MOONHACK 2020

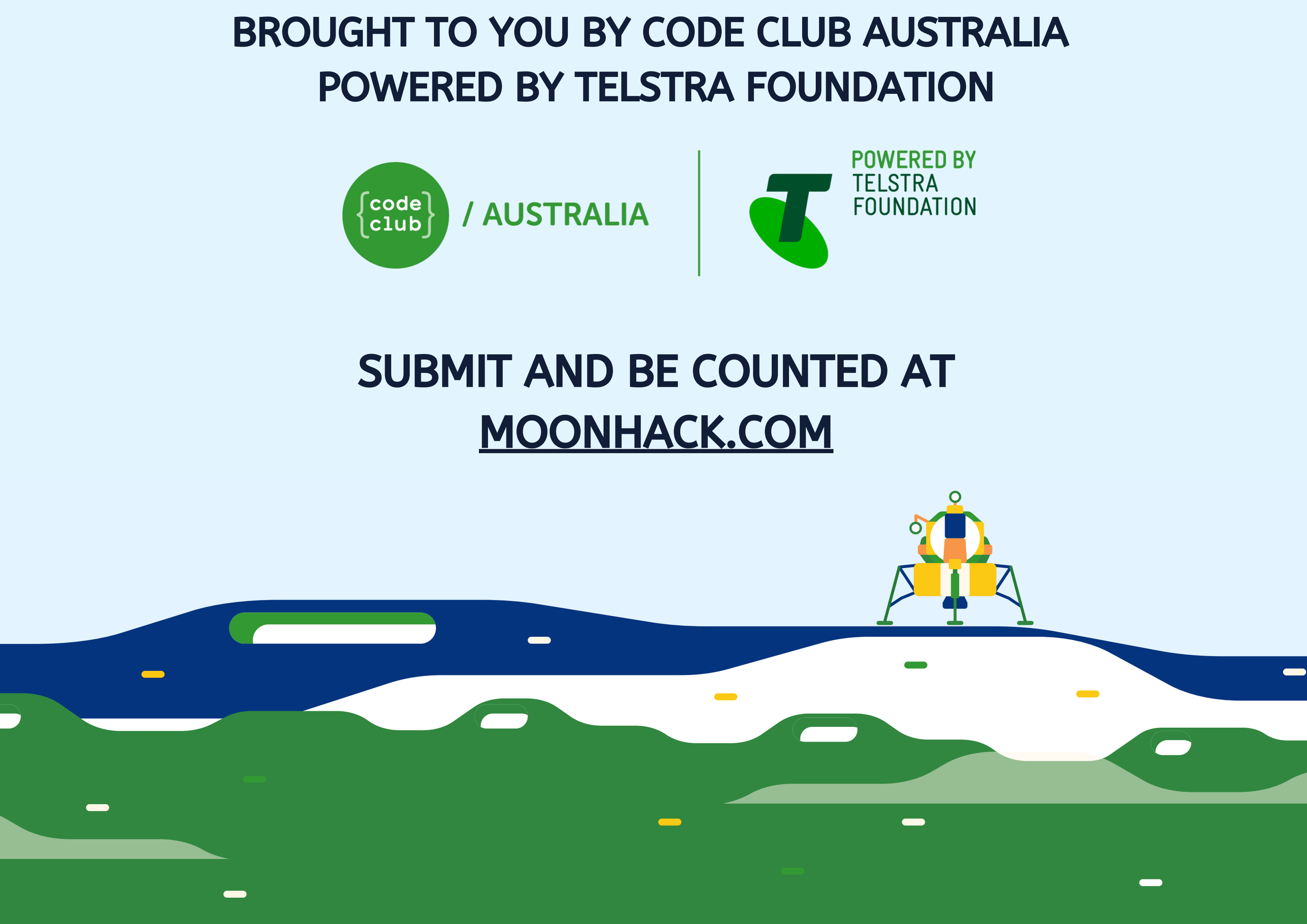
PYTHON WATER ON MARS

JAPANESE

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



PYTHON WATER ON MARS
MOONHACK2020

火星の水
水は宇宙でも信じられないほど貴重な資源です！
火星に必要な水はどのくらいでしょうか？

０　イントロダクション
①　あなたが作るもの
あなたはテキストベースのサバイバルゲームを作ります。
プレイヤーは火星で水を使い果たすことがないよう、できるだけ長く生存できるよう、水を意識する必要があります。

