[**13/06/2019**](http://happyaivietnam.com/?p=193)**BY**[**ANH TUAN**](http://happyaivietnam.com/author/anh-tuan/)

# Chương\_6\_Vòng\_Lặp\_Loops

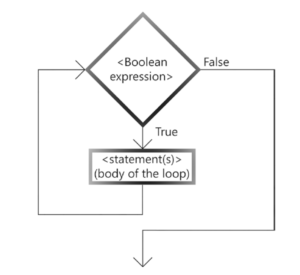
Trong chương này, chúng ta sẽ xây dựng một trò chơi đoán số. Chương trình sẽ lấy một số ngẫu nhiên trong khoảng 1 đến 20, người chơi sẽ dự đoán 5 lần. Mỗi lần đoán sai, chương trình sẽ thông báo kết quả đúng lớn hơn hay nhỏ hơn số đoán. Sau 5 lần đoán sai, chương trình sẽ cho người chơi biết con số đúng.

Thông qua xây dựng trò chơi, chương này sẽ bao gồm những chủ đề sau:

* Vòng lặp là gì
* Câu lệnh While
* Viết 1 vòng lặp đơn giản đầu tiên
* Thế nào là tăng , giảm
* Làm cách nào chạy chương trình nhiều lần
* Hàm vòng lặp trong Python
* Cách tạo 1 số ngẫu nhiên
* Mô phỏng cách búng đồng xu
* Tạo vòng lặp vô tận
* Phong cách mới để xây dựng 1 vòng lặp : while True và break
* Cách hỏi người chơi nếu nhập vào khoảng trắng
* Code giả lập ( Pseudocode)
* Xây dựng chương trình dự đoán số
* Cách chơi game nhiều lần
* Cách kiểm tra lỗi với Try/except
* Câu lệnh continue
* Xây dựng hàm kiểm tra lổi
* Bài tập nâng cao

**1. Giới thiệu trò chơi đoán số**  
>>> Chào mừng đến chương trình đoán số  
>>> Đoán từ 1 đến 20  
>>> Bạn có 5 lấn đoán  
>>> Đoán : 5  
>>> Số bạn đoán nhỏ hơn số đoán  
>>> Đoán : 7  
>>> Số bạn đoán lớn hơn số đoán  
>>> Đoán : 6  
>>> Bạn đã đoán đúng sau 3 lần dự đoán  
>>> Bạn có muốn chơi tiếp (y) hay thoát (Enter) : y  
>>> Đoán : 1  
>>> Số bạn đoán nhỏ hơn số đoán  
>>> Đoán : 2  
>>> Số bạn đoán lớn hơn số đoán  
>>> Đoán : 3  
>>> Số bạn đoán nhỏ hơn số đoán  
>>> Đoán : 4  
>>> Số bạn đoán lớn hơn số đoán  
>>> Đoán : 5  
>>> Bạn đã không đoán đúng sau 5 lần đoán  
>>> Bạn có muốn chơi tiếp (y) hay thoát (Enter) : y

Từ kết quả đầu ra, bạn có thể hình dung bằng cách nào viết chương trình. Bạn đã biết cách gọi hàm input để nhập dữ liệu và cũng biết cách so sánh 2 số dùng câu lệnh if/elif/else , bạn cũng biết cách lưu lại giá trị số lần đoán từ 0 , sau đó mỗi lần tăng 1. Nhưng để xây dựng trò chơi này , cần phải biết thêm vài công cụ khác và bằng cach nào để hiện thực chúng.

**2. Vòng lặp (Loops):**  
Như mô phỏng phần trên, bạn thấy chương trình cho phép người chơi đoán nhiều lần và chơi nhiều lần. Để viết chương trình dang này, người ta đưa ra khái niệm vòng lặp ( loop)  


**Định nghĩa:** 1 vòng lặp là 1 khối code mà được chạy lặp đi lặp lại cho đến khi 1 vài điều kiện chỉ rõ thoả mãn

**3. Câu lệnh lặp while**  
Trong Python, vòng lặp thường được thực hiện với câu lệnh while:

|  |  |
| --- | --- |
|  | while (mệnh đề điều kiện đúng/sai): |
|  | (khối lệnh) |

Chú ý: dấu hai chấm sau mệnh đề điều kiện.  
Khối lệnh có thể ngắn hoặc dài, còn được gọi là thân của chương trình. Trong khi mệnh đề điều kiện còn đúng, khối lệnh trong thân vòng lặp while tiếp tục thực hiện cho đến khi điều kiện là sai và vòng lặp sẽ thoát và thực hiện dòng lệnh đầu tiên ngoài khối lệnh thân vòng while.

**Ví dụ:**

|  |  |
| --- | --- |
|  | looping = **True** |
|  | **while** looping == **True**: |
|  | answer = input("Vui lòng nhập kí tự 'a': ") |
|  | **if** answer == 'a': |
|  | looping = **False** *# hoàn thành nhiệm vụ* |
|  | **else**: |
|  | print("Vui lòng nhập : 'a'!)" |
|  | print("Cám ơn đã nhập 'a'") |

Trước vòng lặp, ta gán biến boolean là đúng để lần đầu tiên vòng while kiểm tra điều kiện thấy đúng mới thực hiện phần khối lệnh thân. Tại cuối vòng lặp, sự thực thi sẽ tự động quay lại dòng đầu nếu biến boolean vẫn còn đúng và dừng khi biến boolean sai , do đó trong thân khối lệnh lặp, ta phải cài đặt một điều thoát. Nếu chúng ta không có làm kĩ điều kiện thoát , điều này sẽ dẫn đến **vòng lặp vô tận(infinite loop)** trừ khi bạn tắt phần mềm hoặc tắt máy tính.

**4. Viết chương trình vòng lặp đầu tiên**

Chương trình hỏi người nhập 1 con số , sau đó nó sẽ tính tổng từ 1 đến con số đó.

Ví dụ: nhập 4, kết quả là 1+2+3+4 = 10.

|  |  |
| --- | --- |
|  | target = input('Enter a target number: ') |
|  | target = int(target) |
|  | total = 0 |
|  | nextNumberToAddIn = 1 |
|  | **while** nextNumberToAddIn <= target: |
|  | *# add in the next value* |
|  | total = total + nextNumberToAddIn *#add in the next number* |
|  | print('Addd in:', nextNumberToAddIn, 'Total so far is:', total) |
|  | nextNumberToAddIn = nextNumberToAddIn + 1 |
|  | print('The sum of the numbers from 1 to', target,'is:', total) |

Thực thi chương trình trên với input = 4 , ta có kết quả sau:

|  |  |
| --- | --- |
|  | Enter a target number: 4 |
|  | Added **in**: 1 Total so far **is**: 1 |
|  | Added **in**: 2 Total so far **is**: 3 |
|  | Added **in**: 3 Total so far **is**: 6 |
|  | Added **in**: 4 Total so far **is**: 10 |
|  | The sum of the numbers **from** 1 to 4 **is**: 10 |

Chú ý: trước vòng while , ta phải lấy input từ người nhập và biến đổi thành dạng số interger, cài đặt biến tổng total = 0 , biến nextNumberToAddIn =1 , biến này sẽ tăng từ 1 đến traget(con số mà người chơi nhập vào).

Kế tiếp, vòng while sẽ chạy cho đến khi nextNumberToAddIn >= target( chính là điều kiện dừng). Cuối cùng, chương trình sẽ in ra số tổng total.

**5. Tăng và Giảm**

Trong đoạn code trên có dòng lệnh:

|  |  |
| --- | --- |
|  | nextNumberToAddIn = nextNumberToAddIn +1 |

Dòng này có nghĩa là lấy giá trị hiện tại của biến nextNumberToAddIn cộng 1 rồi gán lại cho chính biến đó. Phép toán này rất thường được sử dụng, người ta thường gọi là increment.

**Địng nghĩa:** increment là khi một giá trị được cộng vào chính nó.

Ví dụ: counter = counter + 1 tương đương với counter +=1. Tương tự: ta có đếm ngược xuống , gọi là decrement.

**Địng nghĩa**: decrement là khi một giá trị được trừ từ chính nó.

Ví dụ: counter = counter – 1 tương đương với counter -= 1. Trong chương trình đoán số, để lưu giữ bao nhiêu lần đoán, ta sử dụng bộ đếm tăng (increment) cho mỗi lần đoán.

**6. Chạy chương trình nhiều lần**

Trong chương trình trên, chúng ta muốn người dùng có thể chạy nhiều lần bằng cách hỏi yes/no và chương trình sẽ chạy lại lần nữa. Có nhiều cách để làm được điều này nhưng cách đơn giản nhất là đặt 1 hàm tại cuối thân vòng lặp. Đây là đoạn mã:

|  |  |
| --- | --- |
|  | **def** **calculateSum**(target): |
|  | total = 0 |
|  | nextNumberToAddIn = 1 |
|  | **while** nextNumberToAddIn <= target: |
|  | *# add in the next value* |
|  | total = total + nextNumberToAddIn |
|  | *#increment* |
|  | nextNumberToAddIn = nextNumberToAddIn + 1 |
|  | **return** total |
|  | answer = 'y' *# start off with the value 'y' to go through the first time* |
|  | **while** answer == 'y': |
|  | usersTarget = input('Enter a target number: ') |
|  | usersTarget = int(usersTarget) |
|  | thisTotal = calculateSum(usersTarget) *# call our function and get back the answer* |
|  | print('The sum of the numbers 1 to', usersTarget, 'is:', thisTotal) |
|  | answer = input('Do you want to try again (y or n): ') |
|  | print('OK Bye') |

Chú ý: trong cấu trúc này, tất cả việc tính toán được thực hiện trong 1 hàm con, thân chương trình chính làm nhiệm vụ chính là tương tác hỏi người chơi và gọi hàm con ra để tính toán kết quả tổng.

