

PROBLEM!

- There is so much going on in my main function; or
- I keep repeating myself over and over

```
1  from random import randint
2
3  low_number = 1
4  high_number = 100
5
6  max_guesses = 10
7  current_guess = 0
8
9  guess = 0
10 answer = randint(low_number, high_number)
11
12 while guess != answer and max_guesses -
13     current_guess > 0:
14     guess = int(input(f"Guess a number
15         between {low_number} and {high_number}.
16         You have {max_guesses-current_guess}
17         remaining: "))
18
19     if guess < low_number or guess >
20         high_number:
21         continue
22
23     if guess < answer:
24         print(f"{guess} is too low")
25     elif:
26         print(f"{guess} is too high")
27
28     current_guess = current_guess + 1
29 print(f"That's good, {guess} was the right
30 number!!!!")
31
```

WOULDN'T IT BE GOOD IF WE COULD MOVE STUFF OUT OF THE LOOP?

Well, we can. We can also do this to remove repetition.

HOW?

```
1  from random import randint
2
3  min_value = 1
4  max_value = 100
5  answer = randint(min_value, max_value)
6  guess = -1
7  moves = 0
8
9  def get_guess(min_value, max_value):
10     guess = int(
11         input(
12             f'''Guess a number between
13             {min_value} and
14             {max_value} '''
15         )
16     )
17     return guess
18
19 while answer != guess:
20     guess = get_guess(min_value, max_value)
21
22     if guess < min_value or guess >
23     max_value:
24         continue
25
26     moves = moves + 1
27     if guess < answer:
28         return "your guess was too low!"
29     elif guess > answer:
30         return "your guess was too high!"
31
32 print(f"{guess} was the right answer. You
33 won in {moves} moves.")
```

REFACTORING

```
1  from random import randint
2
3  min_value = 1
4  max_value = 100
5  answer = randint(min_value, max_value)
6  guess = -1
7  moves = 0
8
9  def get_guess(min_value, max_value):
10     guess = int(
11         input(
12             f'''Guess a number between
13             {min_value} and
14             {max_value} '''
15         )
16     )
17     return guess
18
19 while answer != guess:
20     guess = get_guess(min_value, max_value)
21
22     if guess < min_value or guess >
23     max_value:
24         continue
25
26     moves = moves + 1
27     if guess < answer:
28         return "your guess was too low!"
29     elif guess > answer:
30         return "your guess was too high!"
31
32 print(f"{guess} was the right answer. You
33 won in {moves} moves.")
```

GETTING RID OF BAD CODE SMELLS

```
1  def get_hint(guess, answer):
2      if guess < answer:
3          return "your guess was too low!"
4      elif guess > answer:
5          return "your guess was too high!"
6
7  while answer != guess:
8      guess = get_guess(min_value,
9      max_value)
10     if guess < min_value or guess >
11     max_value:
12         continue
13     moves = moves + 1
14     print(get_hint(guess, answer))
```

FIXING MORE SMELLS!

```
1 while answer != guess:
2     guess = get_guess(min_value, max_value)
3
4     if guess < min_value or guess >
max_value:
5         continue
6
7     moves = moves + 1
8     print(get_hint(guess, answer))
```

LAST FUNCTION

```
1  from random import randint
2
3  min_value = 1
4  max_value = 100
5  answer = randint(min_value, max_value)
6  guess = -1
7  moves = 0
8
9  def get_guess(min_value, max_value):
10     guess = -1
11     while guess < min_value or guess >
max_value:
12         guess = int(
13             input(
14                 f"Guess a number between
{min_value} and {max_value} "
15             )
16         )
17     return guess
18
19 def hint(guess, answer):
20     if guess < answer:
21         return "your guess was too low!"
22     elif guess > answer:
23         return "your guess was too high!"
24
25 while answer != guess:
26     guess = get_guess(min_value,
max_value)
27     moves = moves + 1
28     print(hint(guess, answer))
29
30
31 print(f"{guess} was the right answer. You
won in {moves} moves.")
```


ONE LAST ODDITY

```
1  from random import randint
2
3  min_value = 1
4  max_value = 100
5  answer = randint(min_value, max_value)
6  guess = -1
7  moves = 0
8
9  def get_guess(min_value, max_value):
10     guess = -1
11     while guess < min_value or guess >
max_value:
12         guess = int(
13             input(
14                 f"Guess a number between
{min_value} and {max_value} "
15             )
16         )
17     return guess
18
19 def hint(guess, answer):
20     if guess < answer:
21         return "your guess was too low!"
22     elif guess > answer:
23         return "your guess was too high!"
24
25 while answer != guess:
26     guess = get_guess(min_value,
max_value)
27     moves = moves + 1
28     print(hint(guess, answer))
29
30
31 print(f"{guess} was the right answer. You
won in {moves} moves.")
```

OKAY, SO PARTS OF A FUNCTION

1. Function signature: must be unique - also don't name it something that python already has
2. Function body: is indented, and runs procedurally from top to bottom
3. Function call: this is where the function is called

```
1 def greet_user():  
2     print("Hello!")  
3 greet_user()
```

CALLING FUNCTIONS

```
1 def greet_user():  
2     print("Hello!")  
3 greet_user()  
4 greet_user()  
5 greet_user()
```

```
1 > "Hello"  
2 > "Hello"  
3 > "Hello"
```

PASSING ARGUMENTS TO FUNCTIONS

- Generally speaking we will want to pass something from our main thread to the function
- Function body: is indented, and runs procedurally from top to bottom

```
1 def my_addition(a, b):  
2     print(a + b)  
3 my_addition(1, 2)  
4 my_addition(4, 1)  
5 my_addition(5, -1)
```

FUNCTIONS SHOULD RETURN

- Generally speak, our functions should return some value to the main thread
- Sometimes we can't, and that's okay, but we should always ask ourselves why aren't we?

```
1 def my_addition(a, b):  
2     return a + b  
3 print(my_addition(1, 2))  
4 print(my_addition(4, 1))  
5 print(my_addition(5, -1))
```

PYTHON CAN RETURN MULTIPLE VARIABLES

- Sometimes your function wants to modify multiple variables at the same time

```
1  def get_guess(min_value, max_value,
2     moves):
3     guess = int(
4         input(
5             f'''Guess a number between
6             {min_value} and
7             {max_value} '''
8         )
9     )
10    moves = moves + 1
11    return guess, moves
12 guess, moves = get_guess(1, 10, guess)
```

Speaker notes