②　あなたが学ぶこと
・テキストベースでの変数の使用と作成

③　必要なもの

ア　ハードウェア
オンラインでTrinketにアクセスできるコンピュータ

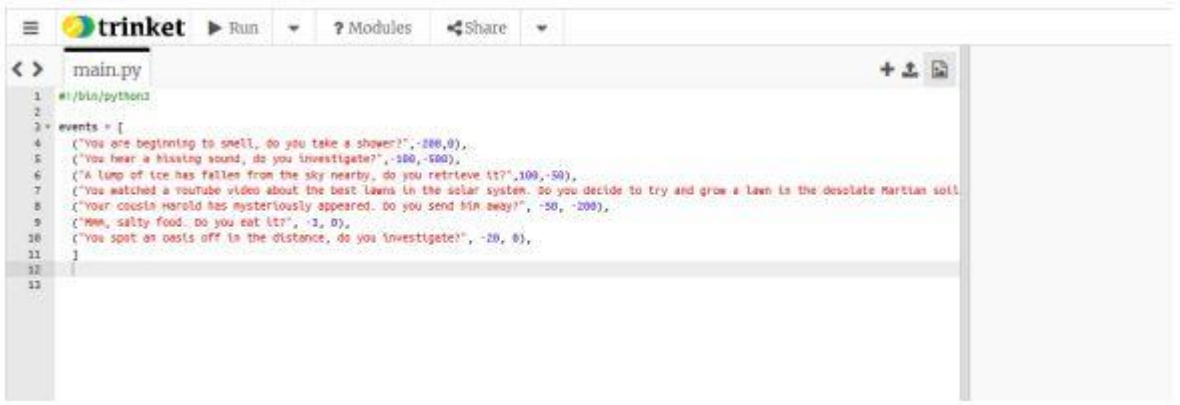
イ　ソフトウェア
スターターコードはここにあります。
bit.ly/pythonwater
（https://trinket.io/embed/python/1d6d37a556）

④　その他の注意事項

教育者向け
このプロジェクトのブログ記事をチェックしてください。
ヒント、カリキュラム、サポート資料
medium.com/@codeclubau

1　水の量の把握
水の量を把握するための変数を設定しましょう。

①　準備
・bit.ly/pythonwater
上記のアドレスで、Trinketのスタータープロジェクトを開きます。
ウィンドウは次のようになります。



②　画面の解説
左側はコードを記述する場所です。 ここにすでにいくつかのコードがあることに気づくでしょう（これについては後で説明します）。
右側は実行する場所です。ゲームが表示されます。
上部には実行ボタンがあります。 このボタンを押すとコードが実行されます。

③　水を供給
開始するにはプレイヤーに水を与える必要があります
キリのいい数字から始めましょう。最初は1000L。
次の水色で強調表示されたコードを、プログラムの最後に書きます。

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

④　メインループ
ユーザーには水がある限りゲームを続けてもらいたいと考えています。
水がなくなるとゲームは終了します。ユーザーが水を持っている限りゲームを続けられるように、メインのループを作りましょう。

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

行末のコロン（:)に注目してください。
この小さな見落としは取るに足らないように見えるかもしれませんが、
Pythonが文句を言うでしょう！

⑤　ループ内に毎日使用する水の量を追加
次に、メインのゲームループに水の使用量を追加してみましょう。これは、プレイヤーの選択に関係なく、プレイヤーは水を飲む必要があるため、毎日減算される量になります。
ループの中に次のコードを追加してください。
16行目の前にある2つのスペースに注目してください。これを「インデント」と呼びます。pythonに、その行の命令がループの中にあることを伝えます。言い換えれば、これはゲームのメインループを回るたびに発生します（Scratchを使っている場合は、オレンジ色のループブロックに似ています)。

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

⑥　出力の時間
さて、いよいよ出力の時間です。水がなくなるとゲームが終了することをユーザーに伝えましょう。以下のコードをゲームループの外側に追加してください（インデントを取り除くことを忘れないでください！）。

コードを実行してください。何が起こりますか？

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

⑦　メインループに出力を追加
Pythonはただ、私たちのコードをできるだけ早く実行しようとします。
メインループに出力がなければ、最後まで何が起こっているのかを知る方法がありません。
メインループに出力を追加してみましょう。

コードをもう一度実行してください。
これで、ゲームが終了するまでに経過したすべての日を見ることができるはずです。

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

⑧　水の量を表示
最後に挿入したコードを変更して 水の量を表示させてみましょう。
+記号を使用して、変数の値を印刷しているテキストと組み合わせることができます。

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

コードをもう一度実行してみてください。エラーは出ましたか？

⑨　TypeError: cannot concatenate 'str' and 'int' objects
プログラミングの大きな部分は、なぜエラーが出ているのかの原因を解明しようとすることです。
今回の場合、「'str'と'int'オブジェクトを連結できません」というエラーが出ています。
プログラミングをしていると、これらの奇妙なエラーメッセージが何を意味しているのかを理解しなければならないことがよくあります。
このエラーは、Python にはない 2 つの異なるものを組み合わせようとしていることを教えてくれています。テキスト（文字列（string）、strに短縮されたもの）、と水の量を把握するために使用している数字(整数（integer）、短縮してint)との組み合わせ方法。
これを修正するには、Python に 整数を次のコードで文字列に変換します。