Một cách khác là chạy chương trình với số lần đã định nghĩa trước. Ví dụ: 3 lần như bên dưới:

|  |  |
| --- | --- |
|  | nTimes = 0 *# initialize a counter* |
|  | **while** nTimes < 3: |
|  | usersTarget = input('Enter a target number: ') |
|  | usersTarget = int(usersTarget) |
|  | thisTotal = calculateSum(usersTarget) |
|  | print('The sum of the numbers 1 to', usersTarget, 'is:', thisTotal) |
|  | nTimes = nTimes + 1 *# increment the counter* |
|  | print('OK Bye') |

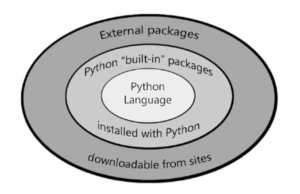
**7. Thư viện trong Python**

Trở lại chương trình đoán số. Việc kế tiếp là tìm hàm sinh số ngẫu nhiên. Để trả lời câu hỏi này, chúng ta tìm hiểu thư viện có sẵn của Python(packages).

Để làm cho chương trình nhỏ gọn, Python chỉ xây dựng vài câu lệnh cơ bản như if, elif, else, while,…, một số hàm(int, str, inout,…). Python cũng có xây dựng 1 một số hàm phổ biến gọi hàm thư viện chuẩn, sẽ được cài đặt vào máy tính khi người dùng cài Python.

Ngoài ra, người lập trình trên khắp thế giới cũng tự xây dựng thư viện riêng của họ, còn gọi là thư viện ngoài. Bạn có thể tải từ internet. Ví dụ: thư viện tên là PyGame dùng để viết game dùng đồ hoạ windows, chuột, âm thanh, …

Hình dưới minh hoạ thư viện được phân ra 3 loại:



**8. Tạo một số ngẫu nhiên**

Một trong những hàm xây dựng sẵn tên là random. Khi bạn muốn dùng thư viện này, bạn phải báo Python bằng cách khai báo thư viện đó trong chương trình của bạn theo cú pháp import &lt;&gt;.Trong ví dụ này là import random

Trong trường thư viện không tồn tại, IDEL sẽ xuất ra một lỗi như sau:

|  |  |
| --- | --- |
|  | **>>> import** NonExistentPackage |
|  | Traceback (most recent call last): |
|  | File "", line 1, **in** |
|  | **import** NonExistentPackage |
|  | ImportError: No module named NonExistentPackage |
|  | >>> |

Thư viện nên đặt ở đầu chương trình của bạn. Để biết tất cả thông tin về thư viện bạn cần dùng: help(random).

IDLE sẽ liệt kê tất các hàm và cách dùng cũng như thông số khai báo. Trong ví dụ trên, ta cần hàm randrange:

|  |  |
| --- | --- |
|  | random.randrange(, ) |

Lưu ý : bạn bắt đầu với thư viện rồi đến dấu chấm tên hàm và thông số của hàm như ví dụ sau :

|  |  |
| --- | --- |
|  | *#random between 1 and 10* |
|  | aRandomNumber = random.randrange(1, 11) |
|  | *#random between 1 and 52, to pick a card number from a deck* |
|  | anotherRandomNumber = random.randrange(1, 53) |

Lưu ý: phạm vi bắt đầu từ lowValue đến upToButNotIncludingHighValue nhưng khong bao gồm giá trị này

Một cách khác:

|  |  |
| --- | --- |
|  | *#random between 0 and 8* |
|  | myRandomNumber = random.randrange(9) |
|  | *#tương đương với random.randrange(0,8)* |

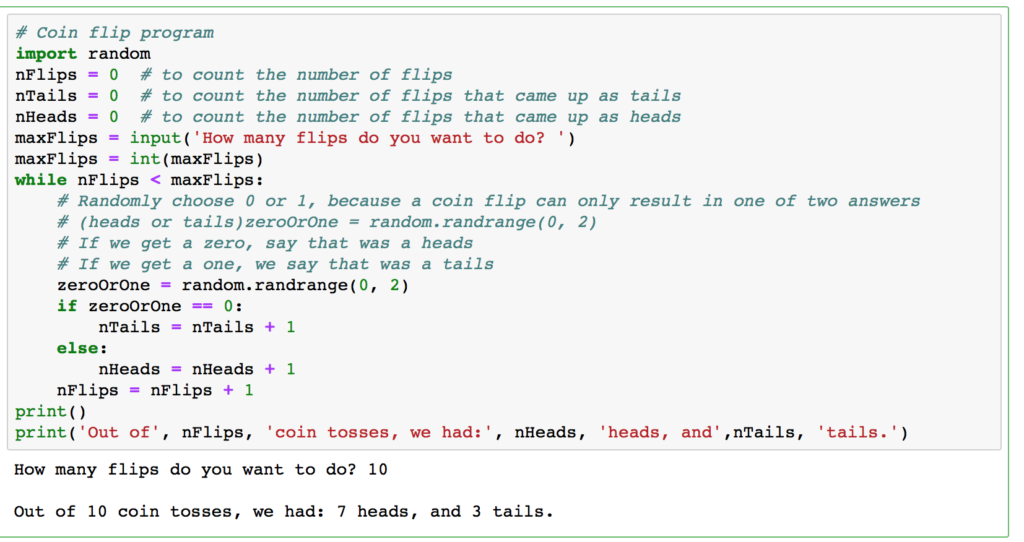
**9. Mô phỏng búng đồng xu**

Đây là một ví dụ điển hình dùng hàm tạo số ngẩu nhiên. ta sẽ mô phỏng việc búng đồng tiền một vài lần. Chương trình sẽ dùng vòng lặp, mỗi lần lặp, ta sẽ tạo ra 1 số ngẫu nhiên 0 hoặc 1, sau đó ánh xạ với mặt xấp và mặt ngữa, kết thúc chương trình sẽ in ra kết quả:

|  |  |
| --- | --- |
|  | *#Coin flip program* |
|  | **import** random |
|  | nFlips = 0 *# to count the number of flips* |
|  | nTails = 0 *# to count the number of flips that came up as tails* |
|  | nHeads = 0 *# to count the number of flips that came up as heads* |
|  | maxFlips = input('How many flips do you want to do? ') |
|  | maxFlips = int(maxFlips) |
|  | **while** nFlips < maxFlips: |
|  | *# Randomly choose 0 or 1, because a coin flip can only result in one of two answers* |
|  | *# (heads or tails)zeroOrOne = random.randrange(0, 2)* |
|  | *# If we get a zero, say that was a heads* |
|  | *# If we get a one, we say that was a tails* |
|  | zeroOrOne = random.randrange(0, 2) |
|  | **if** zeroOrOne == 0: |
|  | nTails = nTails + 1 |
|  | **else**: |
|  | nHeads = nHeads + 1 |
|  | nFlips = nFlips + 1 |
|  | print() |
|  | print('Out of', nFlips, 'coin tosses, we had:', nHeads, 'heads, and',nTails, 'tails.') |

Kết quả:

|  |  |
| --- | --- |
|  | How many flips **do** you want **to** **do**? 10 |
|  | **Out** **of** 10 coin tosses, we had: 7 heads, **and** 3 tails. |



**10. Tạo vòng lặp vô tận**

Như giới thiệu ở trên về vòng lặp while, trong khi biến boolean bằng True,vòng lặp chạy liên tục cho đến khi biến boolean bằng False, bất kỳ vòng lặp đều có thể xây dựng bằng cấu trúc này.  
Biến boolean được xem là điều kiện thoát, thường dùng câu lệnh if để kiểm tra điều kiện thoát có hoạt động hay không.

|  |  |
| --- | --- |
|  | looping = **True** |
|  | **while** looping: |
|  | statement(s) |
|  | **if** found exit condition? : |
|  | looping = **False** *# found the exit condition at this point* |
|  | **else**: |
|  | continuing statement(s) inside the loop |

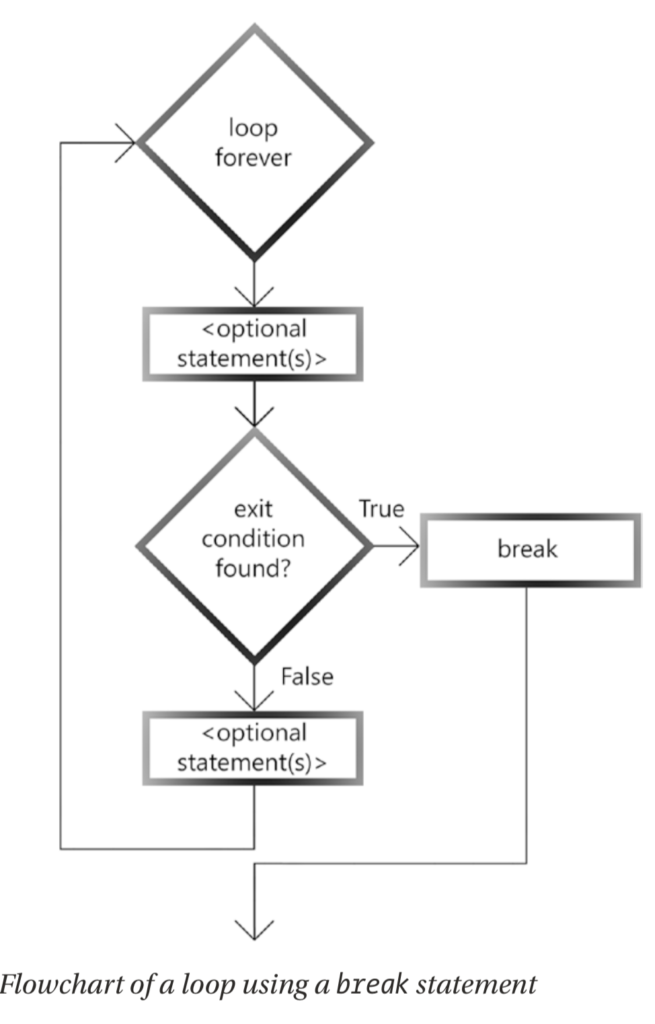
Trong đoạn mã trên có điều kiện thoát nhưng bạn không thể thoát cho đến khi lệnh thực thi quay trở lại đầu câu lệnh while, kiểu viết này thường rất khó viết phần chuỗi câu lệnh sau else và dễ dẫn đến chạy vô tận, ví dụ như câu lệnh thụt lề quá nhiều dẫn đến không kiểm soát được. May mắn có một cách khác xây dựng vào while hiệu quả hơn.

