặp của bạn:

```
15 while water > 0:
16   water -= 5
```

Lưu ý hai khoảng trắng ở trước dòng 16. Chúng tôi gọi đây là **lùi đầu dòng** . Điều này nói với python rằng đoạn mã trên dòng đó nằm trong vòng lặp. Nói cách khác, điều này xảy ra mỗi khi chúng ta lặp lại vòng lặp (nếu bạn đã sử dụng Scratch, nó tương tự như các khối vòng lặp màu cam).

- Bây giờ là lúc cho kết quả. Hãy nói với người dùng rằng trò chơi đã kết thúc khi họ hết nước. Thêm mã sau bên ngoài vòng lặp của bạn (hãy nhớ không lùi đầu dòng):

```
15 while water > 0:
16   water -= 5
17
18 print ("Game Over :(")
```

- Chạy mã của bạn. Chuyện gì sẽ xảy ra?
- Python sẽ chỉ cố chạy mã của chúng ta càng nhanh càng tốt. Không có bất kỳ kết quả nào trong vòng lặp chính, chúng ta không có cách nào biết được chuyện gì đang xảy ra cho đến khi kết thúc. Hãy thêm một số đầu ra vào vòng lặp chính của chúng ta.

```
15 while water > 0:
16   water -= 5
17   print("A day has passed.")
18
19 print ("Game Over :(")
```

- Chạy mã của bạn lần nữa. Bây giờ bạn sẽ thấy tất cả những ngày trôi qua trước khi trò chơi kết thúc.
- Chúng ta hãy thay đổi dòng mã cuối cùng mà chúng ta đã chèn để biết chúng ta đã thêm bao nhiêu nước. Chúng ta có thể kết hợp giá trị của các biến với văn bản chúng ta đang in bằng dấu +.

```
15 while water > 0:
16   water -= 5
17   print("A day has passed. Your Water " + water + "L")
18
19 print ("Game Over :(")
```

- Chạy mã của bạn lần nữa. Bạn có thấy lỗi xuất hiện không?
- Phần lớn của việc lập trình là cố gắng tìm ra lý do tại sao chúng ta gặp những lỗi mà chúng ta gặp phải. Trong trường hợp này, lỗi đang cho chúng ta biết rằng chúng ta không thể ghép

các đối tượng 'str' và 'int'. Khi lập trình, bạn thường sẽ phải tìm ra ý nghĩa của những thông báo lỗi lạ này.
Lỗi này cho chúng ta biết rằng chúng ta đang cố gắng kết hợp hai thứ khác nhau mà Python không biết cách kết hợp, văn bản (chuỗi, rút ngắn thành str) và số chúng ta đang sử dụng để theo dõi lượng nước của chúng ta (số nguyên, rút ngắn thành int). Để khắc phục điều này, chúng ta có thể yêu cầu Python chuyển đổi số nguyên của chúng ta thành một chuỗi với mã sau:

```
15 while water > 0:
16   water -= 5
17   print("A day has passed. Your Water " + str(water) + "L")
18
19 print ("Game Over :(")
```

- Chạy mã của bạn. Bạn sẽ nhận được một kết quả cho chúng ta biết đang sử dụng bao nhiêu nước mỗi ngày, trước khi nhân vật của người chơi hết nước.

Thách thức: Thay đổi lượng nước sử dụng hằng ngày

Hiện tại, chúng ta đang sử dụng 5 lít nước mỗi ngày. Bạn nghĩ bao nhiêu là mức tối thiểu một người cần để tồn tại? Bạn có thể thay đổi mã để người chơi sử dụng lượng nước đó không?

Bước 2: Theo dõi ngày

Đến giờ, chúng ta đã theo dõi lượng nước mà một người chơi đã sử dụng, nhưng để xem họ đã làm tốt như thế nào, chúng ta sẽ theo dõi số ngày đã trôi qua.