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

コードを実行してください。
プレイヤーのキャラクターが水を使い果たす前に、毎日使用している水量を示す出力が得られるはずです。

１－２　チャレンジ：毎日の水を変える
現在、1日あたり5リットルの水を使用しています。
人が一人生き残るのに必要最低限の量を、あなたはどう考えますか？
プレイヤーがその量の水を使うように コードを変更できますか？

２　日々の把握
これまで、プレイヤーが使用した水の量を記録してきましたが、どの程度の成績を残しているかを見るために、経過した日数を記録していきます。

①　日数の初期値を設定
最初に、1日目から始まる日数の変数の初期値を設定します。

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

② 毎日経過するごとに１を追加
次に、1日経過するごとに、現在の日の値を1つずつ増やしていきたいと思います。
以下のコードをゲームのメインループに追加します。

```
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 :(")
```

③ 成績の報告
最後に、プレイヤーに自分の成績を知らせます。これを行うには、最終的なprint文を、経過した日数を含む文に置き換えます。

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

数値をテキストに変換するために、再び str(day) を使用する必要があることに注意してください。

コードを実行してください。毎日出力され、最終的にはプレイヤーが何日続いたかを示す出力が得られるはずです。

２－２ チャレンジ：毎日、何日目かの日にち番号を出力する
現在は最後の最後にday変数を使っているだけです。
毎日、日にち番号を出力するようにコードを変更することはできますか？

現在、出力は次のようになっています。
A day has passed. Your water 950L
これを、次のように変更できますか？
Day 10. Your Water 950L

３ あなたの日々の出来事
今度はゲームプレイを追加します。
毎日、プレイヤーに選択を与え、その選択によって水の量を失うか・得るかが決まるのです。

① イベントリストからの選択
最初にスターターキットを開いたとき、自分で記述していないコードがたくさんあることに気づいたでしょう。
これは火星で起こりうるイベントのリストです。メインループにもう一つprint文を追加してみましょう。
これはイベントリストにアクセスして、ランダムに1つを選択します。

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

② NameError: name 'random' is not defined
ヤバいよヤバいよ！またもやエラーです。これは「名前 'random' が定義されていません」と書いてあります。これはrandomがデフォルトではpythonの一部ではなく、モジュールの一部だからです。
それを使えるようにするにはインポートする必要があります。

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

あなたのコードを実行してください。毎日、ランダムなイベントが出力されているはずです。

③ event変数に格納
与えられた日に複数回イベントを使用する必要があります。
もしイベントを使いたいときに毎回 random.choice（event）を使っていたら、違うイベントが出てきてしまいます。その代わりに、そのイベントを event という変数に格納しておきましょう。

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

もう一度コードを実行してみてください。前回と同じような結果が得られるはずです。

④ インデックス表記
出力されたイベントの周りには、たくさんの余分な情報や記号があることに気づくでしょう。それは、質問内容と「はいyes」か「いいえno」と答えた場合に得られる水の量、または失う水の量の値を格納しているからです。

私たちは、プレイヤーに余分な情報を見せずに質問をしたいだけです。

質問にアクセスするには、インデックス表記法と呼ばれるものを使用します。インデックスは ０から始まる要素の位置を指定しています。
print行を以下のように変更します。

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

⑤　input文
もう一度コードを実行してみてください。これでだいぶ良くなりましたが、プレイヤーは質問に答えられません。
print文の代わりにinput文を使ってみましょう。これは python にユーザーが何かを書くことを期待するように指示します。

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

⑥　ユーザーの入力を変数に保存
さて、その入力を変数に保存してみましょう。

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

コードをもう一度実行してください。これで、あなたのゲームは毎日プレイヤーが反応するのを待つようになりました。

４　.プレイヤーの答えを使って
プレイヤーは質問に答えていますが、その答えを使って水の使用量を判断する必要があります。

①　ユーザーの回答が「yes」の場合の反応
ユーザーは質問に「yes」か「no」のどちらかで答えなければなりません。
「yes」と答えた場合は、イベントの最初の数字を追加します。