**11. Phương cách mới xây dựng vòng lặp: while True, and break**

Như đã nói ở trên, bạn có thể tạo ra vòng lặp vô tận một caách tai nạn khi biến boolean không thay đổi giá trị được. Vì thế , cách đơn giản để tạo ra lặp vô tận như sau:

|  |  |
| --- | --- |
|  | **while** **True**: |
|  | statement(s) |

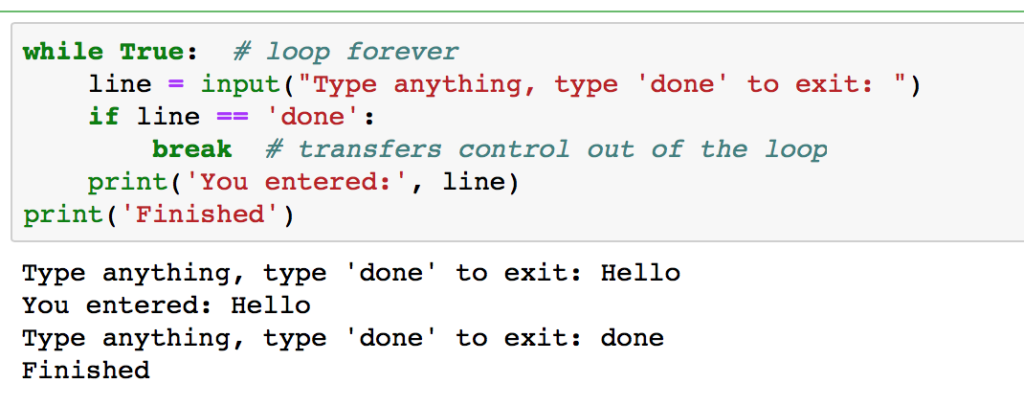
Python cung cấp câu lệnh khác gọi là break. Nếu dòng mã lệnh đang chay, gặp câu lệnh break, bộ điều khiển sẽ chuyễn đến câu lệnh đầu tiên sau dòng cuối cùng của vòng lặp như sơ đồ sau:



Vi dụ:

|  |  |
| --- | --- |
|  | **while** **True**: *# loop forever* |
|  | line = input("Type anything, type 'done' to exit: ") |
|  | **if** line == 'done': |
|  | **break** *# transfers control out of the loop* |
|  | print('You entered:', line) |
|  | print('Finished') |

|  |  |
| --- | --- |
|  | Type anything, type 'done' to exit:Hello |
|  | You entered: Hello |
|  | Type anything, type 'done' to exit: **done** |
|  | Finished |



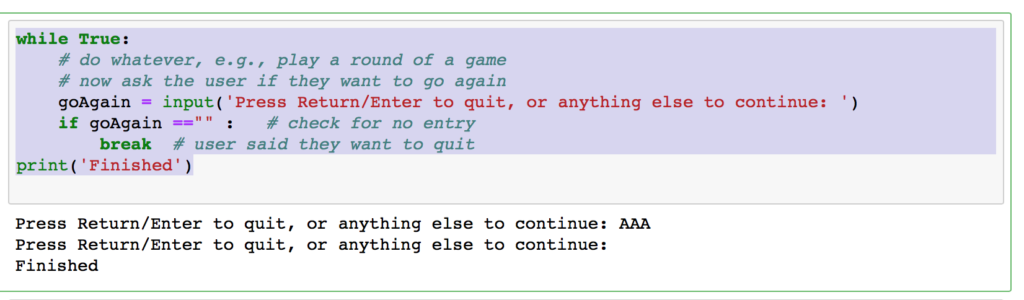
Trong trò chơi đoán số, ta có 2 điều kiện kiểm tra thoát: một là kiểm tra người chơi đã nhập số đúng chưa, hai là kiểm tra số lần nhập vượt quá số lần đoán qui định. Nếu dùng break sẽ làm cho mã code dễ dàng và rõ ràng hơn.

Tóm lại viết vòng lặp while vô tận với câu lệnh break để thoát là rất hiệu quả và rõ ràng. Hâu hết các vòng loop tác giả đều dùng kiểu này.

**12. Hỏi nếu người chơi muốn chơi lần nữa**

Chúng ta gần như xây dựng xong game đoán chữ, trong trường hợp người chơi muốn chơi lại, ta phải lặp lại chương trình này lần nữa theo vòng lặp while như sau:

|  |  |
| --- | --- |
|  | **while** **True**: |
|  | *# do whatever, e.g., play a round of a game* |
|  | *# now ask the user if they want to go again* |
|  | goAgain = input('Press Return/Enter to quit, or anything else to continue: ') |
|  | **if** goAgain =="" : *# check for no entry* |
|  | **break** *# user said they want to quit* |
|  | print('Finished') |



Giải thích đoạn mã trên : biến goAgain sẽ được gán bất kỳ giá trị nhập vào trừ khi nhấn Enter để thoát.

**Mã giả ( Pseudocode):**

Bây giờ, ta đủ thông tin để viết chương trình. Nhưng thay vì nhảy vào viết chương trình liền, ta hãy thảo luận bằng cach nào chương trình chạy, ta sẽ đưa ra một cách tiếp cận tổng thể trước khi viết mã thực tế.

**Định nghĩa:** mã giả là một miêu tả chương trình giống như ngôn ngữ tiếng Anh của một giải thuật trong ngôn ngữ máy tính.

Thông thường, các lập trình viên phát triển một thuật toán và viết thuật toán bằng mã giả trước khi viết nó bằng ngôn ngữ máy tính thực sự. Điều này cho phép các lập trình viên suy nghĩ thông qua cách tiếp cận cấu trúc tổng thể của một giải pháp mà không phải lo lắng về cú pháp chi tiết của ngôn ngữ máy tính.

Ví dụ, mã giả của 1 vòng của trò chơi đoán số:

|  |  |
| --- | --- |
|  | **Show** introduction |
|  | **Choose** random target |
|  | Initialize a guess counter |
|  | **Loop** forever |
|  | Ask the **user** **to** **for** a guess |
|  | **Increment** guess counter |
|  | **If** **user**'s guess is correct, congratulate user, we're done |
|  | **If** **user**'s guess is too low, tell user |
|  | If user's guess **is** too **high**, tell **user** |
|  | **If** reached **max** guesses, tell correct answer, we're done. |
|  |  |

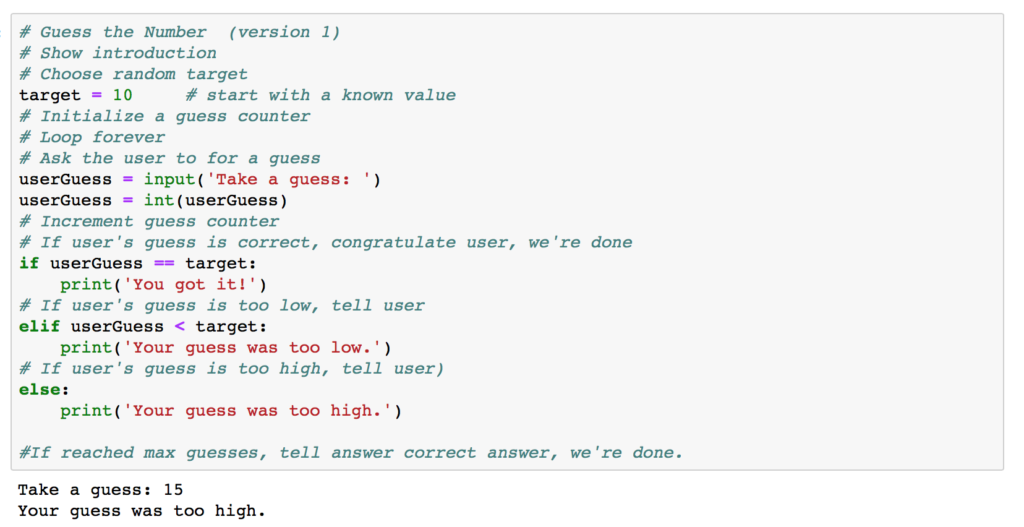
Chúng ta có thể dùng mã giả này để chuyển thành chú thích, nó rất cần thiết như một phác thảo bên trong mã của chương trình.

**13. Xây dựng chương trình đoán số**

Chúng ta sẽ viết từng phần nhỏ vả kiểm tra hơn là viết toàn bộ từ trên xuống. Đây là kỹ thuật chuẩn để viết các chương trình lớn hơn. Viết hay thay đổi nhiều thứ cùng một lúc khiến quá trình xử lý trở nên khó khăn bởi vì nếu có một lỗi xuất hiện, khó mà phát hiện dòng mã nào gây ra lỗi đó và có thể là lỗi logic cần được xử lý.