- Đầu tiên, chúng ta sẽ đặt giá trị ban đầu của biến ngày, bắt đầu từ ngày 1.

```
13 water = 1000
14 day = 1
15
16 while water > 0:
```

- Tiếp theo, mỗi ngày trôi qua, chúng ta muốn cộng thêm 1 ngày cho ngày hiện tại. Thêm mã sau vào vòng lặp của bạn:

```
16 while water > 0:
17   water -= 5
18   print("A day has passed. Your Water " + str(water) + "L")
19   day += 1
20
21 print ("Game Over :(")
```

- Cuối cùng, chúng ta sẽ cho người chơi biết họ đã làm tốt như thế nào. Chúng ta sẽ làm điều này bằng cách thay thế câu lệnh in cuối cùng bằng câu lệnh bao gồm số ngày đã trôi qua.

```
19    day += 1
20
21  print("you lasted " + str(day) + " days")
```

Lưu ý rằng chúng ta phải sử dụng lại str (day) để chuyển đổi số thành văn bản.

- Chạy mã của bạn. Bạn sẽ nhận được một đầu ra cho mỗi ngày, và sau đó là một đầu ra cuối cùng cho biết người chơi đã trải qua bao nhiêu ngày.

Thử thách: Xuất số ngày mỗi ngày

Hiện tại, chúng ta chỉ sử dụng biến ngày lúc cuối cùng. Bạn có thể thay đổi mã để xuất số ngày mỗi ngày không?

Hiện tại, đầu ra sẽ cho biết:

```
Một ngày đã trôi qua. Lương nước bạn có 950L
```

Bạn có thể thay đổi để xuất ra như sau không:

```
Ngày: 10. Lương nước bạn có 950L
```

Bước 3: Sự kiện hàng ngày của bạn

Bây giờ chúng ta sẽ thêm một số trò chơi. Mỗi ngày, chúng ta sẽ cung cấp cho người chơi một lựa chọn và lựa chọn đó sẽ quyết định lượng nước họ mất hoặc thu được.

- Bạn có thể nhận thấy rằng có cả đống mã ở phía trên không phải do bạn viết. Đây là danh sách các sự kiện có thể diễn ra trên Sao Hỏa. Hãy thêm một câu lệnh in vào vòng lặp chính của chúng ta. Điều này sẽ truy cập danh sách sự kiện của chúng tôi và chọn một sự kiện ngẫu nhiên.

```
18    print("A day has passed. Your Water " + str(water) + "L")
19    print(random.choice(events))
20    day += 1
```

Ồ không! Một lỗi khác. Cái này chỉ ra "random" không được định nghĩa. Đó là vì "random" không phải là một phần của python theo mặc định, nó là một phần của mô đun . Chúng ta cần nhập nó để có thể sử dụng nó.

```
1   #!/bin/python3
2   import random
3
4   events = [
```

- Chạy mã của bạn. Mỗi ngày, bạn nên có một sự kiện ngẫu nhiên được in ra.
- Chúng ta sẽ cần sử dụng sự kiện nhiều lần vào một ngày nhất định. Nếu chúng ta sử dụng Random.choice (events) mỗi lần chúng ta muốn sử dụng sự kiện, chúng ta sẽ nhận được một

sự kiện khác nhau! Thay vào đó, hãy lưu trữ sự kiện đó trong một biến gọi là event mà chúng ta có thể sử dụng nhiều lần mà không phải lo lắng.

```
19    print("A day has passed. Your Water " + str(water) + "L")
20    event = random.choice(events)
21    print(event)
22    day += 1
```

- Chạy mã của bạn lần nữa. Bạn sẽ nhận được một kết quả tương tự như lần trước.
- Bạn có thể nhận thấy rằng sự kiện trong đầu ra của chúng ta được bao quanh bởi một loạt các thông tin và biểu tượng bổ sung. Đó là bởi vì chúng ta đang lưu trữ câu hỏi và giá trị cho lượng nước được hoặc mất nếu người chơi trả lời có hoặc không ở cùng một nơi. Chúng ta chỉ muốn hỏi người chơi câu hỏi mà không cho họ xem thêm thông tin. Để truy cập câu hỏi, chúng ta có thể sử dụng ký hiệu **index** (chỉ mục). Chỉ mục chỉ là vị trí mà phần tử nằm ở đó, bắt đầu từ 0. Thay đổi dòng in của bạn thành như sau:

```
20    event = random.choice(events)
21    print(event[0])
```

