Your grade: 100%

Your latest: **100%** • Your highest: **100%** • To pass you need at least 70%. We keep your highest score.

Next item →

1. Consider the following Python function definition: 1 / 1 point

```
def cube_root(val):
    """
    Given number, return the cube root of the number
    """
    return val ** (1 / 3)
```

Which of the expression below is a valid call to the function `cube_root`?

- (•) `cube_root(1.0)`
- ( ) `(cube_root 1.0)`
- ( ) `cube_root.value(1.0)`
- ( ) `cube_root "1.0"`

Correct
Correct.

2. Running the following program results in the error 1 / 1 point

`SyntaxError: bad input on line 5 ('return').`

Which of the following describes the problem?

```
def max_of_3(val1, val2):
    if val1 > val2:
        return val1
    else:
    return val2

def max_of_3(val1, val2, val3):
    return max_of_2(val1, max_of_2(val2, val3))
```

- ( ) Missing colon
- ( ) Misspelled keyword
- ( ) Extra parenthesis
- ( ) Missing parenthesis
- ( ) Wrong number of arguments in function call
- ( ) Misspelled variable name
- ( ) Misspelled function name
- (•) Incorrect indentation

Correct
Correct. The else clause of the function definition for `max_of_2()` should be indented, but it is not.

3. The following code has a number of syntactic errors in it. The intended math calculations are correct, so the only errors are syntactic. Fix these errors. 1 / 1 point

Once the code has been fully corrected, it should print out two numbers. The first should be 1.09888451159. Submit the **second** number printed in CodeSkulptor3. Make sure that you enter at least four digits after the decimal point.

```
define project_to_distance(point_x point_y distance):
    dist_to_origin = (pointx ** 2 + pointy ** 2) ** 0.5
    scale == distance / dist_to_origin
    print point_x * scale, point_y * scale

project-to-distance(2, 7, 4)
```

```
3.846095790563293
```

Correct
Correct.

4. A common error for beginning programmers is to confuse the behavior of `print` statements and `return` statements. 1 / 1 point

- `print` statements can appear anywhere in your program and print a specified value(s) in the console. Note that execution of your Python program continues onward to the following statement. Remember that executing a `print` statement inside a function definition does not return a value from the function.
- `return` statements appear inside functions. The value associated with the `return` statement is substituted for the expression that called the function. Note that executing a `return` statement terminates execution of the function definition immediately. Any statements in the function definition following the `return` statement are ignored. Execution of your Python code resumes with the execution of the statement after the function call.

As an example to illustrate these points, consider the following piece of code:

```
def do_stuff():
    """
    Example of print vs. return
    """
    print("Hello world")
    return "Is it over yet?"
    print("Goodbye cruel world!")

print(do_stuff())
```

Note that this code calls the function `do_stuff` in the last `print` statement. The definition of `do_stuff` includes two `print` statements and one `return` statement.

Which of the following is the console output that results from executing this piece of code? While it is trivial to solve this question by cutting and pasting this code into CodeSkulptor, we suggest that you first attempt this question by attempting to execute this code in your mind.

(•)
```
Hello world
Is it over yet?
```

( )
```
Hello world
```

( )
```
Hello world
Is it over yet?
Goodbye cruel world!
```

( )
```
Hello world
Is it over yet?
Goodbye cruel world!
Is it over yet?
```

Correct

5. Implement the mathematical function $f(x) = -5x^5 + 67x^2 - 47$ as a Python function. Then use Python to compute the function values $f(0)$, $f(1)$, $f(2)$, and $f(3)$. Enter the maximum (largest) of these four values calculated below. 1 / 1 point

A common error for this question is to fail to read the directions above carefully and submit your answer in the incorrect form. As a coder, always remember to note exactly what answers your code (and quiz questions) should produce.

```
61
```

Correct

6. When investing money, an important concept to know is compound interest. 1 / 1 point

The equation $FV = PV(1 + rate)^{periods}$ relates the following four quantities.

- The *present value* (*PV*) of your money is how much money you have now.
- The *future value* (*FV*) of your money is how much money you will have in the future.
- The *nominal interest rate per period* (*rate*) is how much interest you earn during a particular length of time, **before** accounting for compounding. This is typically expressed as a percentage.
- The *number of periods* (*periods*) is how many periods in the future this calculation is for.

Finish the following code, run it, and submit the printed number. Provide at least four digits of precision after the decimal point.

```
def future_value(present_value, annual_rate, periods_per_year, years):
    """
    Input: the numbers present_value, annual_rate, periods_per_year, years
    Output: future value based on formula given in question
    """
    rate_per_period = annual_rate / periods_per_year
    periods = periods_per_year * years

    # Put your code here.


print("$1000 at 2% compounded daily for 4 years yields $", future_value(1000, .02, 365
```

Before submitting your answer, test your function on the following example.

`future_value(500, .04, 10, 10)` should return 745.3177442824

Hint: If you are stuck on this question, try working problem 7 of the Practice Exercises for Functions.

```
1083.284693436586
```

Correct

7. For this final question, your task is to find the formula for a simple geometric problem using Google and then implement that formula in Python. While you may think that it is silly that we don't just give you the formula, scripting in Python often requires one to do a substantial amount of searching for information. This question requires you to practice this important task. 1 / 1 point

Write a Python function that computes the area of an equilateral triangle given the length of one of its sides. Search for a mathematical formula that specifies this relation and translate that formula into Python. **Hint**: The desired formula involves taking a square root. Remember that you compute a square root of a number in Python by raising that number to the 0.5 power using the `**` operator.

As a test, your area function should return an area of 1.73205080757 for an equilateral triangle with sides of length 2. Now, use this function to compute the area of equilateral triangle with sides of length 5. Enter this area as a number (and not the units) with at least four digits of precision after the decimal point.

```
10.8253175473
```

Correct