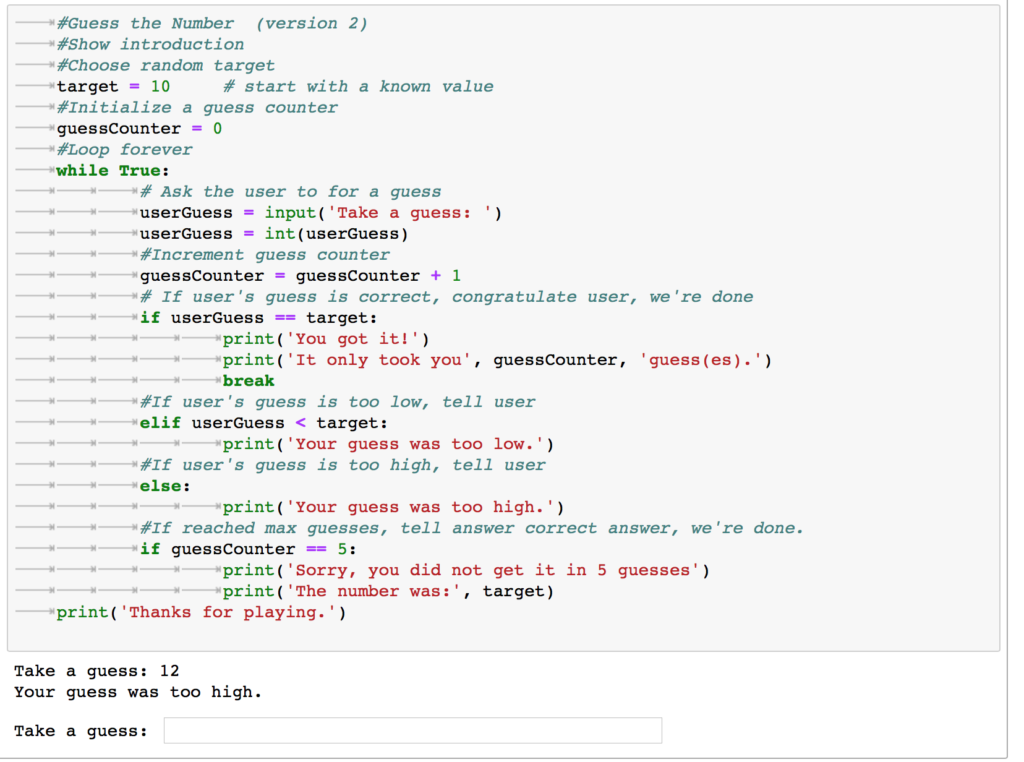
Ta bắt đầu viết phần đoán số người chơi, thật khó đển đoán con số sinh ra ngẫu nhiên khi kiểm tra chương trình chạy đúng hay sai, nên ta gán số cần đoán là 10, sau đó sẽ chỉnh mã sang số ngẫu nhiên khi chương trình chạy hoàn chỉnh:

|  |  |
| --- | --- |
|  | *#Guess the Number (version 1)* |
|  | *#Show introduction* |
|  | *#Choose random target* |
|  | target = 10 *# start with a known value* |
|  | *#Initialize a guess counter* |
|  | *#Loop forever* |
|  | *#Ask the user to for a guess* |
|  | userGuess = input('Take a guess: ') |
|  | userGuess = int(userGuess) |
|  | *#Increment guess counter* |
|  | *#If user's guess is correct, congratulate user, we're done* |
|  | **if** userGuess == target: |
|  | print('You got it!') |
|  | *#If user's guess is too low, tell user* |
|  | **elif** userGuess < target: |
|  | print('Your guess was too low.') |
|  | *#If user's guess is too high, tell user)* |
|  | **else**: |
|  | print('Your guess was too high.') |
|  | *#If reached max guesses, tell answer correct answer, we're done.* |



Bay giờ, ta thêm biến lưu số lần đoán:

|  |  |
| --- | --- |
|  | *#Guess the Number (version 2)* |
|  | *#Show introduction* |
|  | *#Choose random target* |
|  | target = 10 *# start with a known value* |
|  | *#Initialize a guess counter* |
|  | guessCounter = 0 |
|  | *#Loop forever* |
|  | **while** **True**: |
|  | *# Ask the user to for a guess* |
|  | userGuess = input('Take a guess: ') |
|  | userGuess = int(userGuess) |
|  | *#Increment guess counter* |
|  | guessCounter = guessCounter + 1 |
|  | *# If user's guess is correct, congratulate user, we're done* |
|  | **if** userGuess == target: |
|  | print('You got it!') |
|  | print('It only took you', guessCounter, 'guess(es).') |
|  | **break** |
|  | *#If user's guess is too low, tell user* |
|  | **elif** userGuess < target: |
|  | print('Your guess was too low.') |
|  | *#If user's guess is too high, tell user* |
|  | **else**: |
|  | print('Your guess was too high.') |
|  | *#If reached max guesses, tell answer correct answer, we're done.* |
|  | **if** guessCounter == 5: |
|  | print('Sorry, you did not get it in 5 guesses') |
|  | print('The number was:', target) |
|  | print('Thanks for playing.') |



Chương trình trên có vẻ chạy ổn, bây giờ ta thêm hàm sinh số ngẫu nhiên từ 5 đến 20:

|  |  |
| --- | --- |
|  | *#Guess the Number (version 3)* |
|  | **import** random |
|  | MAX\_GUESSES = 5 *# maximum number of guesses allowed* |
|  | MAX\_RANGE = 20 *# highest possible number* |
|  | *#Show introduction* |
|  | print('Welcome to my Guess the Number program.') |
|  | print('Guess my number between 1 and', MAX\_RANGE) |
|  | print('You will have', MAX\_GUESSES, 'guesses.') |
|  | *#Choose random target* |
|  | target = random.randrange(1, MAX\_RANGE + 1) |
|  | *#Initialize a guess counter* |
|  | guessCounter = 0 |
|  | *#Loop forever* |
|  | **while** **True**: |
|  | *# Ask the user to for a guess* |
|  | userGuess = input('Take a guess: ') |
|  | userGuess = int(userGuess) |
|  | *#Increment guess counter* |
|  | guessCounter = guessCounter + 1 |
|  | *#If user's guess is correct, congratulate user, we're done* |
|  | **if** userGuess == target: |
|  | print('You got it!') |
|  | print('It only took you', guessCounter, 'guess(es).') |
|  | **break** |
|  | *#If user's guess is too low, tell user* |
|  | **elif** userGuess < target: |
|  | print('Your guess was too low.') |
|  | *#If user's guess is too high, tell user* |
|  | **else**: |
|  | print('Your guess was too high.') |
|  | *#If reached max guesses, tell answer correct answer, we're done.* |
|  | **if** guessCounter == MAX\_GUESSES: |
|  | print('Sorry, you did not get it in', MAX\_GUESSES, 'guesses.') |
|  | print('The number was:', target) |
|  | **break** |
|  | print('Thanks for playing.') |



Bây giờ , ta thêm mã chơi nhiều lần, có nhiều cách như while loop lồng trong while loop hay xây dưng hàm con rồi gọi trong hàm chính, ta chọn cách 2 trong ví dụ này:

|  |  |
| --- | --- |
|  | *#Guess the Number (version 4)* |
|  | **import** random |
|  | MAX\_GUESSES = 5 *# maximum number of guesses allowed* |
|  | MAX\_RANGE = 20 *# highest possible number* |
|  | *#Show introduction* |
|  | print('Welcome to my Guess the Number program.') |
|  | print('Guess my number between 1 and', MAX\_RANGE) |
|  | print('You will have', MAX\_GUESSES, 'guesses.') |
|  | **def** **playOneRound**(): |
|  | *# Choose random target* |
|  | target = random.randrange(1, MAX\_RANGE + 1) |
|  | *# Initialize a guess counter* |
|  | guessCounter = 0 |
|  | *# Loop forever* |
|  | **while** **True**: |
|  | *# Ask the user to for a guess* |
|  | userGuess = input('Take a guess: ') |
|  | userGuess = int(userGuess) |
|  | *# Increment guess counter* |
|  | guessCounter = guessCounter + 1 |
|  | *# If user's guess is correct, congratulate user, we're done* |
|  | **if** userGuess == target: |
|  | print('You got it!') |
|  | print('It only took you', guessCounter, 'guess(es).') |
|  | **break** |
|  | *# If user's guess is too low, tell user* |
|  | **elif** userGuess < target: |
|  | print('Your guess was too low.') |
|  | *# If user's guess is too high, tell user* |
|  | **else**: |
|  | print('Your guess was too high.') |
|  | *# If reached max guesses, tell answer correct answer, we're done.* |
|  | **if** guessCounter == MAX\_GUESSES: |
|  | print('Sorry, you did not get it in', MAX\_GUESSES, 'guesses.') |
|  | print('The number was:', target) |
|  | **break** |
|  | *#main code* |
|  | **while** **True**: |
|  | playOneRound() *# call a function to play one round of the game* |
|  | goAgain = input('Play again? (Press ENTER to continue, or q to quit):') |
|  | **if** goAgain == 'q': |
|  | **break** |
|  | print('Thanks for playing.') |



**14. Phát hiện lỗi với Try/except**

Trong bất kỳ chương trình mà hỏi người chơi nhập số, đôi khi vì vái ký tự chữ được nhập vào dẫn đến chương trình ngưng chạy, l;úc đó Python sẽ xuất ra lỗi như ví dụ sau:

|  |  |
| --- | --- |
|  | &**gt**;&**gt**;&**gt**; userInput = input('Please enter a number: ') |
|  | Please enter an integer: xyz |
|  | &**gt**;&**gt**;&**gt**; userInput = **int**(userInput) *# convert user's input to an integer* |
|  | Traceback (most recent call **last**): |
|  | File "", line 1, in |
|  | userInput = **int**(userInput) |
|  | ValueError: invalid literal **for** **int**() with base 10: 'xyz' |

Do đó, cách tốt nhất là phát hiện và thông báo người chơi lỗi hơn làn để Python phát hiện và chương trình bị dừng.

Python cung cấp một kỹ thuật để kiểm tra lỗi nhưng mất ít thời gian để làm quen. Về lý thuyết, trước khi chạy mã mà có thể gây ra lỗi, ta yêu cầu Python xem cái gì đang diễn ra, nếu lỗi xảy ra trong quá trình Python o chế độ giám sát, bạn có thể yêu cầu Python chạy thêm một đoạn mã để in ra loại mã lỗi và thông báo người chơi chi tiết lỗi này để họ nhập hay chỉnh lại cho đúng.