- Chạy mã của bạn lần nữa. Tốt hơn nhiều rồi, nhưng người chơi không thể trả lời câu hỏi.
- Thay vì sử dụng câu lệnh in, hãy sử dụng câu lệnh đầu vào. Điều này nói với python cho người dùng viết một cái gì đó.

```
20    event = random.choice(events)
21    input(event[0])
22    day += 1
```

Bây giờ, hãy lưu trữ đầu vào đó trong một biến, bởi vì chúng ta sẽ muốn sử dụng nó sau này.

```
20    event = random.choice(events)
21    response = input(event[0])
22    day += 1
```

- Chạy mã của bạn lần nữa. Trò chơi của bạn bây giờ nên chờ người chơi phản hồi mỗi ngày.

Bước 4: Sử dụng câu trả lời của người chơi

Người chơi đang trả lời câu hỏi, bây giờ chúng ta cần sử dụng câu trả lời của họ để đánh giá việc sử dụng nước của họ.

- Người chơi nên trả lời có hoặc không cho câu hỏi. Nếu họ trả lời "yes", chúng ta cộng số đầu tiên vào sự kiện:

```
21    response = input(event[0])
22    if response == "yes":
23        water += event[1]
24    day += 1
```

Lưu ý rằng ngay cả khi chúng ta cộng, rất nhiều số trong các sự kiện là số âm. Giống như trong toán học, việc cộng một số âm cũng giống như trừ đi số dương của số đó.

- Nếu người chơi trả lời "no", chúng ta nên cộng số thứ hai vào sự kiện.

```
22  if response == "yes":
23      water += event[1]
24  elif response == "no":
25      water += event[2]
26  day += 1
```

- Chạy mã của bạn. Bây giờ người chơi có thể trả lời các câu hỏi. Bạn sẽ nhận thấy rằng lượng nước mà người chơi có sẽ thay đổi theo quyết định của họ. Điều gì xảy ra nếu người chơi trả lời một cái gì đó khác "yes" hoặc "no"?

Bước 5: Xác thực dữ liệu của người dùng

Bây giờ bạn có một trò chơi đang hoạt động, nhưng người chơi có thể trả lời bất kỳ cách nào họ muốn. Chúng ta chỉ chấp nhận "yes" hoặc "no".

- Chỉ có hai tùy chọn, "yes" và "no", vì vậy chúng ta có thể nhắc người dùng trả lời theo cách chúng ta muốn họ trả lời. Sửa đổi dòng phản hồi của bạn thành như sau:

```
20  event = random.choice(events)
21  response = input(event[0] + " (yes/no): ")
22  if response == "yes":
```

- Tốt hơn rồi, nhưng người chơi vẫn có thể trả lời bất kỳ cách nào họ muốn. Người chơi có thể nhập nội dung khác để gian lận trò chơi hoặc nhập sai câu trả lời của họ. Chúng ta hãy thêm một câu lệnh "else" để bắt bất kỳ phản hồi nào khác hơn là "yes" hoặc "no".

```
24  elif response == "no":
25      water += event[2]
26  else:
27      print("Please answer yes or no!")
28  day += 1
```

- Chạy mã của bạn. Điều gì xảy ra nếu người chơi gõ một cái gì đó khác "yes" hoặc "no"?
- Chúng ta hiện đang yêu cầu người dùng sửa câu trả lời của họ, nhưng sau đó chúng ta sẽ hỏi họ câu hỏi tiếp theo! Chúng ta cần tiếp tục hỏi họ cùng một câu hỏi cho đến khi họ đưa ra câu trả lời mà chúng ta muốn nghe. Chúng ta có thể thực hiện việc này bằng một vòng lặp.

```
20    event = random.choice(events)
21    while True:
22      response = input(event[0] + " (yes/no): ")
23      if response == "yes":
24        water += event[1]
25      elif response == "no":
26        water += event[2]
27      else:
28        print("Please answer yes or no!")
29    day += 1
```

Lưu ý rằng chúng ta đã lùi dòng phản hồi và câu lệnh if / else. Điều này có nghĩa là đó là tất cả trong vòng lặp. Bạn có thể làm điều này bằng cách tô sáng các dòng mà bạn muốn lùi đầu dòng và nhấn 'tab' trên bàn phím.