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

足し算をしているにもかかわらず、イベントの数字の多くがマイナスになっていることに注意してください。
数学と同じように、負の数を足すことは、その数の正の数を引くことと同じです。

②　ユーザーの回答が「no」の場合の反応

プレイヤーが「no」と答えた場合は、イベントに2番目の数字を追加する必要があります。

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

コードを実行してください。これでプレイヤーは質問に答えることができるようになります。プレイヤーが持っている水の量は、プレイヤーの決定に応じて変化することに気づくでしょう。
もしプレイヤーが「yes」か「no」以外の答えをしたらどうなるでしょうか？

５　ユーザー入力を検証する
これで動作するゲームになりましたが、プレイヤーはどのように答えても構いません。私たちは答えとして「yes」と「no」だけを受け入れたいと考えています。

①　ユーザーの回答を２パターンに促す
yesとnoの2つの選択肢しかないので、ユーザーに答えて欲しい方法で答えるように促すことができます。回答行を以下のように修正してください。

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

②　″else ″文を追加
それでも良いのですが、プレイヤーは自分の好きなように答えることができます。
プレイヤーはゲームをごまかすために他のものを入力したり、答えを誤入力したりする可能性があります。
yesかno以外の回答もキャッチするために、″else ″文を追加してみましょう。

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

コードを実行してください。プレイヤーが「yes」か「no」以外の何かを入力した場合はどうなりますか？

③　whileループの追加
今はユーザーに回答を修正するよう求めていますが、とにかく次の質問をしています！

ユーザーが私たちの求める答えをくれるまで、同じ質問をし続ける必要があります。ループを使ってこれを行うことができます。
これを行うには、インデントする行を強調表示し、キーボードの「Tab」を押します。
これにより、強調表示されたテキストをすべて、先ほど作成した「while True」ループに追加します。

```
event = random.choice(events)
while True:
  response = input(event[0] + " (yes/no): ")
  if response == "yes":
    water += event[1]
  elif response == "no":
    water += event[2]
  else:
    print("Please answer yes or no!")
day += 1
```

④　実行して確認
あなたのコードを実行してください。yesかno以外の答えをした場合はどうなりますか？もう一度質問をするはずです。
しかし、yesかnoと答えた場合はどうなるのでしょうか？
それでもまだあなたに質問をしてきます！
うああ！！！！！！！！！！！

⑤　breakコマンドの追加
「while True」ループは常にループし続けるループです。
これらは永遠にループし続けたい場合に非常に便利ですが、我々は永遠にループし続けたいのではなく、ユーザーがyesかnoと答えるまでループし続けたいだけなのです。
幸いなことに、ループから抜け出すことができる特別なコマンドがあります。
そのコマンドが「break」です。
プレイヤーが「yes」か「no」と答えたときに breakコマンドを追加してみましょう。

```
    response = input(event[0] + " (yes/no): ")
    if response == "yes":
        water += event[1]
        break
    elif response == "no":
        water += event[2]
        break
    else:
```

⑥　もう一つの問題
あなたのコードをもう一度実行してください。yesかnoで答えなければ、同じ質問が再び出てくるようになっているはずです。
これで物事はうまくいっていますが、一部のプレイヤーが直面するかもしれない小さな問題がもう一つあります。
プレイヤーが「yes」と答えるのではなく、大文字のYで「Yes」と答えたらどうなるでしょうか？

⑦　文字列をすべて小文字にするコマンドの追加
Pythonは大文字と小文字を区別します。
つまり、2つの文字列を比較している場合、大文字小文字が違うだけでPythonはその2つの文字列をは別の文字列と勘違いしてしまいます。
幸いなことに、これは簡単に修正できます。
Pythonには、文字列をすべて小文字にするためのコマンドがあります。
応答行を以下のように変更してください。

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

コードを実行してください。これで、"Yes"、"YES"、"yeS "やその他の大文字小文字の組み合わせにも反応するようになりました。

５－２　チャレンジ：さらにイベントの追加
いくつかの異なるイベントがありますが、もっと追加できますか？

ヒント：　もし難しい場合は、すでにあるイベントをコピーして編集してみてください。

５－３　チャレンジ：地球ゲーム
火星ではなく地球をテーマにしたゲームに修正できますか？
どのような出来事が起こるか、プレイヤーが直面する可能性のある選択肢を考えてみてください。

５－４　高度なチャレンジ：もう一回やってみてください
プレイヤーがゲームに負けた後もプレイを続けるオプションを与えることはできますか？

６　おめでとうございます。
あなたは立派なムーンハック製作者です！
あなたが成し遂げたことを周りの大人や友達と話すことを忘れないでください。