Dưới đây là cú pháp:

|  |  |
| --- | --- |
|  | **try**: |
|  | (câu lệnh có thể gây ra lỗi ) |
|  | **except**: |
|  | (câu lệnh thực thi nếu lỗi xảy ra) |
|  | **else**: *#optional, often not needed* |
|  | ( câu lệnh thực thi nếu không lỗi ) |

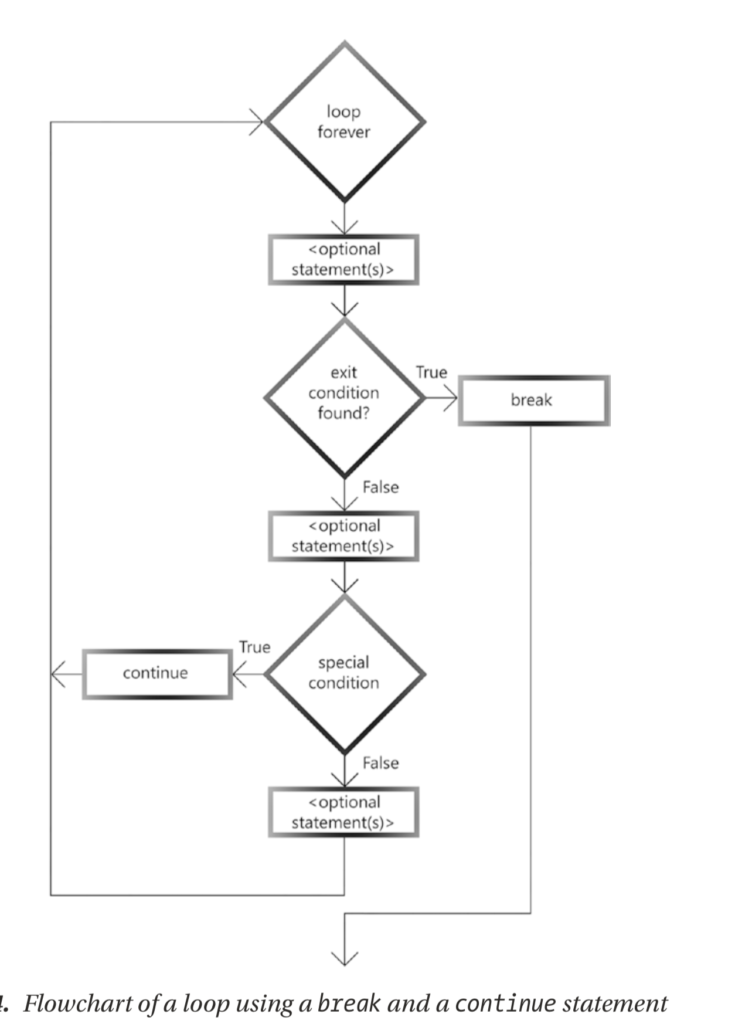
Trong ví dụ này, ta muốn nhập số integer:

|  |  |
| --- | --- |
|  | userGuess = input(' Take a guess: ') |
|  | **try**: |
|  | userInput = int(userInput) |
|  | **except**: |
|  | print('The number you entered was not an integer') |

Khối lệnh try có thể chứa nhiều lệnh thay vì một lệnh như ví dụ trên. Mặt tốt là nếu mã chương trình có lỗi và thông báo, mặt khác chương trình vẫn tiếp tục chạy với biến lỗi dẫn đến việc so sánh giữa số và ký tự gặp lỗi. Do đó, ta cần vài câu lệnh nữa để sữa lỗi này.

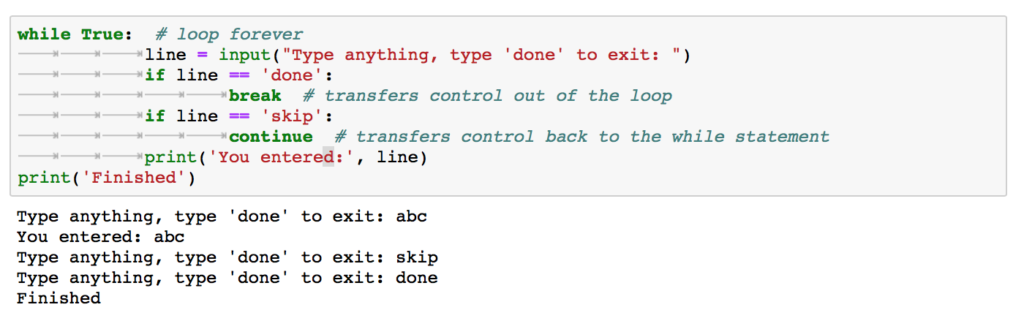
**15. Câu lệnh continue**

Khi ta phát hiện một vài lỗ trong vòng lặp, ta muốn truyền điều khiển ngược trở về đầu vòng lặp while mà không thực thi phần còn lại ben trong vòng lặp, để làm điều này, ta thực thi câu lệnh continue được minh hoạ như hình sau:



Ví dụ:

|  |  |
| --- | --- |
|  | **while** **True**: *# loop forever* |
|  | line = input("Type anything, type 'done' to exit: ") |
|  | **if** line == 'done': |
|  | **break** *# transfers control out of the loop* |
|  | **if** line == 'skip': |
|  | **continue** *# transfers control back to the while statement* |
|  | print('You entered:', line) |
|  | print('Finished') |



**16. Game hoàn chỉnh**

|  |  |
| --- | --- |
|  | *#Guess the Number (version 5 - final)* |
|  | **import** random |
|  | MAX\_GUESSES = 5 *# maximum number of guesses allowed* |
|  | MAX\_RANGE = 20 *# highest possible number* |
|  | *#Show introduction* |
|  | print('Welcome to my Guess the Number program.') |
|  | print('Guess my number between 1 and', MAX\_RANGE) |
|  | print('You will have', MAX\_GUESSES, 'guesses.') |
|  | **def** **playOneRound**(): |
|  | *# Choose random target* |
|  | target = random.randrange(1, MAX\_RANGE + 1) |
|  | *# Initialize a guess counter* |
|  | guessCounter = 0 |
|  | *#Loop forever* |
|  | **while** **True**: |
|  | *# Ask the user to for a guess* |
|  | userGuess = input('Take a guess: ') |
|  | *#Check for potential error* |
|  | **try**: |
|  | userGuess = int(userGuess) |
|  | **except**: |
|  | print('Hey, that was NOT an integer!') |
|  | **continue** *# transfer control back to the while* |
|  | *#Increment guess counter* |
|  | guessCounter = guessCounter + 1 |
|  | *#If user's guess is correct, congratulate user, we're done* |
|  | **if** userGuess == target: |
|  | print('You got it!') |
|  | print('It only took you', guessCounter, 'guess(es).') |
|  | **break** |
|  | *#If user's guess is too low, tell user* |
|  | **elif** userGuess < target: |
|  | print('Your guess was too low.') |
|  | *#If user's guess is too high, tell user* |
|  | **else**: |
|  | print('Your guess was too high.') |
|  | *#If reached max guesses, tell answer correct answer, we're done.* |
|  | **if** guessCounter == MAX\_GUESSES: |
|  | print('Sorry, you did not get it in', MAX\_GUESSES, 'guesses.') |
|  | print('The number was:', target) |
|  | **break** |
|  |  |
|  | *#main code* |
|  | **while** **True**: |
|  | playOneRound() *# call a function to play one round of the game* |
|  | goAgain = input('Play again? (Press ENTER to continue, or q to quit):') |
|  | **if** goAgain == 'q': |
|  | **break** |
|  | print('Thanks for playing.') |



**17. Tổng kết**

Trong chương này, chúng ta đã xây dựng trò chơi đoán số. Để hiện thực, bạn đã học về định ngĩa vòng lặp while tromng Python, khái niệm tăng / giảm biến , dùng vòng loop đễ chạy chương trình nhiều lần, học cách khai báo và sửng dụng hàm trong thư viện chuẩn có sẵng trong Python như sử dụng hàm randrange để sinh số ngẫu nhiên và xây dựng chương trình búng đồng xu cho hàm randragnge này.

Sau đó, bạn biết nhiều cách xây dựng vòng lặp while vô tận và cách thoát khỏi loop bằng cách dùng câu lệnh break và câu lệnh continue để khắc phục từ lỗi nhập sai người chơi. Dùng vòng lặp để hỏi người chơi nhiều lần bằng cách kiểm tra chuỗi rỗng, biết lợi ích của miêu tà mã giả của giải thuật để xây dưng chương trình và kiểm tra từng phần của toàn chương trìnhh.

Cuối cùng, bạn biết cách bắt lỗi bằng dùng khối lệnh try/except và biết cách xây dựng hàm con có thể tái sử dụng để kiểm tra gia trị integer có trong phạm vi qui định.

=========== ===============Ban edit ============================

Trong chương này, chúng ta sẽ xây dựng một trò chơi đoán số. Chương trình sẽ lấy một số ngẫu nhiên trong khoảng 1 đến 20, người chơi sẽ dự đoán 5 lần. Mỗi lần đoán sai, chương trình sẽ thông báo kết quả đúng lớn hơn hay nhỏ hơn số đoán. Sau 5 lần đoán sai, chương trình sẽ cho người chơi biết con số đúng.