- Chạy mã của bạn. Điều gì xảy ra nếu bạn trả lời một cái gì đó *khác với* "yes" hay "no"? Bạn sẽ được hỏi lại lần nữa. Nhưng điều gì xảy ra nếu bạn *đã* trả lời "yes" hoặc "no"? Bạn sẽ được hỏi lại lần nữa! Ôi trời!

- Vòng lặp "While true" là vòng lặp sẽ lặp mãi mãi. Điều này khá hữu ích nếu bạn muốn chạy mãi mãi, nhưng chúng ta không muốn, chúng ta chỉ muốn chạy cho đến khi người dùng trả lời "yes" hoặc "no"! May mắn thay, chúng ta có một lệnh đặc biệt có thể phá vỡ vòng lặp, ngay cả khi đó là vòng lặp mãi mãi. Lệnh đó là "break". Hãy thêm lệnh "break" khi người chơi trả lời "yes" hoặc "no".

```
22      response = input(event[0] + " (yes/no): ")
23      if response == "yes":
24        water += event[1]
25        break
26      elif response == "no":
27        water += event[2]
28        break
29      else:
```

- Chạy mã của bạn lần nữa. Bây giờ bạn chỉ nên nhận lại cùng một câu hỏi nếu bạn không trả lời "yes" hoặc "no".
  Hiện tại mọi thứ đang hoạt động tốt, nhưng có một vấn đề nhỏ nữa mà một số người chơi có thể gặp phải. Điều gì xảy ra nếu, thay vì trả lời "yes", người chơi trả lời "Yes" với chữ Y hoa?

- Python phân biệt chữ hoa và chữ thường. Điều đó có nghĩa là nếu bạn so sánh hai chuỗi, nó sẽ nghĩ rằng chúng là các chuỗi khác nhau chỉ vì chúng có cách viết hoa khác nhau. May mắn thay, điều này có thể dễ dàng sửa chữa. Python có một lệnh mà bạn có thể sử dụng trên

các chuỗi để làm cho tất cả chúng đều viết thường. Sửa đổi dòng phản hồi của bạn thành như sau:

```
21  while True:
22      response = input(event[0] + " (yes/no): ").lower()
23      if response == "yes":
```

- Chạy mã của bạn. Bây giờ nó sẽ chấp nhận “Yes”, “YES”, “yeS” và các biến thể hoa thường khác.

Xin chúc mừng!

Bạn đã hoàn thành dự án này. Làm tốt lắm! Hãy thử chơi trò chơi của bạn và xem liệu bạn có thể sống lâu lâu hơn bạn bè của bạn không. Muốn thêm nữa không? Hãy thử các thử thách dưới đây hoặc xem các dự án Scratch hoặc dự án Moonhack từ những năm trước.

Thử thách: Thêm nhiều sự kiện

Hiện có một vài sự kiện khác nhau, nhưng bạn có thể thêm nhiều hơn không?

Gợi ý: nếu bạn gặp khó khăn, hãy thử sao chép và chỉnh sửa một trong những sự kiện đã có.

Thử thách: trò chơi Trái đất

Bạn có thể sửa đổi trò chơi về Trái đất thay vì sao Hỏa không? Hãy suy nghĩ về các loại sự kiện có thể xảy ra và các lựa chọn mà người chơi có thể phải đối mặt.

Thử thách nâng cao: Chơi lại lần nữa, Sam

Bạn có thể cung cấp cho người chơi một tùy chọn để tiếp tục chơi sau khi họ thua không?

Gợi ý: Bạn sẽ muốn toàn bộ trò chơi của mình được lặp lại.