## Congratulations! You passed!

**Grade received** 100% **Latest Submission Grade** 100% **To pass** 80% or higher

Go to next item

1. Which of the following statements is *false* about Graph approach? 1 / 1 point

- ○ Faster compilation
- ◉ Easier debugging
- ○ Parallelism
- ○ Portability

Correct
Correct! This statement is false. Since operations don't execute until the Graph is fully designed, it can be tricky to debug.

2. Which of the following statements is *true* for *tf.cond*? 1 / 1 point

- ◉ Graph execution does not support *if/else* statements. To replicate that effect you use *tf.cond*
- ○ *tf.cond* is an alternative to using *if/else* statements in Graphs, as its execution is much faster than *if/else* statements.

Correct
Correct!

3. Consider the following code: 1 / 1 point

```
def increment_by_two(x):
  return x + 2

def multiple_increment(x, i):

  k = x
  for j in range(i):
    k = increment_by_two(k)

  return k
```

How do you convert *both* of these functions to execute in *Graph* mode? Check all that are true.

- ☑ By adding the decorator, @tf.function, only above the function definition of *multiple_increment*

Correct
Correct! If a function is decorated with '@tf.function', then the functions that it calls will also be included in graph mode.

- ☐ By adding the decorator, @tf.function, only above the function definition of *increment_by_two*
- ☐ By adding the decorator, @tf.autograph, above the definitions of both of the functions.
- ☑ By adding the decorator, @tf.function, above the definitions of both of the functions.

Correct
Correct!

4. Function written in Eager mode when converted to Graph accommodates different data types all in one, so you don't have to define similar functions for different data types. 1 / 1 point

- ○ False
- ◉ True

Correct
Correct!

5. Which of the following is the correct syntax to display the auto-generated AutoGraph code if your function name is *my_function*? 1 / 1 point

- tf.autograph.to_code(my_function)
- tf.autograph.code(my_function.python_function)
- tf.autograph.to_code(my_function.python_function)
- tf.autograph.code(my_function)

Correct
Correct!

6. Consider the following code, what will be the output? 1 / 1 point

```
def func(str):
  print(str)
  tf.print(str)

for i in range(3):
  func("Hello World!")
```

- Hello World!

  Hello World!

  Hello World!

  Hello World!

  Hello World!

  Hello World!
- Hello World!

  Hello World!

  Hello World!

  Hello World!
- Hello World!

  Hello World!

  Hello World!

Correct
Correct! Even though tf.print is used, we still get 6 print statements because the function is not decorated to run as a Graph.