Thông qua xây dựng trò chơi, chương này sẽ bao gồm những chủ đề sau:

- Vòng lặp là gì

- Câu lệnh While

- Viết 1 vòng lặp đơn giản đầu tiên

- Thế nào là tăng , giảm

- Làm cách nào chạy chương trình nhiều lần

- Hàm vòng lặp trong Python

- Cách tạo 1 số ngẫu nhiên

- Mô phỏng cách búng đồng xu

- Tạo vòng lặp vô tận

- Phong cách mới để xây dựng 1 vòng lặp : while True và break

- Cách hỏi người chơi nếu nhập vào khoảng trắng

- Code giả lập ( Pseudocode)

- Xây dựng chương trình dự đoán số

- Cách chơi game nhiều lần

- Cách kiểm tra lỗi với Try/except

- Câu lệnh continue

- Xây dựng hàm kiểm tra lổi

- Bài tập nâng cao

\*\*1. Giới thiệu trò chơi đoán số\*\*

&gt;&gt;&gt; Chào mừng đến chương trình đoán số

&gt;&gt;&gt; Đoán từ 1 đến 20

&gt;&gt;&gt; Bạn có 5 lấn đoán

&gt;&gt;&gt; Đoán : 5

&gt;&gt;&gt; Số bạn đoán nhỏ hơn số đoán

&gt;&gt;&gt; Đoán : 7

&gt;&gt;&gt; Số bạn đoán lớn hơn số đoán

&gt;&gt;&gt; Đoán : 6

&gt;&gt;&gt; Bạn đã đoán đúng sau 3 lần dự đoán

&gt;&gt;&gt; Bạn có muốn chơi tiếp (y) hay thoát (Enter) : y

&gt;&gt;&gt; Đoán : 1

&gt;&gt;&gt; Số bạn đoán nhỏ hơn số đoán

&gt;&gt;&gt; Đoán : 2

&gt;&gt;&gt; Số bạn đoán lớn hơn số đoán

&gt;&gt;&gt; Đoán : 3

&gt;&gt;&gt; Số bạn đoán nhỏ hơn số đoán

&gt;&gt;&gt; Đoán : 4

&gt;&gt;&gt; Số bạn đoán lớn hơn số đoán

&gt;&gt;&gt; Đoán : 5

&gt;&gt;&gt; Bạn đã không đoán đúng sau 5 lần đoán

&gt;&gt;&gt; Bạn có muốn chơi tiếp (y) hay thoát (Enter) : y

Từ kết quả đầu ra, bạn có thể hình dung bằng cách nào viết chương trình. Bạn đã biết cách gọi hàm input để nhập dữ liệu và cũng biết cách so sánh 2 số dùng câu lệnh if/elif/else , bạn cũng biết cách lưu lại giá trị số lần đoán từ 0 , sau đó mỗi lần tăng 1. Nhưng để xây dựng trò chơi này , cần phải biết thêm vài công cụ khác và bằng cach nào để hiện thực chúng.

\*\*2. Vòng lặp (Loops):\*\*

Như mô phỏng phần trên, bạn thấy chương trình cho phép người chơi đoán nhiều lần và chơi nhiều lần. Để viết chương trình dang này, người ta đưa ra khái niệm vòng lặp ( loop)

<img src="http://happyaivietnam.com/wp-content/uploads/2019/06/python\_Chuong6\_fig1-300x275.png" alt="" width="300" height="275" class="aligncenter size-medium wp-image-198" />

\*\*Định nghĩa:\*\* 1 vòng lặp là 1 khối code mà được chạy lặp đi lặp lại cho đến khi 1 vài điều kiện chỉ rõ thoả mãn

\*\*3. Câu lệnh lặp while\*\*

Trong Python, vòng lặp thường được thực hiện với câu lệnh while:

while (mệnh đề điều kiện đúng/sai):

(khối lệnh)

Chú ý: dấu hai chấm sau mệnh đề điều kiện.

Khối lệnh có thể ngắn hoặc dài, còn được gọi là thân của chương trình. Trong khi mệnh đề điều kiện còn đúng, khối lệnh trong thân vòng lặp while tiếp tục thực hiện cho đến khi điều kiện là sai và vòng lặp sẽ thoát và thực hiện dòng lệnh đầu tiên ngoài khối lệnh thân vòng while.

\*\*Ví dụ:\*\*

```python

looping = True

while looping == True:

answer = input("Vui lòng nhập kí tự 'a': ")

if answer == 'a':

looping = False # hoàn thành nhiệm vụ

else:

print("Vui lòng nhập : 'a'!)"

print("Cám ơn đã nhập 'a'")

```

Trước vòng lặp, ta gán biến boolean là đúng để lần đầu tiên vòng while kiểm tra điều kiện thấy đúng mới thực hiện phần khối lệnh thân. Tại cuối vòng lặp, sự thực thi sẽ tự động quay lại dòng đầu nếu biến boolean vẫn còn đúng và dừng khi biến boolean sai , do đó trong thân khối lệnh lặp, ta phải cài đặt một điều thoát. Nếu chúng ta không có làm kĩ điều kiện thoát , điều này sẽ dẫn đến \*\*vòng lặp vô tận(infinite loop)\*\* trừ khi bạn tắt phần mềm hoặc tắt máy tính.

\*\*4. Viết chương trình vòng lặp đầu tiên\*\*

Chương trình hỏi người nhập 1 con số , sau đó nó sẽ tính tổng từ 1 đến con số đó.

Ví dụ: nhập 4, kết quả là 1+2+3+4 = 10.

```python

target = input('Enter a target number: ')

target = int(target)

total = 0

nextNumberToAddIn = 1

while nextNumberToAddIn <= target:

# add in the next value

total = total + nextNumberToAddIn #add in the next number

print('Addd in:', nextNumberToAddIn, 'Total so far is:', total)

nextNumberToAddIn = nextNumberToAddIn + 1

print('The sum of the numbers from 1 to', target,'is:', total)

```

Thực thi chương trình trên với input = 4 , ta có kết quả sau:

Enter a target number: 4

Added in: 1 Total so far is: 1

Added in: 2 Total so far is: 3

Added in: 3 Total so far is: 6

Added in: 4 Total so far is: 10

The sum of the numbers from 1 to 4 is: 10

Chú ý: trước vòng while , ta phải lấy input từ người nhập và biến đổi thành dạng số interger, cài đặt biến tổng total = 0 , biến nextNumberToAddIn =1 , biến này sẽ tăng từ 1 đến traget(con số mà người chơi nhập vào).

Kế tiếp, vòng while sẽ chạy cho đến khi nextNumberToAddIn &gt;= target( chính là điều kiện dừng). Cuối cùng, chương trình sẽ in ra số tổng total.

\*\*5. Tăng và Giảm\*\*

Trong đoạn code trên có dòng lệnh:

nextNumberToAddIn = nextNumberToAddIn +1

Dòng này có nghĩa là lấy giá trị hiện tại của biến nextNumberToAddIn cộng 1 rồi gán lại cho chính biến đó. Phép toán này rất thường được sử dụng, người ta thường gọi là increment.

\*\*Địng nghĩa:\*\* increment là khi một giá trị được cộng vào chính nó.

Ví dụ: counter = counter + 1 tương đương với counter +=1. Tương tự: ta có đếm ngược xuống , gọi là decrement.

\*\*Địng nghĩa\*\*: decrement là khi một giá trị được trừ từ chính nó.

Ví dụ: counter = counter - 1 tương đương với `counter -= 1`. Trong chương trình đoán số, để lưu giữ bao nhiêu lần đoán, ta sử dụng bộ đếm tăng (increment) cho mỗi lần đoán.

\*\*6. Chạy chương trình nhiều lần\*\*

Trong chương trình trên, chúng ta muốn người dùng có thể chạy nhiều lần bằng cách hỏi `yes/no` và chương trình sẽ chạy lại lần nữa. Có nhiều cách để làm được điều này nhưng cách đơn giản nhất là đặt 1 hàm tại cuối thân vòng lặp. Đây là đoạn mã:

```python

def calculateSum(target):

total = 0

nextNumberToAddIn = 1

while nextNumberToAddIn <= target:

# add in the next value

total = total + nextNumberToAddIn

#increment

nextNumberToAddIn = nextNumberToAddIn + 1

return total

answer = 'y' # start off with the value 'y' to go through the first time

while answer == 'y':

usersTarget = input('Enter a target number: ')

usersTarget = int(usersTarget)

thisTotal = calculateSum(usersTarget) # call our function and get back the answer

print('The sum of the numbers 1 to', usersTarget, 'is:', thisTotal)

answer = input('Do you want to try again (y or n): ')

print('OK Bye')

```

Chú ý: trong cấu trúc này, tất cả việc tính toán được thực hiện trong 1 hàm con, thân chương trình chính làm nhiệm vụ chính là tương tác hỏi người chơi và gọi hàm con ra để tính toán kết quả tổng.

Một cách khác là chạy chương trình với số lần đã định nghĩa trước. Ví dụ: 3 lần như bên dưới:

```python

nTimes = 0 # initialize a counter

while nTimes < 3:

usersTarget = input('Enter a target number: ')

usersTarget = int(usersTarget)

thisTotal = calculateSum(usersTarget)

print('The sum of the numbers 1 to', usersTarget, 'is:', thisTotal)

nTimes = nTimes + 1 # increment the counter

print('OK Bye')

```

\*\*7. Thư viện trong Python\*\*

Trở lại chương trình đoán số. Việc kế tiếp là tìm hàm sinh số ngẫu nhiên. Để trả lời câu hỏi này, chúng ta tìm hiểu thư viện có sẵn của Python(packages).

Để làm cho chương trình nhỏ gọn, Python chỉ xây dựng vài câu lệnh cơ bản như `if`, `elif`, `else`, `while`,..., một số hàm(`int`, `str`, `inout`,...). Python cũng có xây dựng 1 một số hàm phổ biến gọi hàm thư viện chuẩn, sẽ được cài đặt vào máy tính khi người dùng cài Python.

Ngoài ra, người lập trình trên khắp thế giới cũng tự xây dựng thư viện riêng của họ, còn gọi là thư viện ngoài. Bạn có thể tải từ internet. Ví dụ: thư viện tên là PyGame dùng để viết game dùng đồ hoạ windows, chuột, âm thanh, ...

Hình dưới minh hoạ thư viện được phân ra 3 loại:

<img src="http://happyaivietnam.com/wp-content/uploads/2019/06/python\_Chuong6\_fig2\_packages-300x194.png" alt="" width="300" height="194" class="aligncenter size-medium wp-image-217" />

\*\*8. Tạo một số ngẫu nhiên\*\*

Một trong những hàm xây dựng sẵn tên là random. Khi bạn muốn dùng thư viện này, bạn phải báo Python bằng cách khai báo thư viện đó trong chương trình của bạn theo cú pháp `import &lt;&gt;`.Trong ví dụ này là `import random`

Trong trường thư viện không tồn tại, IDEL sẽ xuất ra một lỗi như sau:

```python

>>> import NonExistentPackage

Traceback (most recent call last):

File "", line 1, in

import NonExistentPackage

ImportError: No module named NonExistentPackage

>>>

```

Thư viện nên đặt ở đầu chương trình của bạn. Để biết tất cả thông tin về thư viện bạn cần dùng: `help(random)`.

