1. Write a function that prints a greeting message using a person's name.
Example Output:

```
Hello, Alice!
```

2. **Write a function that takes two numbers and returns their sum.**
Example Output:

```
3 + 5 → 8
```

3. **Write a function that returns the square of a number.**
Example Output:

```
square(4) → 16
```

4. **Write a function that prints whether a number is even or odd.**
Example Output:

```
7 → Odd number
8 → even number
```

5. **Write a function that returns the sum of three numbers.**
Example Output:

```
sum_three(2, 4, 6) → 12
```

6. **Write a function that calculates the area of a circle using the formula 3.14 × radius × radius.**
Example Output:

```
area_of_circle(3) → 28.26
```

7. **Write a function that finds and returns the largest of three numbers.**
Example Output:

```
find_largest(5, 9, 2) → 9
```

8. **Write a function that returns the average of three numbers.**
Example Output:

```
average(3, 6, 9) → 6.0
```

9. **Write a function that returns the sum of all numbers from 1 to n.**
Example Output:

```
sum_to_n(5) → 15
```

10. **Write a function that returns the sum of squares of numbers from 1 to n.**
Example Output:

```
sum_of_squares(3) → 14
```

11. **Write a function that counts how many vowels are in a given word.**
Example Output:

```
count_vowels("banana") → 3
```

12. Sum of Squares

Goal:
Create a program that calculates the sum of squares of all integers from 1 up to the number entered by the user.

Steps:

1. Create a new file called **`squares_calculator.py`**.
2. In that file, create a function named **`calculate_sum_of_squares`** that takes one number as a parameter and returns the sum of squares from 1 to that number. **1x1 + 2x2 + 3x3 + 4x4 = 30**
3. Create another file called **`sum_of_squares.py`**.
4. Import the **`calculate_sum_of_squares`** function from your other file.
5. Ask the user for a number and use your function to calculate and print the result.

Sample output:

```
Enter a number to sum the squares: 4
The sum of squares is 30
```

13. The Price is Right Game

Goal:
Make a simple guessing game where the user tries to guess the price of one of several random items.

Steps:

1. Create a file called **`price_is_right_games.py`**.
2. In that file, create a function named **`guess_items_price`** that takes one parameter named `guess`.
3. Inside that function, generate a list of random item prices and check if the user's guess is in the list.
4. Create another file called **`price_is_right.py`**.
5. Import your **`guess_items_price`** function into this file.
6. Use a loop to keep playing the game until the user chooses to quit (for example, by typing "n").
7. Print whether the user wins or loses after each round.

```
Welcome to the Price is Right!
Guess the price of one of the items (done): 1
The prices were: [3, 3, 3]
Sorry you lose!
Do you want to play again? (y/n): y
Guess the price of one of the items (done): 2
The prices were: [8, 8, 2]
```

14: Car Price Guessing Game

Goal:
Make a game where the user guesses the price of a car and gets feedback based on how close their guess is.

Steps:

1. In the **`price_is_right_games.py`** file, create a new function called **`guess_car_price`** that takes one parameter named `guess`.
2. Inside that function, generate a random actual car price and compare it with the user's guess.
3. Return different messages depending on how close the guess is (exact, within $1000, within $5000, or way off).
4. Create a new file called **`car_price_guessing_game.py`**.
5. Import your **`guess_car_price`** function into this file.
6. Create a loop that keeps asking the user for a guess until they choose to quit (for example, by typing "n").
7. Print the car's actual price and the message returned from your function.

Sample Output

Guess the price of the car (n to quit): 20000

The price of the car was: 13629

Way off!

Guess the price of the car (n to quit): 1111

The price of the car was: 12862

Way off!

Guess the price of the car (n to quit): n

14: Golf Score Calculator

Goal:
Ask the user to enter multiple golf scores, calculate their **average score**, and also calculate their **handicap**.

Steps:

1. Create a file called **`score_calculator.py`**.
2. In that file, create a function named **`calculate_average_scores`** that can take any number of scores (using `*args`) and return the average score.
3. In the same file, create another function named **`calculate_handicap`** that also takes any number of scores (using `*args`) and:
    - Calculates the average of the **best 5 scores**
    - Subtracts 72 from that average
    - Returns the handicap value
4. Create another file called **`golf.py`**.
5. Ask the user to keep entering scores until they type "q" to quit.
6. Use the two functions you created to print both the average score and the handicap.
7. If fewer than 5 scores are entered, display a message that says a handicap cannot be calculated.