IDLE sẽ liệt kê tất các hàm và cách dùng cũng như thông số khai báo. Trong ví dụ trên, ta cần hàm `randrange`:

```python

random.randrange(, )

```

Lưu ý : bạn bắt đầu với thư viện rồi đến dấu chấm tên hàm và thông số của hàm như ví dụ sau :

```python

#random between 1 and 10

aRandomNumber = random.randrange(1, 11)

#random between 1 and 52, to pick a card number from a deck

anotherRandomNumber = random.randrange(1, 53)

```

Lưu ý: phạm vi bắt đầu từ `lowValue` đến `upToButNotIncludingHighValue` nhưng khong bao gồm giá trị này

Một cách khác:

```python

#random between 0 and 8

myRandomNumber = random.randrange(9)

#tương đương với random.randrange(0,8)

```

\*\*9. Mô phỏng búng đồng xu\*\*

Đây là một ví dụ điển hình dùng hàm tạo số ngẩu nhiên. ta sẽ mô phỏng việc búng đồng tiền một vài lần. Chương trình sẽ dùng vòng lặp, mỗi lần lặp, ta sẽ tạo ra 1 số ngẫu nhiên 0 hoặc 1, sau đó ánh xạ với mặt xấp và mặt ngữa, kết thúc chương trình sẽ in ra kết quả:

```python

#Coin flip program

import random

nFlips = 0 # to count the number of flips

nTails = 0 # to count the number of flips that came up as tails

nHeads = 0 # to count the number of flips that came up as heads

maxFlips = input('How many flips do you want to do? ')

maxFlips = int(maxFlips)

while nFlips < maxFlips:

# Randomly choose 0 or 1, because a coin flip can only result in one of two answers

# (heads or tails)zeroOrOne = random.randrange(0, 2)

# If we get a zero, say that was a heads

# If we get a one, we say that was a tails

zeroOrOne = random.randrange(0, 2)

if zeroOrOne == 0:

nTails = nTails + 1

else:

nHeads = nHeads + 1

nFlips = nFlips + 1

print()

print('Out of', nFlips, 'coin tosses, we had:', nHeads, 'heads, and',nTails, 'tails.')

```

Kết quả:

How many flips do you want to do? 10

Out of 10 coin tosses, we had: 7 heads, and 3 tails.

<img src="http://happyaivietnam.com/wp-content/uploads/2019/06/python\_Chuong6\_fig3\_flippingCoin-1-1024x542.png" alt="" width="525" height="278" class="aligncenter size-large wp-image-229" />

\*\*10. Tạo vòng lặp vô tận\*\*

Như giới thiệu ở trên về vòng lặp while, trong khi biến boolean bằng True,vòng lặp chạy liên tục cho đến khi biến boolean bằng False, bất kỳ vòng lặp đều có thể xây dựng bằng cấu trúc này.

Biến boolean được xem là điều kiện thoát, thường dùng câu lệnh if để kiểm tra điều kiện thoát có hoạt động hay không.

```python

looping = True

while looping:

statement(s)

if found exit condition? :

looping = False # found the exit condition at this point

else:

continuing statement(s) inside the loop

```

Trong đoạn mã trên có điều kiện thoát nhưng bạn không thể thoát cho đến khi lệnh thực thi quay trở lại đầu câu lệnh while, kiểu viết này thường rất khó viết phần chuỗi câu lệnh sau else và dễ dẫn đến chạy vô tận, ví dụ như câu lệnh thụt lề quá nhiều dẫn đến không kiểm soát được. May mắn có một cách khác xây dựng vào while hiệu quả hơn.

\*\*11. Phương cách mới xây dựng vòng lặp: while True, and break\*\*

Như đã nói ở trên, bạn có thể tạo ra vòng lặp vô tận một caách tai nạn khi biến boolean không thay đổi giá trị được. Vì thế , cách đơn giản để tạo ra lặp vô tận như sau:

while True:

statement(s)

Python cung cấp câu lệnh khác gọi là break. Nếu dòng mã lệnh đang chay, gặp câu lệnh break, bộ điều khiển sẽ chuyễn đến câu lệnh đầu tiên sau dòng cuối cùng của vòng lặp như sơ đồ sau:

<img src="http://happyaivietnam.com/wp-content/uploads/2019/06/python\_Chuong6\_fig4\_breakwhile-655x1024.png" alt="" width="525" height="821" class="aligncenter size-large wp-image-236" />

Vi dụ:

```python

while True: # loop forever

line = input("Type anything, type 'done' to exit: ")

if line == 'done':

break # transfers control out of the loop

print('You entered:', line)

print('Finished')

```

Type anything, type 'done' to exit:Hello

You entered: Hello

Type anything, type 'done' to exit: done

Finished

<img src="http://happyaivietnam.com/wp-content/uploads/2019/06/python\_Chuong6\_fig5\_breakdemo-1024x406.png" alt="" width="525" height="208" class="aligncenter size-large wp-image-241" />

Trong trò chơi đoán số, ta có 2 điều kiện kiểm tra thoát: một là kiểm tra người chơi đã nhập số đúng chưa, hai là kiểm tra số lần nhập vượt quá số lần đoán qui định. Nếu dùng break sẽ làm cho mã code dễ dàng và rõ ràng hơn.

Tóm lại viết vòng lặp while vô tận với câu lệnh break để thoát là rất hiệu quả và rõ ràng. Hâu hết các vòng loop tác giả đều dùng kiểu này.

\*\*12. Hỏi nếu người chơi muốn chơi lần nữa\*\*

Chúng ta gần như xây dựng xong game đoán chữ, trong trường hợp người chơi muốn chơi lại, ta phải lặp lại chương trình này lần nữa theo vòng lặp while như sau:

```python

while True:

# do whatever, e.g., play a round of a game

# now ask the user if they want to go again

goAgain = input('Press Return/Enter to quit, or anything else to continue: ')

if goAgain =="" : # check for no entry

break # user said they want to quit

print('Finished')

```

<img src="http://happyaivietnam.com/wp-content/uploads/2019/06/python\_Chuong6\_fig6\_repeat\_empty-1024x300.png" alt="" width="525" height="154" class="aligncenter size-large wp-image-246" />

Giải thích đoạn mã trên : biến goAgain sẽ được gán bất kỳ giá trị nhập vào trừ khi nhấn Enter để thoát.

\*\*Mã giả ( Pseudocode):\*\*

Bây giờ, ta đủ thông tin để viết chương trình. Nhưng thay vì nhảy vào viết chương trình liền, ta hãy thảo luận bằng cach nào chương trình chạy, ta sẽ đưa ra một cách tiếp cận tổng thể trước khi viết mã thực tế.

\*\*Định nghĩa:\*\* mã giả là một miêu tả chương trình giống như ngôn ngữ tiếng Anh của một giải thuật trong ngôn ngữ máy tính.

Thông thường, các lập trình viên phát triển một thuật toán và viết thuật toán bằng mã giả trước khi viết nó bằng ngôn ngữ máy tính thực sự. Điều này cho phép các lập trình viên suy nghĩ thông qua cách tiếp cận cấu trúc tổng thể của một giải pháp mà không phải lo lắng về cú pháp chi tiết của ngôn ngữ máy tính.

Ví dụ, mã giả của 1 vòng của trò chơi đoán số:

Show introduction

Choose random target

Initialize a guess counter

Loop forever

Ask the user to for a guess

Increment guess counter

If user's guess is correct, congratulate user, we're done

If user's guess is too low, tell user

If user's guess is too high, tell user

If reached max guesses, tell correct answer, we're done.

Chúng ta có thể dùng mã giả này để chuyển thành chú thích, nó rất cần thiết như một phác thảo bên trong mã của chương trình.

\*\*13. Xây dựng chương trình đoán số\*\*

Chúng ta sẽ viết từng phần nhỏ vả kiểm tra hơn là viết toàn bộ từ trên xuống. Đây là kỹ thuật chuẩn để viết các chương trình lớn hơn. Viết hay thay đổi nhiều thứ cùng một lúc khiến quá trình xử lý trở nên khó khăn bởi vì nếu có một lỗi xuất hiện, khó mà phát hiện dòng mã nào gây ra lỗi đó và có thể là lỗi logic cần được xử lý.

Ta bắt đầu viết phần đoán số người chơi, thật khó đển đoán con số sinh ra ngẫu nhiên khi kiểm tra chương trình chạy đúng hay sai, nên ta gán số cần đoán là 10, sau đó sẽ chỉnh mã sang số ngẫu nhiên khi chương trình chạy hoàn chỉnh:

```python

#Guess the Number (version 1)

#Show introduction

#Choose random target

target = 10 # start with a known value

#Initialize a guess counter

#Loop forever

#Ask the user to for a guess

userGuess = input('Take a guess: ')

userGuess = int(userGuess)

#Increment guess counter

#If user's guess is correct, congratulate user, we're done

if userGuess == target:

print('You got it!')

#If user's guess is too low, tell user

elif userGuess < target:

print('Your guess was too low.')

#If user's guess is too high, tell user)

else:

print('Your guess was too high.')

#If reached max guesses, tell answer correct answer, we're done.

```

<img src="http://happyaivietnam.com/wp-content/uploads/2019/06/python\_Chuong6\_fig7\_version1-1024x527.png" alt="" width="525" height="270" class="aligncenter size-large wp-image-264" />

Bay giờ, ta thêm biến lưu số lần đoán:

```python

#Guess the Number (version 2)

#Show introduction

#Choose random target

target = 10 # start with a known value

#Initialize a guess counter

guessCounter = 0

#Loop forever

while True:

# Ask the user to for a guess

userGuess = input('Take a guess: ')

userGuess = int(userGuess)

#Increment guess counter

guessCounter = guessCounter + 1

# If user's guess is correct, congratulate user, we're done

if userGuess == target:

print('You got it!')

print('It only took you', guessCounter, 'guess(es).')

break

#If user's guess is too low, tell user

elif userGuess < target:

print('Your guess was too low.')

#If user's guess is too high, tell user

else:

print('Your guess was too high.')

#If reached max guesses, tell answer correct answer, we're done.

if guessCounter == 5:

print('Sorry, you did not get it in 5 guesses')

print('The number was:', target)

print('Thanks for playing.')

```

<img src="http://happyaivietnam.com/wp-content/uploads/2019/06/python\_Chuong6\_fig7\_version2-1024x762.png" alt="" width="525" height="391" class="aligncenter size-large wp-image-266" />

Chương trình trên có vẻ chạy ổn, bây giờ ta thêm hàm sinh số ngẫu nhiên từ 5 đến 20:

```python

#Guess the Number (version 3)

import random

MAX\_GUESSES = 5 # maximum number of guesses allowed

MAX\_RANGE = 20 # highest possible number

#Show introduction

print('Welcome to my Guess the Number program.')

print('Guess my number between 1 and', MAX\_RANGE)

print('You will have', MAX\_GUESSES, 'guesses.')

#Choose random target

target = random.randrange(1, MAX\_RANGE + 1)

#Initialize a guess counter

guessCounter = 0

#Loop forever

while True:

# Ask the user to for a guess

userGuess = input('Take a guess: ')

userGuess = int(userGuess)

#Increment guess counter

guessCounter = guessCounter + 1

#If user's guess is correct, congratulate user, we're done

if userGuess == target:

print('You got it!')

print('It only took you', guessCounter, 'guess(es).')

break

#If user's guess is too low, tell user

elif userGuess < target:

print('Your guess was too low.')

#If user's guess is too high, tell user

else:

print('Your guess was too high.')

#If reached max guesses, tell answer correct answer, we're done.

if guessCounter == MAX\_GUESSES:

print('Sorry, you did not get it in', MAX\_GUESSES, 'guesses.')

print('The number was:', target)

break

print('Thanks for playing.')

```

<img src="http://happyaivietnam.com/wp-content/uploads/2019/06/python\_Chuong6\_fig7\_version3-950x1024.png" alt="" width="525" height="566" class="aligncenter size-large wp-image-268" />

Bây giờ , ta thêm mã chơi nhiều lần, có nhiều cách như while loop lồng trong while loop hay xây dưng hàm con rồi gọi trong hàm chính, ta chọn cách 2 trong ví dụ này:

```python

#Guess the Number (version 4)

import random

MAX\_GUESSES = 5 # maximum number of guesses allowed

MAX\_RANGE = 20 # highest possible number

#Show introduction

print('Welcome to my Guess the Number program.')

print('Guess my number between 1 and', MAX\_RANGE)

print('You will have', MAX\_GUESSES, 'guesses.')

def playOneRound():

# Choose random target

target = random.randrange(1, MAX\_RANGE + 1)

# Initialize a guess counter

guessCounter = 0

# Loop forever

while True:

# Ask the user to for a guess

userGuess = input('Take a guess: ')

userGuess = int(userGuess)

# Increment guess counter

guessCounter = guessCounter + 1

# If user's guess is correct, congratulate user, we're done

if userGuess == target:

print('You got it!')

print('It only took you', guessCounter, 'guess(es).')

break

# If user's guess is too low, tell user

elif userGuess < target:

print('Your guess was too low.')

# If user's guess is too high, tell user

else:

print('Your guess was too high.')

# If reached max guesses, tell answer correct answer, we're done.

if guessCounter == MAX\_GUESSES:

print('Sorry, you did not get it in', MAX\_GUESSES, 'guesses.')

print('The number was:', target)

break

#main code

while True:

playOneRound() # call a function to play one round of the game

goAgain = input('Play again? (Press ENTER to continue, or q to quit):')

if goAgain == 'q':

break

print('Thanks for playing.')

```

<img src="http://happyaivietnam.com/wp-content/uploads/2019/06/python\_Chuong6\_fig7\_version4-789x1024.png" alt="" width="525" height="681" class="aligncenter size-large wp-image-270" />

\*\*14. Phát hiện lỗi với Try/except\*\*

Trong bất kỳ chương trình mà hỏi người chơi nhập số, đôi khi vì vái ký tự chữ được nhập vào dẫn đến chương trình ngưng chạy, l;úc đó Python sẽ xuất ra lỗi như ví dụ sau:

&gt;&gt;&gt; userInput = input('Please enter a number: ')

Please enter an integer: xyz

&gt;&gt;&gt; userInput = int(userInput) # convert user's input to an integer

Traceback (most recent call last):

File "", line 1, in

userInput = int(userInput)

ValueError: invalid literal for int() with base 10: 'xyz'

Do đó, cách tốt nhất là phát hiện và thông báo người chơi lỗi hơn làn để Python phát hiện và chương trình bị dừng.

Python cung cấp một kỹ thuật để kiểm tra lỗi nhưng mất ít thời gian để làm quen. Về lý thuyết, trước khi chạy mã mà có thể gây ra lỗi, ta yêu cầu Python xem cái gì đang diễn ra, nếu lỗi xảy ra trong quá trình Python o chế độ giám sát, bạn có thể yêu cầu Python chạy thêm một đoạn mã để in ra loại mã lỗi và thông báo người chơi chi tiết lỗi này để họ nhập hay chỉnh lại cho đúng.

Dưới đây là cú pháp:

try:

(câu lệnh có thể gây ra lỗi )

except:

(câu lệnh thực thi nếu lỗi xảy ra)

else: #optional, often not needed

( câu lệnh thực thi nếu không lỗi )

Trong ví dụ này, ta muốn nhập số integer:

userGuess = input(' Take a guess: ')

try:

userInput = int(userInput)

except:

print('The number you entered was not an integer')

Khối lệnh try có thể chứa nhiều lệnh thay vì một lệnh như ví dụ trên. Mặt tốt là nếu mã chương trình có lỗi và thông báo, mặt khác chương trình vẫn tiếp tục chạy với biến lỗi dẫn đến việc so sánh giữa số và ký tự gặp lỗi. Do đó, ta cần vài câu lệnh nữa để sữa lỗi này.

\*\*15. Câu lệnh continue\*\*

Khi ta phát hiện một vài lỗ trong vòng lặp, ta muốn truyền điều khiển ngược trở về đầu vòng lặp while mà không thực thi phần còn lại ben trong vòng lặp, để làm điều này, ta thực thi câu lệnh continue được minh hoạ như hình sau:

<img src="http://happyaivietnam.com/wp-content/uploads/2019/06/python\_Chuong6\_fig8\_continue-735x1024.png" alt="" width="525" height="731" class="aligncenter size-large wp-image-273" />

Ví dụ:

```python

while True: # loop forever

line = input("Type anything, type 'done' to exit: ")

if line == 'done':

break # transfers control out of the loop

if line == 'skip':

continue # transfers control back to the while statement

print('You entered:', line)

print('Finished')

```

<img src="http://happyaivietnam.com/wp-content/uploads/2019/06/python\_Chuong6\_fig9\_continuedemo-1024x310.png" alt="" width="525" height="159" class="aligncenter size-large wp-image-275" />

\*\*16. Game hoàn chỉnh\*\*

```python

#Guess the Number (version 5 - final)

import random

MAX\_GUESSES = 5 # maximum number of guesses allowed

MAX\_RANGE = 20 # highest possible number

#Show introduction

print('Welcome to my Guess the Number program.')

print('Guess my number between 1 and', MAX\_RANGE)

print('You will have', MAX\_GUESSES, 'guesses.')

def playOneRound():

# Choose random target

target = random.randrange(1, MAX\_RANGE + 1)

# Initialize a guess counter

guessCounter = 0

#Loop forever

while True:

# Ask the user to for a guess

userGuess = input('Take a guess: ')

#Check for potential error

try:

userGuess = int(userGuess)

except:

print('Hey, that was NOT an integer!')

continue # transfer control back to the while

#Increment guess counter

guessCounter = guessCounter + 1

#If user's guess is correct, congratulate user, we're done

if userGuess == target:

print('You got it!')

print('It only took you', guessCounter, 'guess(es).')

break

#If user's guess is too low, tell user

elif userGuess < target:

print('Your guess was too low.')

#If user's guess is too high, tell user

else:

print('Your guess was too high.')

#If reached max guesses, tell answer correct answer, we're done.

if guessCounter == MAX\_GUESSES:

print('Sorry, you did not get it in', MAX\_GUESSES, 'guesses.')

print('The number was:', target)

break

#main code

while True:

playOneRound() # call a function to play one round of the game

goAgain = input('Play again? (Press ENTER to continue, or q to quit):')

if goAgain == 'q':

break

print('Thanks for playing.')

```

<img src="http://happyaivietnam.com/wp-content/uploads/2019/06/python\_Chuong6\_fig10\_demofinal.png" alt="" width="1016" height="1756" class="aligncenter size-full wp-image-278" />

\*\*17. Tổng kết\*\*

Trong chương này, chúng ta đã xây dựng trò chơi đoán số. Để hiện thực, bạn đã học về định ngĩa vòng lặp while tromng Python, khái niệm tăng / giảm biến , dùng vòng loop đễ chạy chương trình nhiều lần, học cách khai báo và sửng dụng hàm trong thư viện chuẩn có sẵng trong Python như sử dụng hàm randrange để sinh số ngẫu nhiên và xây dựng chương trình búng đồng xu cho hàm randragnge này.

Sau đó, bạn biết nhiều cách xây dựng vòng lặp while vô tận và cách thoát khỏi loop bằng cách dùng câu lệnh break và câu lệnh continue để khắc phục từ lỗi nhập sai người chơi. Dùng vòng lặp để hỏi người chơi nhiều lần bằng cách kiểm tra chuỗi rỗng, biết lợi ích của miêu tà mã giả của giải thuật để xây dưng chương trình và kiểm tra từng phần của toàn chương trìnhh.

Cuối cùng, bạn biết cách bắt lỗi bằng dùng khối lệnh try/except và biết cách xây dựng hàm con có thể tái sử dụng để kiểm tra gia trị integer có trong phạm vi qui định.